

USER MANUAL

18.1

No Magic, Inc.
2015

All material contained herein is considered proprietary information owned by No Magic, Inc. and is not to be shared, copied, or reproduced by any means. All information copyright 1998-2015 by No Magic, Inc. All Rights Reserved.

CONTENTS

Contents 3

1 INTRODUCING MAGICDRAW 16
About MagicDraw and UML 16
MagicDraw Editions and Features 17
MagicDraw Editions 17
Other MagicDraw Features and Add-ons 18
MagicDraw Welcome Screen 24
MagicDraw News Reader 27
MagicDraw Documentation and Support 29
New and Noteworthy 29
Manuals and User Guides 29
Help 29
Notifications 29
Hints Associated to User's Actions 36
Productivity Tips Displayed in Progress Window 37
Tutorials 37
Other Documentation 39
Support 39

2 GETTING STARTED 44
Licensing Information 44
Removing Unused Licenses 45
Unlocking MagicDraw 46
Activating the commercial license after purchase 46
Commercial license activation process 48
Commercial license types 50
Host ID 51
Identification 52
Registration 53
Deactivation 54
User Registration 55
Registration Workflow 56
Bug Report 59
Troubleshooting 60
MagicDraw Configuration Files Location 61
Importing Configuration from Earlier Version 62
Disabling Inbound Network Activity 63
Updating 63
Update Information window 63

3 USING MAGICDRAW 65
Customizing and Selecting Perspective 65
Customizing MagicDraw Perspectives 66

Copyright © 1998-2015 No Magic, Inc.

Select Perspective dialog 67
Customize Perspectives dialog 69
Understanding MagicDraw User Interface (UI) 70
Menus 72
Toolbars 72
Model Browser 75
Customizing Environment Options 96
Using Environment Options dialog 97
Common elements in Environment Options dialog 97
Improving Performance 97
Background Tasks Manager 98
Memory Monitor 99
Memory Allocation 100
Look and Feel: Controlling the Interface 103
Assigning Shortcut Keys 104

4 WORKING WITH PROJECTS 105
Creating Projects 105
Creating a new project 105
Working with multiple projects 107
Creating a new project from the existing source code 108
Creating a new Use Case project 109
Working with Project Properties Dialog 110
Saving Projects 113
Autosave 114
Opening Projects 115
Opening projects saved with MagicDraw 17.0 or earlier 115
Importing Data from other UML Tools 116
Exporting Projects 117
Exporting Projects as Eclipse UML2 (v1.x / v2.x / v3.x / v4.x) XMI Files 117
Creating and Using Project Templates 119
Exporting Projects as Templates 119
Creating Custom Project Templates 120
Creating New Projects From Templates 122
NEW! Projects with File Attachments 124
Attaching, Reattaching, and Detaching Files 126
Working with Attached Files 127
Setting Project Options 129
Using Project Options dialog 130
Common elements in Project Options dialog 131
Searching 131
Find dialog 132
Quick Find dialog 137
Find TODO 141
Find and Replace dialog 141
Filtering 143
Project Partitioning 147
Decomposing Model 148
Composing Model 152
Managing Used Projects 155

Copyright © 1998-2015 No Magic, Inc.

Automated Module Usages 161
Maintaining Decomposed Model Integrity 164
Validating Project Integrity 175
MOF Support 178
Exporting projects to MOF files 179
Importing projects from MOF files 180
Ecore Support 180
Creating projects for Ecore modeling 181
Ecore modeling 184
Exporting projects to Ecore files 192
Importing projects from Ecore files 196
Working with Standard Profiles 196
Standard Profiles as System Resources 196
Standard Profiles in Teamwork Server 197

5 DIAGRAMMING 198
Working with Diagrams 198
Creating Diagrams 198
Opening Diagrams 201
Using Diagram Tabs 203
Specifying Diagram Properties 205
Specifying Diagram Style Properties 205
Diagram Name and its Context Name Synchronization 206
Diagram Frame 207
Drawing Diagram Shapes 209
Overviewing Other Diagrams 210
Table with Diagram Information 222
Changing the Diagram Type 223
Working with Symbols of Model Elements 224
Working with Paths and Relationships 227
Inserting a Shape on the Path 231
Creating Relations from the Model 232
Smart Manipulation 233
Smart Manipulators 233
Smart Manipulator Toolbar 235
Compartments 236
Creating Elements in Compartments on Shapes 237
Displaying and Suppressing Compartments on Shapes 238
Displaying or Hiding Elements in Compartments on Shapes 238
Smart Shape Sizing 239
Selection and Multiple Selections 240
Copying Text or Images to Diagrams 242
Nesting Image Shapes 243
Dragging, Copying, Cutting, and Pasting 244
Dragging Objects 244
Dragging Files 244
Dragging Images 246
Dragging Elements in the Specification Window 247
Dragging in Diagrams 249
Other Dragging Actions 252

Copyright © 1998-2015 No Magic, Inc.

Copying and Cutting Objects 253
Zooming 254
Using the Grid 255
Layout 256
Label layout in the diagram 257
Pusher and Magnet 259
Saving as Image 261
Save As Image Dialog 262
Properties Dialog 263
Printing 264
Print Dialog 265
Page Setup Dialog 268
Header/Footer Dialog 269
Print Options Dialog 270
Print Preview Window 271

6 WORKING WITH MODEL ELEMENTS 273
Specification Window 273
Functions of Specification Window 275
Specification Window Toolbar 277
Property Group Toolbar 278
Quick Filter 279
Specification Window Property Groups 281
Buttons Available in Specification Window 296
Default Property Values 296
Editing Property Values 298
Editing textual properties 299
Editing logical values 304
Selecting values 305
Cases of advanced editing 311
Formatting Symbols 317
Symbol Properties dialog 317
Displaying icon or image 319
320
NEW! Displaying rake icon on symbol 322
Copy and paste symbol style 323
Style Engine 324
Symbol Styles option group 324
Working with Properties Styles 326
Properties extension by diagram 327
Properties Inheritance 329
General Style Properties 329
Shape, Path and Diagram Properties 329
Stereotype properties 330
Customizing Lists of Properties and Commands 336
Defining Hyperlinks 340
Adding a hyperlink to the model element 341
Edit Hyperlink dialog 344
Owner of the Model Element 346
Owner display mode 347

Copyright © 1998-2015 No Magic, Inc.

Qualified name starting from model library 348
Changes of Relationships' Ownership when Client or Supplier is Moved to Another Owner 348
NEW! Creation of Elements from other Resources 349
Selecting an Element 352
Element Selection Views 353
Elements Multiple Selection 354
Searching for Elements in Element Selection Dialog 355
Creating New Elements 356
Cloning Selected Element 357
Selecting Port and Part Type 357
Refactoring 359
Converting Elements 360
Replacing Elements 363
Reversing a Relationship Direction 365
Swapping 366
NEW! Redefinition 368
Extracting 368
Extracting in Sequence Diagram 369
Extracting in Activity Diagram 375
Extracting in State Machine diagram 382
Extracting in Composite Structure diagram 389
Generic Numbering Mechanism 394
Working with Generic Numbering Mechanism 395
Element Numbering dialog 399
Smart Packages 402
Creating Smart Packages 403
Managing Contents of Smart Package 404
Smart Package Properties 408
Using Smart Packages in Your Model 408
Favorites 416
Adding, Removing, and Changing Order of Favorites 416
Manage Favorites dialog 419
HTML Editor 422
427
Advanced HTML Editor dialog 428
Copying/Opening Element URLs 429
Parameters Synchronization 430
Resolving Not Synchronized Elements 431
Synchronization Dialog Environment 431

7 TOOLS 434
Model Merge 434
Getting Started with Model Merge 435
Using Model Merge 438
Pattern Wizard 459
Creating Setters/Getters 464
Implementing or Overriding Operations 466
Model Transformation Wizard 468
Available Transformations 468

Copyright © 1998-2015 No Magic, Inc.

Working with Model Transformation Wizard 469
Transformation Type Mapping 474
Controlling Type Mapping Rule Behavior 476
Transformation Traces and Update 479
Profile Migration Transformation 480
Automatic Instantiation Wizard 485
Working with Automatic Instantiation Wizard 486
Resource Manager 494
Resource/Plugin Manager window 495
Spelling Checker 496
Spell checking as you type 497
Spell checking for the whole project or the selected scope 499
Setting the spell checking options 503
Defining properties of the customized element to be spell checked 505

8 MODEL ANALYSIS 507
Displaying Related Elements 508
Analyzing Usages and Dependencies 509
Understanding Used By 510
Understanding Depends On 510
Searching for Usages / Dependent Elements 510
Traceability 514
Creating Traceability Relations 515
Traceability Relations Representation 516
Navigating between Different Levels of Abstraction 522
Analyzing Traceability Relations 523
Predefined Traceability Rules 525
Derived Properties 537
Relation Map 544
Creating Relation Maps 545
Specifying Relation Map Criteria, Layout, and Depth 546
Specifying Other Options 547
Colors and Legend in Relation Map 548
Manipulations in Relation Map 549
Specifying Criteria for Querying Model 551
Getting Started with Specifying Criteria 552
Using Simple Navigation Operations to Specify Criteria 554
Using Metachain Navigation Operations to Specify Criteria 557
Using Find Operation to Specify Criteria 560
Creating New Operations 562
Symbol Usage in Diagrams 576
Projects Comparison 579
Comparing Projects 579
Diagrams Comparison 583
Text comparison 585
Metrics 587
Creating Metric Suites 588
Calculating Metrics 597
Metric Table Environment 601
Using Obsolete Metrics 604

Copyright © 1998-2015 No Magic, Inc.

CONTENTS

Analyzing Package Dependencies 604
Analyzing Dependencies Among Elements 608
Unresolved dependencies 611
Validation 612
Introduction 612
Constraint Types 613
Predefined Validation Suites 613
Validating 614
Validation Results Panel 616
Validation Rules 619
Validation Suites 625
Advanced Topics 628
Active Validation 632
Detecting Invalid Elements in Model 633
Solving Invalid Model 635
Changing Active Validation Options 636
Active Validation Suites 636
Customizing Active Validation 639
Validate element that has no representation in diagram 640
Model Visualizer 641
Class Diagram Wizard 642
Package Dependency Diagram Wizard 644
Package Overview Diagram Wizard 649
Hierarchy Diagram Wizard 653
Realization Diagram Wizard 656
Activity Decomposition Hierarchy Wizard 659
Content Diagram Wizard 662
Sequence Diagram from Java Source Wizard 665

9 UML DIAGRAMS 672
Architectural Views 673
Use Case View 673
Structural View 673
Behavioral View 673
Implementation View 674
Environment View 674
Class Diagram 674
Class Diagram Elements 675
Object Diagram 678
Object Diagram Elements 678
Use Case Diagram 679
Use Case diagram elements 679
Communication Diagram 681
Communication Diagram elements 681
Sequence Diagram 683
Sequence diagram elements 684
State Machine Diagram 689
State Machine Diagram elements 690
Displaying Inner Elements on State Machine Diagrams 693
Protocol State Machine Diagram 694

Copyright © 1998-2015 No Magic, Inc.

Protocol State Machine Diagram elements 694
Activity Diagram 697
Activity Diagram elements 697
Creating Behavior Automatically in Activity Diagram 703
Displaying Inner Elements on Activity Diagrams 704
Smart Activity Diagram Layout 705
Deleting from Activity diagrams 706
Component Diagram 706
Component Diagram elements 708
Deployment Diagram 710
Deployment Diagram elements 711
Package Diagram 713
Package Diagram elements 713
Profile Diagram 714
Profile Diagram elements 715
Composite Structure Diagram 717
Composite Structure Diagram elements 717
Working with Parts in Composite Structure Diagram 719
Mapping ports on a part type change 721
Displaying Internal Structure on Structured Classifiers 722
Interaction Overview Diagram 723

10 EXTENSION DIAGRAMS 725
Common Elements 726
User Interface Modeling Diagram 727
Why Prototyping Is So Important? 728
Working with User Interface Modeling Diagrams 728
User Interface Modeling 729
Case Studies for User Interface Modeling 743
Content Diagram 751
Content Diagram Elements 752
Robustness Diagram 752
Robustness Diagram Elements 753
Web Diagram 753
Web Diagram Elements 754
CORBA IDL Diagram 756
CORBA IDL Diagram Elements 757
WSDL Diagram 758
WSDL Diagram Elements 758
Time Diagram 759
Time Diagram Elements 759
Struts Diagram 760
Struts Diagram Elements 760
Networking Diagram 761
Networking Diagram Elements 762
Free Form Diagram 764
Relation Map Diagram 764
Dependency Matrix 764
Creating Dependency Matrix 765
Using Dependency Matrix 765

Copyright © 1998-2015 No Magic, Inc.

Dependency Matrix Environment 778
Generic Table 785
Creating Generic Tables 786
Using Generic Tables 793
Generic Table Environment 800
Instance Table 803
Creating Instance Tables 803
Manipulations in Instance table 814
Instance Table Environment 816
Glossary 819
Working with Glossary 820
Metric Table 822

11 MODELING ELEMENTS 823
Action 823
Action Properties 825
Accept Event Action 825
Call Behavior Action 827
Call Operation Action 829
Opaque Action 831
Send Signal Action 831
Working with Actions 834
Actor 836
Anchor 838
Association 839
Working with Associations 839
Association End 841
Association in Use Case Diagrams 848
Attribute 848
Behavior 851
Assigning Behaviors 852
Assigning Behavior Diagrams Automatically 853
Class 854
Working with classes 854
Creating A Structured Class 855
Design Patterns 856
Collaboration 856
Collaboration Use 857
Combined Fragment 858
Comment 859
Component 862
Connector 863
Constraint 865
Constraint path 869
Containment 871
Containment Tree 872
Data type 874
Enumeration 874
Primitive 875
Decision Node 875

Copyright © 1998-2015 No Magic, Inc.

CONTENTS

Dependency 875
- Template Binding dependency 876
- Abstraction 876
- Usage 877
- Package Merge 877
- Package Import 877
- Element Import 877
- Access 878
- Deployment 878

Deployment Specification 878
Event 879
Exception Handler 882
Extend 882
Flow Final Node 883
Fragment 883
- Alternative Fragment 883
- Combined Fragment 884

Function Behavior 885
Gate 885
Generalization 888
- Generalization Tree 889

Generalization Set 893
Include 899
Information Flow 900
Information Item 900
Instance Specification 901
Interaction Use 904
Interface 905
- Provided and Required Interfaces 906
- Provided and Required Interfaces in the Composite Structure diagram 906
- Provided/required interfaces in the Component diagram 908

Internal transition 908
Lifeline 909
- Working with Lifelines in Sequence Diagrams 910

Link 911
- Working with Links 911
- Active Validation Rules for Links 916

Manifestations 916
Message 917
- Common Actions with Messages 919
- Messages in Sequence and Time Diagrams 928
- Messages in Communication Diagrams 931

Model 935
Node 936
- Structured activity node 937
- Activity parameter node 938
- Expansion Region and Expansion Nodes 942
- If, Loop and Sequence Conditional Nodes 942

Note 943
Object Flow 945

Copyright © 1998-2015 No Magic, Inc.

Object Node 950
Opaque Behavior 951
Creating executable opaque behaviors 952
Operation 955
Package 957
Working with packages 958
Part 960
Pin 962
Port 965
Profile 972
Pseudo State 974
Initial 974
Final state 974
Terminate 974
Entry Point 974
Exit Point 975
Deep History 975
Shallow History 975
Junction 975
Choice 975
Fork and Join 975
Realization 975
Reception 977
NEW! Report Template 978
Send Signal Action 980
State 981
Changing State to Composite/submachine/orthogonal State 982
Composite State 982
Submachine 983
Adding connection point reference 983
Defining State Invariant 984
Assigning behavior to state 984
Stereotype 985
Subsystem 989
Swimlane 989
Tag 992
Template 994
Working With Templates 996
Transition 1000
Assigning event type 1001
Assigning Behavior Type 1003
Use Case 1004
Use Case Relations 1005
Use Case Description Profile 1006
Use Case Scenario 1009
Value Specification 1027
Auxiliary Diagram Symbols 1029
Image Shape 1030
Diagram Overview 1030
Separator 1032

Copyright © 1998-2015 No Magic, Inc.

Rectangular Shape 1033
Text Box 1034
Diagram Legend 1035

12 WORKING WITH SERVER PROJECTS 1039
Working in MagicDraw Teamwork Server 1040
Teamwork Concepts 1040
Working with Projects in Teamwork Server 1041
Locking Model Elements and Diagrams 1052
Unlocking Model Elements and Diagrams 1065
Teamwork Dialogs 1085
Data Partitioning on Teamwork Server 1100
Project Usage Map 1104
Working with Project Usage Map 1106
Project Usage Map window 1109
Solving Project Usage Problems 1113

13 APPENDIX I: SHORTCUT KEYS, MENUS, BUTTONS, AND ICONS 1124
Commonly Used Shortcut Keys 1125
Menu System 1127
File menu 1127
Edit menu 1129
View menu 1132
Layout menu 1133
Diagrams menu 1135
Options menu 1135
Tools menu 1136
Analyze menu 1138
Collaborate menu 1138
Window menu 1139
Help menu 1139
Main Toolbars 1140
File Toolbar 1141
Switch Projects Toolbar 1142
Perspectives Toolbar 1142
Validation Toolbar 1142
Create Diagram Toolbar 1143
Diagram Toolbars 1143
Model Browser 1145
Floating, Auto-hide, and Close Buttons 1145
Icons of general elements 1146
Icons of relationships 1154
Icons from Modules and Profile mechanism 1158

14 APPENDIX II: UML 2.5 SUPPORT 1159
Introduction 1159
Metamodel Changes 1159
New Property Names Added 1160
Properties Renamed 1160

Copyright © 1998-2015 No Magic, Inc.

Derivation Changes **1160**
Order Changes **1161**
Other Changes **1161**
Properties Removed **1161**
Notation Changes **1162**
Opening UML 2.4.1 models with MagicDraw 18.0 or later **1162**
Index 1163

Copyright © 1998-2015 No Magic, Inc.

1 INTRODUCING MAGICDRAW

In this chapter, you will find the introductory information about MagicDraw:

- "About MagicDraw and UML" on page 16
- "MagicDraw Editions and Features" on page 17
- "MagicDraw Welcome Screen" on page 24
- "MagicDraw News Reader" on page 27
- "MagicDraw Documentation and Support" on page 29

About MagicDraw and UML

Today's graphical software can be extremely complex in its structure and architecture, but that does not mean it must be difficult to use. We have learned much from the hardware industry, where everything you see is scattered pieces. This approach also works well in the software world – objects at a higher abstraction level are treated like "software pieces". To simplify the process further, we may use pictures instead of textual descriptions to show the relationships between objects in a complex system. Though pictures work better than textual descriptions alone, experience has proven that communicating complex ideas effectively requires more than simple flowcharts.

Early methodologies, such as Booch notation, OMT, and others served the same purpose: to graphically express the software's architecture information. However, these methodologies accomplished this in slightly different ways and with different levels of thoroughness. In 1994, Grady Booch, Jim Raumbaugh, and Ivar Jacobson came together to unify their varied methods and experience. The UML (Unified Modeling Language) was the fruit of their joint effort. UML was crafted with two objectives: To reflect the best practices of the industry and to demystify the process of software system modeling.

In short, UML provides standardized pictures of your software applications and allows your development team to quickly grasp the functionality contained within the application. UML is a language and a process with neutral notation. This means that you can use it to design your entire OO system in any programming language and any software development process.

The development of a model for an industrial-strength software system, prior to its construction or renovation, is as essential as having a blueprint for a large building. Good models are vital for effective communication among project teams.

In the early 1990s, the tools for OO software modeling emerged, followed by the development of the visual modeling approach. Visual modeling means that you first design your system by drawing diagrams (blueprints) and then employ tools to convert those diagrams into code. The value of such an approach is that the often tedious framework coding is done automatically, freeing the programmer to focus on design issues. The transition from the design to the implementation phase is smoother and more straightforward. Moreover, using the features of reverse engineering and code generation, the developer can move back and forth between the code and the design that is being expressed in the diagrams.

Today, visual modeling tools provide many features that replace some of the more tedious tasks for the designer, programmer, and documentation writer. Some of the leading tools provide so-called round-trip code engineering capabilities – the structure of reverse engineered code is changed in the modeling tool and is generated back without the implementation of specific information (e.g. method bodies, comments) being lost.

MagicDraw is a visual UML modeling and CASE tool with teamwork support. Designed for Business Analysts, Software Analysts, Programmers, QA Engineers, and Documentation Writers, this dynamic and versatile

Copyright © 1998-2015 No Magic, Inc.

development tool facilitates analysis and design of Object Oriented (OO) systems and databases. It provides the industry's best code engineering mechanism (with full round-trip support for Java, C#, C++, WSDL, XML Schema, and CORBA IDL programming languages), as well as database schema modeling, DDL generation and reverse engineering facilities.



As of version 18.0, MagicDraw supports UML 2.5. For the detailed information about supported changes of UML specification from version 2.4.1 to 2.5, see "Appendix II: UML 2.5 Support" on page 1159.

MagicDraw Editions and Features

A detailed list of MagicDraw features can be found at http://www.nomagic.com/products/magicdraw/magicdraw-features.html

MagicDraw Editions

MagicDraw Personal Edition

MagicDraw Personal Edition contains powerful UML diagramming capabilities, including full UML 2 support and extensibility features, basic reporting functionality, and image export. Exported files are stored in XMI format. As of the version 17.0.1 MagicDraw supports XMI 2.4 format.

All model elements can be accessed via the MagicDraw Open API.

In this edition, you will find everything you need to draw, edit, and publish your UML models.

Personal Edition is available only in a standalone version and is not designed for use with MagicDraw Teamwork Server.

MagicDraw Standard Edition

MagicDraw Standard Edition provides all of the Features of Personal Edition and adds WAE, content, and Robustness diagrams. Standard Edition also adds model analysis and facilitation features, customizable and extendable patterns, integrations with most popular IDEs, and a set of predefined model templates and UML profiles.

Standard Edition supports UNISYS XMI and the latest Model Driven Architecture (MDA) tool offerings. UNISYS XMI diagramming extensions allow the interchange of MagicDraw models with other UML modeling tools. Since of the version 17.0.1 MagicDraw supports XMI 2.4 format.

Standard Edition is available in standalone, floating and mobile license versions and is fully compatible with MagicDraw Teamwork Server.

Standard Edition is ideally suited for analysts and architects who need various model extensions and modeling facilitations.

MagicDraw Professional Edition

Professional Edition is built on the Standard Edition capabilities and is available in one of three programming language specific versions-Java, C++ and C#. In addition to the Standard Edition features, Professional Edition adds code generation and reverse engineering functionality. Depending on the language version selected, the user will receive:

Copyright © 1998-2015 No Magic, Inc.

- **Java version** - Code engineering for Java, Java bytecode. Integration with Java IDEs.
- **C++ version** - Code engineering for C++.
- **C# version** - Code engineering for C#, CIL (MSIL).

Professional Edition is ideal for anyone who wants to generate code from an existing model or create a UML model from an existing project.

MagicDraw Architect Edition

The Architect Edition is specially packaged to provide the optimal price and technical features necessary for architects that do not need the full capabilities of the Enterprise Edition. This edition combines the common functionality of the Standard Edition together with some powerful options from the Enterprise Edition. These include: advanced modeling facilitations and analysis, reverse engineering and code generation for DDL, WSDL, CORBA IDL and XML. Architects have less need for IDE integrations as well as Java and C++ code engineering, so these capabilities are not included.

MagicDraw Enterprise Edition

MagicDraw Enterprise Edition represents the top of the line in the MagicDraw family of products, as well as the ultimate solution for all your modeling needs. Enterprise Edition combines all of the functionality of the Personal and Standard Editions, and all three versions of the Professional Edition, into a comprehensive state-of-the-art UML programming solution. But the Enterprise Edition does not stop there, adding code engineering and diagramming functionality in CORBA IDL, WSDL and XML schema. For working with DB structure, Enterprise Edition not only provides code engineering and diagramming, but also provides structure retrieval via JDBC.

Enterprise Edition is a must when working with multiple development technologies and databases.

The MagicDraw family of award-winning products represents the most powerful and best value in the UML modeling industry today.

MagicDraw Reader Edition

MagicDraw Reader Edition is made for reading and previewing UML models created with MagicDraw and it is free of charge. It is extremely useful when you want to share your ideas expressed in UML with partners, colleagues, or clients, who do not have a copy of MagicDraw. Printing and image export capabilities are also included.

Since MagicDraw version 14.0, Reader Edition has the ability to open and review Teamwork Server projects.

Other MagicDraw Features and Add-ons

Reports Generation

You will find a complete description of the MagicDraw Report Wizard, related OpenAPI, and tutorial in MagicDraw ReportWizard UserGuide.pdf.

Floating License

The Floating license agreement does not limit the number of clients you can install on different computers. It only limits the number of applications that can run at the same time. To control loaded applications, a server is required. The server can be installed on several computers, but simultaneously can be started only on the one of them. The license key of the floating server provides information to the server about how many applications may run simultaneously for the particular MagicDraw edition.

Copyright © 1998-2015 No Magic, Inc.

The Administrator's Console is used to manage client connections and configure the server.



For more information about managing floating licenses, see MagicDraw FloatingLicense UserGuide.pdf.

Teamwork Server

With MagicDraw Teamwork Server, you can assign as many developers as needed to work simultaneously on the same project using multiple workstations. The resulting server project is saved on the server for sharing by other MagicDraw applications. Users with administrator rights can create new users by giving them their own login name and various permissions to work on projects. Depending on permissions, users can update, commit, edit, create, and delete model elements, diagrams, and projects.

To enable Teamwork support, you should install and run MagicDraw Teamwork Server. Each MagicDraw application acts as a client of Teamwork Server.

Teamwork Server functionality is available with MagicDraw Standard, Professional, Architect, and Enterprise editions. MagicDraw Reader Edition is allowed to open and review server projects.

As of version 17.0 you can use the secure connection (SSL) while working with Teamwork Server.



- For more information about working in a collaborative environment, see "Working with Server Projects" on page 1039.
- For MagicDraw Teamwork Server administration instructions, refer to MagicDraw Teamwork UserGuide.pdf.

Code Engineering

MagicDraw code engineering provides a simple and intuitive graphical interface for merging code and UML models, as well as preparing both code skeletons out of UML models and models from code.

MagicDraw code engineering features can be very useful in several situations:

- You already have code that needs to be reversed to a model.
- You wish to have the implementation of the created model.
- You need to merge your models and code.

The tool may generate code from models and create models out of code (reverse). Changes in the existing code can be reflected in the model, and model changes may also be seen in your code. Independent changes to a model and code can be merged without destroying data in the code or model.

MagicDraw code engineering supports Java, Java Bytecode, C++ (ANSI, CLI, Managed), C#, CIL, CIL Disassembler, CORBA IDL, DDL (Cloudscape, DB2, Microsoft Access, Microsoft SQL server, MySQL, Oracle, Pervasive, Pointbase, PostgreSQL, Standard SQL, Sybase), XML Schema, and WSDL.



For more information on working with code engineering, see MagicDraw CodeEngineering UserGuide.pdf.

OpenAPI

This document describes the MagicDraw Open Java API and provides instructions on how to write your own plug-ins, create actions in the menus and toolbars, change UML model elements, and create new patterns.



For more information on working with OpenAPI, see MagicDraw OpenAPI UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

Integrations

MagicDraw supports the following integrations:

- Eclipse
- RAD
- oAW
- AndroMDA
- CVS



For more information about MagicDraw integrations, see MagicDraw Integrations UserGuide.pdf.

MagicDraw Customization

MagicDraw introduces several advanced customization engines, based on UML Profiles:

- Custom Diagram Wizard allows creating your own diagram types for custom profile. You may include your own toolbars, stereotyped elements, symbol styles, and custom smart manipulators. Such customization is saved in the special “diagram descriptor” that could be exchanged between users. This allows others to use your custom diagrams.
- Domain Specific Language Customization Engine (DSL customization engine) allows “tuning” domain specific profiles, customizing multiple GUI, model initialization, and semantic rules, creating your own specification dialogs. DSL customization is model-driven approach, based on UML profiling. Customization is saved as a UML model.
- Advanced UML Profiling allows the use of some profiling enhancements that are not defined in UML, but helps to solve some common problems like tag grouping, unwanted stereotypes, tags hiding, etc.



For more information about MagicDraw customization, see MagicDraw UMLProfiling&DSL UserGuide.pdf.

MagicDraw Plugins

For the complete list of MagicDraw plugins, see http://www.nomagic.com.



For MagicDraw and plugin compatibilities, refer to http://www.nomagic.com/support/compatibility.html.

UPDM plugin

The UPDM plugin for MagicDraw UML/SysML modeling solution fully supports building integrated enterprise architectures meeting DoDAF and MODAF requirements ensuring mission critical project success. The plugin supports all DoDAF 1.5, DoDAF 2.0, and MODAF 1.2 viewpoints and views dependent on the selected user environment. Each user environment provides architecture framework specific concepts, artifacts, new project templates, samples, and architecture framework specific usability features. A user environment can be changed any time by fully converting model to meet requirements of the selected architecture framework.



For more information about UPDM plugin, see http://www.nomagic.com/products/magicdraw-addons/updm-plugin.html

SysML plugin

The SysML plugin includes SysML profile, template, all SysML diagrams, SysML samples project, SysML usability features, and System Engineer perspective. The System Engineer perspective (the specific mode of

Copyright © 1998-2015 No Magic, Inc.

the MagicDraw user interface for SysML modeling) includes SysML specific menus, toolbars, diagrams, specification dialogs and user interface.



For more information about SysML plugin, see http://www.nomagic.com/products/sysml-plugin.html

Cameo DataHub

Cameo DataHub™ is a tool that allows the user to import, export, synchronize, and make references between Cameo Requirements+™, MagicDraw, SysML Plugin, DoDAF Plugin, Telelogic DOORS, Rational RequisitePro, and CSV files.



For more information about Cameo DataHub plugin, see http://www.nomagic.com/products/cameo-datahub.html

Cameo SOA+ plugin

Cameo SOA+™ leverages the Unified Modeling Language® (UML®) along with the latest SOA modeling standard, SoaML™, to provide both architects and developers an integrated solution for creating optimal SOA architectures and implementations. Cameo SOA+™ brings together SOA at both the business and technology levels to address the full spectrum of services. From Enterprise and Business Architectures to implementing, using and composing services on your favorite enterprise service bus (ESB) or application server, this integrated plug-in is versatile enough for both personal and team-based development. SoaML helps create and use services based on new and existing capabilities using composite services.



For more information about Cameo SOA+ plugin, see http://www.nomagic.com/products/cameo-soa.html

Cameo Data Modeler plugin

Cameo Data Modeler™ plugin for MagicDraw® provides support for Entity-Relationship modeling. It expands previous Business Entity-Relationship diagram (a limited form of ER diagram) to full featured Entity-Relationship diagram - including extended entity-relationship concepts - generalization support.



For more information about Cameo Data Modeler plugin, see http://www.nomagic.com/products/magicdraw-addons/cameo-data-modeler.html

Cameo Business Modeler plugin

Cameo Business Modeler™ plugin for MagicDraw provides support for BPMN 2.0 profile, diagrams, user perspective, usability features for BPMN modeling, manual, samples, and import from BPMN 1.1 models that were created with MagicDraw. All four BPMN 2.0 diagrams are supported.



For more information about Cameo Business Modeler plugin, see http://www.nomagic.com/products/magicdraw-addons/cameo-business-modeler-plugin.html

Merge plugin

Model Merge enables copying changes between different project versions. This functionality is usually needed when there are several branches that reflect different releases or versions of the product, e.g. when certain fixes have to be copied from a release branch to the mainstream development.



For more information about Merge plugin, see http://www.nomagic.com/products/magicdraw-addons/merge-plugin.html

Copyright © 1998-2015 No Magic, Inc.

ParaMagic plugin

ParaMagic plugin using the quantitative information and constraint relationships displayed in SysML diagrams, model-builders can run simulations from the earliest stages of system design. In traditional domains of system engineering like aerospace and transportation, users can explore system performance, estimate cost and allocate resources. Developers leveraging MagicDraw's DoDAF and business modeling capabilities can add parametric simulation using SysML submodels for defense planning, business process analysis and computational finance.



For more information about ParaMagic plugin,
see http://www.nomagic.com/products/sysml-plugin/paramagic-plugin.html

Methodology Wizards plugin

Methodology Wizards Plugin automates modeling tasks and extends methodology support in MagicDraw. This Wizard guides you through model creation process according to a set methodology.



For more information about Methodology Wizards plugin,
see http://www.nomagic.com/products/magicdraw-addons/methodology-wizards.html

MARTE Profile

This specification of a UML® profile adds capabilities to UML for model-driven development of Real Time and Embedded Systems (RTES). This extension, called the UML profile for MARTE (in short MARTE), provides support for specification, design, and verification/validation stages. This new profile is intended to replace the existing UML Profile for Schedulability, Performance and Time.



For more information about MARTE Profile,
see http://www.nomagic.com/products/magicdraw-addons/marte-profile.html

CSV Import plugin

The CSV Import plugin is a MagicDraw plugin that will read values in a comma separated values (CSV) file and create model elements, diagrams and relationships from that data. MagicDraw users will have the capability to create MagicDraw models from their data stored in spreadsheets, relational databases and other repositories.



For more information about CSV Import plugin,
see http://www.nomagic.com/products/magicdraw-addons/csv-import-plugin.html

SPEM

SPEM - Adopted standard for software engineering process description. Standard provides generic elements those allow to describe any software development process. The purpose of SPEM is to support the definition of software development processes specifically including those processes that involve or mandate the use of UML. The SPEM plugin includes SPEM profile, template, all SPEM diagrams and properties customization.



For more information about SPEM plugin,
see http://www.nomagic.com/products/magicdraw-addons/spem-plugin.html

SoaML Profile

Service oriented architecture Modeling Language (SoaML) profile.The goals of SoaML are to support the activities of service modeling and design and to fit into an overall model-driven development.



For more information about SoaML Profile, see http://www.nomagic.com/products/magicdraw-no-cost-add-ons/other-no-cost-add-ons.html

Copyright © 1998-2015 No Magic, Inc.

Enterprise Architect Import Plugin

MagicDraw has the capability to import UML models that conform to various XMI versions (including XMI 2.4) from other tools. Sparx Systems Enterprise Architect (EA) is one of those tools. EA provides a XMI export functionality which makes it possible to export UML2.4(XMI2.4). However, the XMI exported from EA contains some XMI conflicts and EA-specific data that do not conform to UML standards. The main purposes of Enterprise Architect Import Plugin are thus to solve XMI conflicts between EA and MagicDraw that cause problems when loading the XMI to MagicDraw and also to transform some EA-specific data to the form of UML elements with stereotypes.

From models that are created with Enterprise Architect, you can import the following items:

- UML models
- Profiles
- Stereotype usage information
- EA-specific data, such as constraints, requirement, scenarios, files, external requirements, changes, and issues
- Diagram information, such as Class diagrams, Package diagrams, Object diagrams, Component diagrams, Deployment diagrams, Use Case diagrams, Activity diagrams, Sequence diagrams, Communication diagrams, State Machine diagrams, Composite Structure diagrams, and Interaction Overview diagram.



For more information about importing an EA exported XMI, see MagicDraw EnterpriseArchitectImportPlugin UserGuide.pdf.

Data Converters



For MagicDraw and data converter compatibilities, refer to http://www.nomagic.com/support/compatibility.html.

Cameo Inter-Op

Cameo Inter-Op™ provides customers both import and export functionality between MagicDraw and IBM Rational® Rhapsody or MagicDraw and System Architect. Cameo Inter-Op supports both semantics and diagrams, ensuring data integrity is maintained between MagicDraw and these products.



For information about the tool, see http://www.nomagic.com/products/cameo-inter-op.

MagicDraw RConverter

MagicDraw RConverter is the most comprehensive tool to convert Rational Rose formats to MagicDraw or other open formats available in the market. MagicDraw RConverter also provides the capability to automatically generate the vast majority of symbol properties, specifications, diagram layouts, user defined properties and stereotype images.

MagicDraw RConverter provides a seamless way to convert a Rational Rose Model format (*.mdl) file to a MagicDraw-supported format (*.xmi) file.



For information about the tool, see http://www.nomagic.com/products/magicdraw-addons/magicdraw-rconverter.html.

MagicDraw RSXConverter

MagicDraw RSXConverter is the most comprehensive tool to convert IBM® Rational® Software Architect (RSA) and IBM® Rational® Software Modeler (RSM) formats to MagicDraw. MagicDraw RSXConverter also

Copyright © 1998-2015 No Magic, Inc.

provides the capability to automatically generate the vast majority of symbol properties, specifications, diagram layouts, stereotype and stereotype images.

MagicDraw RSXConverter provides a seamless way to convert an RSA or RSM format (*.emx, *.epx, *.efx) file to a MagicDraw-supported format (*.mdxml) file.



For more information about the tool, see http://www.nomagic.com/products/magicdraw-addons/magicdraw-rsxconverter.html.

MagicDraw Welcome Screen

The Welcome screen appears on the MagicDraw window by default when all projects are closed. It allows for managing projects as well as provides a quick access to the latest No Magic news, the product related news, available resources, and sample projects.

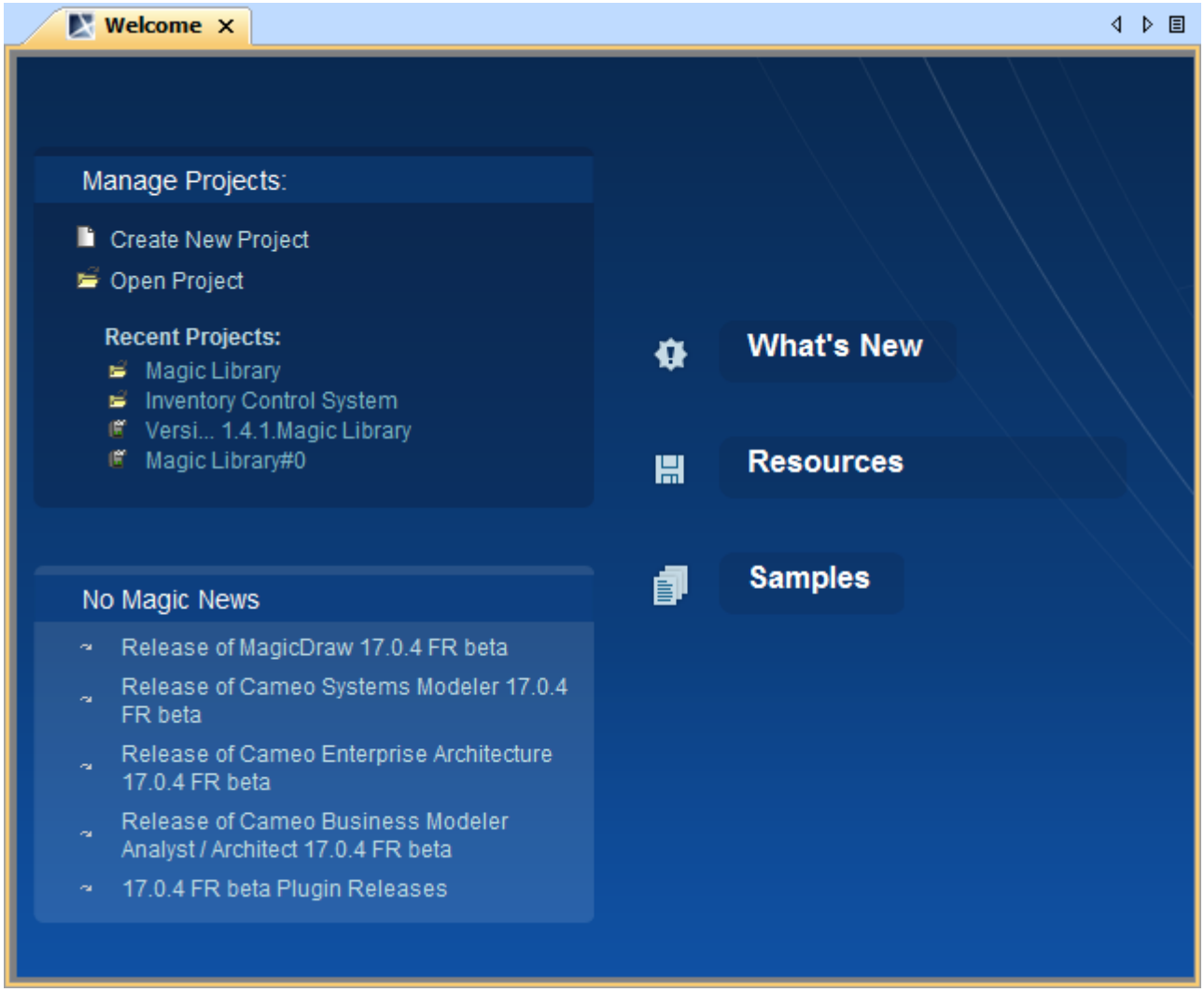


Figure 1 -- MagicDraw Welcome screen

You can turn off showing the Welcome screen by default. To do so, just find the **Show Welcome Screen** option under the **General** options group in the **Environment Options** dialog and change its value to *false*. The Welcome screen will no longer appear on the MagicDraw window after all the projects are closed.

In case showing the Welcome screen by default is turned off, you can open it manually. To do so, you only need to select **Show Welcome Screen** from the **Help** menu after all the projects are closed.

Copyright © 1998-2015 No Magic, Inc.

The Welcome screen includes

- Manage Projects area
- No Magic News list
- What's New page
- Resources page
- Samples page

The first two items are visible at the left side of the Welcome screen. To explore the content of the rest items, you have to open them first. How to do this, learn while reading the following procedures.

To open the **What's New**, **Resources**, or **Samples** page on the Welcome screen

- Click **What's New**, **Resources**, or **Samples** appropriately at the right side of the Welcome screen (see the following figure).

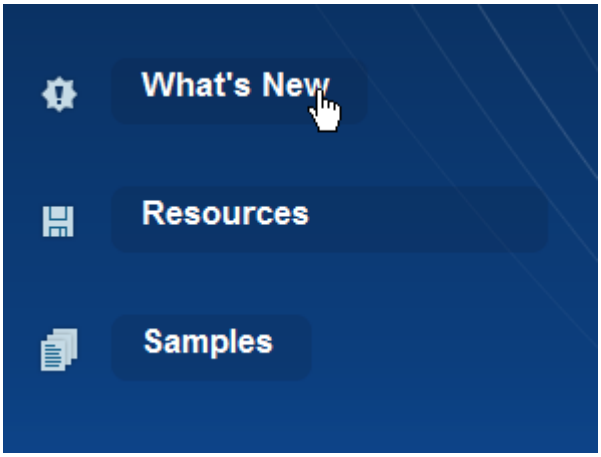


Figure 2 -- Fragment of Welcome screen. Opening What's New page

Copyright © 1998-2015 No Magic, Inc.

To switch between pages on the Welcome screen

- At the top of an active page, click a button with the appropriate page name (see the following figure).

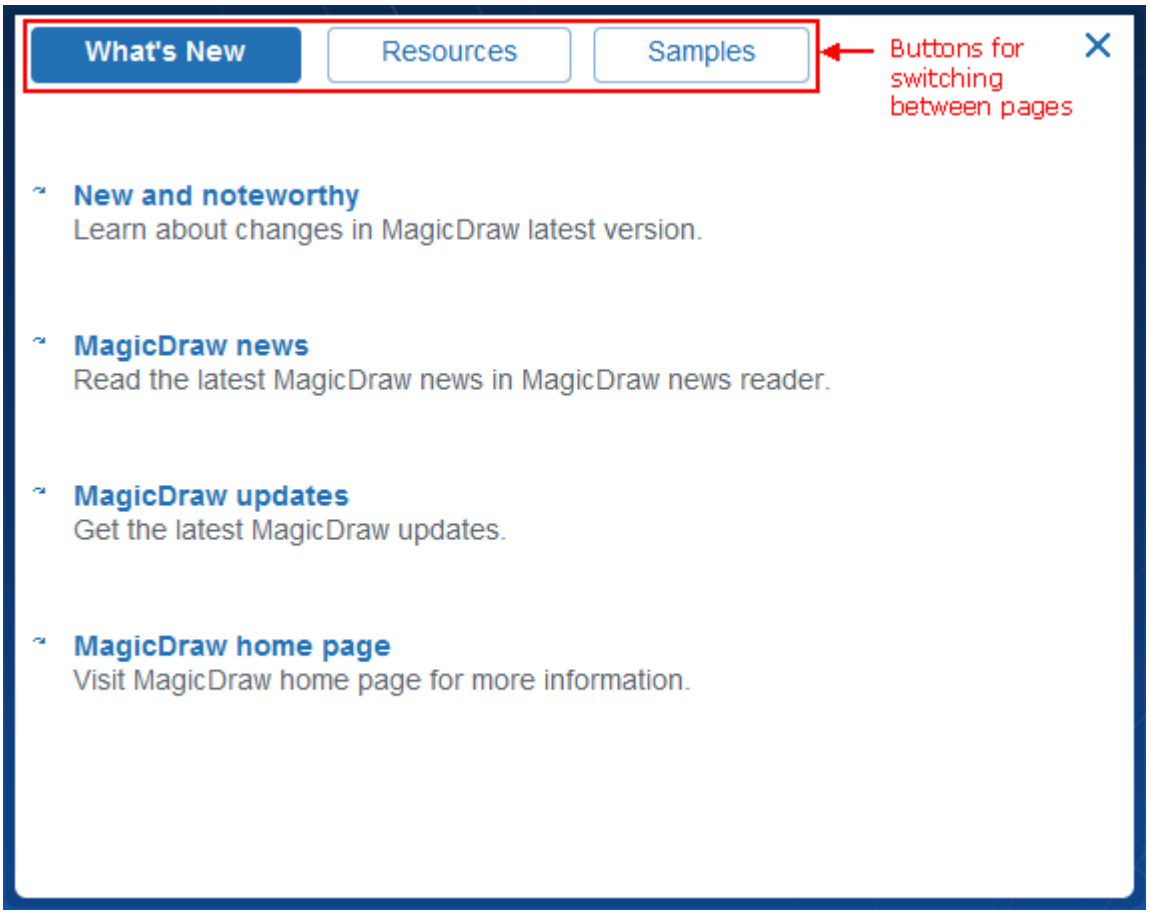


Figure 3 -- What's New page on Welcome screen

All the items of the Welcome screen are described in detail as follows.

Manage Projects area

On the **Manage Projects** area, you can do the following:

- Create a blank project. Click **Create New Project** for this.
- Open an existing project. Click **Open Project** for this.
- Open an existing project that has been recently closed. Click a project name under **Recent Projects**.

No Magic News list

To read a particular item on the **No Magic News** list

- Click the item. The news will be displayed in the News Reader.

TIP!

For more information, see "MagicDraw News Reader" on page 27.

What's New page

The **What's New** page provides a quick access to

Copyright © 1998-2015 No Magic, Inc.

- New and Noteworthy information of the latest product version at the official No Magic website.
- Product related news in the News Reader.



For more information, see "MagicDraw News Reader" on page 27.

- Available updates of the product.
- Product description at the official No Magic website.

Resources page

The **Resources** page provides a quick access to

- Descriptions of available plugins, profiles, templates, and other resources at the official No Magic website.
- Installation of available resources. You will be able to download and install available resources using the **Resource/Plugin Manager** window.



For more information, see "Resource Manager" on page 494.

- The folder wherein the product printable documentation is stored.
- Online demos introducing the variety of product features at the official No Magic website.
- No Magic community forum.

Samples page

The **Samples** page provides a quick access to sample projects that are provided by No Magic. Click a sample project name to open it. The same projects can be found in *<MagicDraw installation folder>\samples*.

MagicDraw News Reader

Information about the latest MagicDraw events is provided in the new MagicDraw News Reader. The News Reader is accessible from the **Help** menu, the **News Reader** command (see Figure 4 on page 28).

MagicDraw News Reader informs about:

- No Magic News. All news regarding company news, product news, new services provided, etc.
- New Versions. Messages about new MagicDraw releases, betas, service packs, plug-ins.
- Resources. Messages about MagicDraw resources updates.

When some news is available, a small envelope icon will be displayed at the right of MagicDraw status bar. Click on this icon to invoke the MagicDraw News Reader (see Figure 5 on page 29).

Reading news in the MagicDraw News Reader

Select the news channel at the left side of the MagicDraw News Reader and then select one of the news in the list. Bellow the title of the selected news, the description is presented. Unread messages are displayed in bold.

Click the **Open in Browser** link to read description on www.nomagic.com website.

To refresh news, click the  **Refresh** button at the left top corner of the **MagicDraw News Reader** window.

Copyright © 1998-2015 No Magic, Inc.

To mark all the selected items as read, click the  **Mark item as read** button.

Setting options of the MagicDraw News Reader

To change the checking period, change the **Check for News** property in the **General** section of the **Environment Options** dialog. Property provides the following options:

- Once a day
- Once a week
- Once a month
- Do not check.

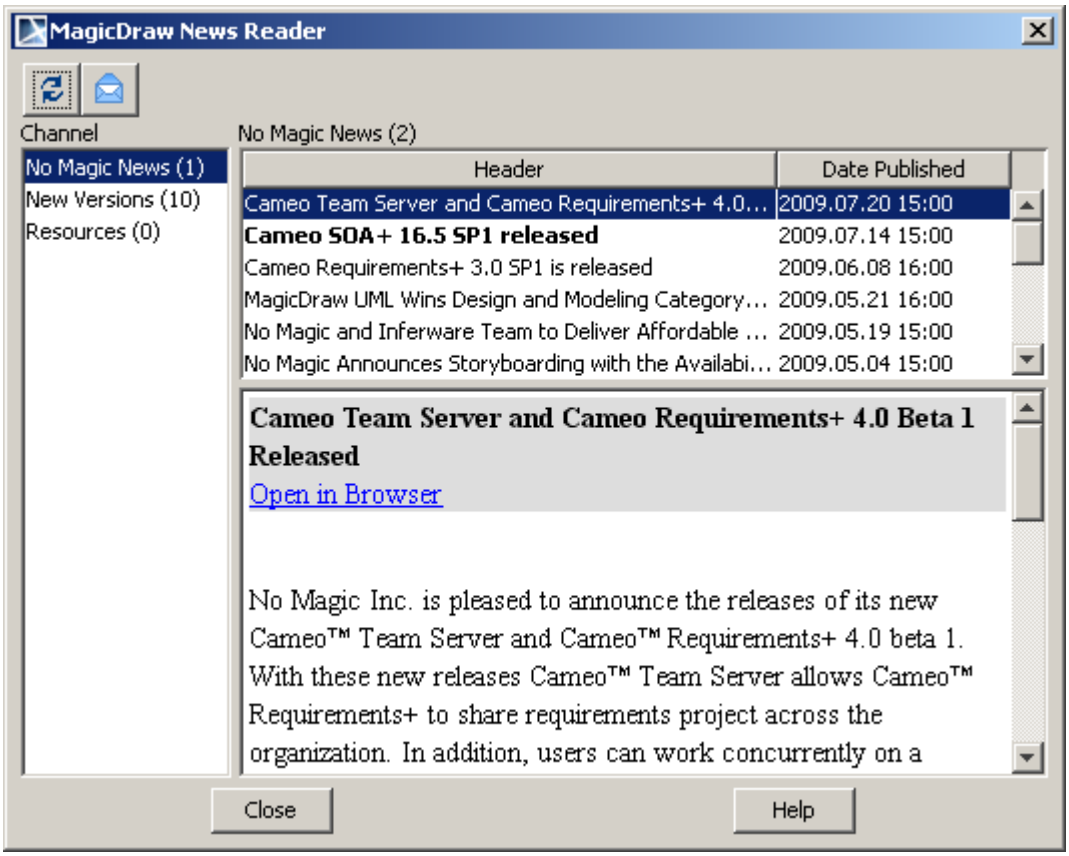


Figure 4 -- MagicDraw News Reader

Copyright © 1998-2015 No Magic, Inc.

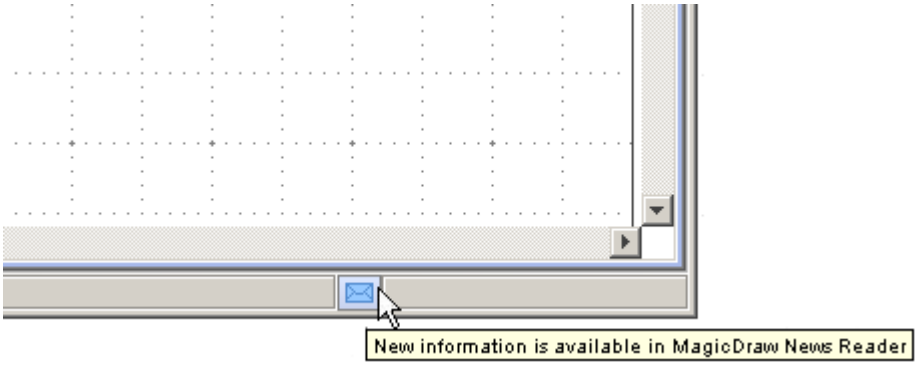


Figure 5 -- Envelope icon informs about news in MagicDraw News Reader

MagicDraw Documentation and Support

MagicDraw provides several kinds of documentation. Choose the way you want to learn.

New and Noteworthy

For the information about what is new in the latest version of MagicDraw, see New and Noteworthy at http://www.nomagic.com/news/new-noteworthy/magicdraw-noteworthy.html.

Manuals and User Guides

You can find the MagicDraw manual and user guides in <*MagicDraw installation directory*>*manual* or download the material from http://www.nomagic.com/support/documentation.html.

Help

The integrated help within MagicDraw is based on JavaHelp. MagicDraw help provides detailed descriptions of all MagicDraw dialog boxes, commands, and shortcut menus. You will also find a How-to list, as well as main descriptions and examples of all UML model elements.

Notifications

Notifications inform about various environment and project specific messages as well as messages received from team members. In addition, it allows you to perform the associated actions with a simple click on the link in the notification message. You may receive notification messages in the lower-right corner of MagicDraw window and in various dialogs.

To open the Notification Window, which stores the history of all notifications, select **Window** > **Notification Window** from the main menu.

Copyright © 1998-2015 No Magic, Inc.

See an example of the Notification Window and the notification message that appears in the lower-right corner of MagicDraw window.

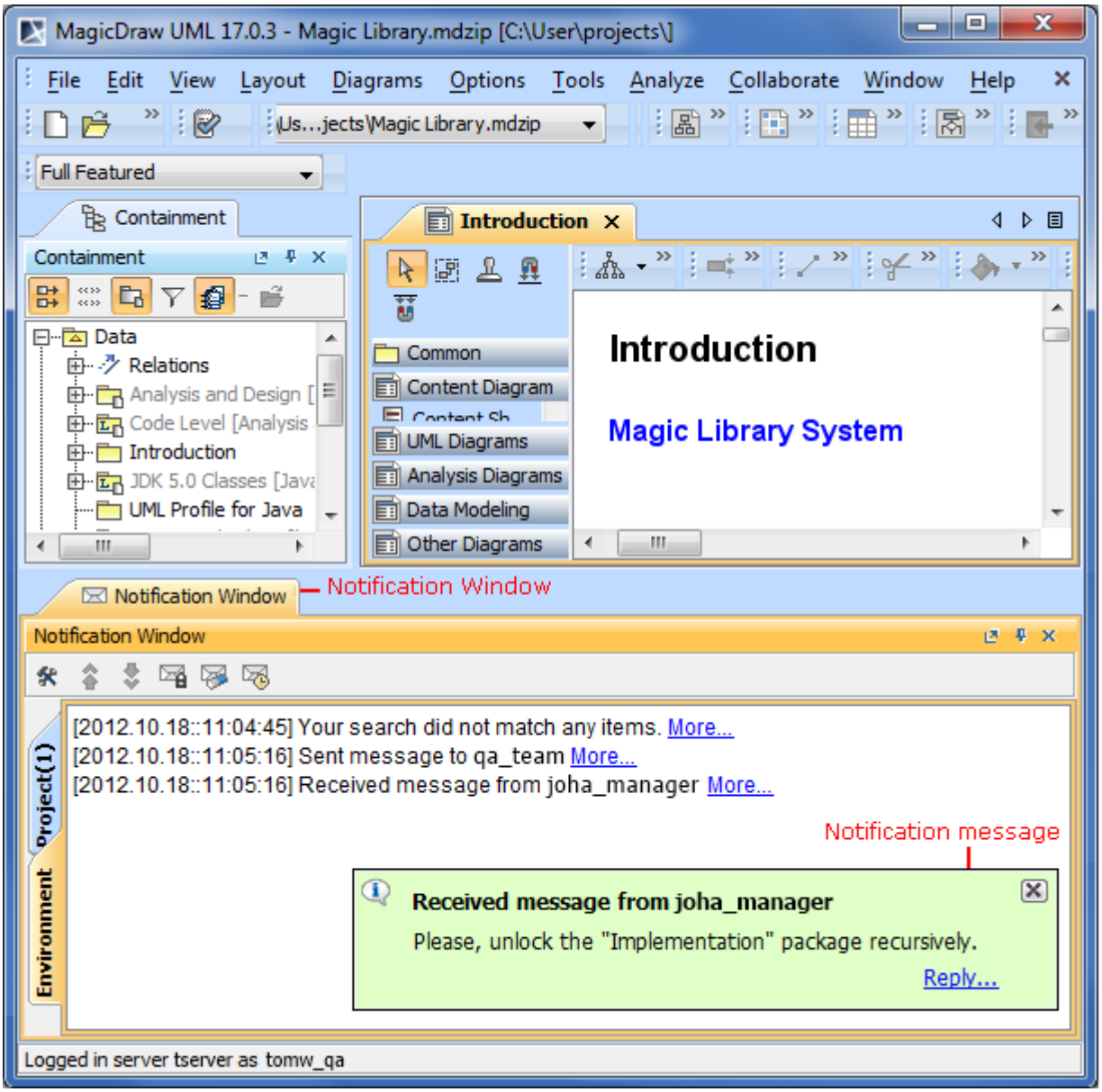


Figure 6 -- Example of Notification Window and notification message

There are two sections about the notifications:

- "Notification messages" on page 31
- "Notification Window" on page 32

Copyright © 1998-2015 No Magic, Inc.

Notification messages

See an example of the notification message that appears in the lower-right corner of MagicDraw and the notification message that is displayed in the dialog.

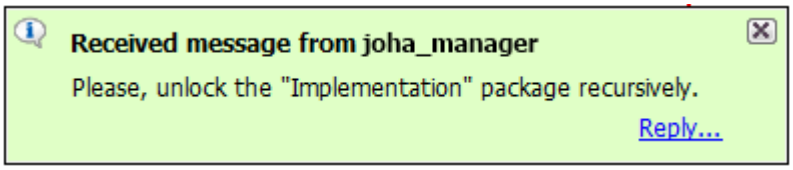


Figure 7 -- Example of notification message

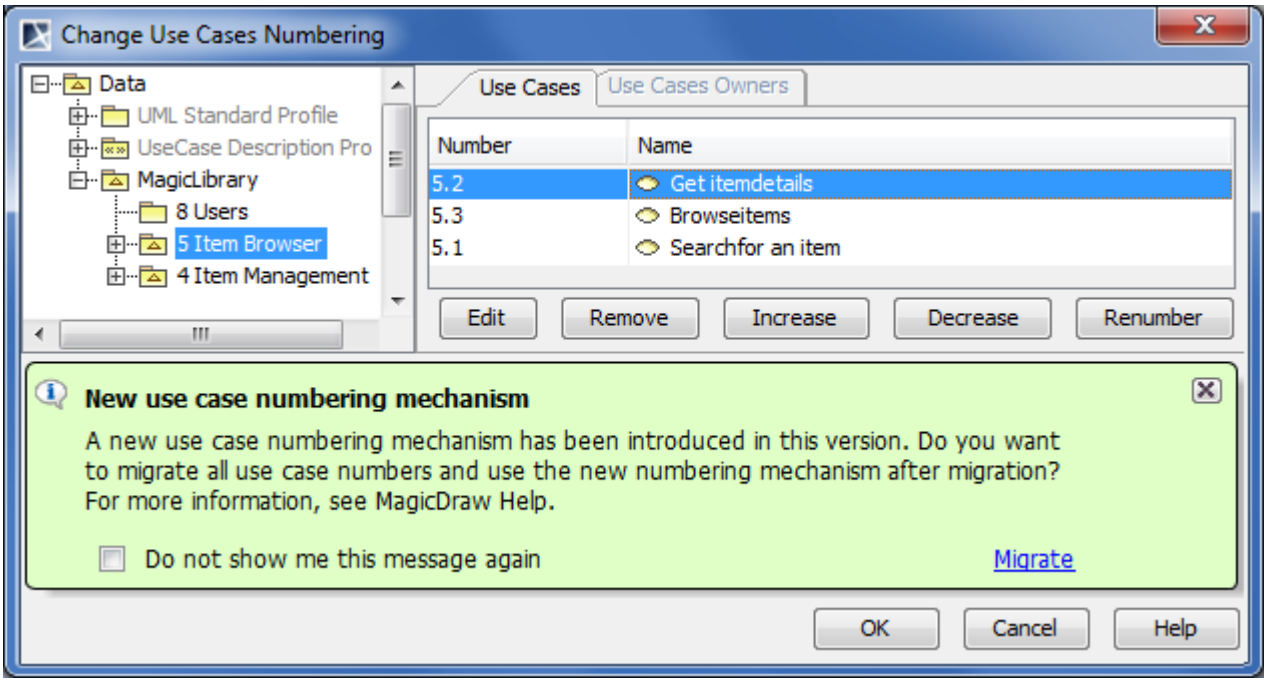


Figure 8 -- Example of notification message that is displayed in the dialog

There are the following types of notification messages:

1. information (light green)
2. warning (light yellow)
3. error (light red)

You can disable selected notification messages by clicking the **Do not show me this message again** check box in the notification message. Note that this check box exists in some messages only.

TIP!

For more information about the notification message display mode, see "To change the notification messages display mode" on page 31.

When the notification message opens, after some time it closes automatically. To leave the notification message opened, move the mouse cursor on the notification message and hold it there.

TIP!

For more information, see "To change the notification messages display time" on page 32.

To change the notification messages display mode

1. From the **Options** menu, select **Environment**. The **Environment Options** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

2. In the **General** options group, the **Display** category, change the **Notifications Display Mode** option value to one of the following:
    - **Display custom notification sets**. This is the default value, that is, the set of notifications will be displayed, except the ones that were selected by the user not to be displayed (using the **Do not show me this message again** check box).
    - **Display all notifications**. All notifications about specific events happening in the program will be displayed.
    - **Do not display**. All notifications are hidden.

To change the notification messages display time

1. From the **Options** menu, select **Environment**. The **Environment Options** dialog opens.
2. In the **Notifications** options group, under the **General** category, type the time on how long the notification message will be displayed.

TIP!

As of MagicDraw version 17.0.3, the notification messages are displayed not only in the lower-right corner of MagicDraw, but in the Notification Window too. For more information about the Notification Window, see "Notification Window" on page 32.

Related concepts

MagicDraw Documentation and Support

Related references

Notification Window

Notification Window

As of version 17.0.3, MagicDraw introduces the Notification Window, which is designed to store the history of all environment and project specific messages and messages received from other team members.

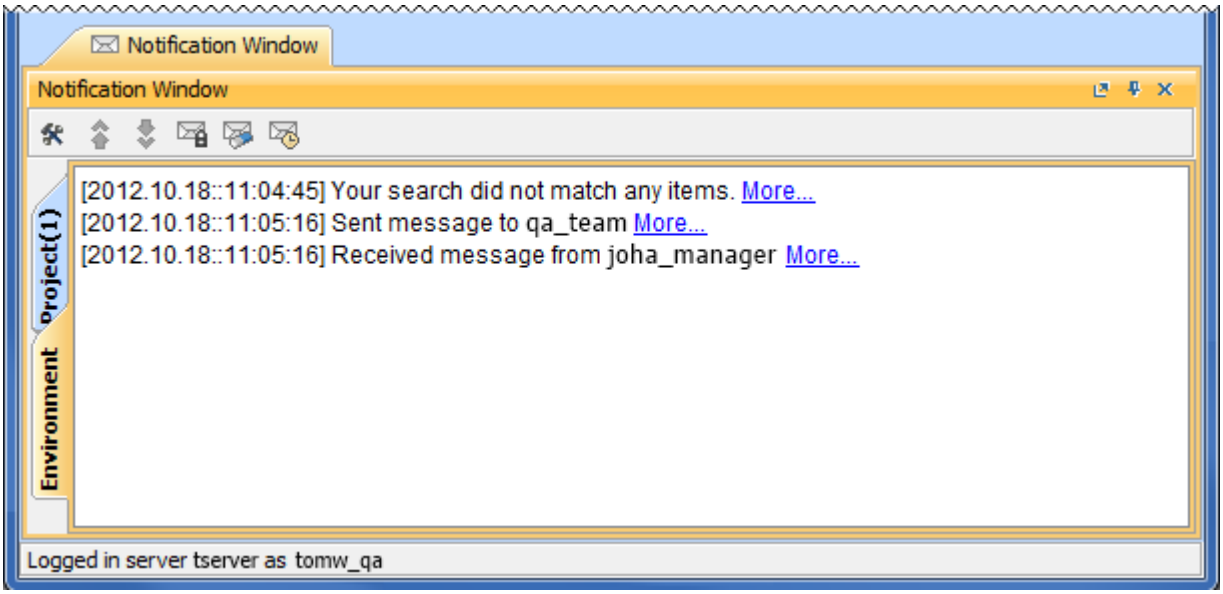
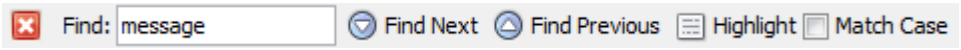


Figure 9 -- Notification Window

In the Notification Window, only the message title is displayed. To read the rest of the message text, click *More...* The message text is expanded.

Copyright © 1998-2015 No Magic, Inc.

By using the search toolbar, you can perform a search in the Notification Window. To open the search toolbar, press CTRL+F.



For more information about the Notification Window environment, see the following sections:

- "Grouping messages in Notification Window" on page 33
- "Notification Window toolbar" on page 34
- "Notifications option group in the Environment Options window" page 35

To open the Notification Window

Do one of the following:

- From the **Window** menu, select **Notification Window**.
- Press the CTRL+M shortcut keys.



Small icon  is displayed in a status bar after notification message hides. Click on this icon, to open the Notification Window.

Related references

Grouping messages in Notification Window

Notification messages

Grouping messages in Notification Window

In the Notification Window there are two tabs:

- **Project** tab. In this tab the project specific messages of the active project are displayed.
- **Environment** tab. In this tab the environment specific messages are displayed.I

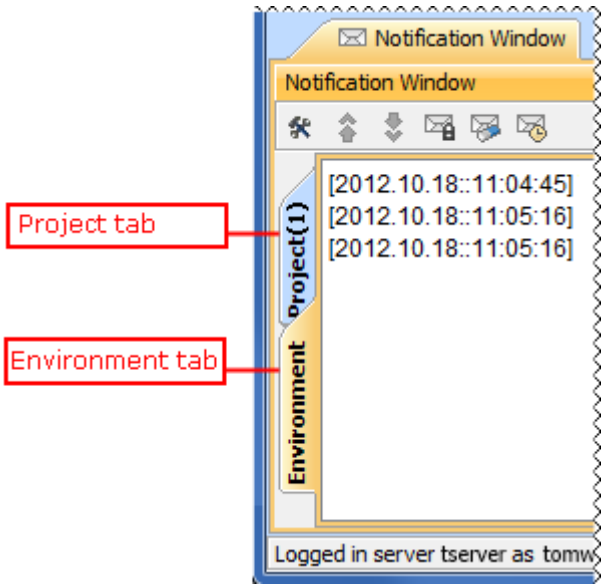


Figure 10 -- Environment and Project tabs in the Notification Window

In the example above you can see that next to the **Project** tab title, there is a number written in brackets. The number shows how many unread messages there are listed in the **Project** tab when the **Environment** tab is active.

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Notification Window

Notification Window toolbar

The Notification Window has the toolbar for configuring the messages display.



Figure 11 -- Notification Window toolbar

Button	Button name	Description
	Open Configuration	Click to configure the messages in the Notification Window. The **Environment Options** window opens. For more information, see "Notifications option group in the Environment Options window" on page 35.
	Scroll to the Start	Click to display the first message in the list. Note: The **Scroll to the Start** button is available if the first message is not visible.
	Scroll to the End	Click to display the last message in the list. Note: The **Scroll to the End** button is available if the last message is not visible.
	Lock Position	Click to lock the current message list position.
	Clear Messages	Click to remove all the messages from the Notification Window. Note that messages remains in the history.
	Show Full History	Click to display all the messages that are kept in the history. Note that hyperlinks that existed in the messages are not working in the history messages anymore.

Related concepts

Notification Window

Copyright © 1998-2015 No Magic, Inc.

Notifications option group in the Environment Options window

In the **Environment Options** dialog, you can configure the messages that are listed in the Notification Window.

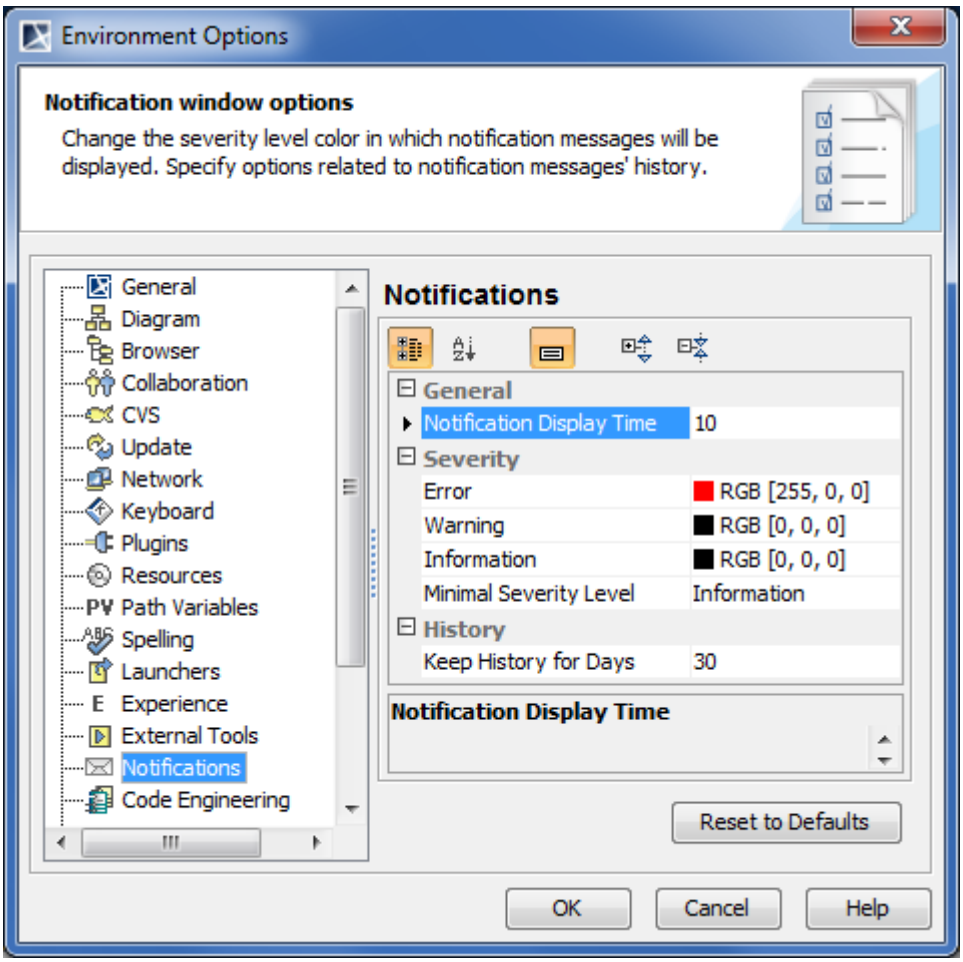


Figure 12 -- Notifications option group in the Environment Options window

To configure the notification messages

1. From the **Options** menu, select **Environment**. The **Environment Options** dialog opens.
2. In the **Environment Options** dialog, click the **Notifications** option group. Now you can specify the options for the notification messages.

Option name	Type	Description
Notification Display Time	Text box	Specify the duration of displaying notification messages (in seconds). For more information about the notification messages, see "Notification messages" on page 31.
Error, Warning, Information	Color picker	Click the **Error**, **Warning**, or **Information** box and then click the ... button to change the text color for the messages of the different severity level (error, warning, or information). The **Color** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

Minimal Severity Level	Drop-down list	Click to open the drop-down list in which you can select the minimal severity level of the messages that will be displayed in the Notification Window. There are the following options: • Error. Only the error messages will be listed. • Warning. The error and warning messages will be listed. • Information (default). The error, warning and information messages will be listed.
Keep History for Days	Text box	Specify the duration of keeping messages in the history list (in days).

Related concepts

Notification Window

Related references

Customizing Environment Options

Hints Associated to User's Actions

MagicDraw now provides hints to help you MagicDraw (Figure 13 on page 36). Hints related to your actions will open and inform you about the functionality that is available in MagicDraw and show you how to perform some operations more effectively.

Hints are displayed in the lower right-hand corner of the application window. Figure below shows an example of a hint.

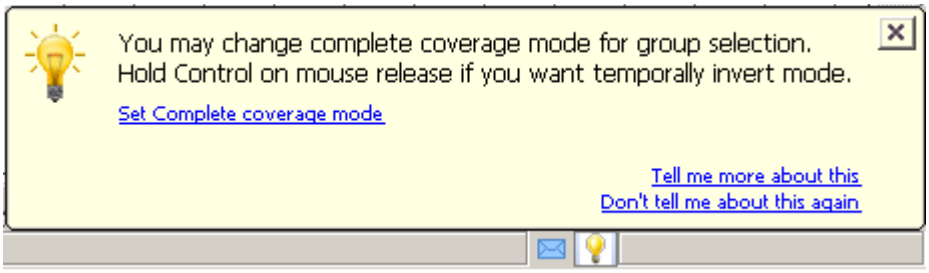


Figure 13 -- Hints Associated to User's Actions

Hint is displayed for a 10 seconds or for a period while mouse is rolled on hint. Small icon  will be displayed in a status bar after hint will hide. Clicking on this icon, you can open hint again.

To change hints display mode

Change the **Hints display mode** property in the **Environment Options** dialog, **General** branch, **Display** group.

The **Hints display mode** property specifies whether hints on MagicDraw functionality related to user actions will be displayed. Custom hint set does not include hints, that are asked not to be shown by user. Select option **Display all hints again** to reset custom hint set to show all hints. Selecting value **Show all hints again** will delete the list of the hints that should not be displayed.

Copyright © 1998-2015 No Magic, Inc.

Productivity Tips Displayed in Progress Window

MagicDraw now displays productivity tips in a progress window whenever it performs a long task (Figure 14 on page 37).

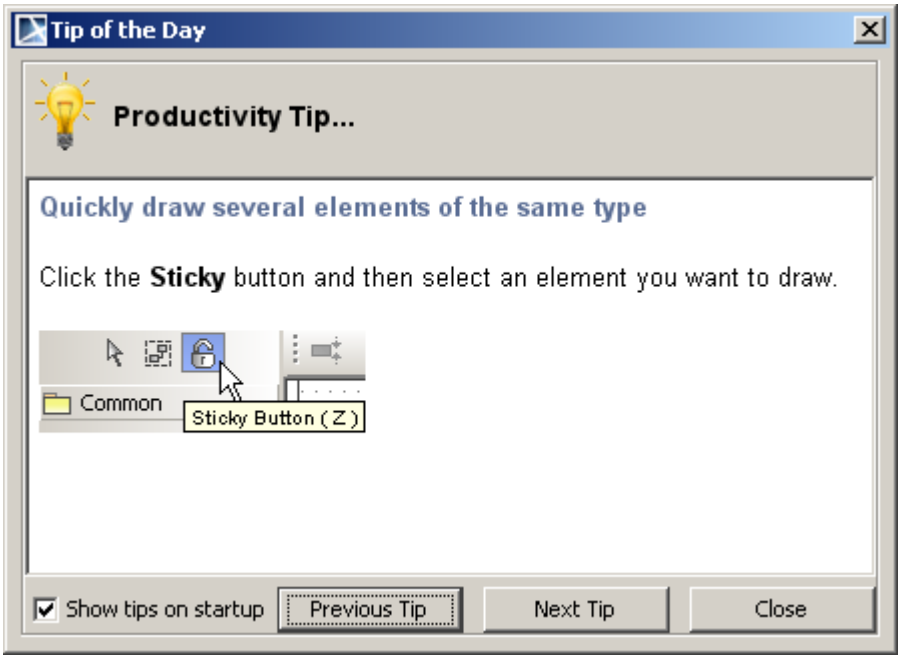


Figure 14 -- Productivity Tip

To show or hide productivity while running long task

Select or clear the **Show tips while running long tasks** property in the **Environment Options** dialog box, **General** branch, **General** group.

Tutorials

Quick Start tutorials for UML diagrams provide the step-by-step on-screen instructions on how to work with UML diagrams and start modeling with MagicDraw. They also give the links to the associated information in the other MagicDraw user's documentation and to the worldwide tutorials sources as well.

Try the Quick Start tutorials for the new diagram, Class diagram, Use Case diagram, Activity diagram, Sequence diagram, and others.

To open the Quick Start tutorials:

1. From the **File** menu, select **New Project**. The **New Project** dialog will open (Figure 15 on page 38).
2. Select the **Project from Template** icon on the left-hand side of the **New Project** dialog.
3. Select **Guide to UML Diagrams Project** > **Guide to UML Diagrams Project** in the Select template tree.
4. Click **OK**. The MagicDraw project with its tutorials will open (Figure 16 on page 39).

Copyright © 1998-2015 No Magic, Inc.

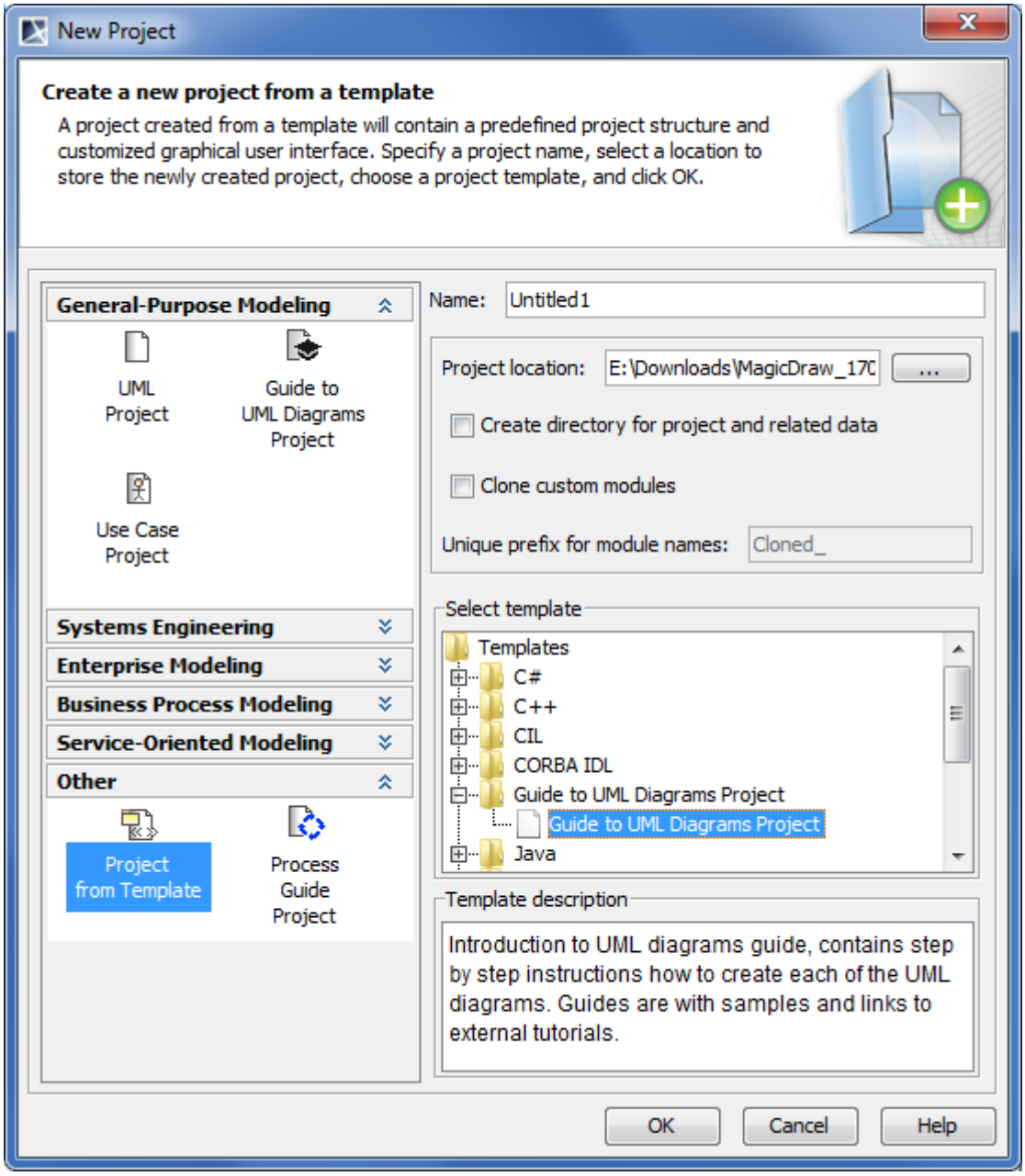


Figure 15 -- The New Project Dialog

Copyright © 1998-2015 No Magic, Inc.

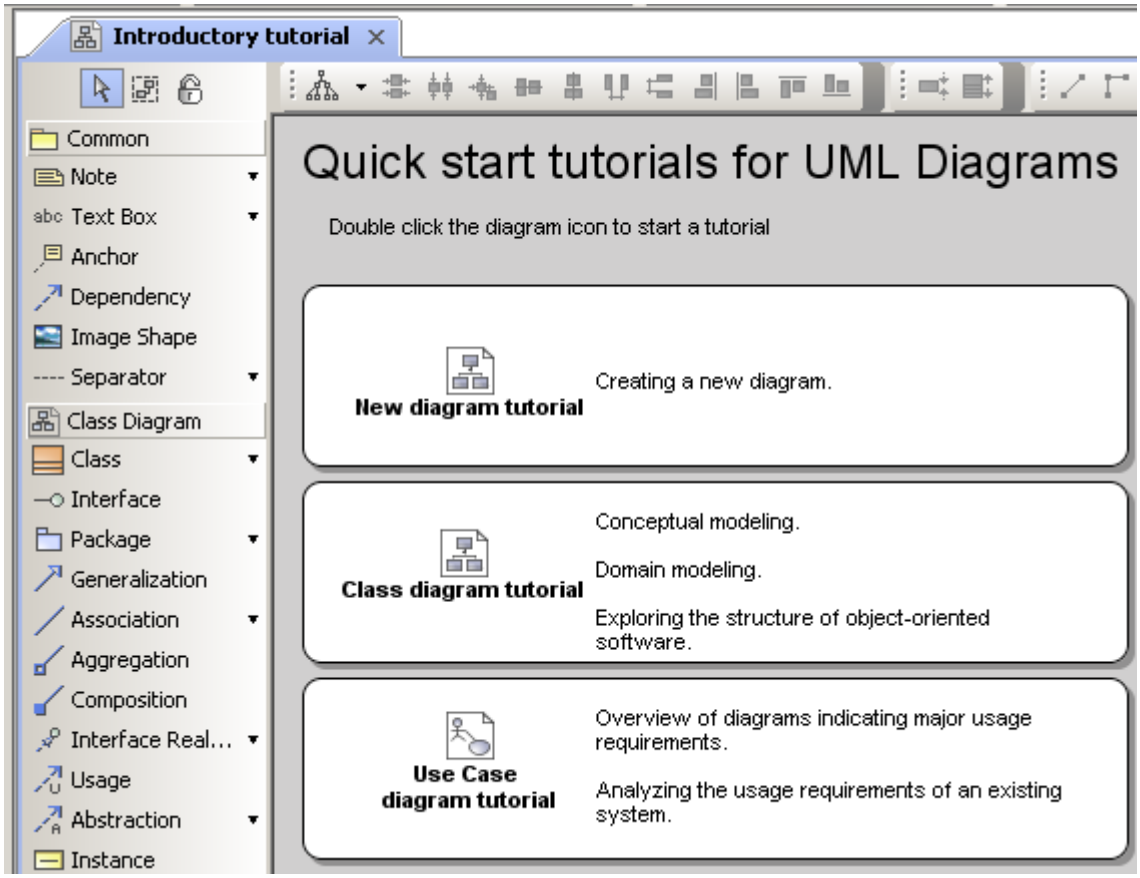


Figure 16 -- Quick Start Tutorials for UML Diagrams

Other Documentation

- The readme.html file is located in the main MagicDraw installation directory. Readme documents are also available for MagicDraw integrations.
- *Viewlets*. View online demos introducing MagicDraw and particular functionalities. You may find online demos at http://www.nomagic.com/support/demos.html.
- *Samples*. In the MagicDraw installation directory (samples directory), you will find the samples of MagicDraw projects.

Support

FAQ

Before calling or registering an issue, please have a look in our FAQ section. It is constantly updated and may contain an answer to your question.

Forum

Discuss and get answers about No Magic products in No Magic Community Forum. With respect to the growing interest in No Magic products, we have expanded the capabilities that had been provided by our newsgroups and have moved the content to the online Forum.

You are welcome to post your comments and questions here.

Copyright © 1998-2015 No Magic, Inc.

Note: If you are interested in old discussions from the newsgroups, you can find them in the Forum as well as in the newsgroups in a read-only mode.

- If you are a new user, ask questions and get started on learning about UML and MagicDraw by the NEW USER section.
- Discuss issues related to technologies and processes in the STANDARDS/METHODOLOGIES section.
- Share your professional experience in terms of using MagicDraw, including installation and running, MagicDraw OpenAPI, scripting, integrations with other 3rd party tools, etc. in the MAGICDRAW section.
- Find new product versions, updates, and major events in the ANNOUNCEMENT section.
- Post your suggestions for improvements and new features that you would like to see added to the Forum in the OTHER section, the Suggestions subsection.

Knowledge Base. New issue registration

- Knowledge base and New issues registration - https://support.nomagic.com.
- sales@magicdraw.com - e-mail address for questions regarding academic or site license discounts and quotes.
- contact@magicdraw.com - e-mail address for the other contacts.

The support system https://support.nomagic.com provides:

- Submitted issues status tracking.
- Ability to submit private and public questions, suggestions, improvements, and problems.
- Ability to search through the existing public issues, view status of your issues, provide your vote for suggestions.

We provide free professional support for:

- Registered users with a valid No Magic Software Assurance contract (SA). SA provides you with technical support, software upgrades and maintenance releases at no additional cost for a contract period.
- Pre-sales users during the evaluation period.
- New customers for 30 days.
- Inquiries about registration, licensing and product updates.

Customer support levels

A customer support level determines customer access rights and the response time for support requests. To find out yours customer support level, please, visit the knowledgebase - https://support.nomagic.com.

Support level	Applies for	Ensured support
Limited support	Customer with no Software Assurance (SA)	Self-help: FAQ, Knowledge base, documentation, online demos, No Magic Community Forum.
Full support	• Customer with SA • Inquiries about registration, licensing, and product updates. • With valid evaluation key. • 30 days after purchase.	Response within business hours, with 24 - 48 hours response time.
Premium support	VIP customer* (marked as VIP)	Response within business hours, with 24 hours response time.

Copyright © 1998-2015 No Magic, Inc.

* VIP customer usually is our partner.



During the support period covered by our SA, you can also report any software problems or errors. If it is determined that a reported reproducible issue in the software actually exists, and this significantly impacts the usability of the software, No Magic agrees to make reasonable efforts to provide a usable workaround solution or to correct the issue in an upcoming maintenance release or update.

End-of-life policy

Public service packs are primarily released for the newest version. As we understand that switching from one version to another can take some time, in case of significant issues we provide service packs (or private patches) for up to one year old versions when applicable.

We support and provide patches to older versions only when you own Software Assurance (SA) contracts http://www.nomagic.com/support/sales-and-licensing/software-assurance-maintenance-contracts.html only on your purchased products, so please be sure you are covered.

We always strongly recommend using the newest version as it contains all newest fixes, usability features, new capabilities, and support for standards and technologies.

Reporting issues directly from MagicDraw

We highly recommend to submit report any problem, suggest improvements, or ask about new features directly from MagicDraw. Report will create an issue in to our Online Customer Support System https://support.nomagic.com.

These reports help us address issues in a more timely manner, as well as speeding up maintenance releases that are free of known defects.

To report an issue directly from MagicDraw

1. On the help menu, click **Report an Issue**.
2. Fill in the first and last names together with your email address.



If you supplied the email address used when registering at www.magicdraw.com, you will be able to track the status of your submitted issue in our Online Customer Support System located at https://support.nomagic.com. If you used another email address, you will only receive email notifications of status changes.

3. Choose the product, issue type, and component.
4. Describe your issue by providing as much information as possible.



Please note that we provide a professional support for registered MagicDraw users with the valid No Magic Software Assurance Contract (SA). SA provides you with technical support together with the major software updates and maintenance releases at no cost throughout the contract period. Issues are normally handled within one or two business days during regular business hours.

5. In the **Attachments** tab, select files you would like to send together with your issue report:
   - **Attach log file** - the log file will be sent together with your issue report.
   - **Attach project file** - the opened and last saved project file will be sent together with your issue report.
   - **Attach used modules** - modules that have been used in the opened project file will be sent together with your issue report.

Copyright © 1998-2015 No Magic, Inc.

- **Attach diagrams images** - choose diagrams, whose images you would like to send together with your issue report. Also specify the images format.

6. If you are registered user, track you issue at https://support.nomagic.com.



If you are already registered personal information will be filled into the Report an Issue dialog.

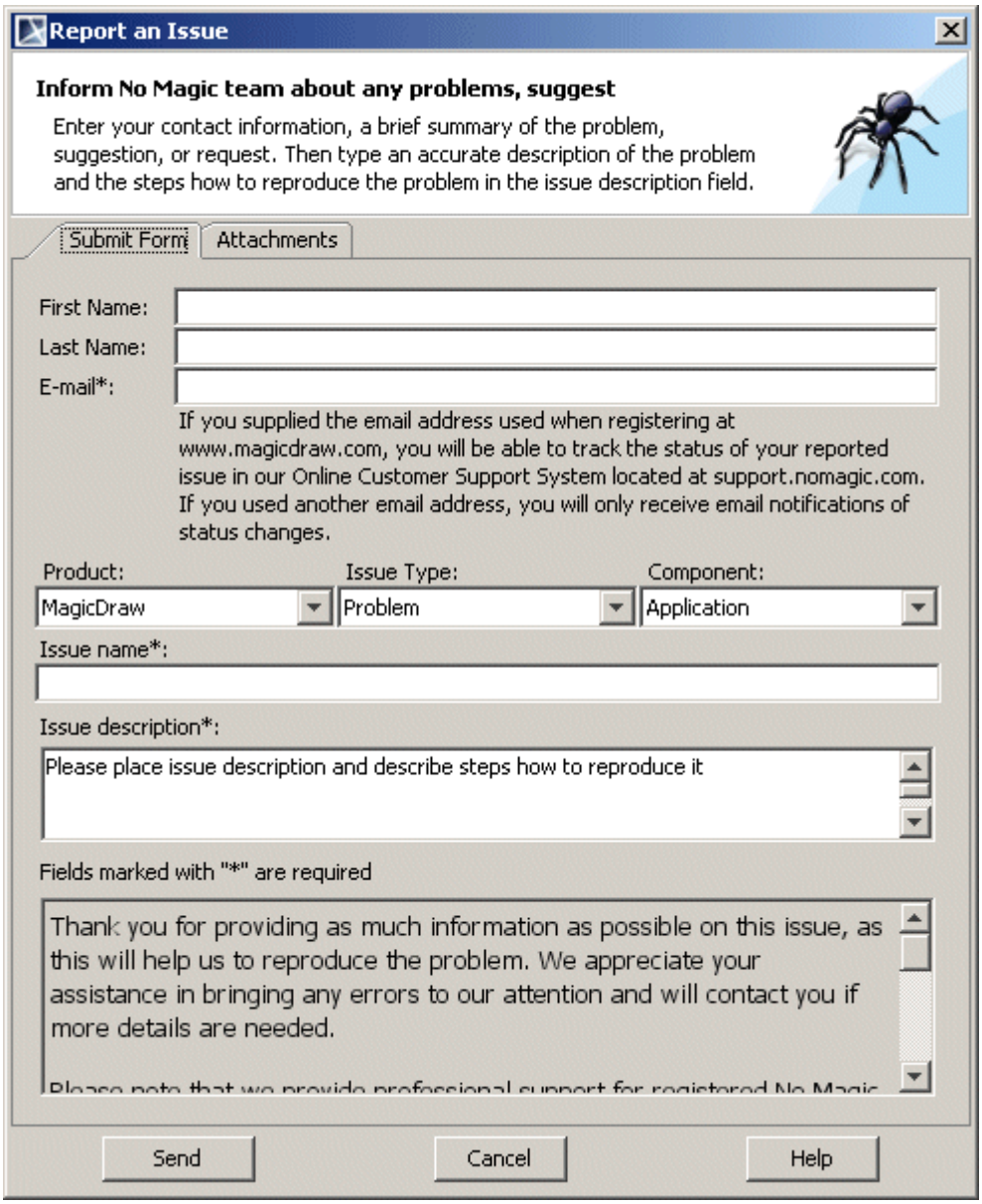


Figure 17 -- Report an Issue dialog box, Submit Form tab

To report an issue when application is unresponsive

If MagicDraw becomes unresponsive, a separately executable tool is provided for analyzing the status of the process to aide in bug submission. In these situations, manually start the *submitbug.exe* file (located in the <*MagicDraw installation directory*>*bin* folder) and follow directions. After *submitbug.exe* is started, the **Report an Issue** dialog opens. For more information about the **Report an Issue** dialog, see "To report an issue directly from MagicDraw" on page 41.

Copyright © 1998-2015 No Magic, Inc.

View and submit internal errors

If an error occurs, an error message will appear at the bottom of the MagicDraw window.

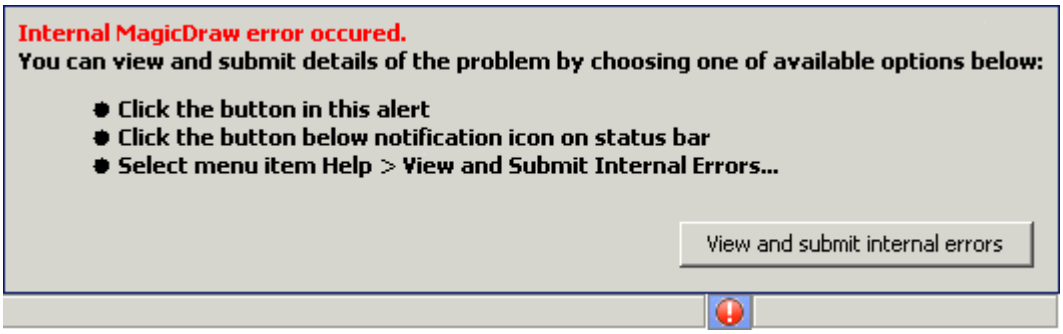


Figure 18 -- Message, about MagicDraw internal error

To view internal errors

To view internal errors you have to open the **Unhandled Errors** dialog. There are three methods for to open the **Unhandled Errors** dialog.

- Click the **View and submit internal errors button** in the Notification Window.
- From the **Help** main menu, choose the **View and Submit internal errors** command.
- Click the notification icon on the status bar.

Note: The **View and Submit internal errors** command is only active in the **Help** menu and the red button at the bottom of the status bar only exists if the **Submit errors** dialog contains errors.

To submit an error

1. Open the **Unhandled Errors** dialog.
2. Click the **Submit** button in the **Unhandled Errors** dialog box. The **Submit Error** dialog appears.
3. Fill in **Your name**, **Your e-mail** and **Bug description** fields and click the **Send** button. The error will be sent to the MagicDraw support team.

Copyright © 1998-2015 No Magic, Inc.

2 GETTING STARTED

In the chapter, you will find the following sections:

1. "Licensing Information" on page 44
2. "Unlocking MagicDraw" on page 46
3. "Activating the commercial license after purchase" on page 46
4. "User Registration" on page 55
5. "MagicDraw Configuration Files Location" on page 61
6. "Importing Configuration from Earlier Version" on page 62
7. "Disabling Inbound Network Activity" on page 63
8. "Updating" on page 63

Licensing Information

Information about installed or needed licenses and the status is presented in the **About** screen with the ability to remove unused licenses.

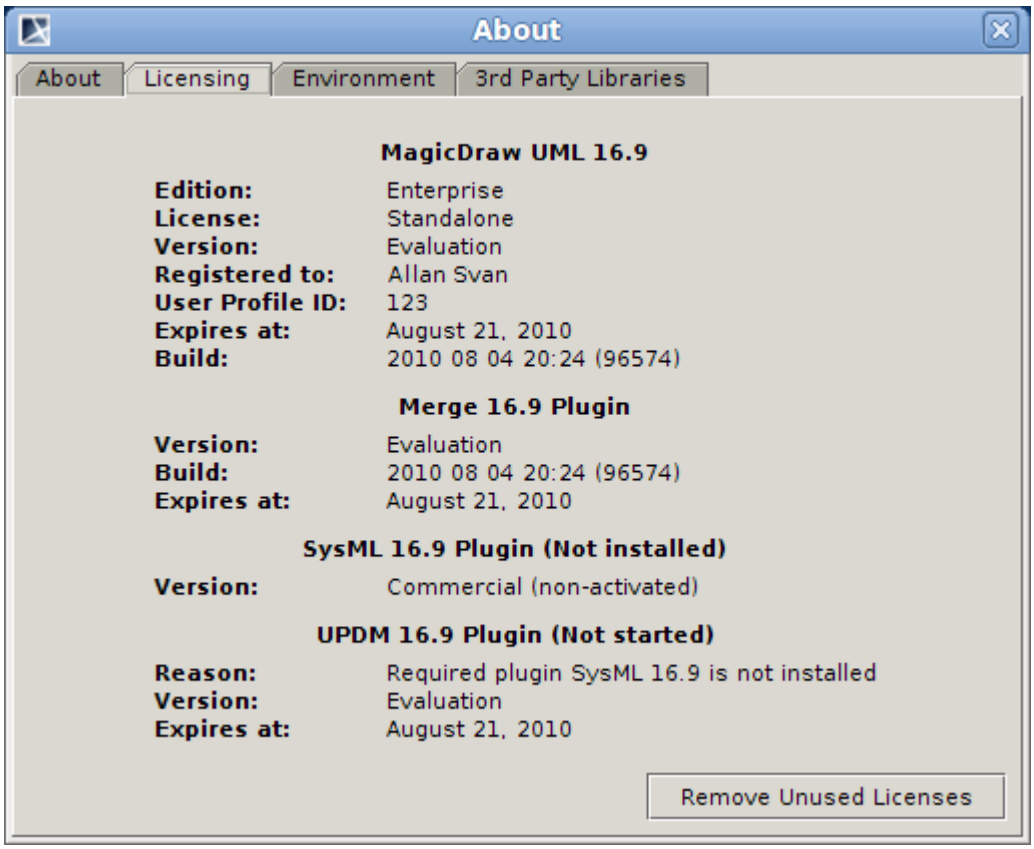


Figure 19 -- About screen, Licensing

The following table describes the main information in the **Licensing** tab on the **About** screen.

Information item	Description
User Profile ID	Please deliver your User Profile ID when contacting support or sales.

Copyright © 1998-2015 No Magic, Inc.

Information item	Description
Not installed indication	If license is not installed, the text *Not installed* is displayed next to the license name.
Not started indication	If license is installed, but not started, the text *Not started* is displayed next to the license name, and the reason is given. Possible reasons: • Required resource is not installed • Plugin startup failed • Other

Removing Unused Licenses

Click the **Remove Unused Licenses** button in **Licensing** tab on the **About** screen, to remove licenses of not installed plugins.

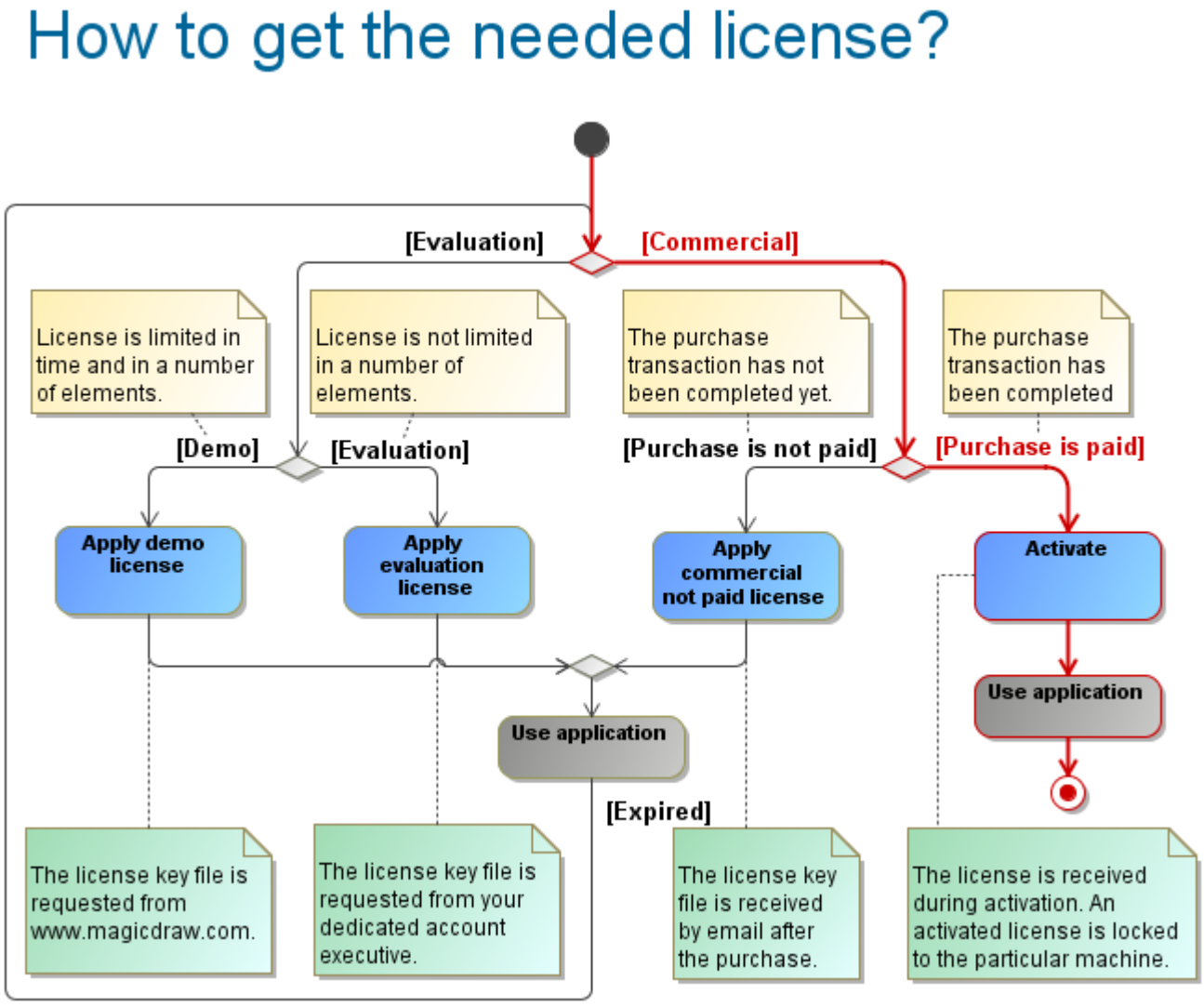


If you have any questions or issues, please report them using the **Report an Issue** dialog. For more information, see "View and submit internal errors" on page 43.

Copyright © 1998-2015 No Magic, Inc.

Unlocking MagicDraw

Unlock MagicDraw with evaluation, demo, or commercial license key file.



To unlock MagicDraw

1. After downloading MagicDraw, you will get license key file to your mailbox.
2. Save this key file into the local file system before starting MagicDraw.
3. When starting MagicDraw for the first time, you will be asked to choose the license key file. Click the **Select Licenses Key Files** button and select the file from the directory you have saved the file.
4. System will automatically start MagicDraw and you will be able to use it.

Activating the commercial license after purchase

To improve software protection and to prove our customers' investments, No Magic uses the solid product licensing. Licenses are locked to the machine. You will be able to monitor and control, on what machines the commercial licenses are used.

Copyright © 1998-2015 No Magic, Inc.

After updating to version 16.9 or later, you will be requested to activate and receive the commercial licenses dedicated for the particular machine.

Flexera Software FLEXnet licensing system, the industry leader helping to manage and secure flexible software licensing, is used to manage No Magic products licenses.

How to get the commercial Activated license?

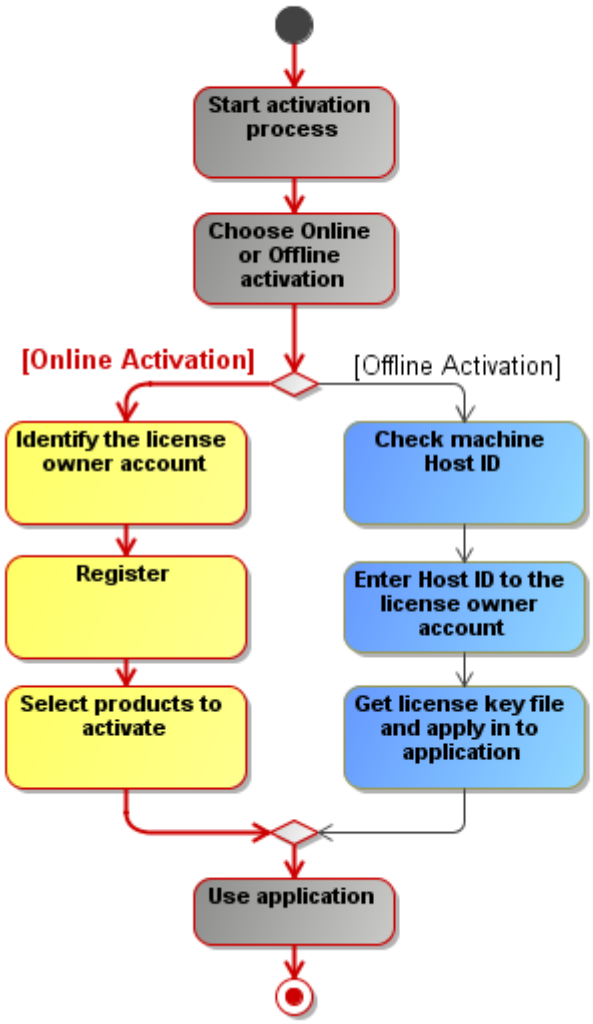


Figure 20 -- Activation process. How to get commercial license?

All the updated information about the commercial license activation you may also find at http://www.nomagic.com/support/activation.html.

Copyright © 1998-2015 No Magic, Inc.

Commercial license activation process

The commercial license activation is required after the purchase transaction has been completed. The activation process allows receiving the commercial license dedicated for the particular machine.

NOTE

- Only commercial activated licenses are locked to the particular machine.
- During the activation process, identification and registration is required. Any key file (evaluation, demo, commercial not activated and other) can be used for the identification of the license owner.
- Commercial not activated keys allow using the application for 7 days according to obtained licenses, before requesting activation. Not activated keys can be used during activation process for identification of the account on which the purchases are.

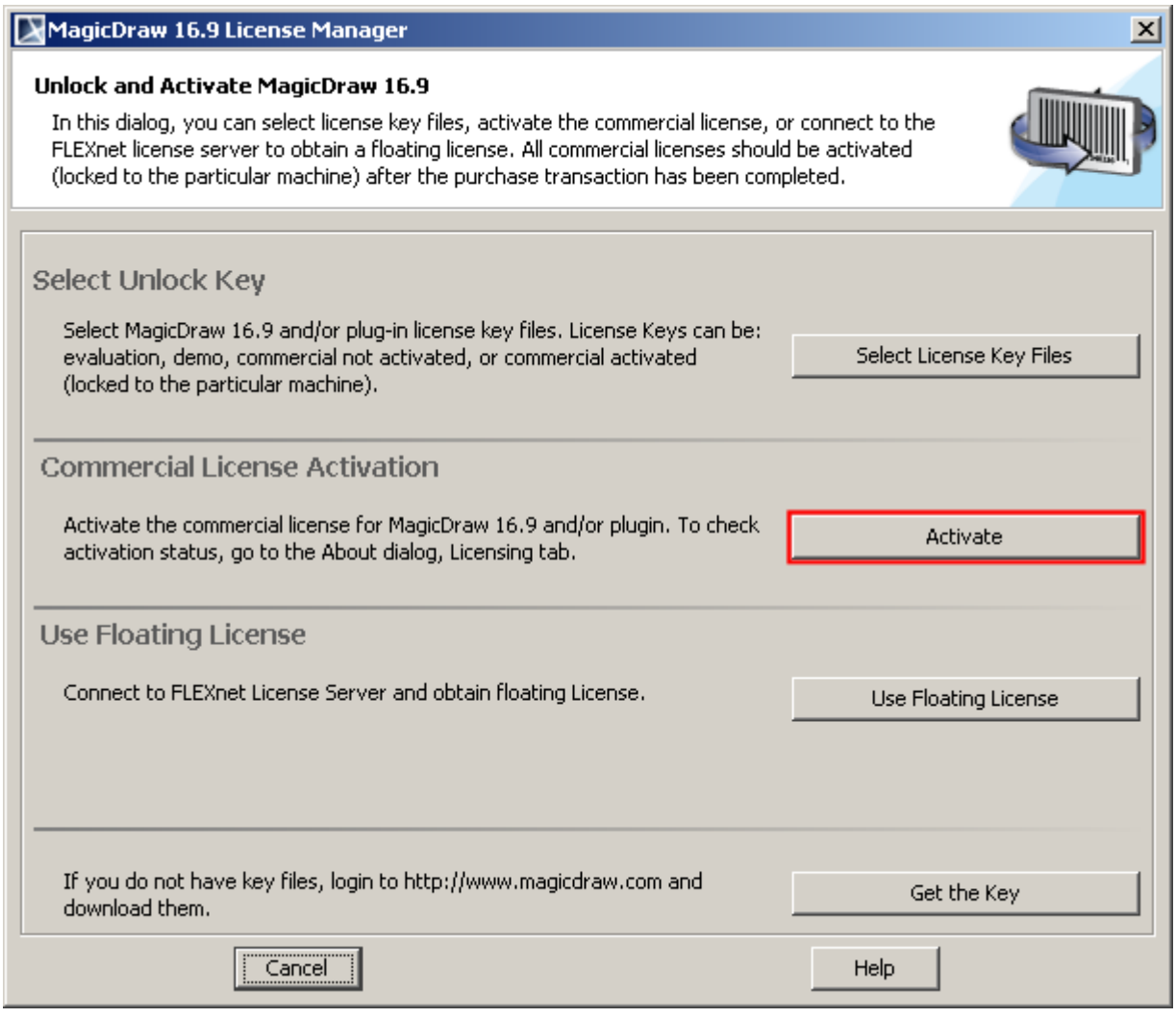
Online and offline activation is available.

Online activation

Choose online activation and enable commercial activated (without expiration) licenses automatically.

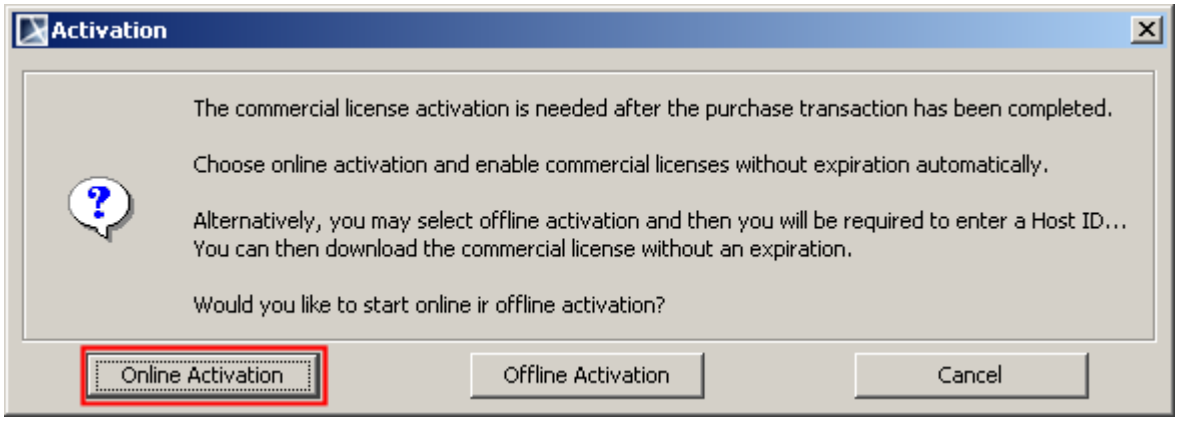
To activate the license online

1. Click the **Activate** button in the **License Manager** dialog of the application.



Copyright © 1998-2015 No Magic, Inc.

2. Click **Online Activation**.



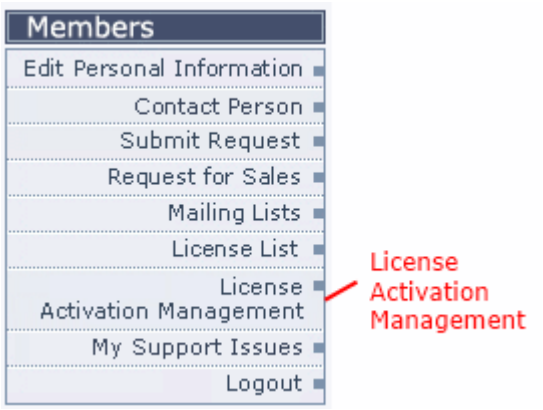
3. If you will be requested, identify yourself and register.
4. Select licenses you need to activate.

Offline activation

Alternatively, you may select offline activation, get the Host ID, enter it in the license owner account and then download the commercial activated license.

To activate the commercial license offline

1. Login to the license owner account on www.nomagic.com.
2. Click **License Activation Management** in the **Members** area at the right side of the web page.



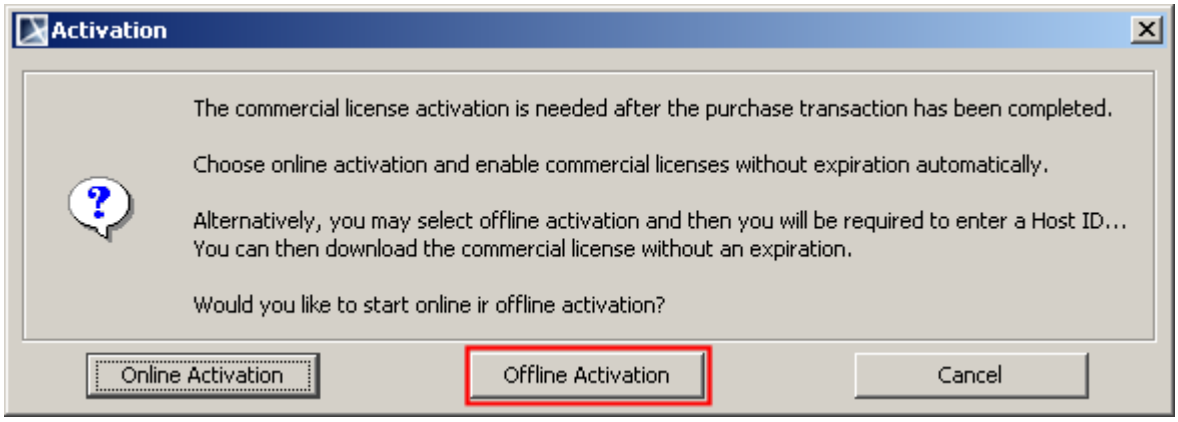
3. Click the **Create New License Activation Record** button and enter user details and Host ID of the machine.
4. Download or send the key to your e-mail.

Determining a Host ID (for offline activation)

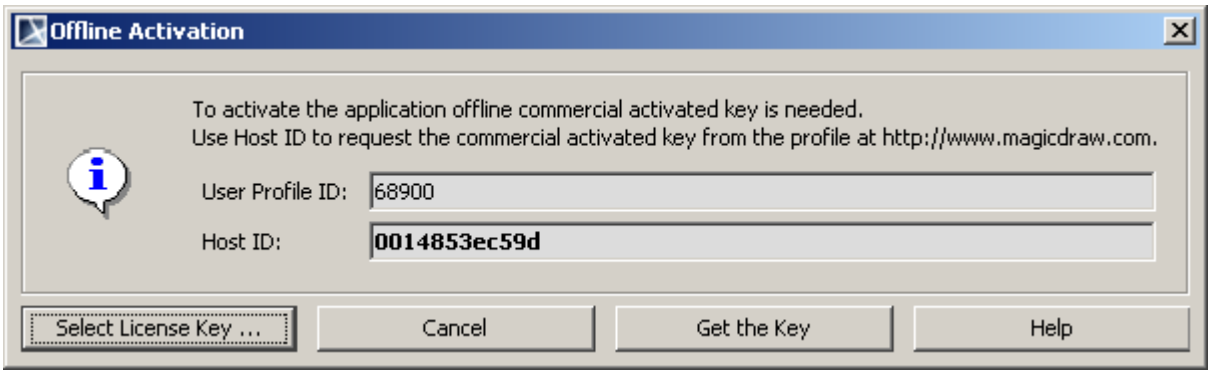
1. Click the **Activate** button in the **License Manager** dialog of the application.

Copyright © 1998-2015 No Magic, Inc.

2. Click **Offline Activation**.



3. Host ID is shown.



NOTE

Host ID is the Ethernet address of the machine on which application will run.

Related topics

Commercial license types
Host ID
Identification
Registration
Deactivation

Commercial license types

Definitions

Commercial not activated license is used only to determine the account on www.nomagic.com and allows using the application for 7 days according to obtained licenses, before requesting the activation.

EXAMPLE

MagicDraw_16_9_Professional_C#_Standalone_ Not activated key.txt

Commercial activated license is the license without expiration, locked to the particular machine, and can be deactivated.

EXAMPLE

MagicDraw_16_9_Professional_C#_Standalone_
Activated_key_for_<name>_<surname>.txt

Copyright © 1998-2015 No Magic, Inc.

After the purchase transaction has been completed, you will receive the **commercial not activated license**. They are used ONLY to determine the account on www.nomagic.com from which activated keys should be requested. Any other license, evaluation, demo, or commercial can be used for identification of the license owner account during activation.

Commercial not activated license requires activation. It allows using the application for 7 days according to obtained licenses, before requesting the activation. After the online or offline activation you will get the **commercial activated license**.

After the online activation, commercial activated license will be applied automatically and you will be able to use the application on the particular machine.

During the offline activation you will be required to enter a Host ID <link to Host ID section> in the license owner account at www.nomagic.com. You can then download the commercial activated license and select it from the application.

Related topics

Commercial license activation process
Host ID
Identification
Registration
Deactivation

Host ID

A Host ID is the value of a specific system attribute (MAC / Ethernet address) that uniquely identifies the host / machine under which an application is running. A Host ID is used for the locked to particular machine license. Only commercial activated licenses are locked to the particular machine.

For the Host ID determining instructions refer to “Determining a Host ID (for offline activation)” on page 49.

Determining a Host ID using lmhostid

One of the ways for determining Ethernet address (Host ID) is using lmhostid utility. This way can be useful when the list of Host IDs of multiple machines should be discovered, even before installing the applications itself. If you are a system administrator, you can provide end users with commercial activated licenses without requesting licenses from them.

The lmhostid is a command line utility, which prints the Host ID that is required. The Lmhostid utility is available as an execule called lmutil. Download it for a particular OS from http://www.globes.com/support/fnp_utilities_download.htm.

Run lmutil with lmhostid parameter from the command line. Lmhostid displays the default hostid for the current platform.

For exact commands for specific OS refer to: License Administration Guide (Obtaining System Hostids).



If two or more Ethernet addresses are available on the particular machine, the first one printed by the hostid command should be used.

Related topics

Commercial license activation process
Commercial license types
Identification

Copyright © 1998-2015 No Magic, Inc.

Registration
Deactivation

Identification

To determine a license owner, the user identification is requested during activation process.

Identification is requested only if you have not added any license key file, have not registered your installation, or have not performed identification.

There are two methods available for the user identification:

- **License file based identification**. Select the license key file to identify the license owner account. Evaluation, demo, or commercial license files can be used for the identification.
- **Login name based identification**. Use the license owner login name and password to identify on which account on www.nomagic.com your purchases are.

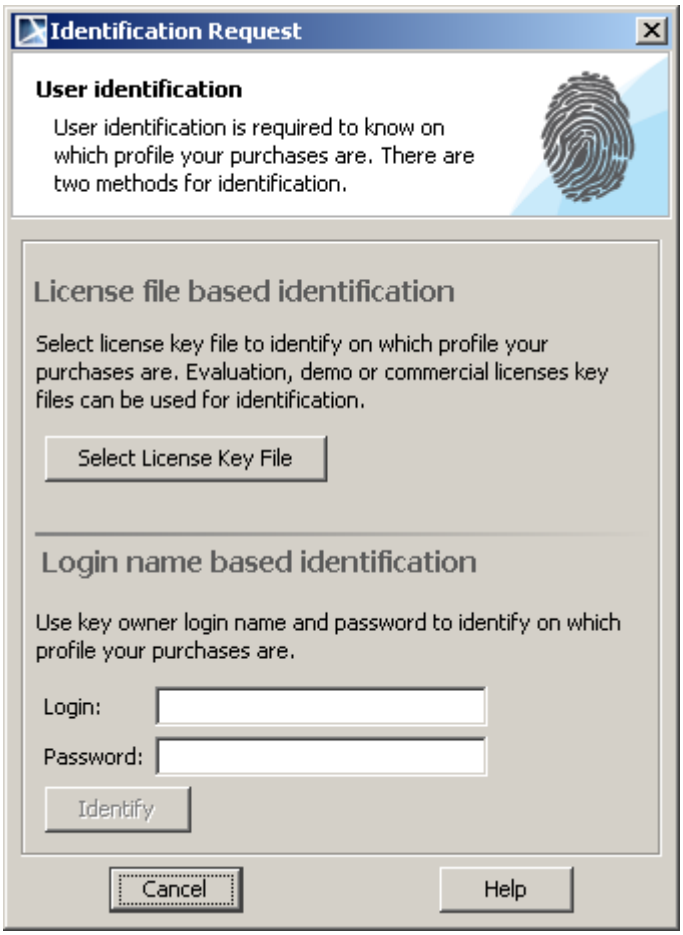


*Figure 21 -- **Identification Request** dialog*

Ones identification is performed, it will not be requested the next time during activation.

NOTE

In case you need to change identification records, just apply the new license key file. The last applied license key file is used for the identification.

Related topics

Commercial license activation process
Commercial license types

Copyright © 1998-2015 No Magic, Inc.

Host ID
Registration
Deactivation

Registration

The user registration is required during activation process in order to obtain the commercial activated license.

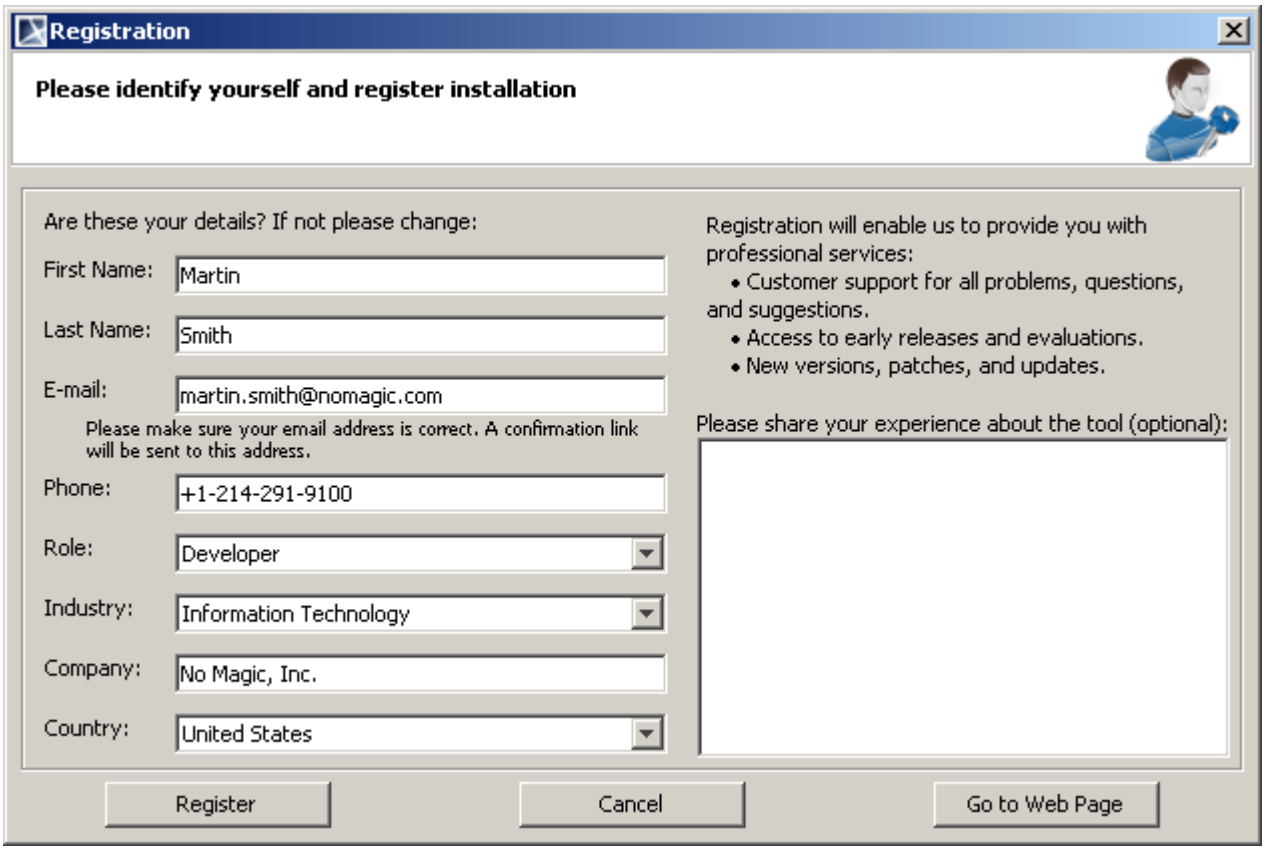


Figure 22 -- Registration window

The registration will help us to provide you with:

- Customer support. Issues, questions, and suggestions can be reported and existing ones tracked on https://support.nomagic.com.
- Access to early releases and evaluations.
- New versions, patches, and updates.

Related topics

User Registration
Commercial license activation process
Commercial license types
Host ID
Identification
Deactivation

Copyright © 1998-2015 No Magic, Inc.

Deactivation

The deactivation is the process, which allows returning commercial activated licenses. You may remove and return all activated (locked to the particular machine), not used commercial licenses by deactivating them.

To deactivate a license in the application

1. Click the **Deactivate** button in the **License Manager** dialog of the application or uninstall the application.
2. Commercial licenses will be deactivated automatically and returned online.
If the online licenses return is not available, licenses will be deactivated and offline deactivation - licenses return message with License Deactivation ID - will be shown. To update the account information manually, return the used license by entering the License Deactivation ID to the license owner account at www.nomagic.com.
3. The application is deactivated and licenses are returned.



- The deactivated license still will be active for 7 days.
- The returned licenses can be activated on the same or another machine.
- Online deactivation from the application side or the deactivation with the License Deactivation ID is treated as the confirmed deactivation case and does not decrees an available rehost limit.

Deactivating from the License Activation Management screen

You can deactivate licenses not only from application, but also from the License Activation Management screen in the license owner account at www.nomagic.com.



The deactivation from the License Activation Management screen should be used only if the deactivation from application side is not available: installation has been lost or corrupted.

To deactivate an application from the license owner account:

1. Go to license owner account at www.nomagic.com.
2. Remove all products assigned for the particular installation in the **Activated Products** column.
3. Application will be deactivated on the next start-up. Licenses will be returned.



The number of available, not confirmed, deactivations from the License Activation Management screen is 1 for the each obtained license per year. The Confirmed deactivation case from the application side will not decrease the rehost limit.

License Deactivation ID

License Deactivation ID is the value of a specific system attribute that uniquely identifies license deactivation/return from the particular host/machine under which application was running. The License Deactivation ID is used for the license offline deactivation.

To determine License Deactivation ID (for offline deactivation):

1. Click the Deactivate button in the License Manager dialog of the application.
2. The license Deactivation ID will be shown if the online deactivation will not be available.

Copyright © 1998-2015 No Magic, Inc.

Confirmed deactivation case

The license deactivation, about which confirmation is received in the license owner account at www.nomagic.com is treated as confirmed deactivation case.

The deactivation confirmation can be received either online, during online deactivation, or offline, with the help of the License Deactivation ID, which is shown during the offline deactivation.

The deactivation from the License Activation Management screen without the License Deactivation ID, is treated as NOT confirmed deactivation and decreases the rehost limit. An exception is, if after such the deactivation, automatic confirmation is received from the application side that the license has been also deactivated on the application side.

Rehost limit

The rehost limit is the number of available, not confirmed, deactivations from the License Activation Management screen. By default such number is 1 for the each obtained license per year. The confirmed deactivation case from the application side will not decrease the rehost limit.

If the rehost limit is exceeded you can request extension from your dedicated account executive by filling the reason of the request and clicking the Request Rehost Increase button.

Rehosting - moving the license from one machine to another

You may moving of the activated (locked to the particular machine) licenses from one machine to another is called rehosting.

To rehost a license

1. Deactivate the license on the old installation.
2. Install an application on the new machine.
3. Activate the license on the new installation.

Related topics

Commercial license activation process
Commercial license types
Host ID
Identification
Registration

User Registration

User Registration allows you to access dedicated resources on the No Magic website such as dedicated online support, answers database, new products evaluation, and beta products.

The **Registration** dialog will open the first time MagicDraw is started (see the following figure). You can complete Product Registration at any time by selecting **Register** from the **Help** menu.



- No Magic, Inc. respects your privacy. We will only use your personal information for communications and management of your online account, and the products you register with your account.
- Registration for the owners of commercial licenses is mandatory.

Copyright © 1998-2015 No Magic, Inc.

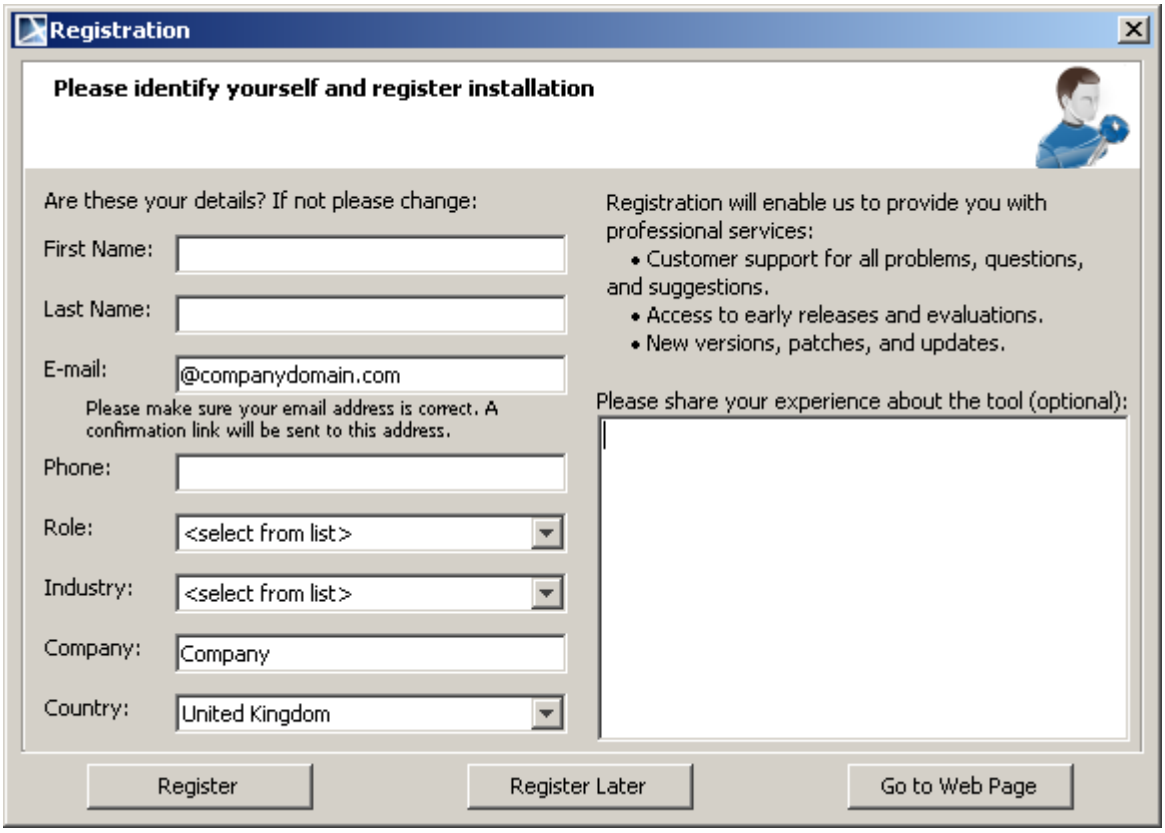


Figure 23 -- Registration dialog

Registration Workflow

The registration process is straightforward. After a new key application, you will be requested to register your installation. If you have successfully sent the online registration form, you will receive an e-mail with a link to confirm your registration and the correctness of information furnished through online registration.

Upon confirmation, a dedicated account will be created for you at www.nomagic.com.

NOTE

If you have an existing profile at www.nomagic.com you may register with the same user information and the same profile information will be used for registration.

Copyright © 1998-2015 No Magic, Inc.

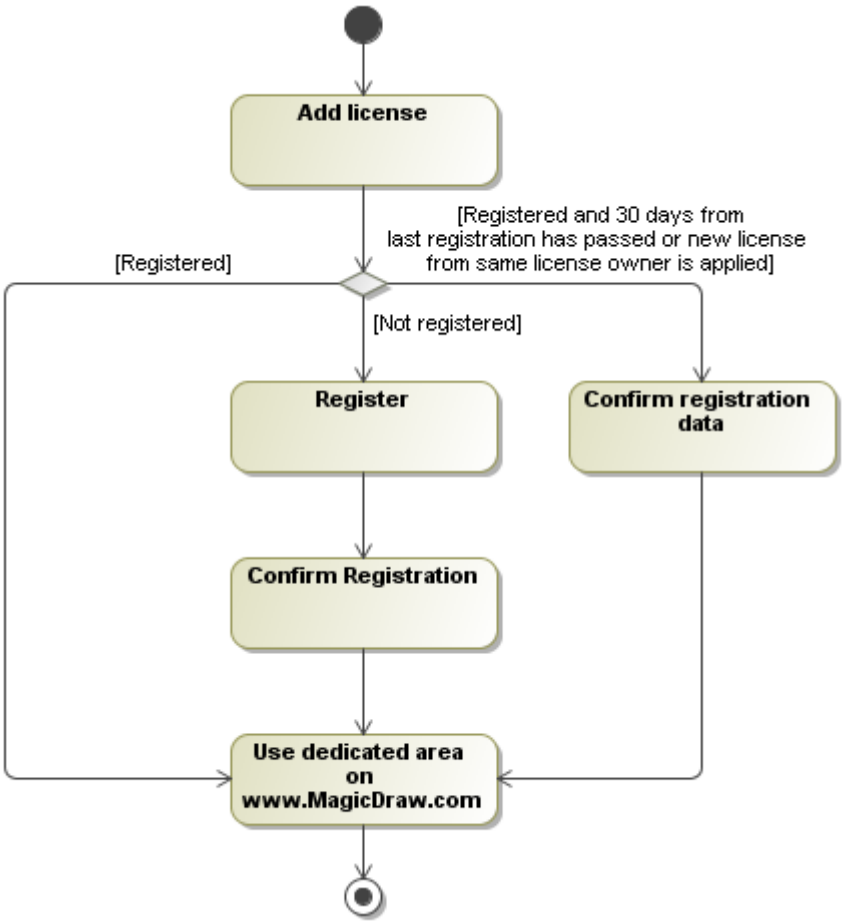


Figure 24 -- Registration Workflow

Adding a License

MagicDraw always checks the registration status at startup after license has been provided, an unlock key has been added, or a Floating server has been selected.

Registering

If you are not a registered user, MagicDraw will open the **Registration** dialog at startup, prompting you to register.

NOTE

You do not have to complete the online User Registration to use MagicDraw, however, it is recommended to do so to receive the benefits available for a registered user. You can complete Product Registration at any time by clicking **Register** on the **Help** menu.

To complete Product Registration:

1. On the main menu, click **Help** > **Register**. The **Registration** dialog will open (see the following figure).
2. Provide the requested information in the **Registration** dialog (some information has been profiled from the key owner profile in order to provide a more usable registration process).
3. Share your experience about the tool (optional).
4. Click **Register** to send data to the server and to receive a confirmation email later on.

Copyright © 1998-2015 No Magic, Inc.

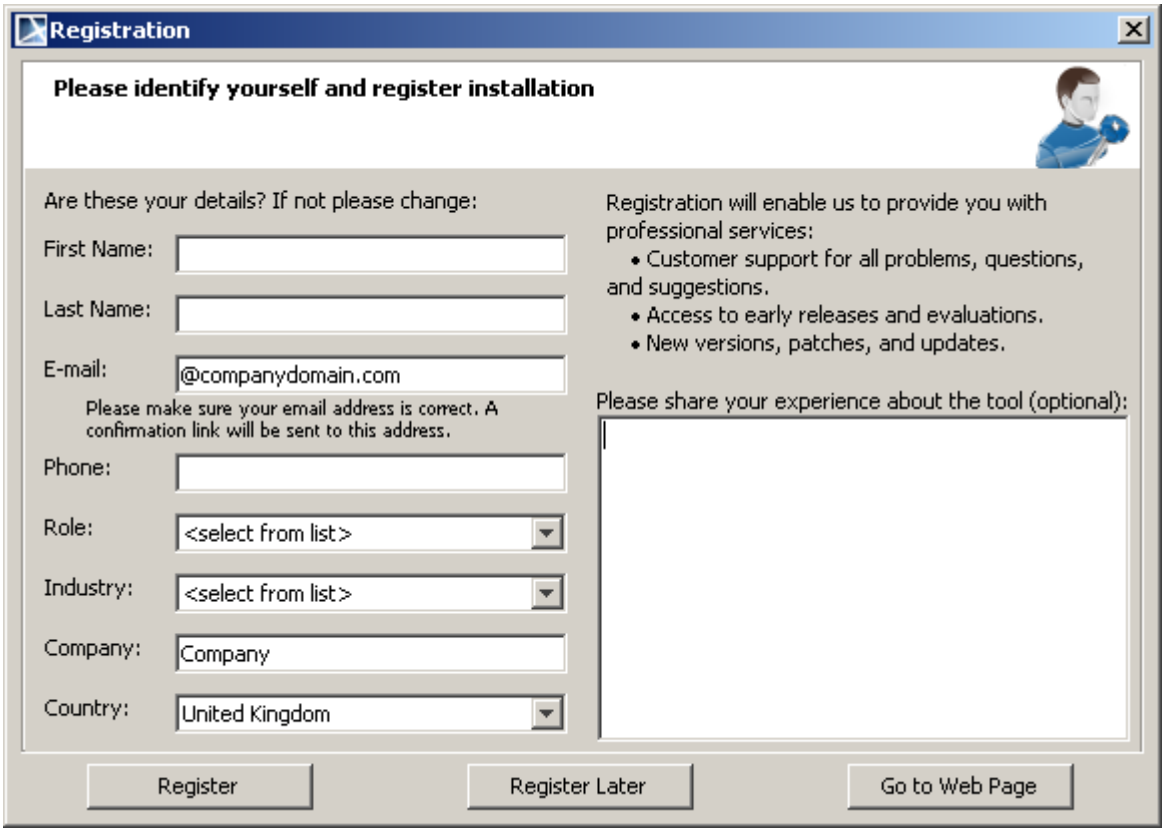


Figure 25 -- The Registration Dialog

NOTE

- You can click the **Confirm Later** button to register at a later time and close the **Registration Confirmation** dialog.
- A message reminding about Registration Confirmation will appear at MagicDraw startup if you do not complete the registration process.
- No Magic, Inc. respects your privacy. We will only use your personal information for communications and management of your online account, and the products you register with your account.

Confirming Your Registration

An e-mail with registration data and a confirmation link will be sent to the e-mail address provided during registration. Click the confirmation link to confirm the registration process and create or navigate (if it has been created) to your dedicated area at www.nomagic.com.

Logging in to Your Dedicated Area at www.nomagic.com

Use your login name and password received upon confirming your registration to log in to the user dedicated area at www.nomagic.com.

Your registration will enable us to provide you with the following professional services:

- Dedicated customer support for all problems, questions, and suggestions.
- Access to early releases and evaluations.
- New versions, patches, and updates.

Copyright © 1998-2015 No Magic, Inc.

Registration Data Confirmation

After a period of 30 days has passed since the date of your registration or a new license from the same licensed owner has been applied, you will be requested to confirm that you are the one using the software installation at MagicDraw startup. A profiled **Registration Confirmation** dialog will open. Check the correctness of data and click the **Confirm** button.

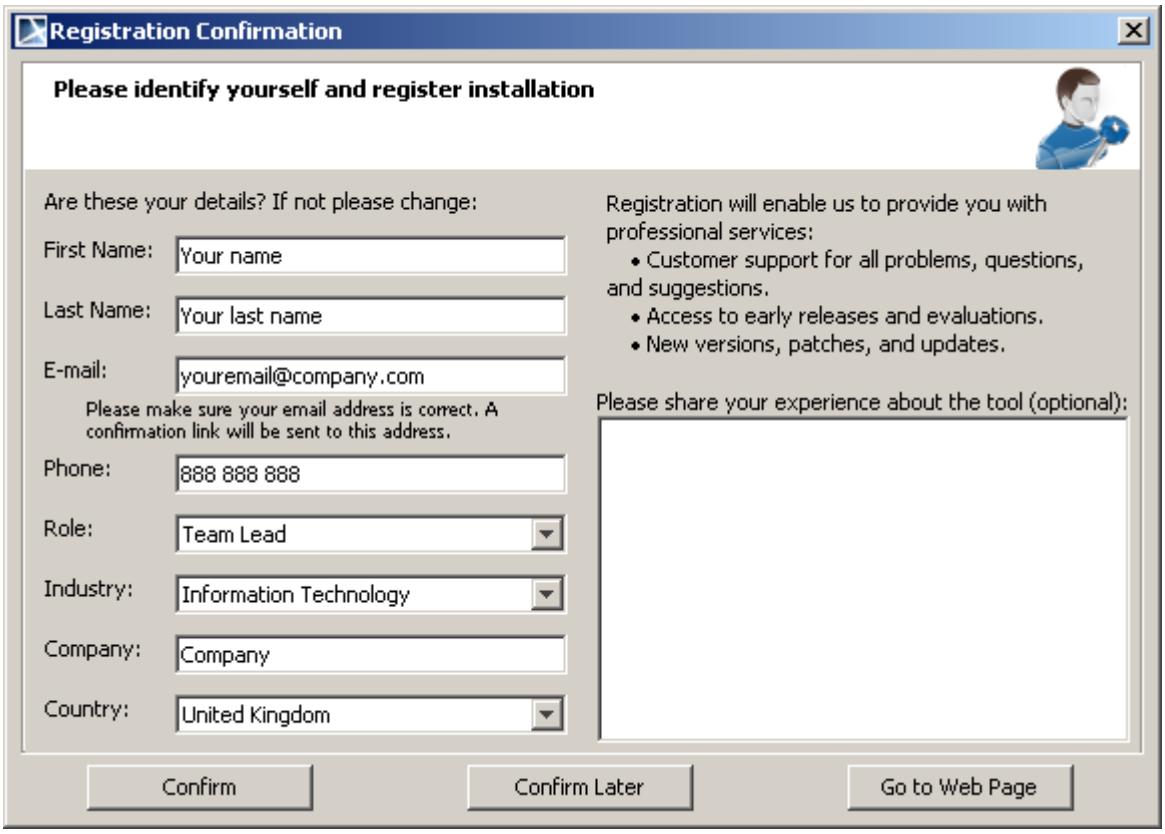


Figure 26 -- Registration Confirmation

NOTE

You will receive no email upon confirming your registration.

If you are registering as a new user, with different profile information, the **Update Existing Profile** question will appear. You can either update your existing profile or identify yourself as a different user using the product installation.

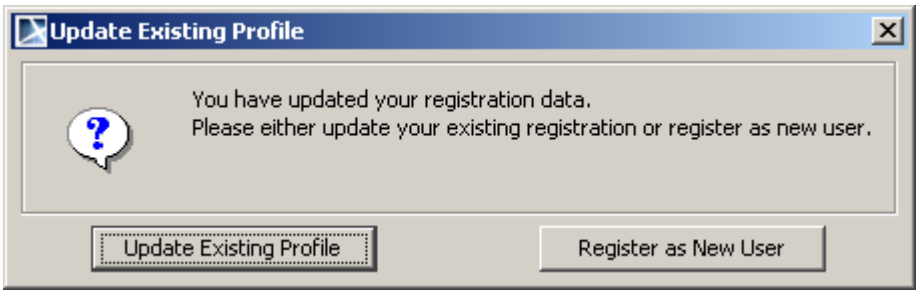


Figure 27 -- Updating Existing Profile

Bug Report

If you are a registered user, your personal information will be provided in the **Report an Issue** dialog. Submit notifications of software errors dialog is available from **Help** menu > **Report an Issue**.

Copyright © 1998-2015 No Magic, Inc.

If the provided information does not correspond to the information you have provided during registration, once you click the **Send** button to submit the bug, you will be asked to register or update your information. The **Registration** dialog will open and profiled with your personal information from previous registration with changes from the **Report an Issue** dialog or details from the **Report an Issue** dialog if you have not yet registered.



For more information about bug reporting,
see "Reporting issues directly from MagicDraw" on page 41.

Troubleshooting

Issue	Solution
Registration is requested on each startup	The **Registration Confirmation** dialog will open each time MagicDraw starts until you register the product installation. The registration process is straightforward and requires minimum data for you to access dedicated online support, answers database, new products evaluation, and beta products. Clicking the **Confirm Later** button will close the **Registration Confirmation** dialog.
Registration confirmation is requested at each startup	If you do not click the registration confirmation link, installation will not be confirmed. You may request a new confirmation email from the registration confirmation message.
No email with a confirmation link is received	Your spam or virus filter may filter out the e-mail with a confirmation link. If that is the case, you can request a new confirmation e-mail by clicking **Help** > **Register** on the main menu.
The No connection to registration server message opens	Check your internet connection and try to reconnect. User Registration is not required to use MagicDraw.
You are getting "cannot connect to server" message on registration dialog invocation from **Help** menu -> **Registration**	Check your internet connection and try to reconnect. User Registration is not required to use MagicDraw.
The Registration dialog does not opens at Startup	• You have already registered and the period of 30 days since the date of your last registration has not been used up. • There are network limitations to check your registration status. User Registration is not required to use the MagicDraw.



If you encounter problems during the registration process, please contact us registration@nomagic.com

Copyright © 1998-2015 No Magic, Inc.

MagicDraw Configuration Files Location

By default MagicDraw configuration and auxiliary files are stored in the following location:

OS	Configuration files storage
Windows Vista/7/8	C:\Users\<USERNAME>\AppData\Local\.magicdraw\<md.version.number>
Windows 2000/XP	C:\Documents and Settings\<USERNAME>\Local Settings\Application Data\.magicdraw\<md.version.number>
Windows NT4	C:\WINNT\Profiles\<USERNAME>\Local Settings\Application Data\.magicdraw\<md.version.number>
Other OS	<user.home>/.magicdraw/<md.version.number>

To get the exact path to the configuration files

1. From the **Help** menu, select **About MagicDraw**. The **About** screen opens.
2. Click the **Environment** tab.
3. Click the hyperlink next to **Configuration files**. The folder containing MagicDraw configuration files opens.

You can also save configuration files in any of the following locations:

- <MagicDraw installation directory>
- Common Application Data location (on Windows OS)
- Other directory

To store MagicDraw configuration files in MagicDraw installation directory

1. Open the *magicdraw.properties* file, which is located in <MagicDraw installation directory>\bin.
2. Change the parameter value from

```
-DLOCALCONFIG=true
```

to

```
-DLOCALCONFIG=false
```

3. On Windows OS, add the additional parameter

```
-DWINCONFIG=false
```



It is important to add the *-DWINCONFIG=false* parameter on Window OS, otherwise the configuration files will not be stored at MagicDraw installation directory.

To store MagicDraw configuration files in Common Application Data location on Windows OS

1. Open the *magicdraw.properties* file, which is located in <MagicDraw installation directory>/bin.
2. Change the parameter value from

```
-DLOCALCONFIG=true
```

to

Copyright © 1998-2015 No Magic, Inc.

```
-DLOCALCONFIG=false
```

NOTE

The Common Application Data location is the following:

- On Windows NT4 - C:\WINNT\Profiles\All Users\Application Data
- On Windows 2000/XP - C:\Documents and Settings\All Users\Application Data
- On Windows Vista/7/8 - C:\ProgramData

To store MagicDraw configuration files to your chosen location

Do either:

1. Go to the folder, where the configuration files are stored (for the exact location, see the preceding table.
2. Create a new file named *magicdrawredirect*.
3. In the file, type the absolute path to the folder, wherein MagicDraw configuration files will be saved, e.g., C:/<directory name>.

Or:

4. Open the *magicdraw.properties* file, which is located in <MagicDraw installation directory>\bin.
5. Change the parameter value:

```
-Dlocalconfig.location=<absolute path to a chosen location>
```

NOTE

If you have defined to store files in the MagicDraw installation directory (see section "To store MagicDraw configuration files in MagicDraw installation directory"), files will not be stored in the defined location.

Importing Configuration from Earlier Version

When you start MagicDraw for the first time, the **Import Configuration** dialog opens. Using this dialog, you can choose one of the following:

- Use the default MagicDraw configuration.
- Import the settings and environment options from the earlier MagicDraw installation.

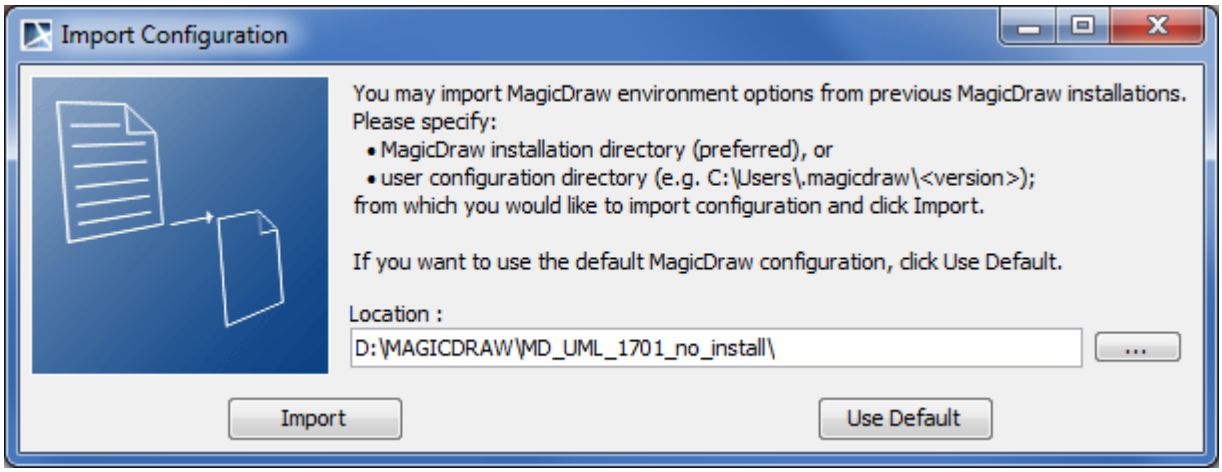


Figure 28 -- Import Configuration dialog

Copyright © 1998-2015 No Magic, Inc.

Disabling Inbound Network Activity

To disable MagicDraw inbound network activity

1. Open the *magicdraw.properties* file, which can be found in *<MagicDraw installation directory>\bin*.
2. In the JAVA_ARGS line, add the following argument:

```
-JAVA_ARGS: - DDISABLEINBOUNDCON=true
```



After disabling the inbound network activity you will not be able to

- Open MagicDraw Help
- Submit an issue

Also be advised that double-clicking a project file (*.mdzip*) will open the project in a new MagicDraw window.

Updating

An automatic updates feature is implemented in MagicDraw. Notification and update of all the service packs can be done automatically.

To check for updates

- On the **Help** menu, click the **Check for Updates** command.

To enable an automatic checking for MagicDraw updates

1. On the **Options** menu, click **Environment**. The **Environment Options** dialog appears.
2. In the **Update** options list, change the **Check for Updates** option value to one of the following:
   - **Manually**
   - **On startup**
   - **Once a day**
   - **Once a week**
   - **Once a month**

   It is recommended to check for updates once a month.

Update Information window

The **Update Information** window appears if updates are found. Do one of the following:

- Click **Update to New Version**, to apply the latest released product version.
- Click **Apply Patch**, to apply the latest released patch.
- Click **Close**, to close the dialog.

Copyright © 1998-2015 No Magic, Inc.

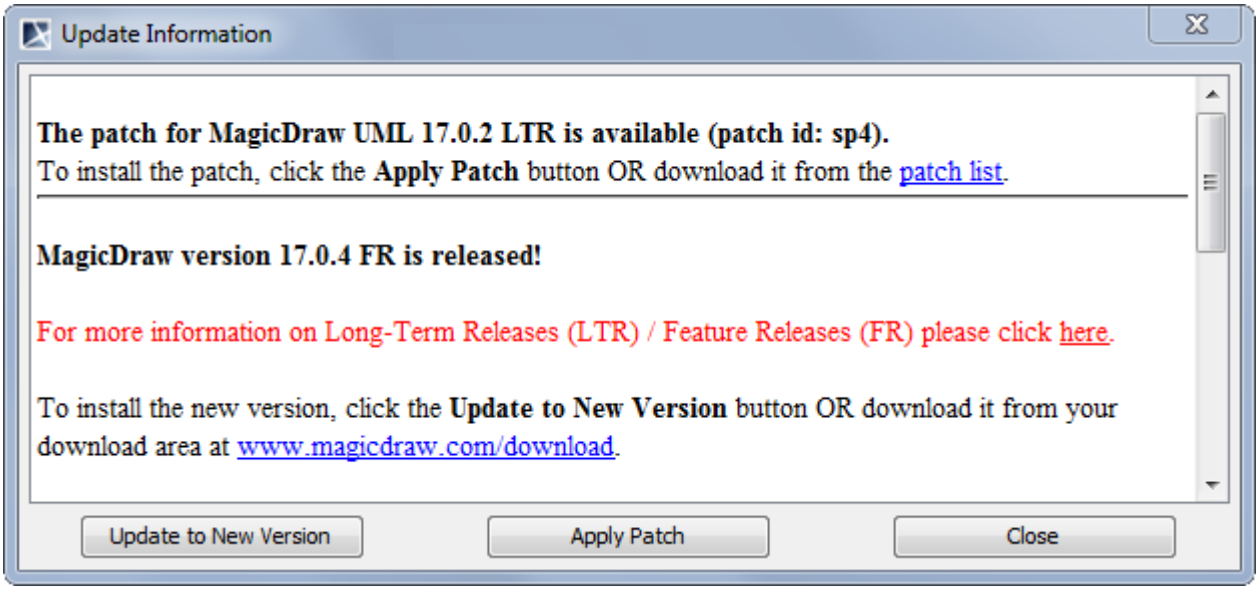


Figure 29 -- Update Information window

Copyright © 1998-2015 No Magic, Inc.

3 USING MAGICDRAW

In this chapter, you will find the following sections:

- "Customizing and Selecting Perspective" on page page 65
- "Understanding MagicDraw User Interface (UI)" on page page 70
- "Customizing Environment Options" on page page 96
- "Improving Performance" on page 97
- "Look and Feel: Controlling the Interface" on page page 103
- "Assigning Shortcut Keys" on page page 104

Customizing and Selecting Perspective

Launch MagicDraw for the first time and after the application starts, the **MagicDraw Startup** dialog appears. In this dialog you can select your work perspective.

Due to the growing number of MagicDraw features, many features may be configured for standard or expert user. MagicDraw can satisfy the needs of different software development process roles. In order to better satisfy user needs, MagicDraw configuration depends on Perspective.

Perspectives allow:

- The selection of a predefined MagicDraw configuration and features according to your software development process role.
- Finding features faster, because there are less of them.
- To choosing a suitable MagicDraw experience mode with a single click.
- Customizing a set of predefined features and configuration based on user needs.

There are four perspectives in MagicDraw:

- **Full Featured** - Perspective provides all features available in MagicDraw and installed plugins.
- **Quick Start** - Quick Start perspective provides basic features dedicated for modelling and not overcrowded interface for quick learning. Code engineering, transformations and other advance features are hidden, however easily reachable in expert mode of this perspective.
- **Software Architect** - this perspective provides features primarily involved in designing and implementing projects. It is a set of roles consisting of Software Architect, Designer, Interface designer, and Database designer. This is the default MagicDraw configuration. All functionalities are available for expert mode.
- **System Analyst** - this perspective provides features primarily dedicated to obtaining requirements and modeling the system. Analysis features are highlighted. Configuration is modeling oriented. Code engineering, transformations and other features are hidden.

To set the perspective for the MagicDraw environment

- Launch MagicDraw for the first time. The **MagicDraw Startup** dialog will appear with the possibility to switch between perspectives. Select the desired perspective from the list and click the **OK** button.

Copyright © 1998-2015 No Magic, Inc.

- From the **Options** menu, choose **Perspectives** and then click **Perspectives**. The **Select Perspective** dialog opens. Select the desired perspective and click the **Apply** button.
- On the **Perspectives** toolbar, select the desired perspective from the available perspectives list.



For more information about the Perspectives toolbar, see "Validation Toolbar" on page 1142.

Customizing MagicDraw Perspectives

Perspective customization allows the grouping of functional MagicDraw features to standard/expert modes. Customization also allows the user to hide unnecessary commands, which makes MagicDraw simpler and faster to use.

To open the **Customize Perspectives** dialog

- From the **Options** menu, choose **Perspectives** and then **Customize**.
- In the **Perspectives** dialog, click the **Customize** button.

MagicDraw has seven customizable areas in different perspectives. They are as follows:

- Main menu
- Main toolbars
- Diagram toolbars
- Diagram modeling elements toolbar
- Smart manipulators
- Shortcut menu actions
- Create Diagram dialog

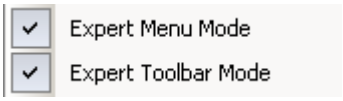
Each of these areas has a set of commands, which can be shown in standard/expert mode, or hidden.

To customize the selected MagicDraw area in a predefined perspective

1. In the **Customize Perspectives** dialog, select the perspective and click the **Edit** button near the selected MagicDraw area. The appropriate **Customize** dialog opens.
2. Expand tree sections and select radio buttons beside items in the tree depending on your choice for **Standard and Expert**, **Expert** only or **Hidden** modes.
3. Click **OK** to save changes, then **OK** in the **Customize Perspectives** dialog, and then **Apply** in the **Select Perspective** dialog (if needed).

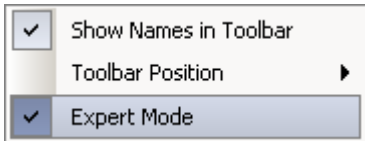
To switch between Standard/Expert menu, toolbar, or diagram toolbar modes

- From the toolbar shortcut menu, select/clear the **Expert Menu Mode** or **Expert Toolbar Mode** check box.



Copyright © 1998-2015 No Magic, Inc.

- From the diagram buttons toolbar menu, select/clear **Expert Mode** check box.



Select Perspective dialog

Launch MagicDraw for the first time. The **Select Perspective** dialog appears with the possibility to switch between perspectives.

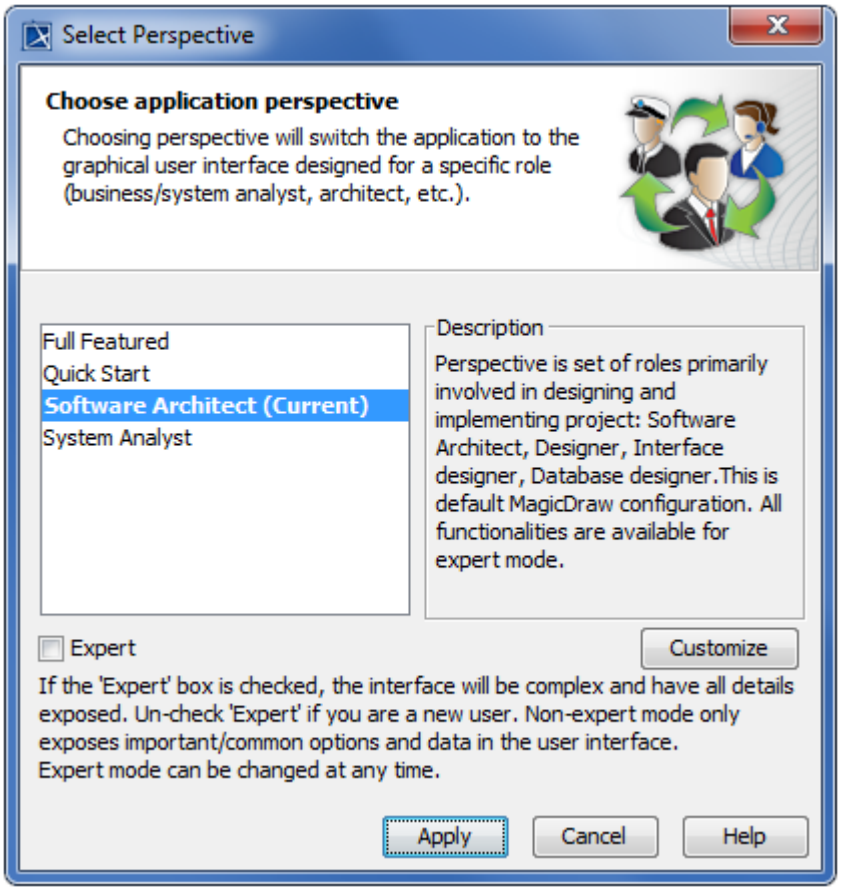


Figure 30 -- Select Perspective dialog

Element Name	Function
Full Featured **Quick Start** **Software Architect** **System Analyst**	Possible perspectives to set, which will load a predefined MagicDraw configuration.
Expert	If not selected, a simplified MagicDraw interface with the most popular items and features will be opened and the specification properties will be shown in Standard Mode.
Customize	Opens the **Customize Perspectives** dialog wherein you can customize the selected perspective for making the program simpler and faster to use, for example, to group functional features, to show or hide selected commands.
Apply	Applies changes to the system.
Cancel	Discards all changes and closes the dialog.

Copyright © 1998-2015 No Magic, Inc.

Element Name	Function
Help	Displays MagicDraw Help.

You can also change the perspective while working with MagicDraw.

To change the MagicDraw perspective

1. From the **Options** menu, choose **Perspectives** > **Perspectives**. The **Select Perspective** dialog opens.
2. In the opened dialog, select a desired perspective and click **Apply**.

Copyright © 1998-2015 No Magic, Inc.

Customize Perspectives dialog

To open the Customize Perspectives dialog

- From the **Options** menu, choose **Perspectives** and then **Customize**.

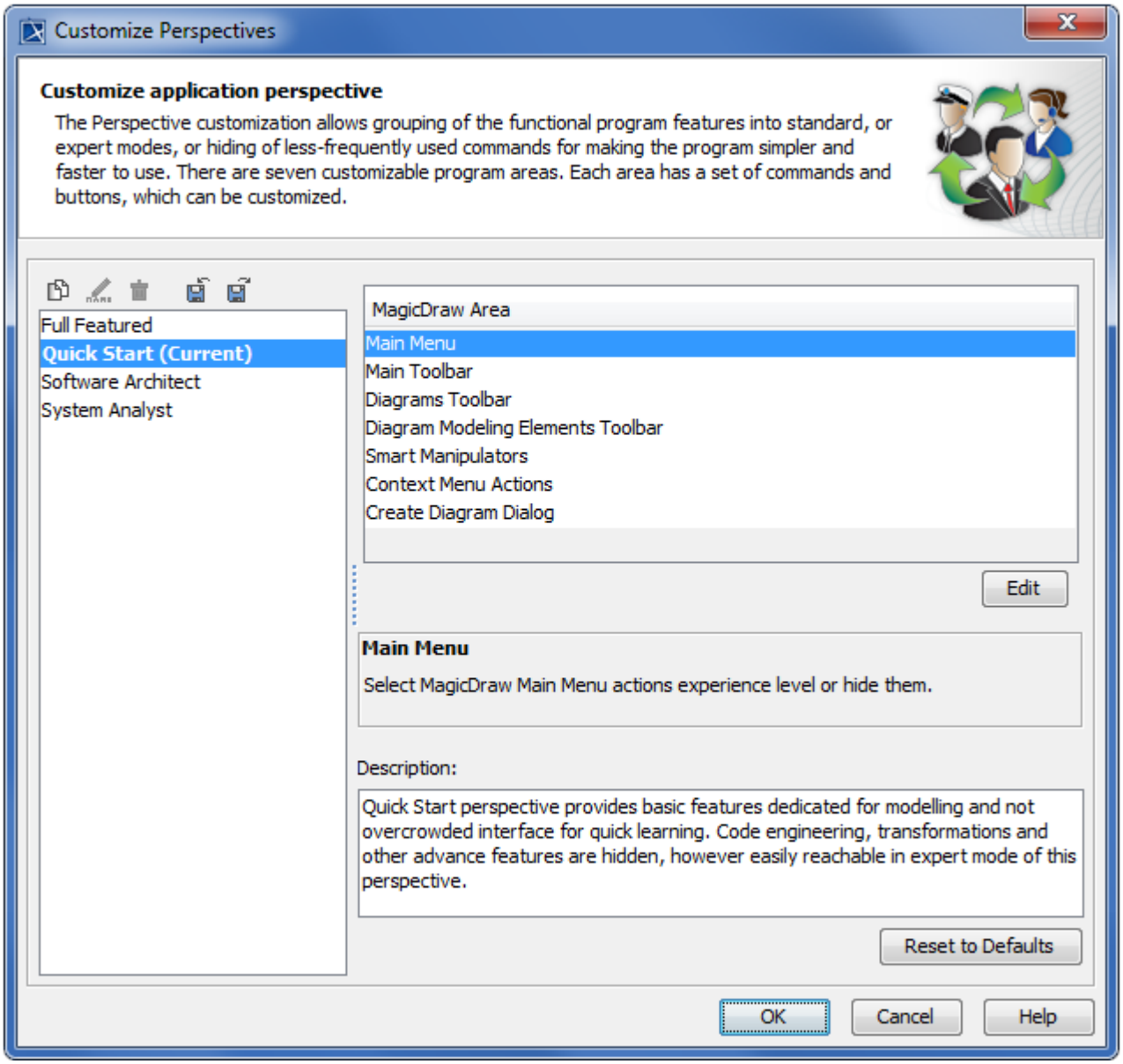


Figure 31 -- Customize Perspectives dialog

Element Name	Function
Clone Selected Perspective 	Copies the selected perspective to a new one.
Rename Selected Perspective 	The **Enter Perspective Name** dialog opens. Change name of the perspective and click **OK**. Renaming can also be performed using the F2 shortcut key.

Copyright © 1998-2015 No Magic, Inc.

Element Name	Function
Remove Selected Perspective 	Deletes the selected perspective from the list.
Import New Perspective 	The **Open** dialog appears. Select *.umd extension file and click **Open** to import the perspective into the MagicDraw environment.
Export Selected Perspective 	The **Save** dialog opens. Type a name for the created perspective and click **Save** to store it as *.umd extension file.
Full Featured **Quick Start** **Software Architect** **System Analyst**	List of possible perspectives, which will load the predefined MagicDraw configuration.
MagicDraw Area	List of customizable toolbars and command sections.
Edit	Click the **Edit** button to open the **Customize Main Menu** dialog in which a commands mode could be changed by selecting radio buttons.
Description	Displays short description about each selected area.
Reset to Defaults	Resets changes back to the default configuration.

Understanding MagicDraw User Interface (UI)

MagicDraw window has the following parts:

- Main menu
- Main toolbars
- Model Browser
- Diagram toolbars
- Diagram pallet
- Diagram pane

Copyright © 1998-2015 No Magic, Inc.

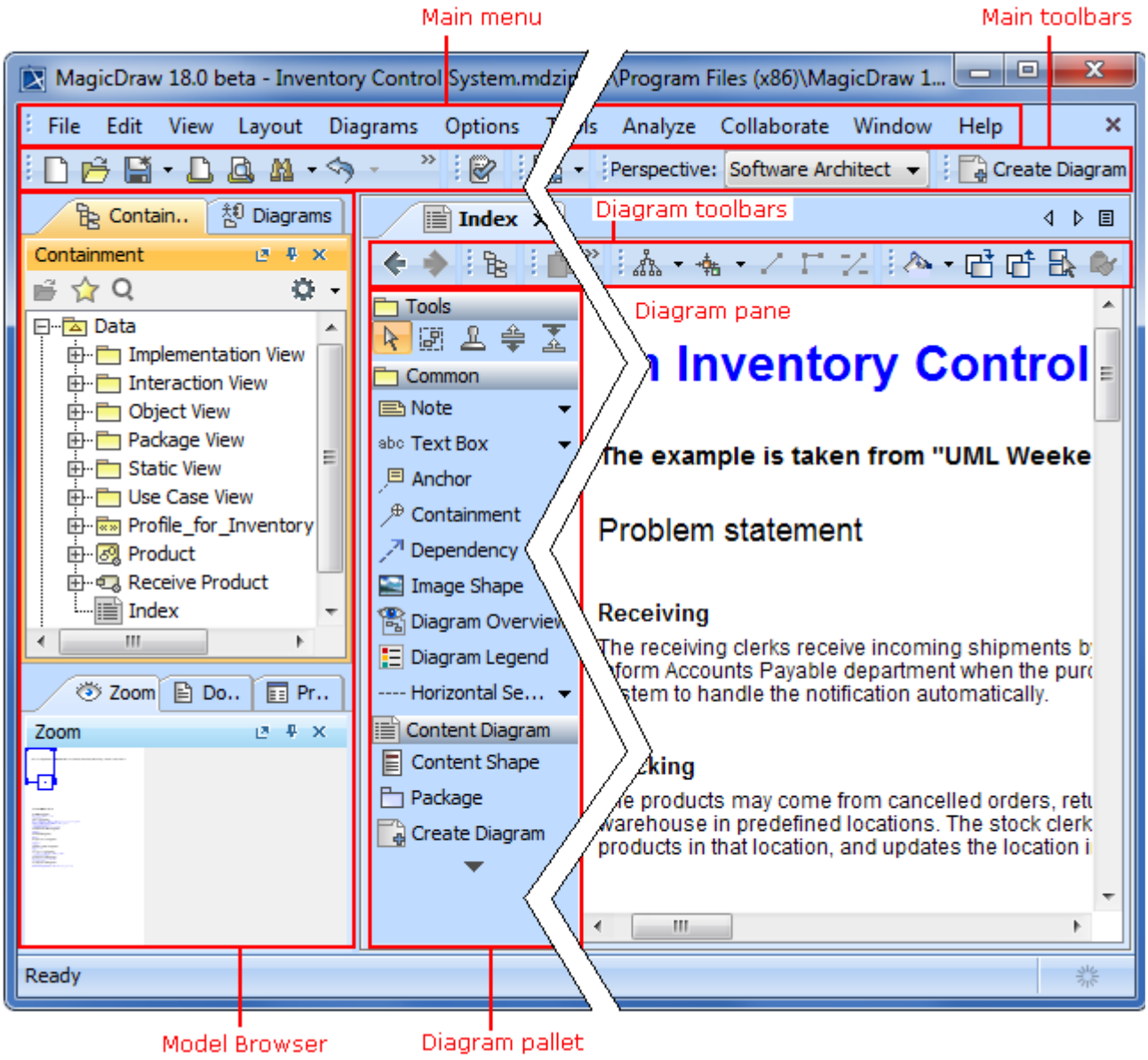


Figure 32 -- Structure of MagicDraw window

Nearly all MagicDraw commands can be accessed from multiple places within MagicDraw:

- Main menu
- Main toolbars
- Model Browser
- Diagram toolbars
- Shortcut menus (accessible by right-clicking)
- Shortcut keys
- Smart manipulators (accessible by selecting a symbol on the diagram pane)

Copyright © 1998-2015 No Magic, Inc.

The following table shows the accessibility of several commands in different ways

Function	Accessibility
Main operations of editing (copy, cut, paste, delete)	• **Edit** menu. • Main toolbar buttons. • Shortcut keys. • Shortcut menu commands.
Opening of the Specification window	• By double-clicking the model element. • **Specification** command from the element shortcut menu on a diagram or in the Model Browser. • When adding one model element to another model element from the Specification window or the Model Browser.
Defining symbols properties (font, color, etc.)	• Symbol shortcut menu > **Symbol Properties**. • **Edit** menu > **Symbol** > **Symbol Properties**. • **Options** menu > **Project**. • Main toolbar buttons



The toolbar of a particular diagram presents the paths and shapes available for the corresponding diagram. If an arrow is placed on the diagram toolbar button, select a button representing the corresponding model element by right-clicking the button.

Menus

The description of all menu commands you can find in "Menu System" on page 1127.

You can customize menu items by selecting and/or modifying perspectives. More information about customizing perspectives, you can find in "Customizing and Selecting Perspective" on page 65.



Various plugins might bring additional menu items.

Toolbars

Toolbars help to speed up your work with MagicDraw, when performing commonly used tasks. There are main toolbars and diagram toolbars in the MagicDraw window.

To show or hide different toolbars

- Right-click the toolbars area and then select or clear the check boxes of the toolbars you want to be displayed or hidden.



You can also save your own toolbars configuration and set it as a default one (for more information see "Customizing toolbars" on page 73).

Main toolbars

The main window of MagicDraw contains the following main toolbars:

- **File** (main)
- **Switch Projects**
- **Perspectives**

Copyright © 1998-2015 No Magic, Inc.

- **Collaboration**
- **External Tools**
- **Validation**
- **Create Diagram**



For detailed information about the commands of the main toolbars, see "Main Toolbars" on page 1140.

Diagram toolbars

The main window of MagicDraw contains the following diagram toolbars:

- **Navigation**
- **Layout**
- **Edit**
- **Symbol Editing**
- **Shape Editing**
- **Path Editing**
- **View**



For detailed information about the commands of the diagram toolbars, see "Diagram Toolbars" on page 1143.

Customizing toolbars

Toolbars configuration shortcut menu has the following commands:

Check box	Function
Rearrangable	If selected, it is possible to change the toolbar position by selecting the dotted line in front of the desired toolbar group and dragging it to a new location.
Hidable	If selected, there is no possibility to close a separately opened toolbar group (for example, dragged diagram pane) with the X button on the right top corner.
Floatable	If selected, the toolbar group can be dragged to any desirable position inside the MagicDraw borders.
Expert Menu Mode	If selected, all menu commands will be listed on the menu. Otherwise, the command list will be shortened and you can expand it by clicking the arrow on the bottom.
Expert Toolbar Mode	If selected, displays all toolbar buttons, which were marked to be shown in the Expert mode perspective.
Customize	Opens the **Customize Toolbars** dialog.

To add a new toolbar

1. From the toolbars configuration shortcut menu, choose **Customize**. The **Customize Toolbars** dialog opens.
2. Click **Add** > **New Toolbar**.
3. Type the name for a new toolbar.
4. Click **OK**.

Copyright © 1998-2015 No Magic, Inc.

To add a new button to the selected toolbar

1. From the toolbars configuration shortcut menu, choose **Customize**. The **Customize Toolbars** dialog opens.
2. Click **Add** > **Button**.
3. Select the desired command.
4. Click **OK**.

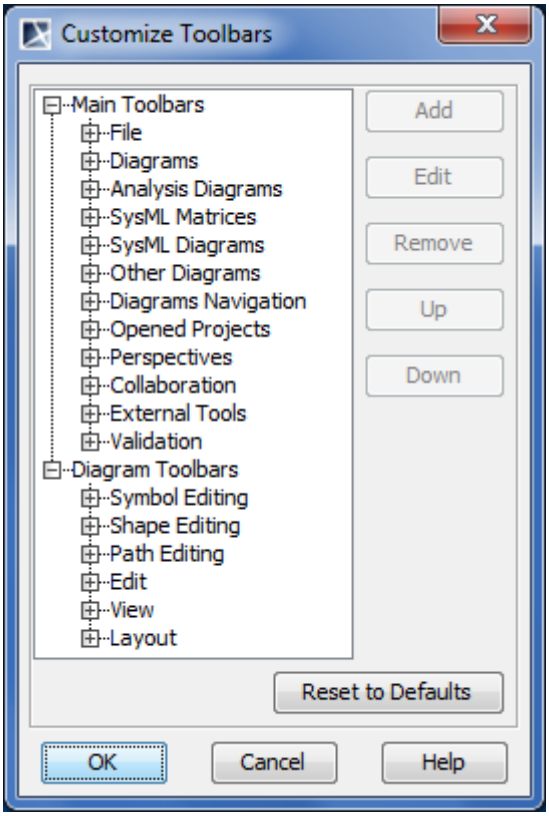


Figure 33 -- The Customize Toolbars dialog

Button	Function
Add	Adds a new button or a toolbar.
Edit	The **Edit Icon** dialog opens. Click the "..." button to add an icon to the selected toolbar button.
Remove	Removes the selected button from the toolbar section.
Up	Moves the selected button up the toolbar list.
Down	Moves the selected button down the toolbar list.
Reset to Defaults	Resets changes made to the toolbar back to the default settings.

Copyright © 1998-2015 No Magic, Inc.

Model Browser



MagicDraw Basics

The Model Browser provides a visual representation of the hierarchy of your model elements.

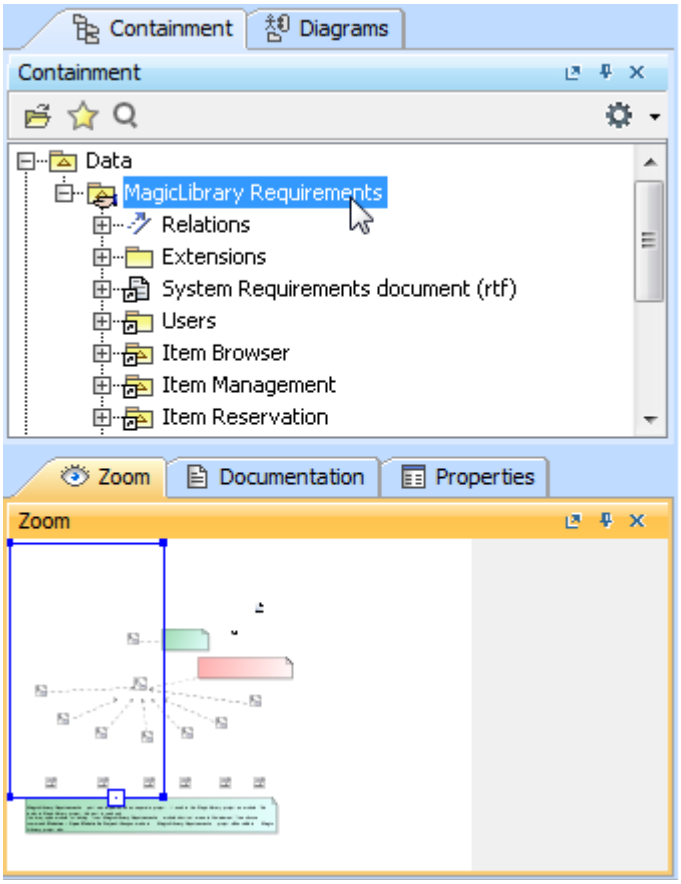


Figure 34 -- Example of Model Browser

The items in the Model Browser hierarchy are either:

- Compressed - a plus sign next to an icon indicates that the icon is compressed. It means that the element contains other model elements. This is the default setting when you start the MagicDraw. Click the plus sign to expand the icon and view its subordinate items.
- Expanded - a minus sign next to an icon indicates that the icon is fully expanded. Click the minus sign to collapse the item.

If there is no plus or minus sign next to an icon, it does not contain other model elements.

The Model Browser is a hierarchical navigation tool that allows you to manage your model data, including packages, components, classes, all UML diagrams, extension mechanisms, and other data. The Model Browser may be used as an alternative tool to the menus and toolbars that are in MagicDraw. It is easier to work with project diagrams and data elements using the Model Browser. In the Model Browser, you can perform the following operations:

- Create and specify model elements.
- Copy, cut, and paste model elements.
- Delete of model elements.
- Drag model elements to the Diagram pane and inside the Model Browser.

Copyright © 1998-2015 No Magic, Inc.

- Drag data in the Code engineering sets (you may create data in the Data branch, drag it to the Code Engineering sets, and then the round trip object is created automatically).
- View the hierarchy of all model elements.
- Create a symbol for the selected model element in the current diagram.
- Manage diagrams.
- Manage extension mechanisms, such as constraints, stereotypes, and tagged values.
- Java reverse of a class from the classpath.
- Adjust the code engineering sets.
- Generate code for particular sets.
- Filter visible items by any model type (for example, class, package, operation, component, state, and others - for both views and dates).
- Sort the visible items of the selected model element.
- Sort all model elements.
- Display search results.
- Display locked elements in the server project.

The Model Browser window is divided into two parts:

1. **Containment** tab / **Inheritance** tab / **Diagrams** tab / **Model Extensions** tab / **Search Results** tab / **Lock View** tab:
   - The Containment tab groups data into the logical sets.
   - The Inheritance tab represents the model hierarchy of the project.
   - The Diagrams tab groups diagrams according to the diagrams type or shows them as a list.
   - The Model Extensions tab represents all predefined and created constraints and stereotypes.
   - The Search Results tab displays search results.

   The Search Results tab is not displayed by default. It appears after you perform searching.

   - The Lock View tab represents locked elements of the server project.

   The **Lock View** tab is not displayed by default. It can appear if you are working on a server project. For more information about locked elements, see "Representation of locked elements" on page 1053.
2. **Documentation** / **Zoom Control** / **Properties** tabs.
   - The Documentation tab shows documentation associated with the selected item.
   - The Zoom tab is responsible for zooming the current diagram.
   - The Properties tab shows the specification and symbol properties of the selected element or diagram.

Related procedures

Using the Model Browser

Working with model elements in the Model Browser

Using the Model Browser

The Model Browser is displayed when at least one project is open. By default, the Model Browser is placed at the left side of the main window. It is possible to move the Model Browser to any place on the MagicDraw window. Also, all tabs can be viewed separately and you can set up the Model Browser according to your needs

Copyright © 1998-2015 No Magic, Inc.

by hiding the desired tabs. See the following procedures for more information on how to manage the Model Browser and items in it.

To change the size of either part of the Model Browser

- Drag the bar that separates the two parts.

To change the Model Browser position

- Using the **Option** menu:
  1. From the **Options** menu, select **Environment**. The **Environment Options** dialog opens.
  2. Click on the **Browser** option group.
  3. In the Browser environment options pane, change the **Browser Position** property to **Right** or **Left**.
- In the upper-right corner of the selected tab in the Model Browser, click the Toggle auto-hide or Toggle floating button.



This action changes the layout of the MagicDraw interface.

To close or reopen the desired tab of the Model Browser

- From the **Window** menu, choose the Model Browser tab you want to close or open.

To reset all Model Browser tabs to the default position

- From the **Window** menu, choose the **Reset Windows Configuration** command.

To sort items in the Model Browser alphabetically

1. From the **Options** menu, select **Environment**. The **Environment Options** dialog opens.
2. Click the **Browser** option group.
3. In the Browser options pane, set the **Sort Always** value to *true* (the default value is *true*).

Related concepts

Model Browser

Copyright © 1998-2015 No Magic, Inc.

Containment tab

The **Containment** tab displays model data grouped in logical sets. Using the Containment tab, you can quickly access each model element, change a model element specification, or create a new model element.

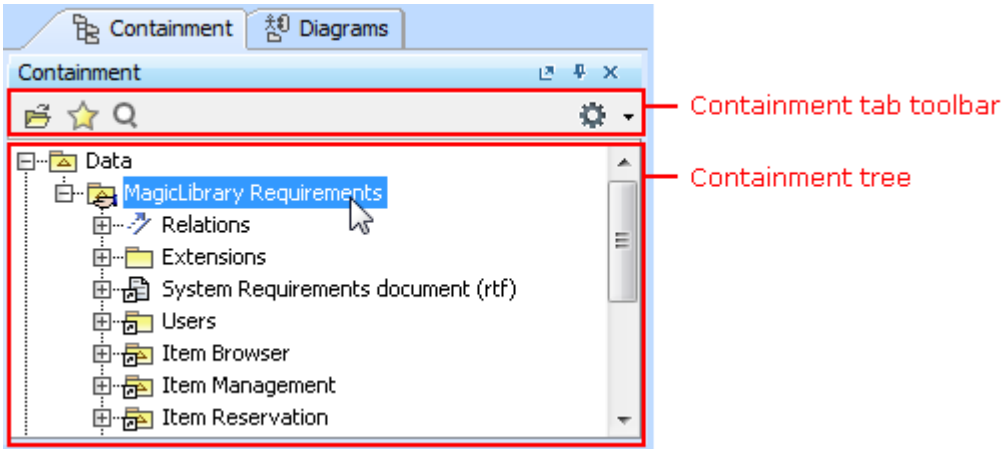


Figure 35 -- Containment tab

To open the Containment tab

Do one of the following:

- At the top of the Model Browser, click the **Containment** tab.
- If the Containment tab is hidden, from the **Window** menu, select **Containment**.

TIP!

You can change some Containment tab properties in the **Environment Options** dialog the **Browser** options group. For more information about using the **Environment Options** dialog, see "Customizing Environment Options" on page 96.

The Containment tab consists of the following two components (see the figure above):

- Containment tab toolbar
- Containment tree (see the following figure)

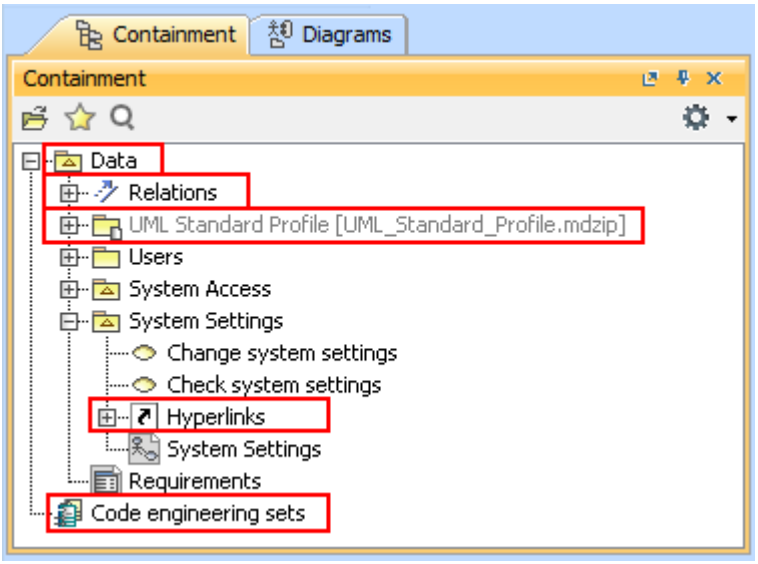


Figure 36 -- Containment tree

The Containment tree includes

Copyright © 1998-2015 No Magic, Inc.

- Data model:
  - The Relations branch
  - The UML Standard Profile
  - The Hyperlinks branch
- Code engineering sets



For more information about working with the Containment tree, see "Working with model elements in the Containment tree" on page 81.

Data model

The *Data* model in the Containment tree is the top model element wherein the entire model structure is created. The *Data* model represents the model element that is named *Data*. All model elements are stored in the *Data* model. You can create the structure of your project by creating packages in the *Data* model. This helps to distribute model elements into the logical groups. For more information about working with a model, see "Model" on page 935. For more information about packages, see "Package" on page 957.

Along with model elements that are created by the user, the *Data* model may contain the following default items:

- The **Relations** branch. This branch appears when at least one relationship is created in the project. The *Relations* branch collects all relationships from the particular owner. For example, if the *Relations* branch is in the *Data* model, that means, all relationships from the *Data* model are listed in the *Relations* branch owned by the **Data** model. The *Relations* branch can exist in other elements which may have inner elements and relations among these elements. For example, in packages, classes, and others.
- The **UML Standard Profile** package. This is a profile which is included to every MagicDraw project. It contains a list of stereotypes, data types, and other elements from the UML 2 metamodel. By default, the *UML Standard Profile* is hidden. To display the *UML Standard Profile*, click the **Show Auxiliary Resources** button. There can be more standard profiles (such as, SysML, UPDM, Java, and others). For more information about the standard profiles, see "Working with Standard Profiles" on page 196.

  We highly recommend not to modify our provided standard profiles and libraries as it could cause problems on version updates, plugins, core MagicDraw tool malfunctions, and model corruptions.
- The **File View** package. This package appears when at least one code engineering set is created. The *File View* package contains components that are created on code reverse and represents source files. For more information about working with code engineering sets in the Containment tab, see "Code engineering sets" on page 80.
- The **Hyperlinks** branch. This branch appears when at least one hyperlink is added to the particular element. The Hyperlinks branch contains the list of hyperlinks to the file, element/symbol, or web page. For more information about defining hyperlinks, see "Defining Hyperlinks" on page 340.

For more information about managing model elements in the Model Browser, see "Working with model elements in the Model Browser" on page 91.

Copyright © 1998-2015 No Magic, Inc.

Code engineering sets



Code engineering is available in Professional, Architect, and Enterprise editions.

Code engineering sets can be considered a gateway between your source code and model data. Using these sets, you can perform Java, C++, C#, IDL, CIL, CORBA IDL, EJB 2.0, and WSDL round-trip code engineering, that is, code generation and reverse engineering.

To display Code engineering sets

- In the Containment tab toolbar, click the **Options** button, and then click to select the **Show Code Engineering Sets** check box.

In Code engineering sets branch, you can create a new code engineering set, generate code, and perform other actions.



For more information about the code engineering, see "MagicDraw CodeEngineering UserGuide.pdf".

Containment tab toolbar

The Containment tab toolbar contains the following buttons:

Button	ToolTip text	Description
	Open in New Tab	Opens the contents of the selected element in a new tab of the Model Browser.
	Favorites	Opens the Favorites menu for choosing to do one of the following: • Add a selected element to favorites (see "To quickly add an element to favorites" on page 417). • Open the dialog for managing favorites (see "Manage Favorites dialog" on page 419). • Navigate to a desired favorite in the Containment tree.
	Quick Find	Opens the **Quick Find** dialog. For more information, see "Quick Find dialog" on page 137.
	Options	Click to open the Options menu and then select: • **Filter** - opens the **Items Filter** dialog wherein you can select element types to be visible in the Containment tree. For more information about the filtering, see "Filtering" on page 143. • **Show Auxiliary Resources** - shows or hides modules and profiles that are used in your project. For more information, see "Data model" page 79. • **Show Applied Stereotypes** - shows or hides the applied stereotypes next to the element name in the Containment tree. For more information about working with stereotypes, see "Stereotype" on page 985. • **Show Full Types** - shows or hides the full types of typed elements (for example, operations, attributes, relationships, and other) in the Containment tree. • **Show Code Engineering Sets** - shows or hides the code engineering sets. For more information, see "Code engineering sets" on page 80.

The same actions you may perform using the shortcut menu of the Containment tree.

Copyright © 1998-2015 No Magic, Inc.

Working with model elements in the Containment tree

You can work with elements in the Containment tree using the commands available from the selected element's shortcut menu. The shortcut menu contains the following commands: .



Particular actions are available for the particular elements.

Command	Function
Create Element	Creates a new element inside the selected element. **NOTE**: This command is available if the selected element can be editable. **TIP!** For more information, see "Working with model elements in the Model Browser" on page 91.
Create Diagram	Creates a new diagram inside the selected element. **NOTE**: This command is available if the selected element can be editable. **TIP!** For more information, see "Diagramming" on page 198.
Create Relation > Outgoing / Incoming	Creates a new relation to or from the selected element. **TIP!** For more information, see "Working with Paths and Relationships" on page 227.
Open in New Tree	In the Model Browser, opens a new tab for the selected element.
Specification	Opens the element Specification window wherein you can specify various element properties. **TIP!** For more information about the Specification window, see "Specification Window" on page 273.
Go To	Opens or selects the item associated with the selected element. The item may be diagram, model element, or hyperlink. **TIP!** For more information, see: • "Symbol Usage in Diagrams" on page 576. • "To add a hyperlink from the Model Browser" on page 343. • "Traceability properties in Go To submenu" on page 521.
Refactor	Converts the selected element to the element indicated by a user. **TIP!** For more information, see "Refactoring" on page 359.
Related Elements	Opens a list of functionalities available for the selected element. **TIP!** For more information, see "Relation Map" on page 544, and "Analyzing Usages and Dependencies" on page 509.
Tools	Opens a list with tools that are available for the selected element. Depends on the selected element. **TIP!** For more information, see "Tools" on page 434.
Stereotype	Opens the drop down list, wherein you can apply an existing stereotype, or create a new one. **TIP!** For more information, see "Stereotype" on page 985.
Find	Opens the **Find** dialog. **TIP!** For more information, see "Find dialog" on page 132.

Copyright © 1998-2015 No Magic, Inc.

Apply Profiles	Opens the list of the available to apply profiles. **TIP!** For more information, see "Profile" on page 972. **NOTES:** • This command is available in the package, model, and system boundary shortcut menu. • To see this command you have to expand the shortcut menu, that is, click on the little arrow at the bottom of the shortcut menu.
Modules	Opens the list with the commands for working with modules. **TIP!** For more information, see "Project Partitioning" on page 147. **NOTES:** • This command is available in the package, model, and system boundary shortcut menu. • To display this command in the shortcut menu you have to expand the shortcut menu, that is, click on the little arrow at the bottom of the shortcut menu.
Generate Code Framework	Generates code for the selected item. Opens the Message Window with the information. **TIP!** For more information, see "MagicDraw CodeEngineering UserGuide.pdf".
Check Syntax	Checks syntax in the model according to the default code engineering language. Opens the Message Window with the information. **TIP!** For more information, see "MagicDraw CodeEngineering UserGuide.pdf".
Generate Report	Opens the list of the available report templates. **TIP!** For more information, see "MagicDraw ReportWizard UserGuide.pdf".
Reverse from Classpath	This command is available in the class shortcut menu. **TIP!** For more information, see "MagicDraw CodeEngineering UserGuide.pdf".

Related concepts

Model Browser

Inheritance tab

The Inheritance tab represents classifiers, packages, data types, and stereotypes hierarchy within your project. Inheritance according to the UML Specification is shown using the generalization relationship.



If a model element has no generalization relationship, it will not be represented in the Inheritance tab.

Copyright © 1998-2015 No Magic, Inc.

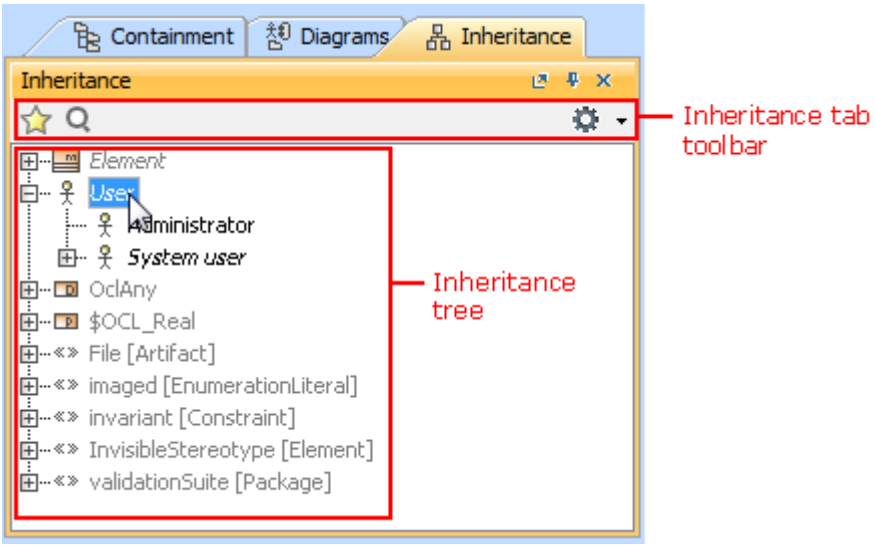


Figure 37 -- Inheritance tab

To open the Inheritance tab

Do one of the following:

- At the top of the Model Browser, click the **Inheritance** tab.
- If the **Inheritance** tab is hidden, from the **Window** menu, select **Inheritance**.

TIP! You can change some Inheritance tab properties in the **Environment Options** dialog the **Browser** options group. For more information about using the **Environment Options** dialog, see "Customizing Environment Options" on page 96.

The Inheritance tab consists of the following two components:

- Inheritance tab toolbar
- Inheritance tree

TIP! For more information about working with the Inheritance tree, see "Working with model elements in the Inheritance tree" on page 84.

Inheritance tab toolbar

The Inheritance tab toolbars contains the following buttons:

Button	ToolTip text	Description
	Favorites	Opens the Favorites menu for choosing to do one of the following: • Add a selected element to favorites (see "To quickly add an element to favorites" on page 417). • Open the dialog for managing favorites (see "Manage Favorites dialog" on page 419). • Navigate to a desired favorite in the Containment tree.
	Quick Find	Opens the **Quick Find** dialog. For more information, see "Quick Find dialog" on page 137.

Copyright © 1998-2015 No Magic, Inc.

Button	ToolTip text	Description
	Options	Click to open the Options menu and then select: • **Show Applied Stereotypes** - shows or hides the applied stereotypes next to the element name in the Containment tree. For more information about working with stereotypes, see "Stereotype" on page 985. • **Show Only Hierarchies** - at the top of the hierarchy shows the parent element. • **Invert Tree** - Inverts the tree hierarchy - shows the child element at the top of the hierarchy.

The same actions you may perform using the shortcut menu of the Inheritance tree.

Working with model elements in the Inheritance tree

You can work with elements in the Inheritance tree using the commands available from the selected element's shortcut menu. The shortcut menu contains the following commands:

Command	Function
New Specific Classifier	Creates a new element that is connected with the current element with the generalization relationship.
Select in Containment Tree	Opens the Containment tab and selects the current element in the Containment tree.

TIP! For more information about commands available from the shortcut menu in the Model Browser, see "Working with model elements in the Model Browser" on page 91.

Related concepts

Model Browser

Diagrams tab

The Diagrams tab in the Model Browser represents diagrams created in your project.

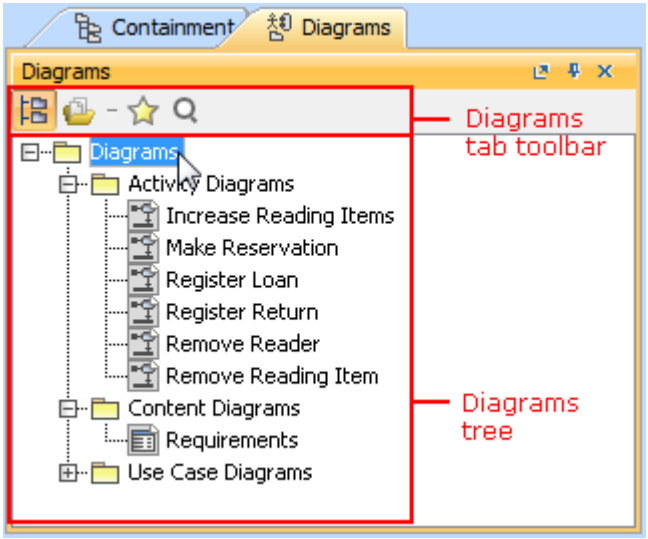


Figure 38 -- Diagrams tab

Just diagrams grouped by types that are used in your project are listed in the **Diagrams** tab in the Model Browser, not all diagram types as it was in earlier versions.

Copyright © 1998-2015 No Magic, Inc.

To open the Diagrams tab

Do one of the following:

- At the top of the Model Browser, click the **Diagrams** tab.
- If the Diagrams tab is hidden, from the **Window** menu, select **Diagrams**.



You can change some Diagrams tab properties in the **Environment Options** dialog the **Browser** options group. For more information about using the **Environment Options** dialog, see "Customizing Environment Options" on page 96.

The Diagrams tab consists of the following two components (see the figure above):

- Diagrams tab toolbar
- Diagrams tree



For more information about working with the Diagrams tree, see "Working with model elements in the Diagrams tree" on page 85.

Diagrams tab toolbar

The Diagrams tab toolbars contains the following button:

Button	ToolTip text	Description
	Group by Diagram Type	Groups diagrams into packages by diagram types.
	Group by Diagram Name	Groups diagrams into packages by the first symbol of their name, when the model contains more than 99 diagrams.
	Favorites	Opens the Favorites menu for choosing to do one of the following: • Add a selected element to favorites (see "To quickly add an element to favorites" on page 417). • Open the dialog for managing favorites (see "Manage Favorites dialog" on page 419). • Navigate to a desired favorite in the Containment tree.
	Quick Find	Opens the **Quick Find** dialog. For more information, see "Quick Find dialog" on page 137.

The same action you may perform using the shortcut menu of the Diagrams tree.

Working with model elements in the Diagrams tree

You can work with diagrams in the Diagrams tree using the commands available from the selected diagram's shortcut menu. The shortcut menu contains the following commands:

Command	Function
Open	Opens the selected diagram.
Print	Opens the **Print** dialog. For more information, see "Printing" on page 264.
Group by Diagram Type	Groups diagrams into the packages according to the diagram type.
Select in Containment Tree	Opens the Containment tab and selects the current element in the Containment tree.



For more information about commands available from the shortcut menu in the Model Browser, see "Working with model elements in the Model Browser" on page 91.

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Model Browser

Model Extensions tab

The Model Extensions tab contains all Stereotypes that are predefined and created manually in the project.

In this tree, you can create, review, copy/paste, and delete extension mechanisms.

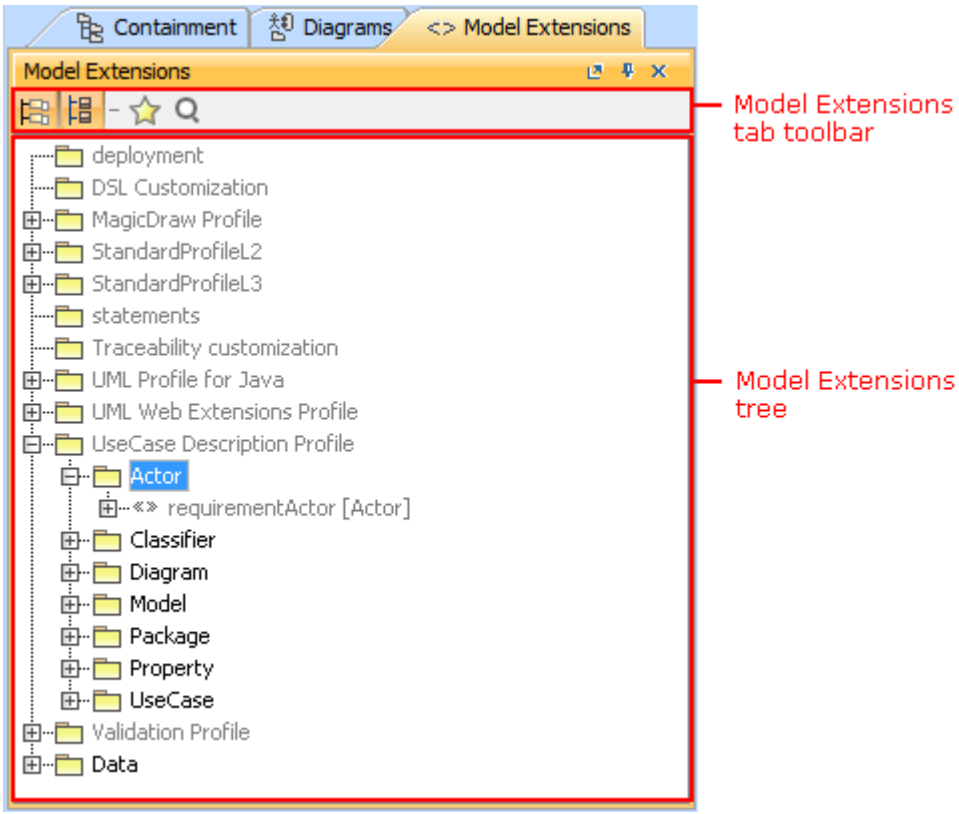


Figure 39 -- Model Extensions tab

To open the Model Extensions tab

Do one of the following:

- At the top of the Model Browser, click the **Model Extensions** tab.
- If the Model Extensions tab is hidden, from the **Window** menu, select **Model Extensions**.

TIP!

You can change some Model Extensions tab properties in the **Environment Options** dialog the **Browser** options group. For more information about using the **Environment Options** dialog, see "Customizing Environment Options" on page 96.

In the Model Extensions tab consists of the following two components (see the figure above):

- Model Extensions tab toolbar
- Model Extensions tree

TIP!

For more information about working with the Model Extensions tree, see "Working with model elements in the Model Extensions tree" on page 84.

Copyright © 1998-2015 No Magic, Inc.

Model Extensions tab toolbar

The **Model Extensions** tab toolbars contains the following buttons:

Button	ToolTip text	Description
	Group by Profiles	Groups extensions by the profiles. For more information about the profiles, see "Profile" on page 972.
	Group by Metaclasses	Groups extensions by the metaclasses.
	Favorites	Opens the Favorites menu for choosing to do one of the following: • Add a selected element to favorites (see "To quickly add an element to favorites" on page 417). • Open the dialog for managing favorites (see "Manage Favorites dialog" on page 419). • Navigate to a desired favorite in the Containment tree.
	Quick Find	Opens the **Quick Find** dialog. For more information, see "Quick Find dialog" on page 137.

The same actions you may perform using the shortcut menu of the Model Extensions tree.

Working with model elements in the Model Extensions tree

You can work with elements in the Model Extension tree using the commands available from the selected element's shortcut menu.



For more information about commands available from the shortcut menu in the Model Browser, see "Working with model elements in the Model Browser" on page 91.

Related concepts

Model Browser

Search Results tab

The **Search Results** tab is used to display results of a search. This tab opens after the search has been performed.

Copyright © 1998-2015 No Magic, Inc.

The **Search Results** tab consists of two components:

- Search Results tab toolbar
- Search Results tree

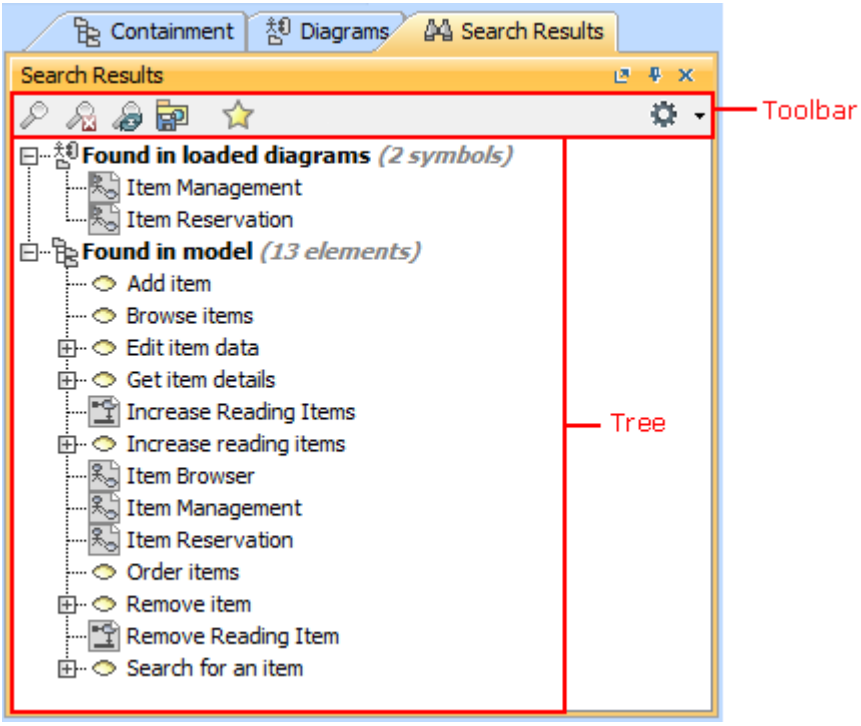


Figure 40 -- Search Results tab

Search Results tab toolbar

The Search Results tab toolbars contains the following buttons:

Button	ToolTip text	Description
	Find	Opens the **Find** dialog.
	Clear Results	Clears the results of the previous search.
	Rerun Search	Reruns a search with the already specified search criteria. Search results are updated according to changes done in the model.
	Save Search Results	Saves as a smart package either the search options or the search results. For more information, see "To create a smart package from the search results" on page 403.
	Favorites	Opens the Favorites menu for choosing to do one of the following: • Add a selected element to favorites (see "To quickly add an element to favorites" on page 417). • Open the dialog for managing favorites (see "Manage Favorites dialog" on page 419). • Navigate to a desired favorite in the Containment tree.

Copyright © 1998-2015 No Magic, Inc.

Button	ToolTip text	Description
	Options	Click to open the Options menu and then select: • **Show Applied Stereotypes** - shows or hides the applied stereotypes next to the element name in the Containment tree. For more information about working with stereotypes, see "Stereotype" on page 985. • **Show Full Types** - shows or hides the full types of typed elements (for example, operations, attributes, relationships, and other) in the Containment tree.



The same actions you may perform using the shortcut menu of the Search Results tree. For more information about commands available from the shortcut menu in the Model Browser, see "Working with model elements in the Model Browser" on page 91.

Working with model elements in the Search Results tree

The search results in the **Search Results** tree are displayed in the following two packages:

- **Found in loaded diagrams** - symbols that have been found in the diagram(s) are displayed in this package.
- **Found in model** - elements that have been found in the model data are displayed in this package.

Related concepts

Model Browser

Related procedures

Searching

Find TODO

Related references

Find dialog

Lock View tab



The Lock View tab is displayed only when working with the MagicDraw Teamwork Server.

The **Lock View** tab is used to display what was locked in the project (locked elements, project structure, symbol styles) and by which users. Next to the locked element, you can see the name of the user who has

Copyright © 1998-2015 No Magic, Inc.

locked it. Using the Filter by user list in the Lock View tab toolbar, you can see the elements, locked by the particular user.

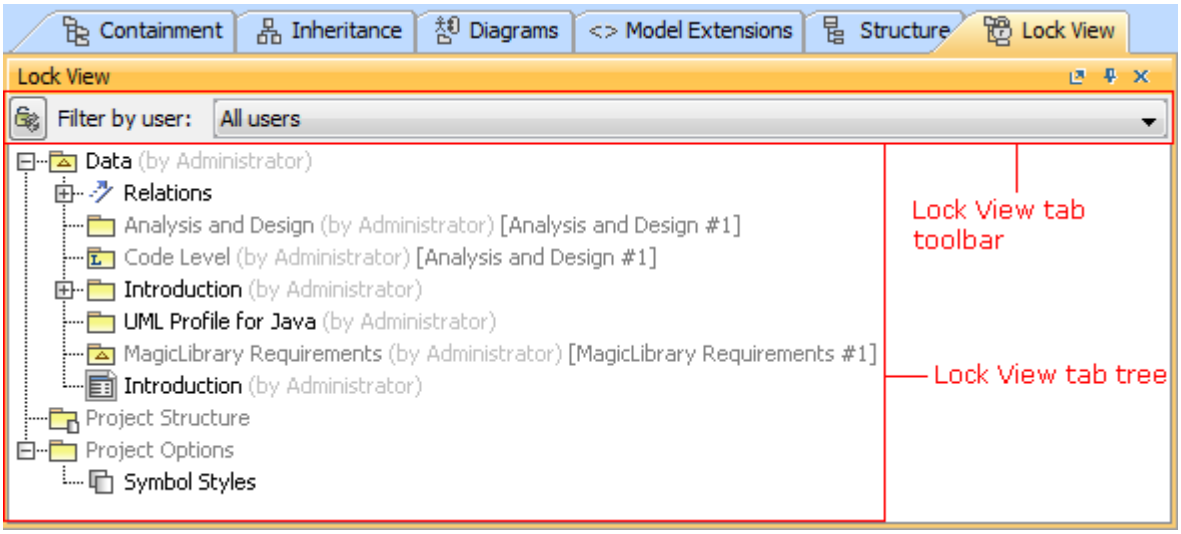


Figure 41 -- Lock View tab

To open the Lock View tab

Do one of the following:

- At the top of the Model Browser, click the **Lock View** tab.
- If the Containment tab is hidden, from the **Window** menu, select **Lock View**.

The Locked Elements tab consists of the following two components (see the figure above):

- Lock View tab toolbar
- Lock View tree



For more information about working with the Lock View tree, see "Working with model elements in the Lock View tree" on page 91.

Lock View tab toolbar

The Lock View tab toolbar contains the following:

- Update Lock Information button

Copyright © 1998-2015 No Magic, Inc.

- **Filter by user** list

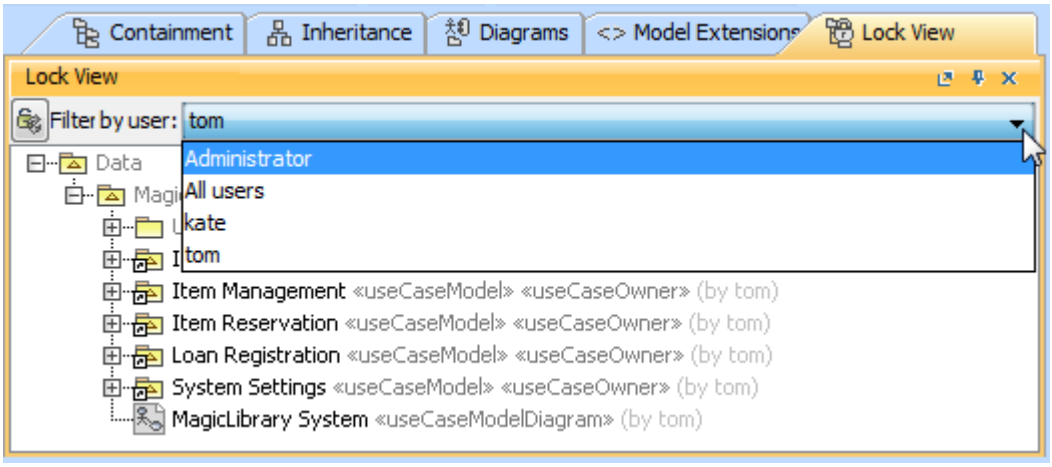


Figure 42 -- Lock View tab, Filter by user list

Using the **Filter by user** list you can select to display only the elements locked by the particular user.

Working with model elements in the Lock View tree

You can work with elements in the Lock View tree using the commands available from the selected element's shortcut menu. The shortcut menu contains the following commands:.

Command	Function
Lock	Locks or unlocks the particular element for edit.
Select in Containment Tree	Opens the Containment tab and selects the current element in the Containment tree.
Select in Inheritance Tree	Opens the Inheritance tab and selects the current element in the Inheritance tree.

TIP! For more information about commands available from the shortcut menu in the Model Browser, see "Working with model elements in the Model Browser" on page 91.

Related concepts

Model Browser

Related procedures

Locking Model Elements and Diagrams

Working with model elements in the Model Browser

In the Model Browser you can do the various actions with model elements, such as, rename, cut, copy, delete, and others. These actions are available from the particular model element shortcut menu. See the list of the concrete actions in the following table.

Command	Function
Rename	Switches the element name into the edit mode. You can also press F2 for this.
Copy	Copies the selected element.
Copy URL	Copies a model element URL to a clipboard. For more information, see "Copying/Opening Element URLs" on page 429.
Paste	Pastes the copied element.

Copyright © 1998-2015 No Magic, Inc.

Command	Function
Cut	Cuts the selected element.
Delete	Deletes the selected element.

Using model browser you can perform the following procedures:

- To copy/cut and paste the selected model element between different projects
- To change the location of an element in the Model Browser

To copy/cut and paste the selected model element between different projects

1. Select the model element in the Model Browser.
2. From the model element shortcut menu, select **Copy** or **Cut**.
3. Open the project to which you want to paste the selected element(s).
4. Select the container, where you wish to paste the element(s).
5. From the container's shortcut menu, select **Paste**.

To change the location of an element in the Model Browser

1. Select the element in the Model Browser.
2. Drag it to the new location (possible owner).

Related concepts

Model Browser

Related procedures

Copying Text or Images to Diagrams
Dragging, Copying, Cutting, and Pasting

Related references

Selection and Multiple Selections

Copyright © 1998-2015 No Magic, Inc.

Zoom tab

The Zoom tab enables you to preview any selected diagram. To select a diagram go to Model Browser > Containment / Diagram tab. You can also use the Zoom tab for zooming in or out the active diagram.

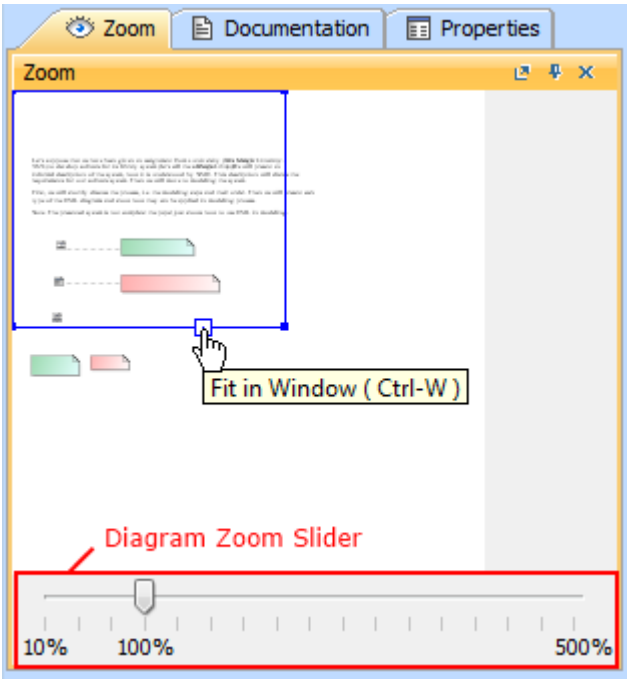


Figure 43 -- Zoom tab

To open the Zoom tab

Do one of the following:

- In the Model Browser, click the **Zoom** tab.
- On the **Window** menu, click **Zoom**.

To zoom in or out the active diagram

- In the Zoom tab, drag corners of the blue square.

To access quickly any part of the diagram

- In the Zoom tab, drag the blue square to the desired part of the diagram. The desired diagram part will be displayed on the diagram pane.

To fit the diagram to the window

- In the Zoom tab, on the blue square, click the **Fit in Window** button or press the CTRL+W shortcut keys.

To show the diagram zoom slider

1. From the **Options** menu, select **Environment**. The Environment Options window opens.
2. Select the Browser options group.
3. In the Browser category, select the **Show Diagram Zoom Slider** check box. The diagram zoom slider is displayed in the Zoom tab (see the figure above).

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Model Browser

Documentation tab

The Documentation tab shows the information associated with the selected model element in the Model Browser or on a diagram pane.

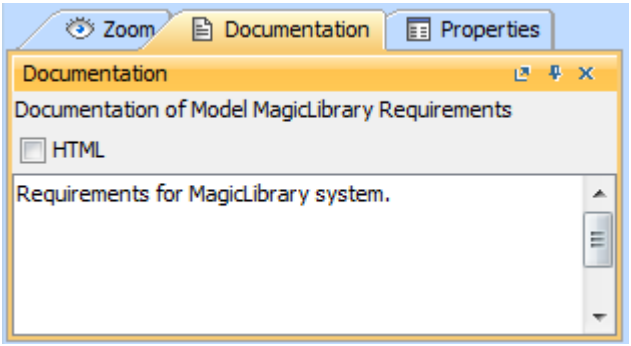


Figure 44 -- Documentation tab

To open the Documentation tab

Do one of the following:

- In the Model Browser, click on the **Documentation** tab.
- On the **Window** menu, click **Documentation**.

NOTE

If there is no documentation for the selected element, the text area is empty.

To write documentation for the selected model element

- In the **Documentation** tab, click on the text area and type the text.

To turn on the HTML text

- In the Documentation tab, click the **HTML** check box. The HTML toolbar opens and you can type the HTML text. For more information about using the HTML text, see "HTML Editor" on page 422.

Related concepts

Model Browser

Copyright © 1998-2015 No Magic, Inc.

Properties tab

The Properties tab at the bottom of the Model Browser allows you to quickly access the basic information about the selected element or diagram.

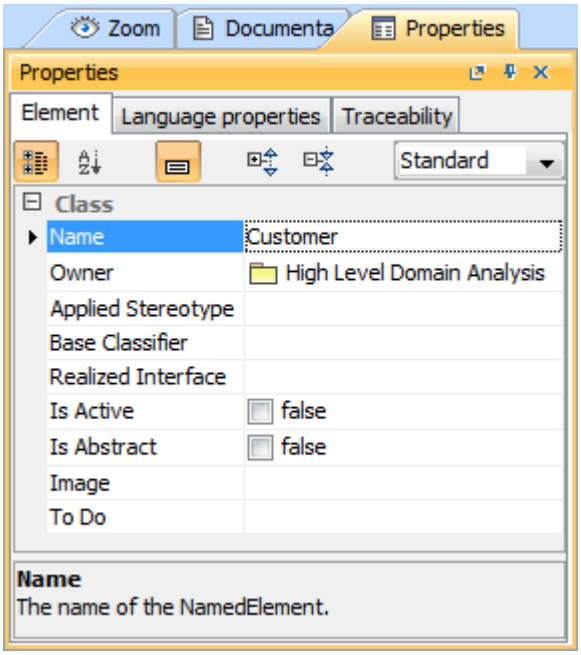


Figure 45 -- Properties tab

To open the Properties tab

Do one of the following:

- In the Model Browser, click the **Properties** tab.
- On the **Window** menu, click **Properties**.

The **Properties** tab includes the following tabs:

Tab name	Description
Element	Contains the main properties of the element's or diagram's specification. For more information about editing values of different property types, refer to "Editing Property Values" on page 298.
Symbol	Contains the element symbol's properties. **NOTE:** This tab is available for symbols only. For more information, see "Formatting Symbols" on page 317.
Language properties	Contains the same data as the **Language Properties** property group in the element's Specification window. **NOTE:** This tab is available only for the elements, which are used for code generation, that is, class, attribute, operation. For more information, see MagicDraw CodeEngineering UserGuide.pdf.
Traceability	Contains the same data as the **Traceability** property group in the element's Specification window. **TIP!** For more information about traceability, see "Traceability" on page 514.

Related concepts

Model Browser

Copyright © 1998-2015 No Magic, Inc.

Customizing Environment Options

You can customize the application environment according to your preferences via the **Environment Options** dialog.

To open the **Environment Options** dialog

- From the **Options** menu, select **Environment**.

The **Environment Options** dialog contains various project-independent options grouped by different features (e.g., diagrams, Model Browser, code engineering). Each option group is available in a different tab. Tabs are displayed in the tab tree.

An option value can be simply changed by typing a new value, setting a value to true / false, or selecting a value from the list.

Learn more about the **Environment Options** dialog in the following sections:

- "Using Environment Options dialog" on page page 97.
- "Common elements in Environment Options dialog" on page page 97.

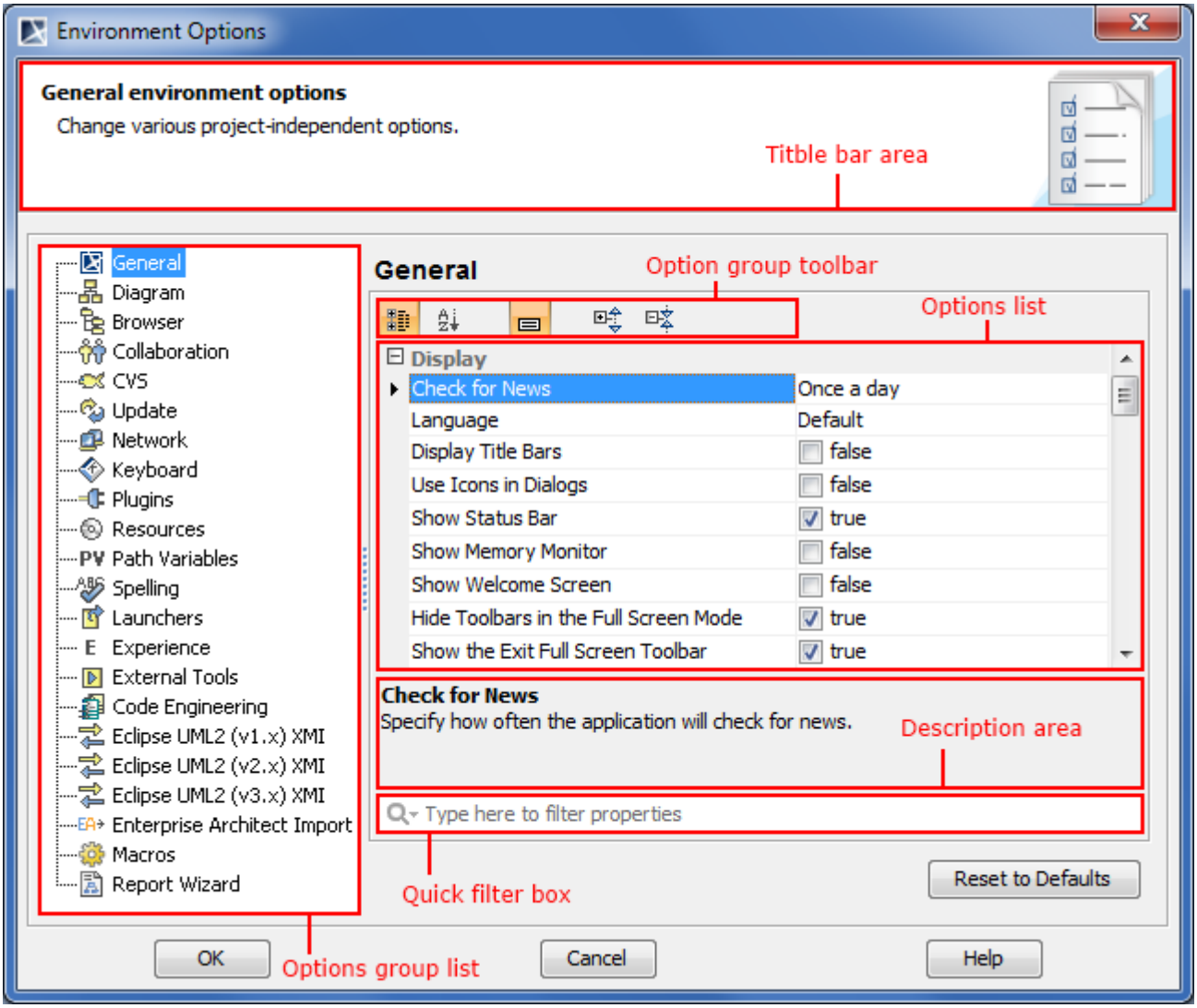


Figure 46 -- Structure of Environment Options dialog

Copyright © 1998-2015 No Magic, Inc.

Using Environment Options dialog

In order to change a desired environment option, first of all you may need to find it. It can be rather difficult to find a desired option, if the tab's option list contains 10 or more options. In this case the Quick filter appears in the dialog. Using the Quick filter you can quickly find the desired option in the list. For more information about the Quick filter please refer to "Quick Filter" on page 279.

For better understanding an option you can read its description that tells what is the effect of changing the option value.

To read the option description

IMPORTANT!

Make sure that the Show Description mode is turned on in the **Environment Options** dialog. To turn the Show Description mode on or off, click the Show Description button on the tab toolbar.



1. Click an option, whose value you want to change.
2. Read the option description in the area below the tab options list. You are ready now to change the option value.

Common elements in Environment Options dialog

For the common element descriptions look in the following table.

Element	Element Type	Description
Quick filter	Text box	Type an option name or its fragment. For more information about the Quick filter box please refer to "Quick Filter" on page 279.
Reset to Defaults	Button	Resets all options to their default values.
OK	Button	Saves changes and closes the dialog.
Cancel	Button	Closes the dialog without saving changes.
Help	Button	Opens MagicDraw Help.

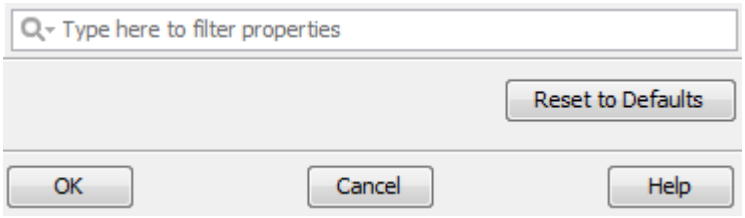


Figure 47 -- Fragment of Environment Options dialog. Quick filter and common buttons

Improving Performance

When you work with very large models or use a lot of diagrams at a time, the performance of MagicDraw may become slow. To increase an efficiency of modeling, we suggest the following solutions:

Copyright © 1998-2015 No Magic, Inc.

- **Increase memory allocation (java heap) size**. See the section "Memory Allocation" on page 100.
- **Do not keep unused diagrams open**. Perform the procedure "To open project without loading diagrams" described bellow this list. Your projects will be opened over a shorter period of time without opening a diagram as well as use less memory.
- **Increase an active validation period**. Perform the procedure "To increase an active validation period" described bellow this list. Reduced active validations using takes less memory.
- **Split the project to read only modules**. Keep read only modules not loaded. This may help only if your project contains several parts with minimal dependencies between them. For more information about working with partially loaded projects, see "MOF Support" on page 178. You can also find the “Project Decomposition Description” sample in <MagicDraw installation directory>\samples\product features\ project decomposition.
- **Use Garbage Collector to free unused memory**. See the procedure "To free unused memory" on page 100.

To open project without loading diagrams

1. On the **Option** menu, click **Environment**. The **Environment Options** dialog opens.
2. In the **General** option list, expand the **Save / Load** options group.
3. To the **Diagrams Lode Mode** option, assign the **Do not load diagrams** value.
4. Restart MagicDraw.

To increase an active validation period

1. On the **Option** menu, click **Environment**. The **Environment Options** dialog opens.
2. In the **General** option list, expand the **Active Validation** options group.
3. Increase the **Active Validation Period (seconds)** value.



Be aware about limits of the **Active Validation Period** size, as a long period may be an alternative of switching an active validation off.

4. Restart MagicDraw.

Related references

Background Tasks Manager

Memory Monitor

Background Tasks Manager

Low-priority or routine tasks are moved to separate threads and are performed as background tasks. Such tasks do not stop MagicDraw running, and you may work with your model as usual. Background tasks are displayed in the Background Task Manager appearing at the bottom right corner of the MagicDraw status bar. As of the MagicDraw version 17.0.2, the searching process is recomposed as a background task. While a

Copyright © 1998-2015 No Magic, Inc.

search process is running, the searching progress bar is displayed in the Background Task Manager. See an example in the following figure.

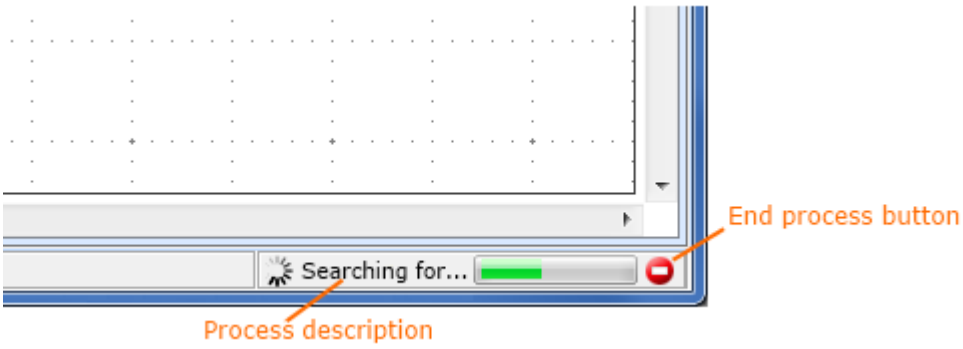


Figure 48 -- Task manager with searching process

You can stop a background task while the process is running.

To end a process

- Click the End process button to cancel the selected process. If the last or only one existing process is canceled, Background Task Manager is turned off.

Related procedures

Searching

Memory Monitor

In order to monitor the memory used by MagicDraw while working with larger projects, you can turn the **Memory Monitor** on.

Memory Monitor shows two values: currently used memory and a current java heap size. While working with MagicDraw, the java heap size increases until it reaches a limit.

The Garbage Collector is designed to find and free unused memory.

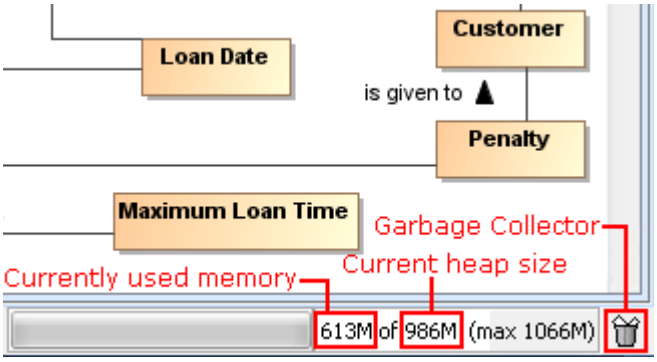


Figure 49 -- Memory monitor bar

Typically, when working with a program, used memory grows up because actions are stored in the undo list, opened diagrams are not unloaded.The Memory Monitor bar becomes red when used memory takes more than

Copyright © 1998-2015 No Magic, Inc.

85% of the total heap. Red bar shows that MagicDraw may run out of memory soon. Even if all heap size is used, the Garbage Collector may recover enough memory to save a project, but this may take few minutes.

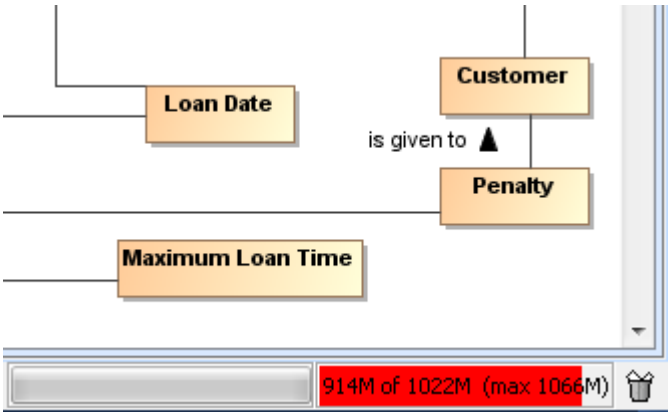


Figure 50 -- Memory Monitor with increased memory usage

When the total heap size is exceeded, you can get an out of memory error message. You can increase the heap size directly in this message. Changes will be applied after you restart MagicDraw. For more information, see "Memory Allocation" on page 100.

To turn Memory Monitor on

- On the **View** menu, select **Status Line** > **Show Memory Monitor**. The **Memory Monitor** bar appears in the right bottom corner of the MagicDraw window.

To free unused memory

- Click the Garbage Collector several times and wait few seconds:

  1. If the Memory Monitor bar is still red, save the project and restart MagicDraw and reload the project.
  2. If the Memory Monitor bar is red after reloading the project, the maximum heap size should be increased.

IMPORTANT!

When almost all heap memory is used, the Garbage Collector starts to free unused memory after each action automatically. It slows down the program as the most CPU power is used for the Garbage Collector.

Memory Allocation

If the OutOfMemory error occurs when running the program, the cause is that you haven't got enough memory available for the workings of the application. You can change the amount of allocated memory (java heap size and/or the MaxPermSize) directly in the OutOfMemory dialog or see "Changing amount of allocated memory"

Copyright © 1998-2015 No Magic, Inc.

on page 102.

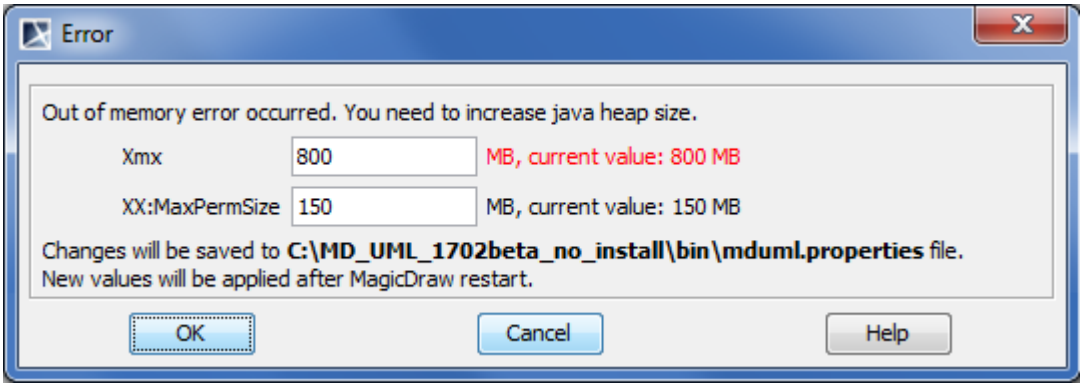


Figure 51 -- Out of memory error message

NOTE

The “memory allocation for the product” concept means the same as “java heap size”. That is you can increase the memory allocation for the product by increasing java heap size.

Related concepts

Improving Performance

Related references

Recommended amount of allocated memory

Changing amount of allocated memory

Memory Monitor

Recommended amount of allocated memory

The amount of allocated memory (or java maximum heap size) should be set to less than the available physical RAM on the machine, that is the RAM minus the amount of memory taken by any other processes that will run concurrently. Otherwise your JVM process will likely swap and that will slow down the application. If the heap size is increased too much, you will get the “Could not create the Java virtual machine” error message and MagicDraw will not start.

- The maximum theoretical heap limit for the 32-bit JVM is 4G. Due to various additional constraints such as available swap, kernel address space usage, memory fragmentation, and VM overhead, in practice the limit can be much lower. On most modern 32-bit Windows systems the maximum heap size will range from 1.4G to 1.6G. On 32-bit Solaris kernels the address space is limited to 2G. On 64-bit operating systems running the 32-bit VM, the max heap size can be higher, approaching 4G on many Solaris systems. For more information, see JAVA HotSpot VM FAQ.
- On 64-bit VMs, you have 64 bits of addressability to work with resulting in a maximum Java heap size limited only by the amount of physical memory and swap space your system provides. For more information, see JAVA HotSpot VM FAQ.

Related concepts

Memory Allocation

Related references

Changing amount of allocated memory

Memory Monitor

Copyright © 1998-2015 No Magic, Inc.

Changing amount of allocated memory

You can set the memory allocation for MagicDraw in one of the following ways:

- In the MagicDraw installation wizard
- In the **Environment Options** dialog
- In the magicdraw.properties file

Changing amount of allocated memory in the MagicDraw installation wizard

The right amount of allocated memory (or java heap size) for MagicDraw, is automatically set according to the physical memory (RAM) size of your computer during MagicDraw installation.

To automatically set the amount of allocated memory during MagicDraw installation

1. Double-click the installer file, for example, MD_UML_<version number>_win.exe.
2. In the Setup Wizard, in the **Memory Allocation** step, leave the **Automatic** option button selected and click the **Next** button. The amount of allocated memory is set automatically.



- By default, in MagicDraw the default amount of allocated memory is 30% of total physical memory (RAM) size.
- The automatic memory detection on Mac OSX, Linux or when using no_install packages is performed on the first MagicDraw startup. For more information on how to set the amount of allocated memory, see "Changing amount of allocated memory in the Environment Options dialog" on page 102 and "Changing amount of allocated memory in the magicdraw.properties file" on page 102.

To manually set the amount of allocated memory during MagicDraw installation

1. Double-click the installer file, for example, MD_UML_<version number>_win.exe.
2. In the Setup Wizard, in the **Memory Allocation** step, click the **Custom** option button. Type the amount of allocated memory for the product.

Changing amount of allocated memory in the Environment Options dialog

To set amount of allocated memory in the Environment Options dialog

1. Open the **Environment Options** dialog.
2. In the **General** options group, under the **Memory Settings** category, change the values of the following properties: **Maximum Heap Size**, **Permanent Gen Max Size**, **Thread Stack Size**.

Changing amount of allocated memory in the magicdraw.properties file

To set amount of allocated memory in the magicdraw.properties file

1. Open the <MagicDraw installation directory>\bin\magicdraw.properties file for edit.
2. In the `JAVA_ARGS` line, increase the value next to `-Xmx, -XX:MaxPermSize or -Xss`. For example, change the `-Xmx800M` value to `-Xmx1066M`.

Copyright © 1998-2015 No Magic, Inc.

3. Save the file.



If MagicDraw is not started yet, in the magicdraw.properties file, there are the following two lines:

```
DEFAULT_MEMORY_SETTINGS_64=-Xmx[30%,1200,4000]M -XX\:PermSize\=60M -XX\:MaxPermSize\=200M -Xss1024K
DEFAULT_MEMORY_SETTINGS_32=-Xmx[30%,800,1400]M -XX\:PermSize\=40M -XX\:MaxPermSize\=150M -Xss640K
```

Here you can specify the relative, minimum, and maximum amount of the allocated memory, for example, `-Xmx[30%,1200,4000]M`

Both lines are removed from the magicdraw.properties file on the first MagicDraw start up.

Related concepts

Memory Allocation

Related references

Recommended amount of allocated memory

Memory Monitor

Look and Feel: Controlling the Interface

The appearance of MagicDraw windows, dialog boxes, menus, and everything inside them can be changed. The **Look and Feel** submenu allows you to personalize the user interface of the MagicDraw. You can set your favorite colors and fonts.

To make changes to the interface

- From the **Options** menu, choose **Look and Feel** and then choose the style you wish to apply.

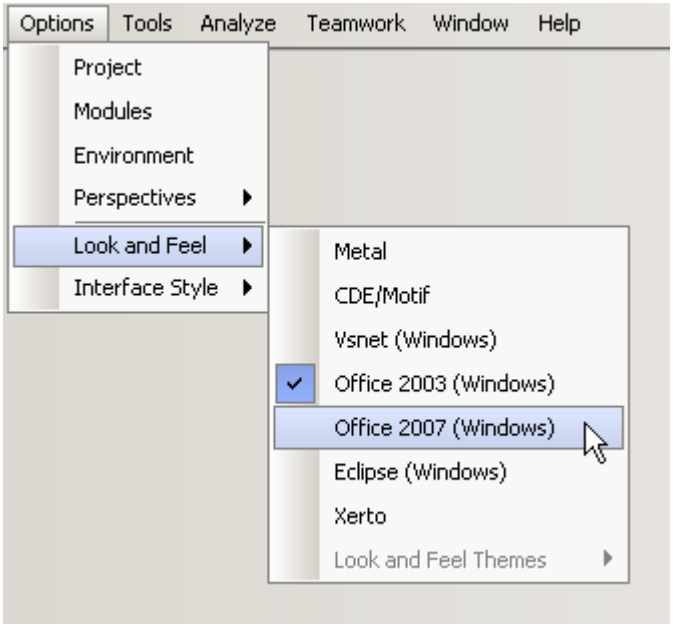


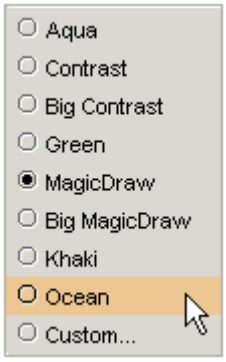
Figure 52 -- Interface styles for MagicDraw

The chosen style will not look exactly like the applications in those operating systems because every style of graphical interface is implemented within a Swing library, but it will look quite similar.

Copyright © 1998-2015 No Magic, Inc.

Depending on the operating system you use, some choices might be unavailable for you. For example, Windows9x/NT users may not switch to the Mac interface style.

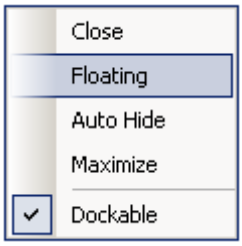
Note that themes listed in **Look and Feel Themes** are valid only for the **Metal** style. You can choose any of the following themes:



After choosing the Custom theme, the **Properties** dialog for setting your own options will be opened.

To make the Model Browser a separate window

- From the Model Browser shortcut menu, select **Floating** and move the window to any desired position.





If the **Dockable** check box is selected, the floating tab window will appear in a fixed edge position after trying to move it outside the MagicDraw window borders.

Assigning Shortcut Keys

To assign or change a command shortcut key

1. From the **Options** menu, select **Environment**.
2. The **Environment Options** dialog opens.
3. Select the **Keyboard** tab and assign the desired shortcut keys in the right pane of the dialog.

Copyright © 1998-2015 No Magic, Inc.

4 WORKING WITH PROJECTS

Concepts

For better understanding the further material, first of all read the following descriptions.

Project

A physical working unit that consists of model, model visualizations (diagrams, tables, matrices, etc.), configuration data, and references to other elements residing in used projects.

Used project
(up to 18.0 – Module)

A project having one or more shared packages. Used projects are created for the purpose of decomposing projects into parts in order to manage each part separately and/or reuse those parts in other projects.



Used project is a role that one project plays in the composition of another project. It is NOT a special kind of project. Each used project can be opened and modified separately.

In this chapter, you will find the following sections:

1. "Creating Projects" on page 105
2. "Saving Projects" on page 113
3. "Opening Projects" on page 115
4. "Importing Data from other UML Tools" on page 116
5. "Exporting Projects" on page 117
6. "Creating and Using Project Templates" on page 119
7. "NEW! Projects with File Attachments" on page 124
8. "Setting Project Options" on page 129
9. "Searching" on page 131
10. "Filtering" on page 143
11. "Project Partitioning" on page 147
12. "MOF Support" on page 178
13. "Ecore Support" on page 180
14. "Working with Standard Profiles" on page 196

Creating Projects



MagicDraw Basics

Creating a new project

All project information is stored in a single file. A project name matches the file name where the project is saved.

The newly created project consists of the following packages:

- The root package *Data* is empty and stores all model elements.

Copyright © 1998-2015 No Magic, Inc.

- **File View** package contains components that are created during code engineering and represent source files. Adding a resident element to a particular component causes that element to be generated within the source file.
- **UML Standard Profile** contains stereotypes that are necessary for working with MagicDraw, primitive data types and constraints (which are UML standard), and UML 2 metamodel elements. The following data types are specified in MagicDraw: boolean, byte, char, date, double, float, int, Integer, real long, short, void, and string.

You can also create your own packages for holding the model elements. By default, packages cannot be deleted or renamed in a project (except for the File View package).

To start a new project, you must create a new workspace for it.

You can select a project type from the following domains:

- **General-Purpose Modeling** (UML, Use Case, Guide to UML Diagrams projects, Project from Existing Source Code)
- **System Engineering** (SysML project)
- **Enterprise Modeling** (DoDAF, DoDAF 2.0, MODAF projects)
- **Business Process Modeling** (BPMN 2.0 project)
- **Service-Oriented Modeling** (Cameo SOA+ project)
- **Other** (Project from Template, Process Guide project)

On a new project creation the **General-Purpose Modeling** domain opens by default.

To create a new workspace for a blank project

1. Do one of the following:
   - From the **File** menu, select **New Project**.
   - On the main toolbar, click the **New Project** button.
   - Press Ctrl+Shift+N.

Copyright © 1998-2015 No Magic, Inc.

In all cases, the **New Project** dialog opens.

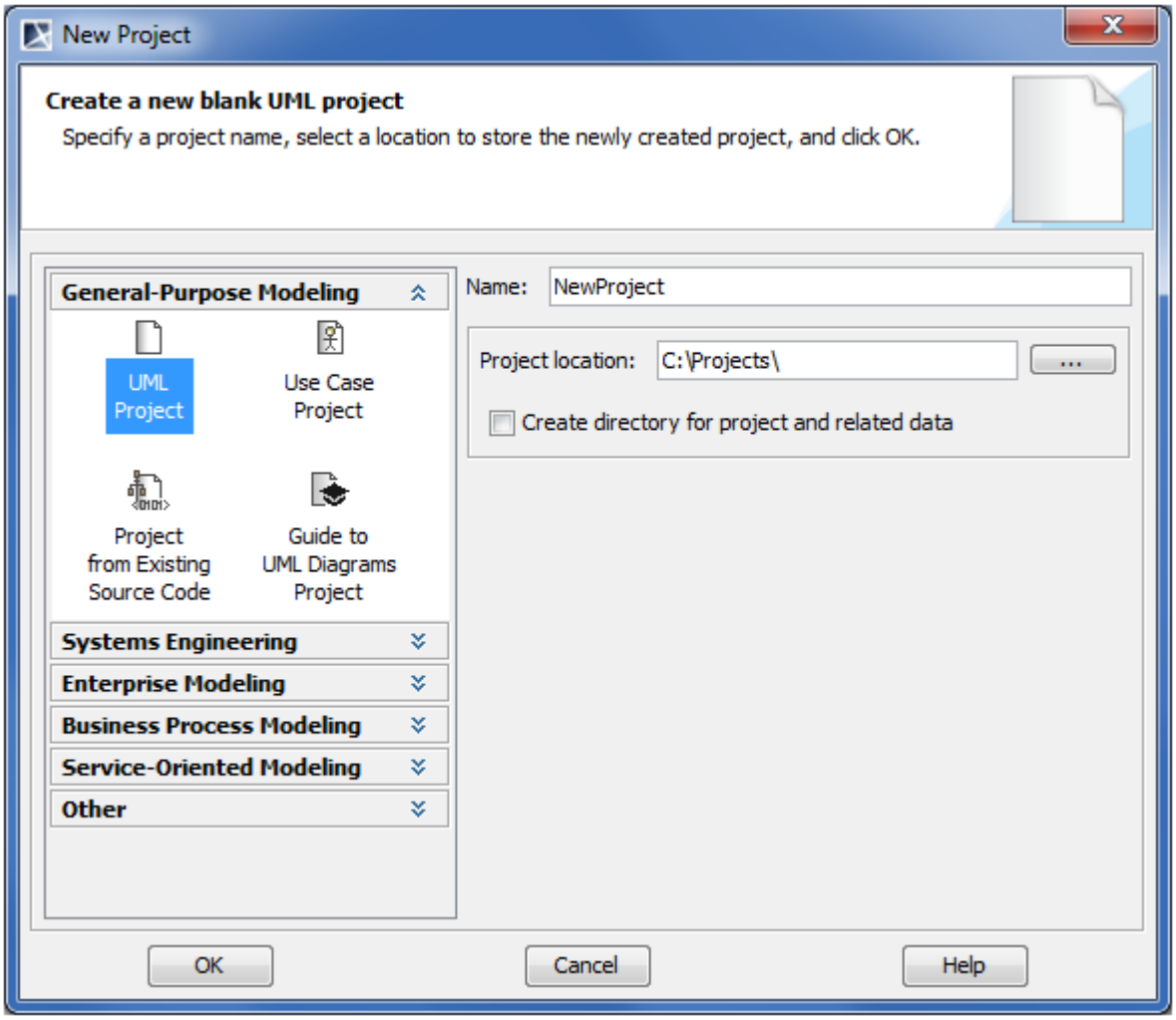


Figure 53 -- New Project dialog

2. Select the **UML Project** icon in the **General-Purpose Modeling** domain.
3. Specify the file name in the **Name** box.
4. Click the **...** button to select the location to store a newly created project in your computer.
5. Click **OK**.

Working with multiple projects

Because you may need to manage several projects at the same time, MagicDraw allows you to work with several projects simultaneously.

All open projects are held in separate workspaces. Different active projects may exchange data. Entities from one project can be copied or moved to another.

To switch between open projects

Do one of the following:

- From **File** menu, select **Switch Projects** and then click the name of the project you want to switch to.
- On the main toolbars, click the Switch Projects button and then select the name of the project you want to switch to.

Copyright © 1998-2015 No Magic, Inc.

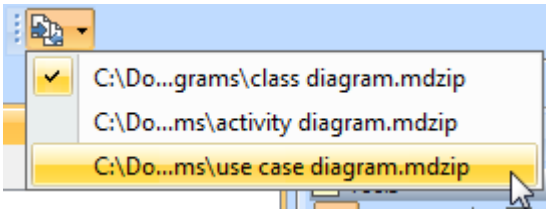


Figure 54 -- Using toolbar button to switch between open projects

To close all open projects

- Select **Close All Projects** from the **File** menu. The **Question** message box appears.

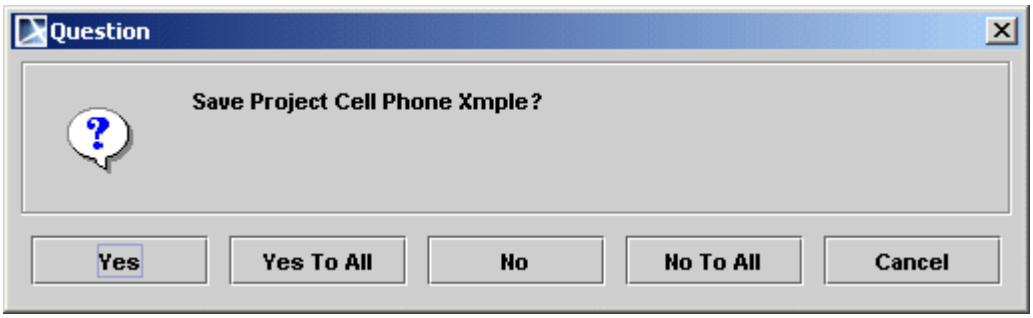


Figure 55 -- Question message

Choose the way your projects will be closed:

Yes	The project you are currently closing will be saved (its name appears in the question). The dialog is displayed again when the next project closes.
Yes To All	Save all projects without prompting. The **Save** dialog will not appear for each open project.
No	Project you are currently closing will not be saved. The dialog is displayed again when the next project closes.
No To All	All the projects will be closed without saving or further prompting.
Cancel	Cancel saving projects.

To exchange model elements between open projects

- Use the **Cut**, **Copy**, and **Paste** commands from the **Edit** menu or the selected element's shortcut menu, or the appropriate shortcut keys: Ctrl+X, Ctrl+C, Ctrl+V.

Creating a new project from the existing source code

To create a new project from existing source files

1. Do one of the following:
   - From the **File** menu, select **New Project**.
   - On the main toolbar, click the **New Project** button.
   - Press Ctrl+Shift+N.
     In all cases, the **New Project** dialog opens.
2. In the **General-Purpose Modeling** domain, select the **Project from Existing Source** icon.

Copyright © 1998-2015 No Magic, Inc.

3. Specify the file name in the **Name** text box.
4. Click the **...** button to select the location to store a newly created project in your computer.
5. Select a code engineering language from the list by clicking the **...** button and click **OK**.

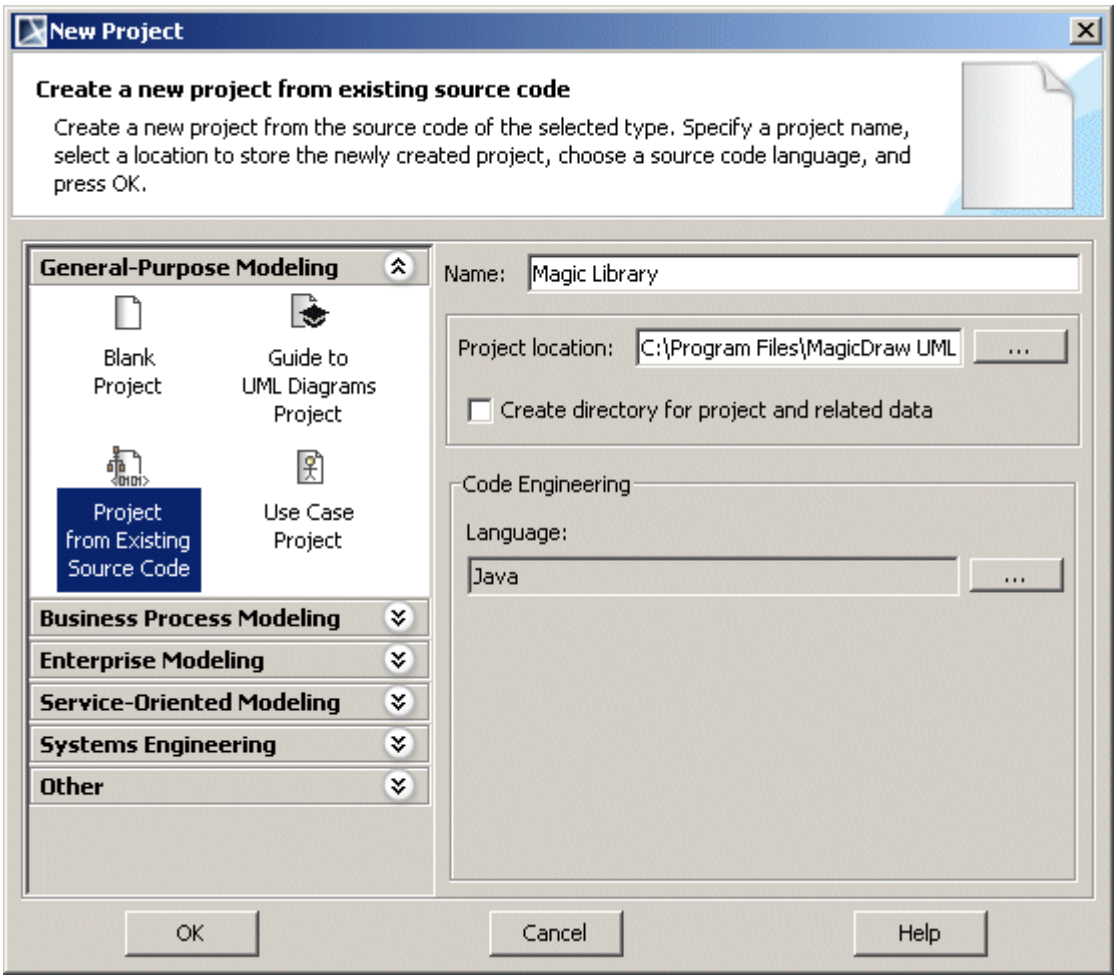


Figure 56 -- New Project dialog. Creating project from existing source

6. The **Round Trip Set** dialog opens. Add the source files to enable code engineering to reverse them into a newly created project.

Creating a new Use Case project

To create a new project from existing source files

1. Do one of the following:
   - From the **File** menu, select **New Project**.
   - On the main toolbar, click the **New Project** button.
   - Press Ctrl+Shift+N.
     In all cases, the **New Project** dialog opens.
2. In the **General-Purpose Modeling** domain, select the **Use Case Project** icon.
3. Specify the file name in the **Name** text box.

Copyright © 1998-2015 No Magic, Inc.

4. Click the **...** button to select the location to store a newly created project in your computer.

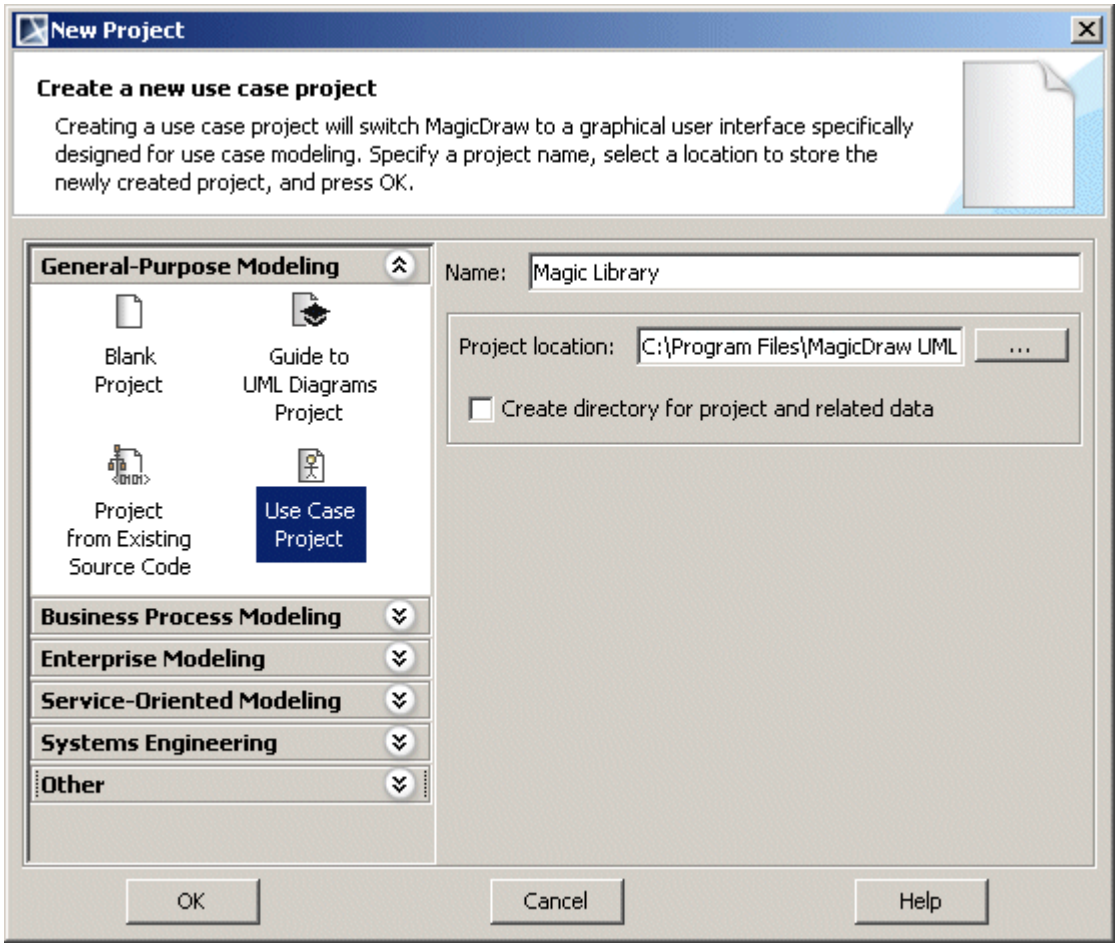


Figure 57 -- New Project dialog. Creating use case project

The newly created project will automatically load the **UseCase Description Profile**. Also *Actor*, *High-Level Use Case* and *System-Level Use Case* packages will be created under the root package *Data*. Additional properties will be displayed in the newly created use cases **Specification** window.

Working with Project Properties Dialog

The **Project Properties** dialog is designed to provide general information about a project. Using this dialog you can also add title, author, version, and description for projects.

To open the **Project Properties** dialog

1. Open the project.

Copyright © 1998-2015 No Magic, Inc.

2. From the **File** menu, select the **Project Properties** command.

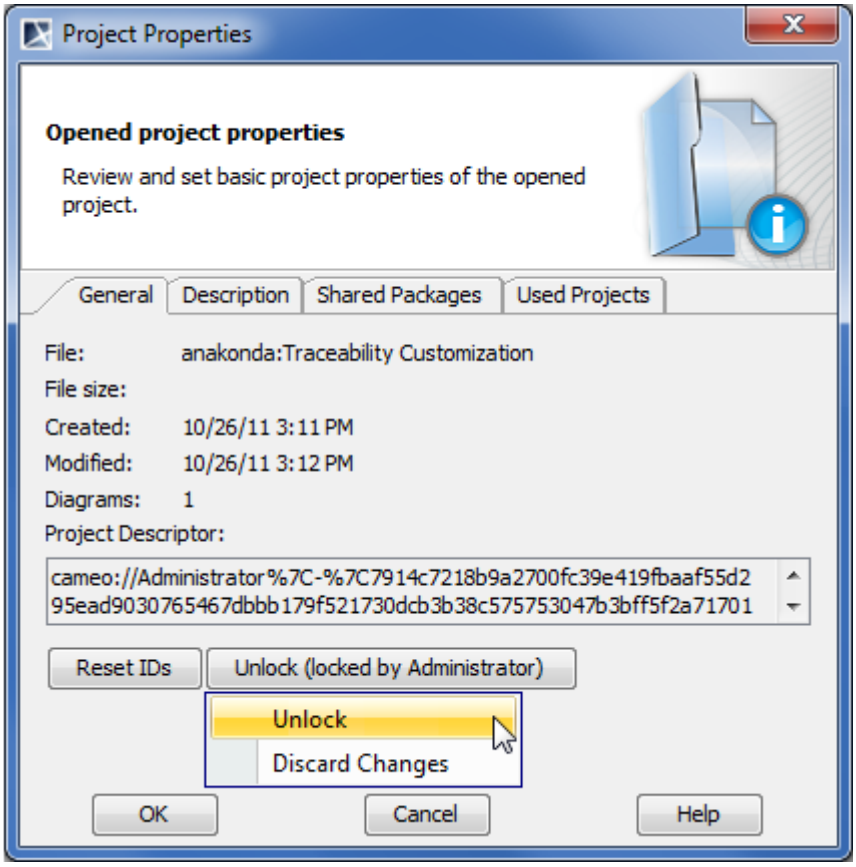


Figure 58 -- Project Properties dialog

The Project Properties dialog consists of the following tabs:

- General tab
- Description tab
- Shared Packages tab
- Used Projects tab

General tab

In the General tab, you can see the general information about a project.

Element Name	Description
File	The location of the file and the file name.
File Size	The size of the current project.
Created	The date the project was created.
Modified	The date of the last project modification.
Diagrams	The number of all project diagrams.
Project Descriptor	A unique project ID.

Copyright © 1998-2015 No Magic, Inc.

The **General** tab buttons are described in the following table:

Element Name	Description
Reset IDs	Click to reset all project element IDs. All project elements have their ID (a unique identificator). While working with several related projects, you can get a message about the duplicated element IDs. After clicking the **Reset IDs** button, new IDs will be created for all project elements. **NOTE:** Make sure the project, for which you are resetting IDs, is not used in other projects.
Unlock / Discard Changes / Lock Project Structure	Click the **Unlock** to unlock the project structure. This button is available, when the project structure is locked. Click the **Discard Changes** button to unlock the project structure discarding changes made after the last commitment. This button is available, when the project structure is locked. Click the **Lock Project Structure** button to lock the project structure. This button is available when the project structure is unlocked. For more information about locking, see Locking project structure on page 1060. **NOTE:** These buttons are available when working with server projects.

Description tab

In this tab, you can add a project name, author, and comment about that project.

Element Name	Description
Title	Type project name.
Author	Type the author.
Comment	Type information about the project.

Shared Packages tab

In this tab, you can see information about the project shared packages.

Element Name	Description
Shared Package	The list of shared packages that are included in the current project.
Preferred Path	The path to the shared package. Click the **...** button to select the package.
Project Description	The description of the open project.
Standard/System Profile	If you want to set the project as standard/system profile, select the check box.

Used Projects tab

In this tab, you can see the projects and standard/system profiles that are used in the open project.

Related references

Locking project structure

Copyright © 1998-2015 No Magic, Inc.

Saving Projects



The native MagicDraw format is *.mdzip and *.mdxml.
You can also save a project as *.xml and *.xml.zip.

To save changes for later sessions, revised projects must be saved. While saving, you can edit the name of the project and the file format.

To save the project

1. From the **File** menu, select the **Save Project** or **Save Project As** command. Alternatively, you can click the **Save** button on the main toolbar or press the shortcut keys CTRL+S. The **Save** dialog opens.
2. Select the destination directory (where you wish to save the project) and type the chosen file name.



The default project saving location is set according to your operating system:

- Windows: My Documents/
- Linux: /home/<username>/
- OS X: Users/Documents/

3. Select the format for saving a project:
   - **Packed MagicDraw File Format** (*.mdzip) (default). *.mdzip is a MagicDraw native format. It contains various parts of the project compressed. You should select it if you intend only to work with a project in the MagicDraw environment.
   - **MagicDraw File Format** (*.mdxml). You should choose a *.mdxml format if you intend to analyze the project's structure, xml, or for other purposes. You can even open the *.mdxml project with another UML tool.
   - **XML** (*.xml).*.xml is the same format as *.mdxml. The difference is that *.mdxml files can be opened with MagicDraw or other UML tool and *.xml files can be opened with an XML editor.

Copyright © 1998-2015 No Magic, Inc.

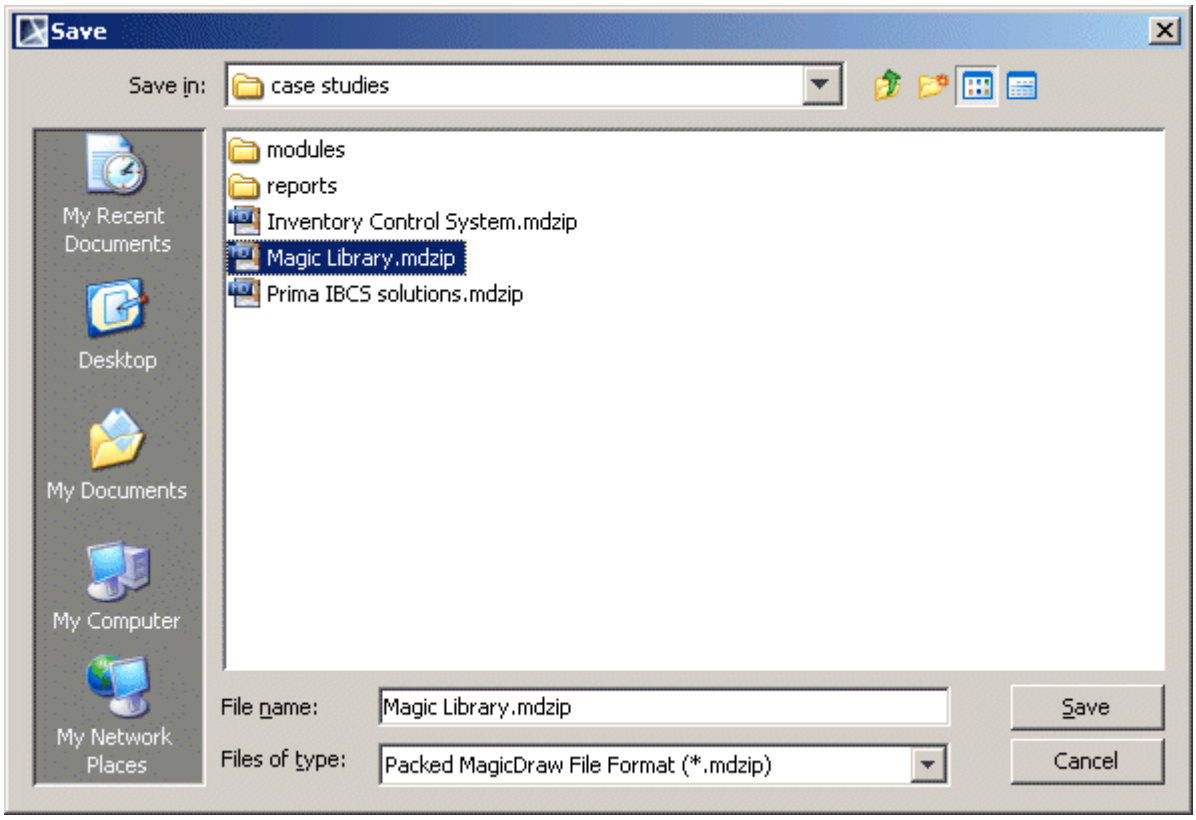


Figure 59 -- Save dialog

NOTE

- If the **Create Backup File** check box is selected in the **Environment Options** dialog, MagicDraw always creates a backup file of the previously saved project. The backup is held in a file with a name identical to that of the project. For a detailed description of the **Environment Options** dialog, see "Customizing Environment Options" on page 96.
- If you want to save maximum additional information to an xmi file (not required in loading to MagicDraw load, but may be useful when using other tools), select the **Rich XMI** check box in the **Environment Options** dialog.

Autosave

After you stop working with MagicDraw, an idle time passes and the current project is saved to a special file called the AutoRecovery file.

If the application is terminated normally, the AutoRecovery file is removed. If the application crashes, the AutoRecovery file is left. On startup, MagicDraw checks for an AutoRecovery file. If it exists, MagicDraw suggests loading the project from this file.

To save an AutoRecovery file of the open project(s) when a system is not in use

1. Open the **Environment Options** dialog.
2. In the **General** pane, select the **Save Project Recovery Data on Idle** check box**.** Enter the system idle time (in minutes) in the **Idle Time to Activate Recovery Save** text box. This is the length of time the system must be idle in order to activate an AutoRecovery save.

Copyright © 1998-2015 No Magic, Inc.

Opening Projects



The native MagicDraw format is *.mdxml or *.mdzip.



If you are about to open a project that has been saved with 17.0 or earlier MagicDraw version, it is highly recommended to read "Opening projects saved with MagicDraw 17.0 or earlier" on page 115 first.

To edit or review previously created projects, do one of the following

- From the **File** menu, select **Open Project**. In the **Open** dialog, select the project and click **Open**.
- On the main toolbar, click the **Open Project** button.
- Drag the project from the open window of your file system to MagicDraw. The project starts immediately.
- Double-click a project file with the *.mdxml or *.mdzip extension. A new MagicDraw window opens.



To start MagicDraw with the last project you worked on, select **Options** > **Environment** and set **Open Last Project on Startup** to *true*.

XMI 2.4 / UML 2.4 is the main file format, used by MagicDraw for storing models. This format does not specify how to store diagrams, so MagicDraw stores and opens diagram data in XMI extension sections. Thus if you use MagicDraw to open an XMI file exported from another UML tool, only the model will be open, but not diagrams or views.

Correspondingly, if you open a MagicDraw file in another UML tool, diagrams or views will not be open in that tool (unless the tool understands MagicDraw specific file extensions).

You may open more than one project within the same MagicDraw window. A separate workspace will be created for the each opened project.

Opening projects saved with MagicDraw 17.0 or earlier

Projects saved with MagicDraw 17.0 or earlier can be as well opened with MagicDraw 17.0.1 or later.

To update older models, simply open them with MagicDraw 17.0.1 or later and then re-save them. Models will be automatically converted to the latest format.

In case your model is stored in a project that uses other projects, you should open and re-save the used projects first and only then do the same with the main project. Be advised that only having the used projects successfully converted to the latest format, you may expect the smooth conversion of the project that uses them.

Copyright © 1998-2015 No Magic, Inc.

If you convert only the project and leave the used projects saved in an old format, you will not be able to modify the projects used in the *read-write* accessibility mode.

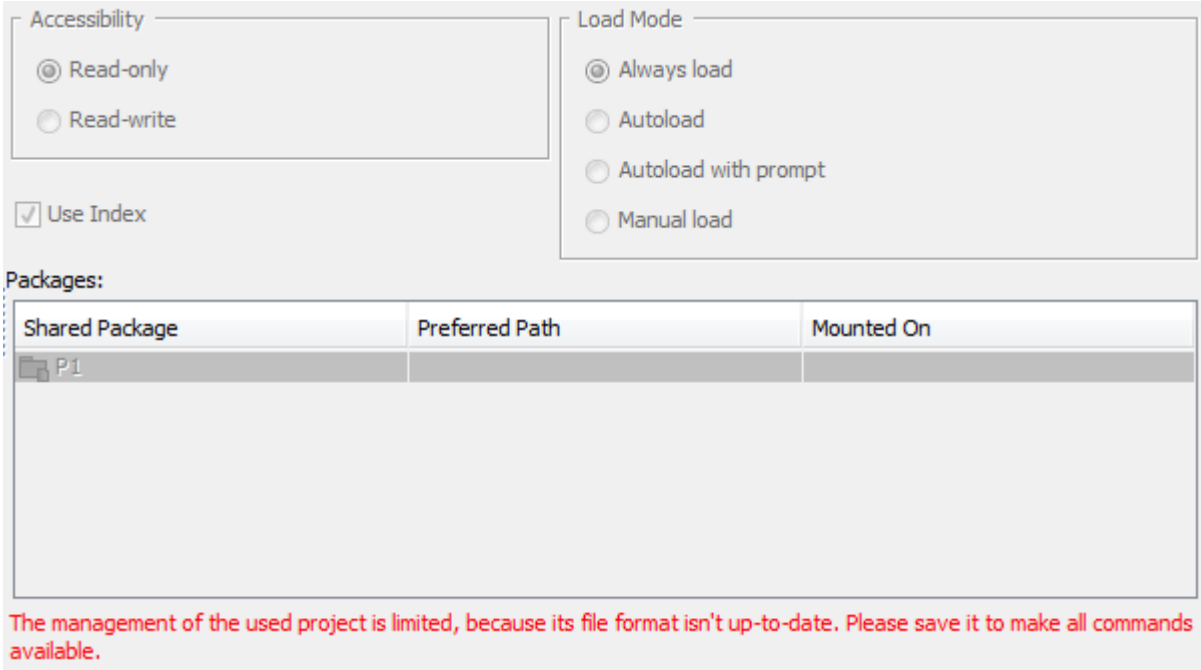


Figure 60 -- Fragment of Used Projects dialog showing options of non-converted project usage in converted project

The appropriate procedure should be applied in case your model is stored in a server project with composite structure: the used projects should be migrated to Teamwork Server 17.0.1 or later first and only then should be migrated the main project.

TIP! Read more in "Migrating server projects to MagicDraw Teamwork Server 17.0.1 or later" on page 1049.

Importing Data from other UML Tools

If you need to open a model which was created with another UML tool, you have to import the model to MagicDraw instead of simply opening it. MagicDraw can import

- Most of the model data from a Enterprise Architect 7.1, 7.5, or 8.0 file.

  NOTE Enterprise Architect does not export 100% standard UML 2.1 XMI, and this causes some data loss during the import.

- Model (without diagrams) from a IBM Rational Software Architect/Modeler 6.x file. Select **File** > **Eclipse UML2 (v1.x) Import** for this.
- Model from IBM® Rational® Software Architect/Modeler 7.x files. You can use MagicDraw RSXConverter or select **File** > **Eclipse UML2 (v2.x) Import** for this.

  TIP! For more information about MagicDraw RSXConverter, see "MagicDraw RSXConverter" on page 23.

- Model from a IBM® Rational Rose® Modeler file. You can use MagicDraw RConverter for this.

  TIP! For more information about MagicDraw RSXConverter, see "MagicDraw RConverter" on page 23.

- Model files (without diagrams) from Together 2006 files. Select **File** > **Eclipse UML2 (v1.x) Import** for this.

Copyright © 1998-2015 No Magic, Inc.

- UML diagrams from Microsoft Visio using XMI files.



Microsoft Visio 2007 and earlier requires extra add-ons for exporting UML model to XMI files.

Exporting Projects

MagicDraw allows for exporting projects to the following file formats:

- UML XMI 2.4 File. You can export a project to file formats that are supported by MagicDraw.
- MagicDraw Native XML File. You can export a project to an .xml file format.
- EMF Ecore File. You can export either the whole project or selected packages to an .ecore file.
- MOF XMI File You can export a project to the MOF (both CMOF and EMOF) XMI file.
- Eclipse UML2 (v1.x, v2.x, v3.x, v4.x) XMI File. You can export a project to an Eclipse based UML2 (v1.x / v2.x / v3.x / v4.x) compatible XMI file.



You can export a part of a project and share it with other users or projects. For the description of the exporting procedure, see "MOF Support" on page 178.



You can also export your project as a template. For the description of the exporting procedure, see "Exporting Projects as Eclipse UML2 (v1.x / v2.x / v3.x / v4.x) XMI Files" on page 117.

To export a project

1. On the main menu, click **File** > **Export To**.
2. Select a file format you want to export your project.
3. The following actions depends on the dialog that corresponds to the selected file format. Exporting procedures in details are described in the following sections:
   - "MOF Support" on page 178.
   - "Exporting Projects as Eclipse UML2 (v1.x / v2.x / v3.x / v4.x) XMI Files" on page 117.
   - "Exporting projects to Ecore files" on page 192.
   - "Exporting projects to MOF files" on page 179.

Exporting Projects as Eclipse UML2 (v1.x / v2.x / v3.x / v4.x) XMI Files

The export of a MagicDraw model to an Eclipse based UML2 (v1.x / v2.x / v3.x / v4.x) compatible XMI file enables the interchange of the UML2 models for the further manipulations and transformations with the most popular MDA tools, such as AndroMDA, OpenArchitectureWare, and other.

To export a project as an Eclipse UML2 (v1.x / v2.x / v3.x / v4.x) XMI file

1. Open a project you want to export as an Eclipse UML2 (v1.x / v2.x / v3.x / v4.x) XMI file.
2. On the **File** menu, click **Export To** and select one of the following command:
   - **Eclipse UML2 (v1.x) XMI File**.
   - **Eclipse UML2 (v2.x) XMI File**.
   - **Eclipse UML2 (v3.x) XMI File**.
   - **Eclipse UML2 (v4.x) XMI File**.

Copyright © 1998-2015 No Magic, Inc.

3. Specify a location for exported project files.
4. Click **Export**.



A project exported as an Eclipse UML2 (v1.x) XMI file is saved with the .uml2 file extension.

A project exported as an Eclipse UML2 (v2.x / v3.x / v4.x) XMI file is saved with the .uml file extension.

To change export property values

1. From the **Options** menu, select **Environment**.
2. Click the one of the following options group:
   - **Eclipse UML2 (v1.x) XMI**.
   - **Eclipse UML2 (v2.x) XMI**.
   - **Eclipse UML2 (v3.x) XMI**.
   - **Eclipse UML2 (v4.x) XMI**.
3. In the property list, specify desired property values. You can see descriptions of each property in the description area below the property list.

Copyright © 1998-2015 No Magic, Inc.

Creating and Using Project Templates

Project template is a customized project pattern. It serves as a starting point for creating a new project in a predefined format.

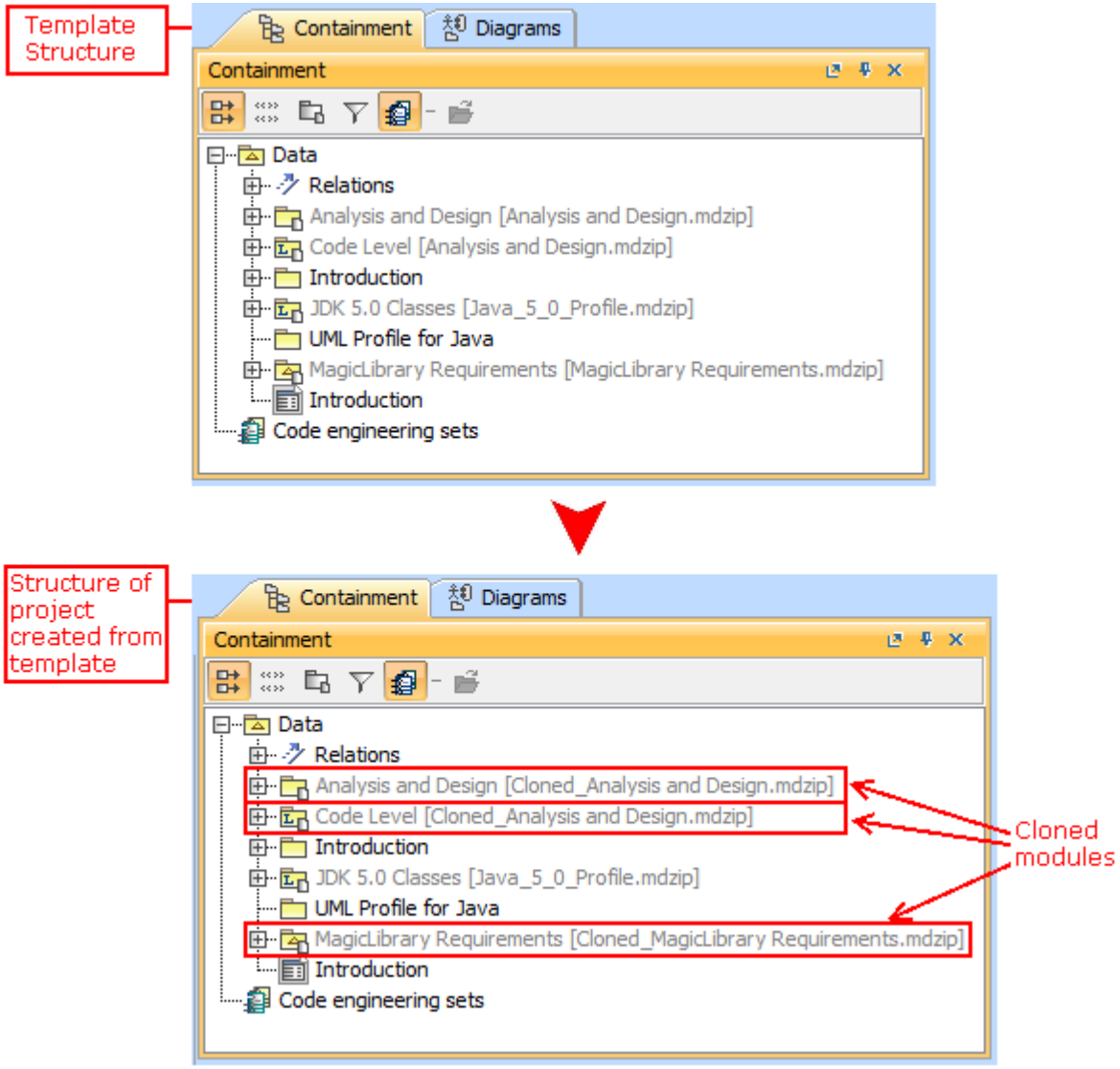


Figure 61 -- Example of a template structure and of a new project that is created from the template

In this section you can find information on how to prepare project template and then how to use it. There are the following sections:

- Exporting Projects as Templates
- Creating Custom Project Templates
- Creating New Projects From Templates

Exporting Projects as Templates

NOTE

This functionality is available in Standard, Professional, Architect, and Enterprise editions.

You can save (or export) the created project as a template and then use the same project for creating other new projects.

Copyright © 1998-2015 No Magic, Inc.

To export a project as a template

1. Open a project you want to export as a template.
2. From the **File** menu, select **Export** > **Template**. The **Export Template** dialog opens.
3. Type the name and the description of the template.
4. Click **OK**. The project is exported as template.

NOTE

- According to the template name in the **Export Template** dialog, the *.mdzip* file will be created in *<MagicDraw installation folder>\templates*. For example, type the name *LibraryTemplate* to save file as *LibraryTemplate.mdzip*.
- As of MagicDraw 17.0.3, when creating a new project from template, you can clone the custom used projects that exist in the template. For more information, see "Creating New Projects From Templates" page 122.

Related concepts

Creating and Using Project Templates

Related references

Creating Custom Project Templates

Creating New Projects From Templates

Creating Custom Project Templates

You can create your custom project template that appears in the **New Project** dialog.

In MagicDraw, each project template is saved in a separate folder in *<MagicDraw installation directory>\templates*. The template folder should consist of the following files:

- A MagicDraw project file containing predefined initial project data.
- An .xml file describing project template data. See the following figure wherein all customizable template data are depicted.
- An image of a dialog title bar icon that is displayed in the title bar of the **New Project** dialog. The recommended maximum image size is 80 x 80 pixels. Recommended image file formats are .gif or .png.
- An image of a template icon that is displayed in a particular project template category of the **New Project** dialog. The recommended maximum image size is 24 x 24 pixels. Recommended image file formats are .gif or .png.

IMPORTANT!

The folder, the MagicDraw project file, and the .xml file names must have the same.

Copyright © 1998-2015 No Magic, Inc.

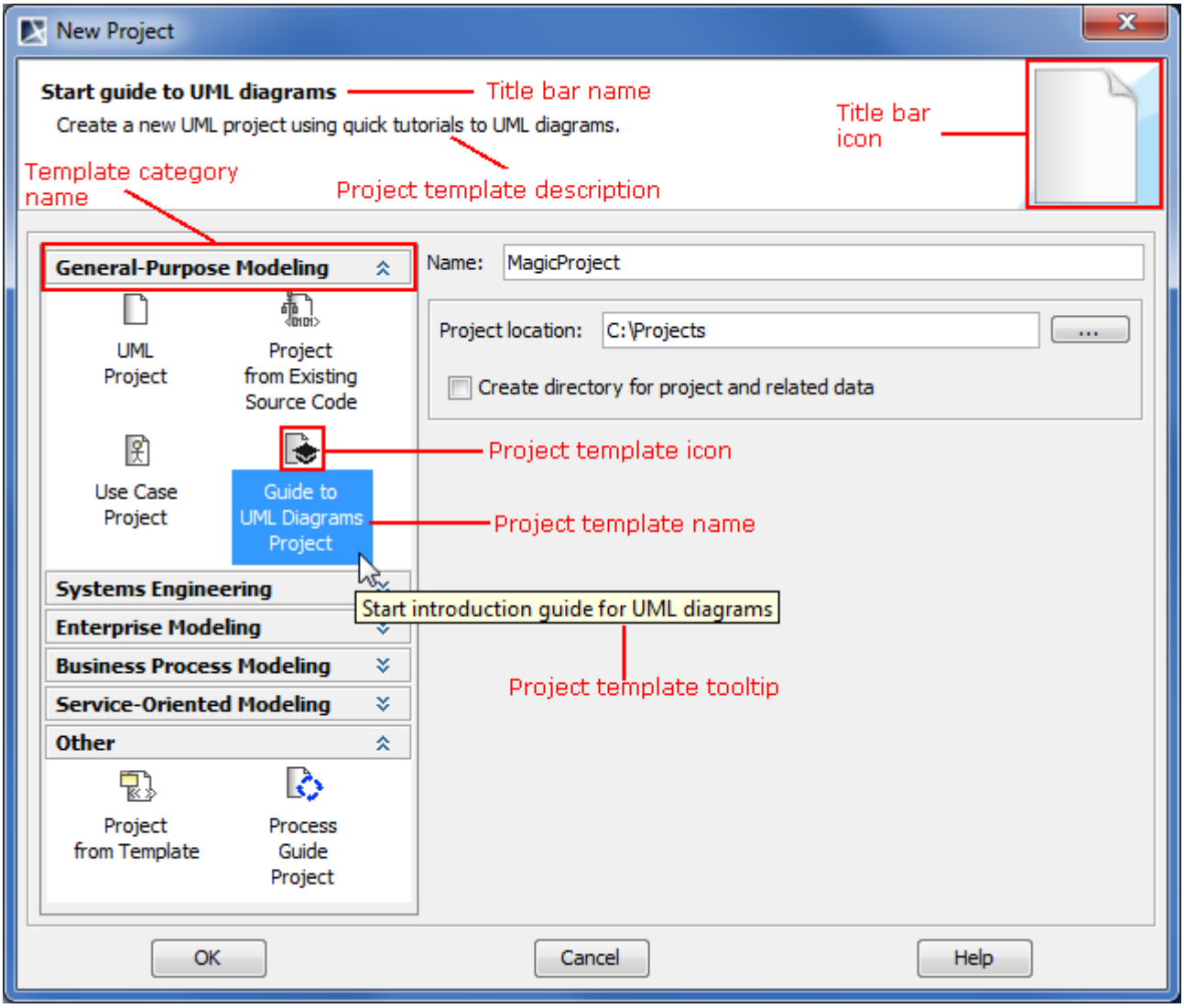


Figure 62 -- Customizable data for project template

While creating a project template, you can also create a category wherein your template can be located. If you need to have several project templates under the same category, in the .xml file, define the same category name for these templates.

To create a custom project template

1. Close MagicDraw, if it is started.
2. In *<MagicDraw installation directory>\templates* create a new folder for your template files.
3. Paste a MagicDraw project file and icon image files to the created folder. The project file name should be the same as a folder name.
4. In the same folder, create an .xml file and name it the same name as a folder name. The .xml file content should be as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<custom_template>
    <icon>Project_Template_Icon.gif</icon>
    <iconLabel>Project template name</iconLabel>
    <perspective>MagicDraw perspective name</perspective>

<!-- The default MagicDraw installation may have Full Featured, Quick Start,
Software Architect, System Analyst. For more information about MagicDraw
```


Copyright © 1998-2015 No Magic, Inc.

```
perspectives, see "Customizing and Selecting Perspective" on page 65 -->

    <tooltip>Project template tooltip text</tooltip>
    <banner_title>Title bar name</banner_title>
    <banner_description>Project template description</banner_description>
    <category>Template category name</category>
    <banner_icon>Title_Bar_Icon.png</banner_icon>
    <cloneModules/>
<!-- The cloneModules tag enables cloning of the existing modules from the template
project to the new project. Moreover the Clone custom modules check box and the
Unique prefix for module names text box are available in the New Project dialog.
For more information, see "Creating New Projects From Templates" on page 122. -->
</custom_template>
```

You can copy this example and change the tag values according to your data.

5. Save the file and start MagicDraw. Your created project template is displayed in the **New Project** dialog in the category defined by you. Select your template and create a new project.



The easiest way to create a custom template is to copy and paste the **Guide to UML Diagrams Project** folder with the different name and edit its inner files. This folder comes with the default MagicDraw installation and is located in *<MagicDraw installation directory>\templates*.

Related concepts

Creating and Using Project Templates

Related procedures

Creating Projects

Customizing and Selecting Perspective

Exporting Projects as Templates

Creating New Projects From Templates

Creating New Projects From Templates



This functionality is available in Standard, Professional, Architect, and Enterprise editions.

By default, the following templates are available in the **New Project** dialog: C#, C++, CIL, CORBA IDL, Guide to UML Diagrams Project, Java, Metamodeling, RUP, Use Case modeling, WAE, and WSDL.



You can also create your own customized templates. For more information on how to create templates, see the following sections: "Exporting Projects as Templates" on page 119 and "Creating Custom Project Templates" on page 120.

A newly created project from a template will contain specific structure, model elements, and stereotypes that were already created in the template. As of MagicDraw 17.0.3 when creating a new project from a template, you can select to clone the custom used projects that exist in the template.

To create a new project from a template

1. Do one of the following:
   - From the **File** menu, select **New Project**.
   - On the main toolbar, click the **New Project** button.
   - Press Ctrl+Shift+N.
     In all cases, the **New Project** dialog opens.
2. Expand the **Other** group and select the **Project from Template** icon.

Copyright © 1998-2015 No Magic, Inc.

3. Specify the new project name in the **Name** box.
4. Click the **...** button to select the location in your file system to store the new project.
5. Select the template from the templates tree and click **OK**.

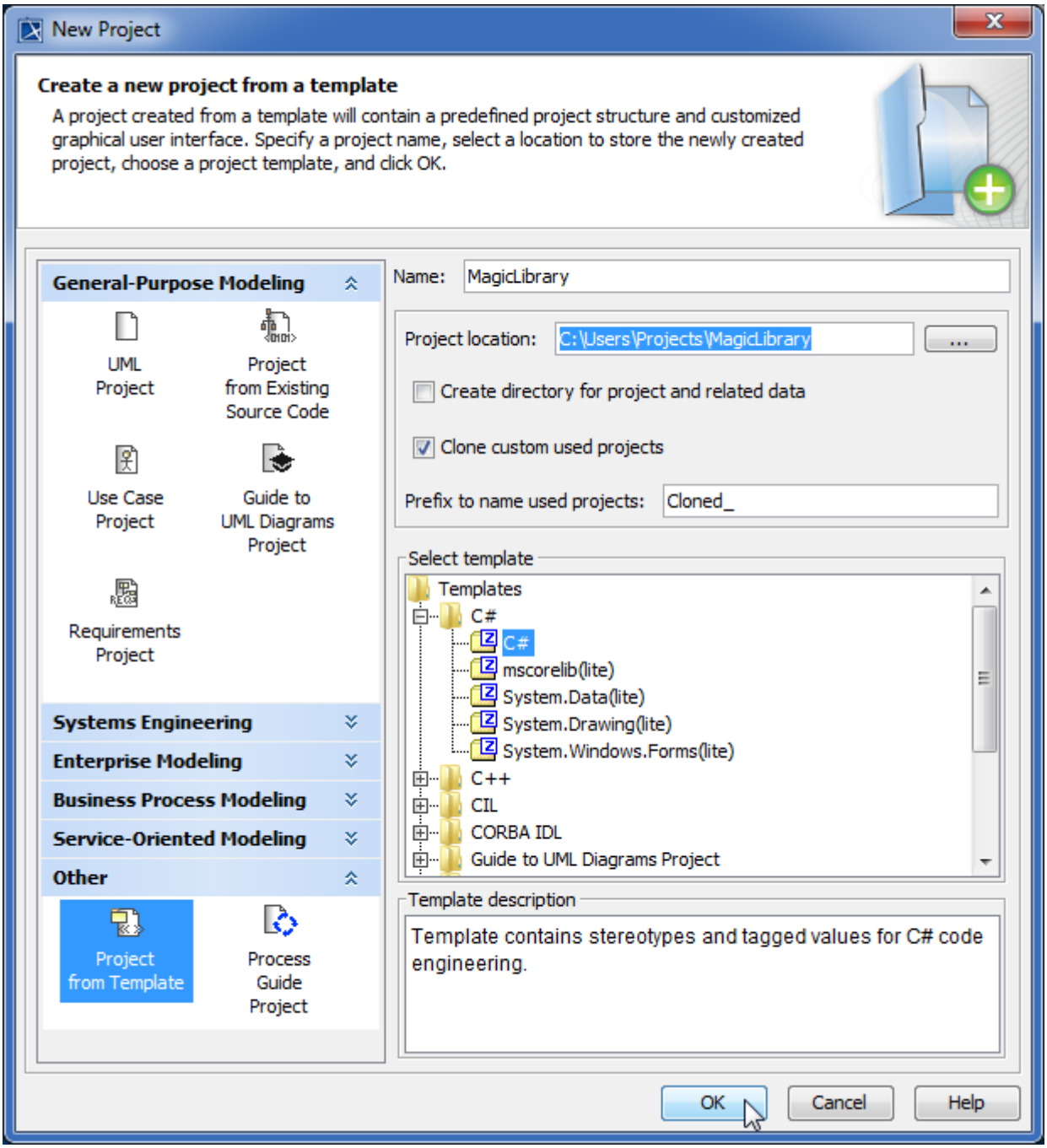


Figure 63 -- New Project dialog. Creating a new project from template

In the following table, see the **New Project** dialog description when under the **Other** category, the *Project from Template icon* is selected.

Element name	Type	Description
Other	Category	Select the **Other** category, to see the list of the available templates.
Project from Template	Icon	Select the **Project from Template** icon, to create a new project from template.

Copyright © 1998-2015 No Magic, Inc.

Element name	Type	Description
Name	Text box	Type the name of the new project that will be created from template.
Project location	Text box	Click the **...** button, to select a new project location. The **Select Location** dialog opens.
Create directory for project and related data	Check box	Click the check box to create a new folder for the new project. The folder name will be the same as the specified project name.
Clone custom used projects	Check box	Click to clone the used projects from the template project. In the new project from the template, the copies of the used projects will be created.
Prefix to name used projects	Text box	Type the prefix for the name of each cloned used project. The default value is *Cloned_*. For example, if a used project of the template project is named *Requirements,* then the name of the cloned used project in the newly created project is *Cloned_Requirements*. **NOTE:** The text box is available only if the **Clone custom used projects** check box is selected.
Select template	List	Select a template from the list. The list represents the folders and files, that are located in *<MagicDraw installation folder>\templates*.
Template description	Text area	The predefined template description is displayed. Note, that this field is not editable.



You can also import the desired template into your project. Select **File** > **Import From** > **Another Project** for this.

Related concepts

Creating and Using Project Templates

Related references

Exporting Projects as Templates

Creating Custom Project Templates

NEW! Projects with File Attachments

You can attach any type of file from the local file system to your MagicDraw project.

Attached files are bundled with the project and can be safely transfered to another computer together with this project. Files are not missing after transferring the project, and project references to these files are not broken.

Copyright © 1998-2015 No Magic, Inc.

The contents of the attached file can be modified and the changes are instantly saved to the project. Different versions of server project can refer to different versions of the same attached file.

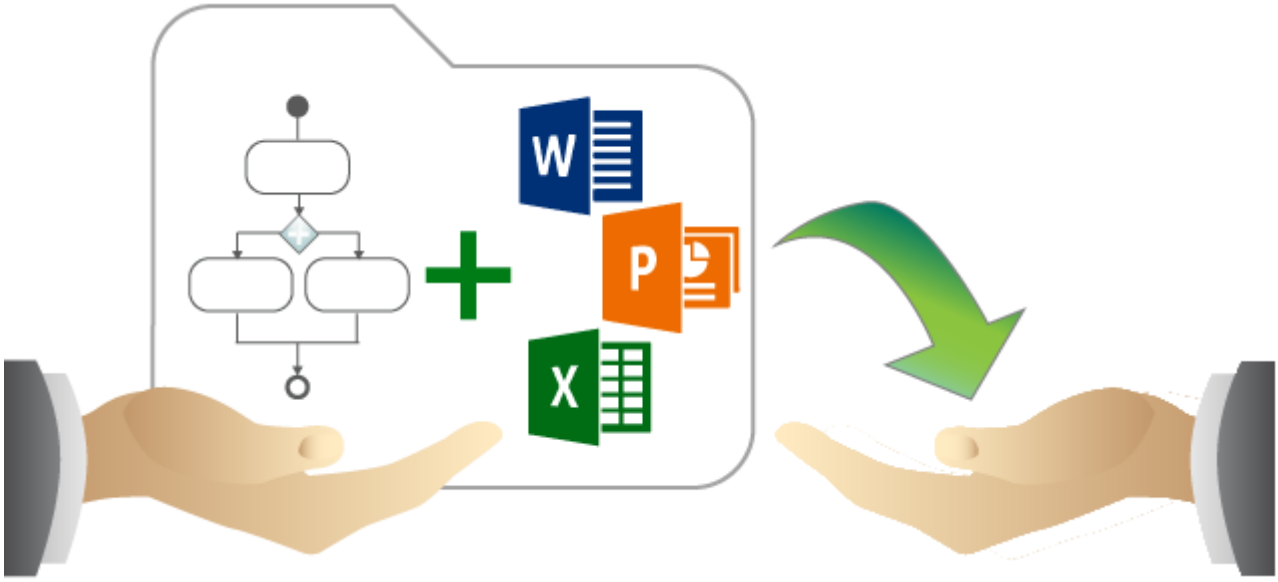


Figure 64 -- Concept of Project with File Attachments

To learn more about the attached files, read:

- "Attaching, Reattaching, and Detaching Files" on page 126
- "Working with Attached Files" on page 127

Copyright © 1998-2015 No Magic, Inc.

Attaching, Reattaching, and Detaching Files

When attached to the project, file becomes a model element of the Attached File type and is displayed in the Model Browser.

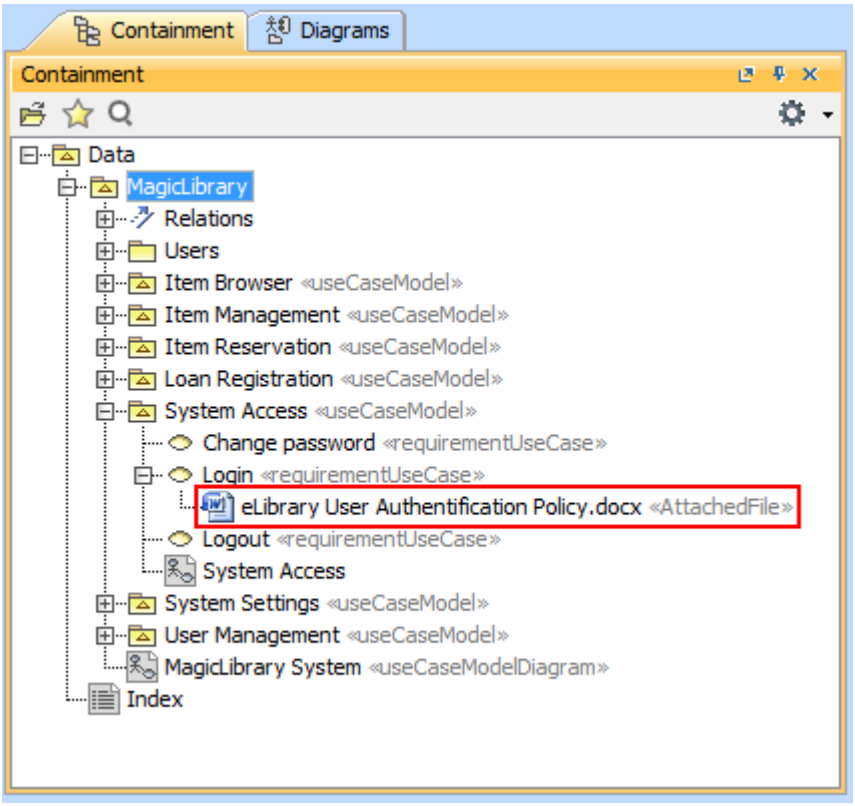


Figure 65 -- Attached file in Containment tree

A file can be attached in one of the following ways:

- By dragging the file icon to the diagram pane.
- By dragging the file icon to the Containment tree.
- By using a command from the shortcut menu of an element selected in the Containment tree.

To attach a file by dragging the file icon to the diagram pane

1. In the file system, select the icon of a file you need to attach.
2. Drag it to the diagram pane.

The shape of a new Attached File element appears on the diagram pane. The diagram owner becomes the owner of the newly attached file too.

To attach a file by dragging the file icon to the Containment tree

1. In the file system, select the icon of a file you need to attach.
2. Drag it onto an element in the Containment tree. The shortcut menu opens.
3. Select **Create Attached File**.

A new Attached File element appears in the Containment tree. The element on which you dragged the file icon becomes the owner of the newly attached file.

Copyright © 1998-2015 No Magic, Inc.

To attach a file by using a command from the shortcut menu of an element

1. In the Containment tree, right-click a possible owner of the attached file. The shortcut menu opens.
2. Select **Create Element** > **Attached File**. The **Select File to Attach** dialog opens.
3. Select the file to attach and then click **Attach**. A new Attached File element with its name in the edit mode, appears in the Containment tree.
4. If you do not need to change the file name, press Enter. Otherwise, type a new name and only then press Enter.

The Attached File element created in either way possess the properties, such as **Attached At**, **Modified At**, **Size**, **Author**, and **File**. You can see these properties, their descriptions, and values in the Specification window of the Attached File element. If you need to attach another file instead of it, you can do this with help of the **File** property value editor.

To reattach the file

1. In the Containment tree, select the attached file you need to replace.
2. Press Enter to open its Specification window.
3. In the Quick Filter box, type *file*.
4. Click the cell of the **File** property value. The button  appears.
5. Click the button. The **Select File to Attach** dialog opens.
6. Select another file to attach and then click **Attach**.
7. Click **Close** in the Specification window.

The file is reattached, and the **Modified At** property value of the appropriate Attached File element updates.

If you no longer need the attached file, you can simply remove it from the model as any other element. Just select it in the Containment tree and press Delete. If you have its shape on the diagram, select the shape and press Ctrl+D.

Working with Attached Files

Attached files can be:

- Reviewed and modified (see "Modifying the contents" on page 128).
- Inserted into HTML text and displayed on diagrams (image files only) (see "Inserting into HTML text" on page 128).
- Exported back to the local system ("Exporting back to the local file system" on page 129).
- Included in reports. The attached file is saved in the file system together with the generated report. In this report, any hyperlink to that attached file defined in the model is replaced with reference to the appropriate file in the file system.

Copyright © 1998-2015 No Magic, Inc.

Modifying the contents

Attached files can opened directly from MagicDraw and reviewed or modified with appropriate programs, for example, documents – with MS Word, worksheets – with MS Excel, HTML files – with any HTML editor.

To open an attached file

Do either:

- In the Containment tree, double-click the attached file.
- In the Containment tree, right-click the attached file and select **Open File**.
- Double-click either the element that has an active hyperlink to this attached file, or the shape of that element (see the following figure).



For more information, see "Adding a hyperlink to the model element" on page 341.

The file opens with appropriate program in your operating system. You can review or modify it.

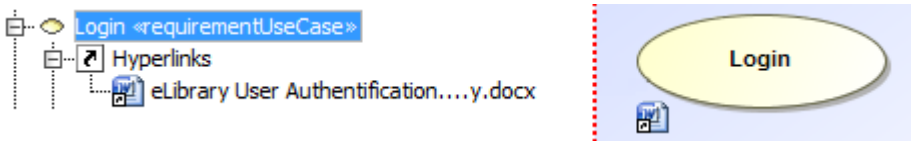


Figure 66 -- Element (left) and its shape (right) with active hyperlink to attached file

After closing the modified file, you can choose whether to save it in MagicDraw or not. Saving the latest version of the file in MagicDraw updates the **Modified At** property value of the appropriate Attached File element.



If the attached file is read-only, the changes made in it cannot be saved. To be able to modify the file, do one of the following:

- If the file is attached to used project, change the accessibility mode of this used project from *read-only* to *read-write*. For more information, refer to "Managing Used Projects" on page 155.
- If the file is attached to server project, lock the Attached File element for edit. For more information, refer to "Locking Model Elements and Diagrams" on page 1052.

Inserting into HTML text

Attached *image* files can be inserted into HTML texts and displayed on diagrams.

To insert the attached image file into HTML text

1. In the Containment tree, right-click the attached image file and select **Copy Image URL** from the shortcut menu.
2. On a diagram pane, draw a text box with the HTML mode on.
3. Open the **Advanced HTML Editor** dialog from the text box shape.



For more information, see "Advanced HTML Editor dialog" on page 428.

4. On the toolbar, click the Insert Image button. The **Picture Properties** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

5. Click the **Picture source** box and press Ctrl+V (Cmd+V on OS X). The attached image file URL is inserted in the text box.
6. Click **OK** twice to close both the **Picture Properties** and **Advanced HTML Editor** dialogs.

The inserted image appears on the diagram. Now if you transfer the project to another computer, the diagram continues displaying the image.

Exporting back to the local file system

To export the attached file back to the local file system

1. In the Containment tree, right-click the attached file and select **Save File As** from the shortcut menu. The **Save** dialog opens.
2. Select the location in the local file system to save file and click the **Save** button.

The file is saved in the local file system.

Setting Project Options

Use the **Project Options** dialog to do the following:

- Specify general project-specific options.
- Specify summarizing information (for example, diagram author, diagram creation and modification dates) that will be displayed on each diagram.
- Specify symbol property styles for shapes, paths, diagrams, and stereotypes within the project.



For the instructions how to create, edit, clone, import /export, or remove symbol property styles, please refer to "Style Engine" on page 324.

- Change default element property values.



For the instructions about setting the default element property values, see "Default Property Values" on page 296.

- Set general code generation or code reversing options as well as code formatting styles for selected programming languages.

The **Project Options** dialog includes option groups, each designated for one of the above mentioned features and containing lists of corresponding options. Groups are represented using the tree structure.

An option value can be simply changed by typing a new value, setting a value to true / false, or selecting a value from the list.

Learn more about the **Project Options** dialog in the following sections:

- "Using Project Options dialog" on page 130.
- "Common elements in Project Options dialog" on page 131.

Copyright © 1998-2015 No Magic, Inc.

To open the **Project Options** dialog

- From the **Options** menu, select **Project**.

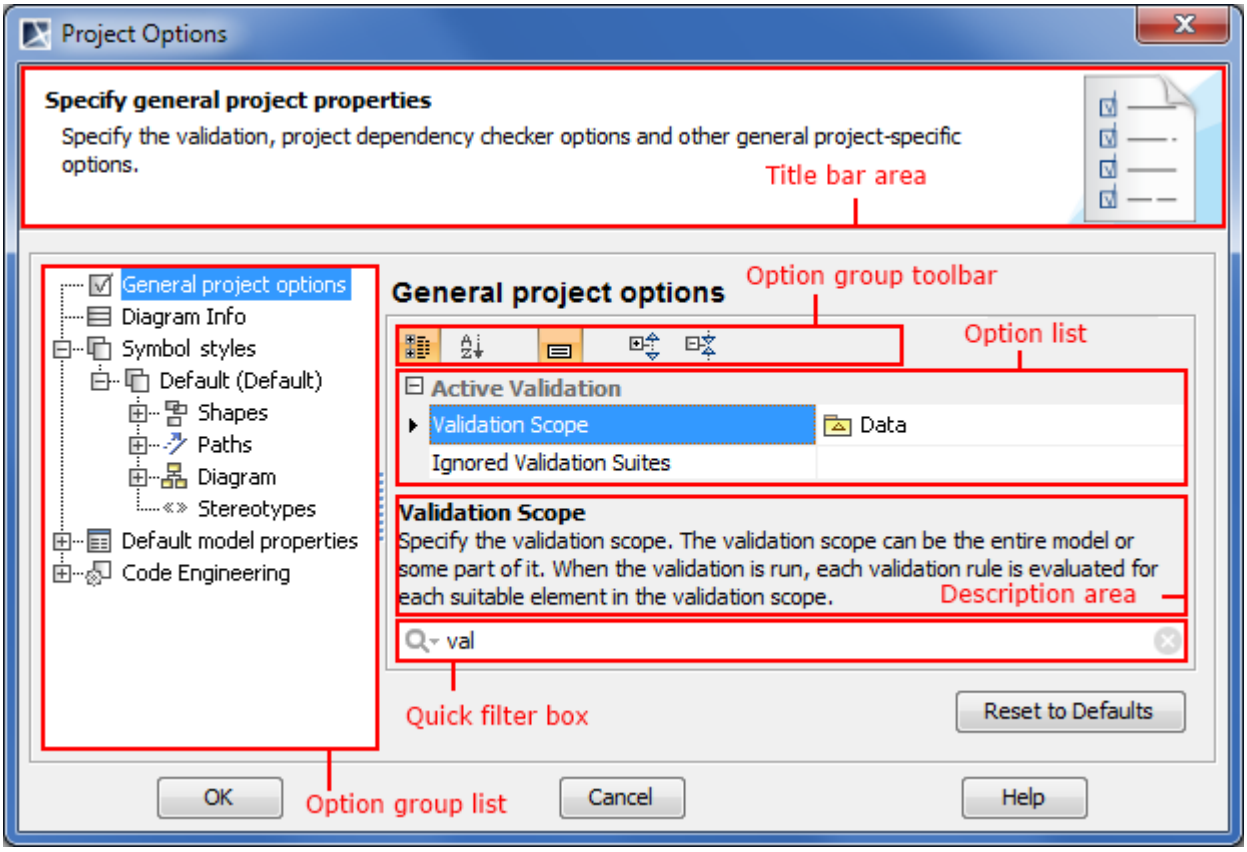


Figure 67 -- Structure of Project Options dialog

Using Project Options dialog

In order to change a desired project option, first of all you may need to find it. It can be rather difficult to do this, if the options list contains 10 or more options. In this case the Quick filter box appears in the dialog. Using the Quick filter box you can quickly find the desired option in the list.

TIP! For more information about the Quick filter box please refer to "Quick Filter" on page 279.

For better understanding an option you can read its description that tells what is the effect of changing the option value.

To read the option description

IMPORTANT! Make sure that the Show Description mode is turned on in the **Environment Options** dialog. To turn the Show Description mode on or off, click the Show Description button on the following toolbar in the **Project Options** dialog.



1. Click an option, whose value you want to change.
2. Read the option description in the area below the options list. You are ready now to change the option value.

Copyright © 1998-2015 No Magic, Inc.

Common elements in Project Options dialog

For the common element descriptions look in the following table.

Element	Element Type	Description
Quick filter	Text box	Type an option name or its fragment. **TIP!** For more information about the Quick filter box please refer to "Quick Filter" on page 279.
Reset to Defaults	Button	Resets all options to their default values.
OK	Button	Saves changes and closes the dialog.
Cancel	Button	Closes the dialog without saving changes.
Help	Button	Opens MagicDraw Help.

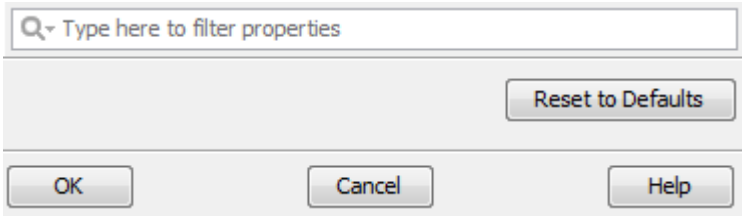


Figure 68 -- Fragment of Project Options dialog. Quick filter box and common buttons

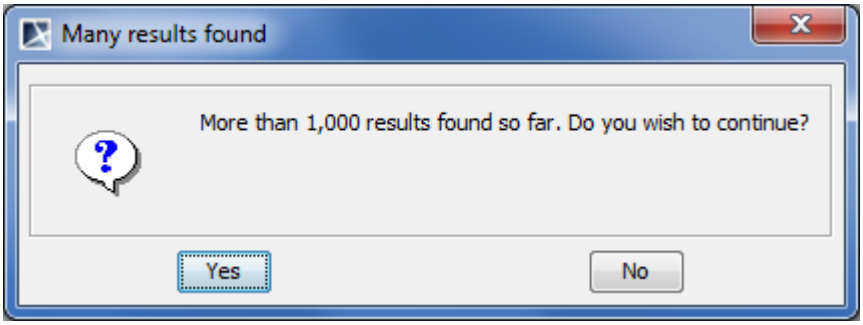
Searching

The MagicDraw search mechanism is designed for search within model elements, element properties. You can also perform a quick search, that is a search without defining any advanced options, as well as you can replace any text quickly.

These are the types of search dialogs:

- Find dialog
- Quick Find dialog
- Find TODO
- Find and Replace dialog

If there is a large number of search results to display (more than 1000), the following message appears.



Click **Yes** to continue searching, or click **No** to stop searching and correct the query.

Copyright © 1998-2015 No Magic, Inc.

A search is performed as a background task. That is while searching, you may read the model, browse through it. After you make changes in the model, the search is stopped, and incomplete search results will be displayed in a Search Results tab in the Model Browser. You need to restart the search to get the full list of results.



For more information about the background tasks, see "Background Tasks Manager" on page 98.

Find dialog

The **Find** dialog is designed to enter data and define options to search for elements according to their property values. In this dialog you can

- Type a phrase to search for
- Choose an element type you are searching for
- Select properties and their values for the search
- Specify the search scope
- Define other search criteria

To open the **Find** dialog, do one of the following

- On the main menu, click **Edit** > **Find**.
- Press Ctrl+F.
- On the main toolbar, click the  button.

To open the **Find** dialog for searching in an active diagram, do either

- Right-click the active diagram and select **Find in Diagram** from its shortcut menu.
- Press CTR+SHIFT+D.

Copyright © 1998-2015 No Magic, Inc.

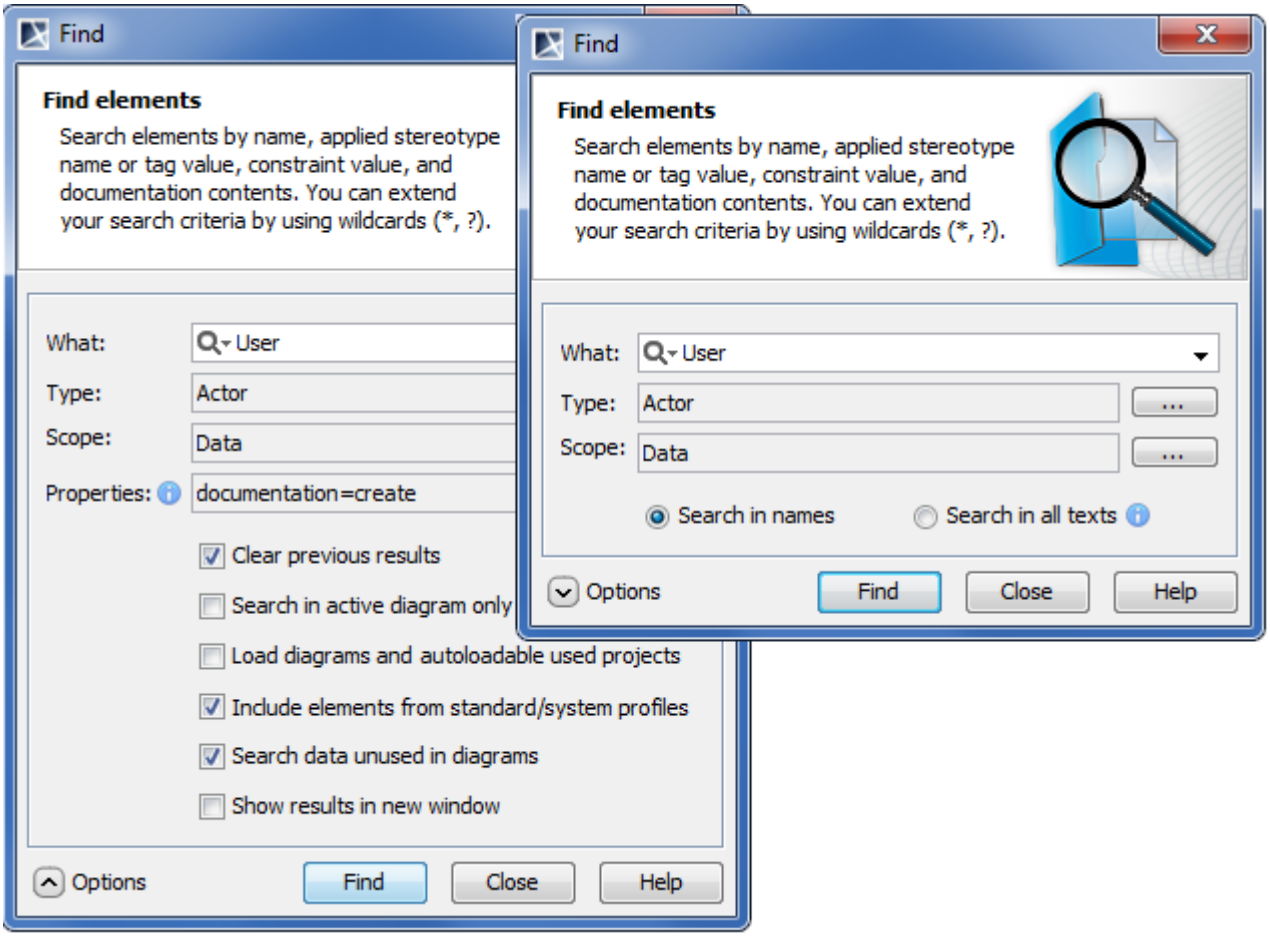


Figure 69 -- Find dialog with collapsed (on the right) and extended (on the left) Options area

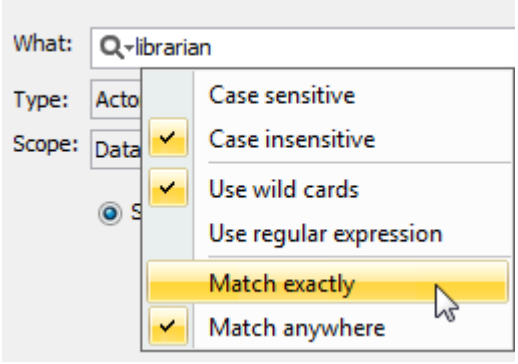
Search results are displayed in the **Search Results** tab on the Model Browser. You can select to display the results of every subsequent search in a new **Search Results** tab on the Model Browser.

TIP!

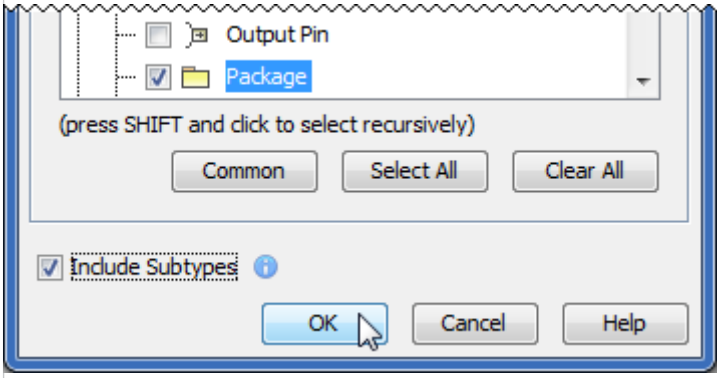
For more information about the **Search Results** tab, see "Search Results tab" on page 87.

Copyright © 1998-2015 No Magic, Inc.

See the description of the **Find** dialog components in the following table.

Dialog element name	Dialog element type	Description
What	Text box with drop-down list	Type or select from the list a phrase to search either in element names or in textual properties. To search in element names, click **Search in names**. To search in textual properties including Documentation, click **Search in all texts**. Note that both above mentioned options are available only if the **Options** area is collapsed. Use the text box shortcut menu to make your search phrase more specific.  You may use wildcard characters (*,?) in your search phrase. “*” means zero, one or more characters. So if your search phrase is “*dd”, all the elements containing “dd” in their names can be found. For example, “Address” and “ThreadDeath”. “?” means exactly one character. So if your search phrase is “?dd”, only the elements with “dd” in their names starting from the second character can be found. For example, “Address”. “ThreadDeath” cannot be found in this case, because it contains more than one character before the search phrase “dd”. Be sure the **Match exactly** command is checked when using wildcard characters!

Copyright © 1998-2015 No Magic, Inc.

Dialog element name	Dialog element type	Description
Type	Text box with ...	Click **...** and select one or more element types from the proposed items in the opened dialog, or leave the default value <any>. **NEW!** Standard and custom subtypes of the selected element type can be optionally included into the search results. For example, if you need to search for packages as well as for profiles, models, smart packages, and other custom subtypes of the Package type, select this type and then click to select the **Include Subtypes** check box. 
Scope	Text box with ...	Click **...** and select a package wherein the content search will be performed. You can specify the search scope even before opening the **Find** dialog. For this do one of the following: • Right-click a package in the Model Browser and select **Find** from its shortcut menu. • Select a package in the Model Browser and press CTRL+F.
Properties	Text box with ...	Click **...** and select properties and specify their values for the search. The property list in the opened dialog contains only the properties of one or more element types that are selected in the **Type** box. Select at least one element type to make the button available. Keep in mind that property list contains primitive value properties (for example, string, boolean, integer) and a non-primitive value property named **Applied Stereotype**.
Options area		
Clear Previous Results	Check box	Select to remove all previous search results from the **Search Results** tab on the Model Browser.
Search in active diagram only	Check box	Select to search the symbols in the active diagram only. The check box is not available, if all diagrams are closed or if the active diagram is empty. Note that the check box is automatically selected, if the **Find** dialog is opened by using the **Find in Diagram** command on the diagram shortcut menu.

Copyright © 1998-2015 No Magic, Inc.

Dialog element name	Dialog element type	Description
Load diagrams and autoloadable used projects	Check box	Select to load all elements to be included in the search if the model has diagrams or used projects that are not loaded. Used project will not be loaded and elements from it will not be included into the search, if the load mode of that used project is **Manual load**. For more information about working with used projects, see "Managing Used Projects" on page 155.
Include elements from standard/ system profiles	Check box	Select to include elements from standard/system profiles.
Search data unused in diagrams	Check box	Select to search only for those elements that do not contain symbols in any diagram. Note that the check box is unavailable, if the **Find** dialog is opened by using the **Find in Diagram** command on the diagram shortcut menu.
Show results in new window	Check box	Select to display the results of every subsequent search in a new **Search Results** tab on the Model Browser.

How to...

This section gives some remarks that can be useful to gain some knowledge in performing a specific search.

How to find all abstract classes in a model?

Solution

1. Select class as element type to search for.
2. Expand the **Options** area if it is not yet expanded.
3. Select the **Is Abstract** property to search in and set its value to true.

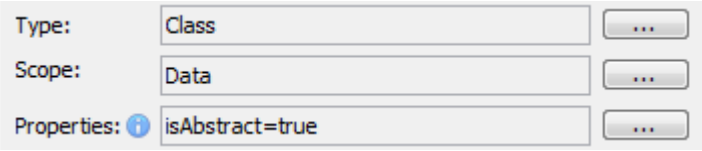


Figure 70 -- Fragment of Find dialog illustrating above described case

How to find all elements with specific stereotype applied and having special phrase in their documentation?

Solution

1. Select actor as element type to search for and select to include its subtypes.
2. Expand the **Options** area if it is not yet expanded.
3. Select the **Applied Stereotype** property to search in and select some custom stereotype, for example, «BusinessActor» as its value.

Copyright © 1998-2015 No Magic, Inc.

4. Select the **Documentation** property to search in and type a phrase, for example, “Corporation Manager” to search for.

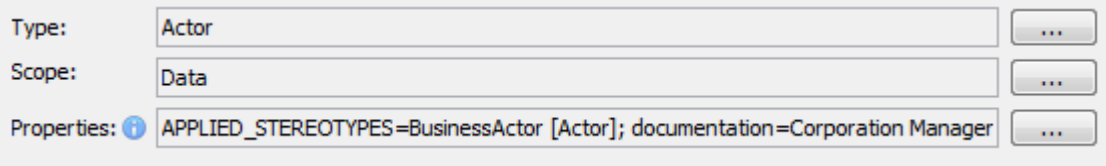


Figure 71 -- Fragment of Find dialog illustrating above described case

Related concepts

Searching
Filtering

Related references

Search Results tab
Quick Find dialog
Find and Replace dialog

Quick Find dialog

You can use a quick search to find an element by its name. A very simple and easy to use dialog is designed to perform a quick search. In this dialog, you just need to type a name of the element you are searching for and select, if it’s a type, diagram, or element.

To open the **Quick Find** dialog

Do either:

- On the toolbar of the **Containment**, **Diagrams**, **Inheritance**, or **Model Extensions** tab, click  .
- On the main menu, click **Edit** > **Quick Find**.
- Press CTRL+ALT+F.
- On the main toolbar, click the arrow nearby the  button and then select **Quick Find**.

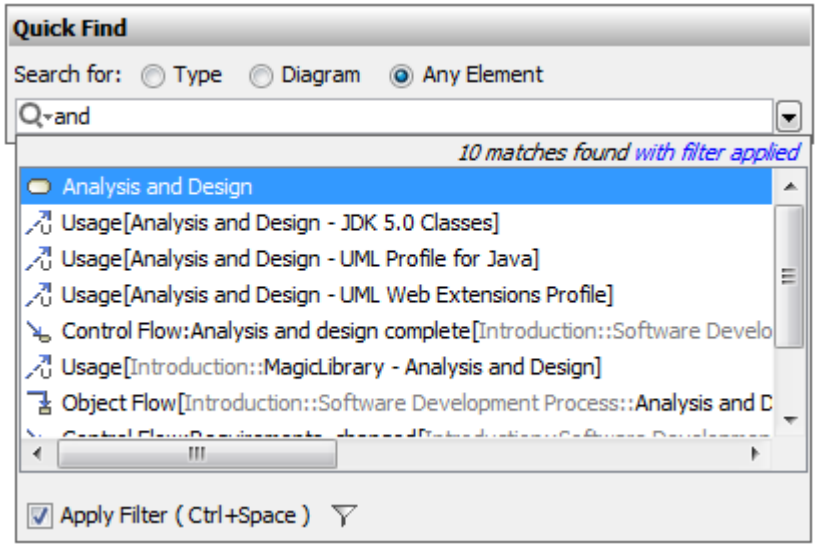


Figure 72 -- Quick Find dialog

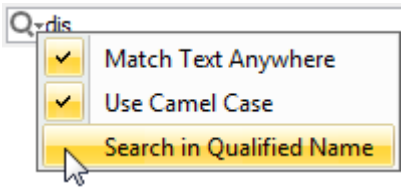
Copyright © 1998-2015 No Magic, Inc.

Search results are displayed in the drop-down list, which opens when you start typing a search phrase. Click the element you need in this list or correct the search query to perform a new search. The element will be selected in the Containment tree on the Model Browser as well as on the active diagram, if the element has symbols on the active diagram.

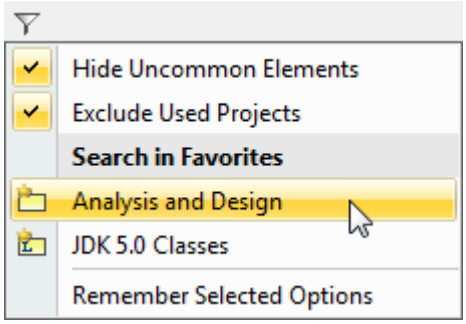
Elements of the **Quick Find** dialog are described in the following table.

Dialog element name/image	Dialog element type	Description
Search for	Options group	• Click **Type** to search only in types, that is, classes, interfaces, use cases, and others. • Click **Diagram** to search only in diagrams. • Click **Any Element** to search in all elements of your project.
Search By Name	Text box with a drop-down list	Type a phrase to search for in element names. See the search results in the drop-down list. You may use wildcard characters (*,?) in your search phrase: • “*” substitutes any range of characters. So if your search phrase is “*dd”, all the elements containing “dd” in their names can be found. For example, “Address” and “ThreadDeath”. • “?” substitutes exactly one character. So if your search phrase is “?dd”, only the elements with “dd” in their names starting from the second character can be found. For example, “Address”. “ThreadDeath” cannot be found in this case, because it contains more than one character before the search phrase “dd”. Be sure the **Match Text Anywhere** command is unchecked when using wildcard characters!

Copyright © 1998-2015 No Magic, Inc.

Dialog element name/image	Dialog element type	Description
NEW! 	Menu button	Click the button and customize search options.  **Match Text Anywhere** • To search for matching phrase in any part of element name, click the command to add the check mark. • To explicitly search from the beginning of element name, click the command to remove the check mark. **Use Camel Case** • To search for matching phrase typed in camel case, click the command to add the check mark. • To search for matching phrase typed in regular case, click the command to remove the check mark. You can use camel case as follows: • Type only the first letters (either capitalized or not) of each word, for example, “and” or “AND” to find “Analysis and Design”. • Type skipping spacers, for example, “cv3” to find “CV-3”. • Perform partial search, for example, “ibd” to find SysML Internal Block diagram. Be sure the **Match Text Anywhere** command is checked! **Search in Qualified Name** • To search for matching phrase in qualified names of elements, click the command to add the check mark. • To search for matching phrase in element names only, click the command to remove the check mark.
NEW! Apply Filter (Ctrl+Space)	Check box	Click to clear the check box, if you need to remove the search scope filter.

Copyright © 1998-2015 No Magic, Inc.

Dialog element name/image	Dialog element type	Description
NEW! 	Menu button	Click the button and customize search scope filter options. Be sure the **Apply Filter** check box is selected!  **Hide Uncommon Elements** To reduce the search results list by hiding uncommon elements, click the command to add the check mark. Elements, such as Interface, Association Class, Class, and Component, remain in the list. **Exclude Used Projects** To reduce the search results list by excluding elements from the used projects, click the command to add the check mark. **Search in Favorites** Select one or more favorite packages as the search scope. Be sure you have at least one package marked as favorite! For more information, see "Favorites" on page 416. **Remember Selected Options** To make the **Quick Find** dialog remember the specified search scope filter options for the next time it opens, click the command to add the check mark. The same scope filter settings appear in autocompletion lists and the element Selection dialog as well.

Related concepts

Searching

Related procedures

Find dialog

Find TODO

Find and Replace dialog

Related references

Containment tab

Inheritance tab

Copyright © 1998-2015 No Magic, Inc.

Diagrams tab
Model Extensions tab

Find TODO

Performs a quick search for all To Do values.

You can specify To Do values in the element Specification window, the **To Do** property cell. You can use the **To Do** property cell for adding the notes about work you need to do. Then you can quickly find the elements that have the **To Do** properties defined.

Results of the To Do search are displayed in the **Search Results** tab on the Model Browser.



For more information about the **Search Results** tab, see "Search Results tab" on page 87.

To start the **Find TODO** search, do one of the following

- On the main menu, click **Edit** > **Find TODO**.
- On the main toolbar, click the arrow nearby the  button and then select **Find TODO**. The **Search Results** tab opens with results represented in it.

Related concepts

Searching

Related references

Search Results tab
Find dialog
Quick Find dialog
Find and Replace dialog

Find and Replace dialog

The Find and Replace functionality allows for replacing one specified model value with another.

You can change the values for the following properties:

- Names
- Documentation
- Tag values
- Text included to Notes
- Text included to Text Boxes
- Expressions

To open the **Find and Replace** dialog, do one of the following

- On the main menu, click **Edit** > **Find and Replace**.
- Press Ctrl+R.

Copyright © 1998-2015 No Magic, Inc.

- On the main toolbar, click the arrow nearby the  button and then select **Find** > **Find and Replace** button.

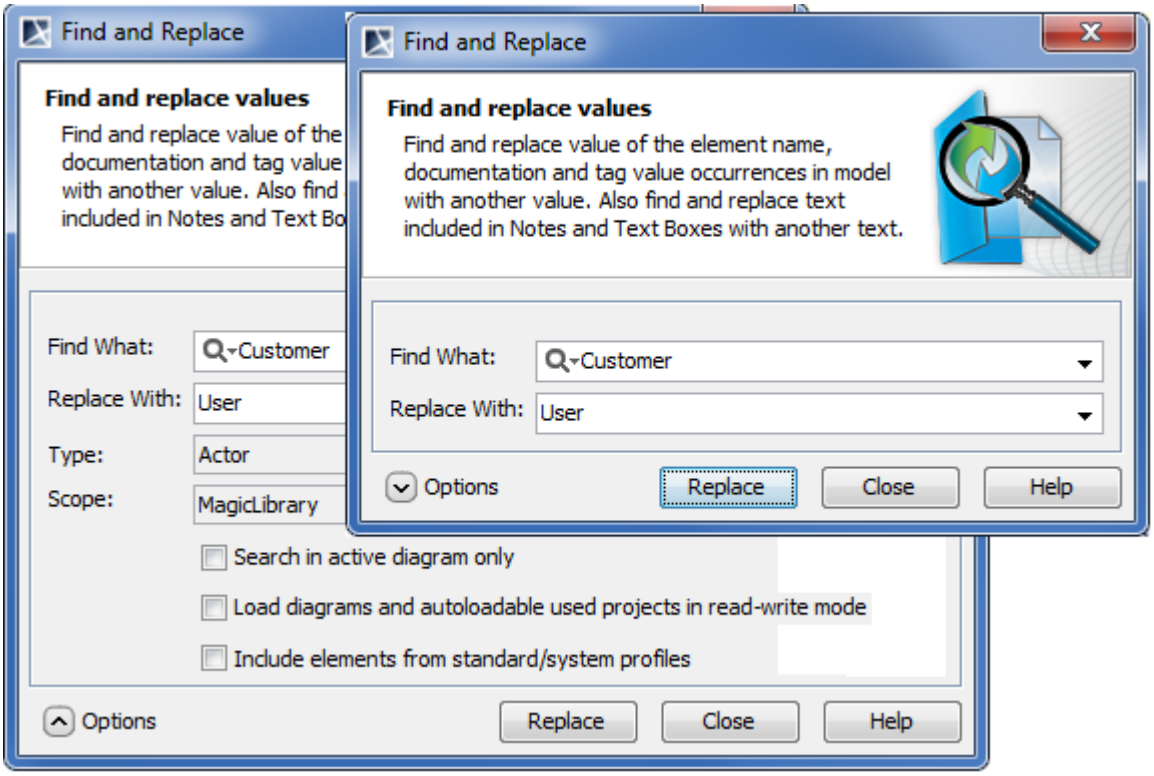


Figure 73 -- Find and Replace dialog with collapsed (on the right) and extended (on the left) Options area

See the description of the **Find and Replace** dialog components in the following table.

Dialog element name	Dialog element type	Description
Find What	Text box with drop-down list	Type or select from the list to search the value of the element to be replaced.
Replace With	Text box with drop-down list	Type or select from the list the value of element that will replace the found value.
Options area		
Type	Text box with ...	Click **...** and select one or more element types from the proposed items in the opened dialog, or leave the default value <any>.
Scope	Text box with ...	Click **...** and select a package wherein the content search will be performed.
Search in active diagram only	Check box	Select to search the symbols in the active diagram only. The check box is not available, if all diagrams are closed or if the active diagram is empty.
Load diagrams and autoloadable used projects in read-write mode	Check box	Select to load all elements to be included in the search if the model has diagrams or used projects that are not loaded. Used project will not be loaded and elements from it will not be included into the search, if the load mode of that used project is **Manual load**. For more information about working with used projects, see "Managing Used Projects" on page 155.

Copyright © 1998-2015 No Magic, Inc.

Dialog element name	Dialog element type	Description
Include elements from standard/ system profiles	Check box	Select to include elements from standard/system profiles.



You may see the error when trying to replace a value with a not valid one.

- For example, if the value type is *boolean*, you may replace values from *true* to *false* or from *false* to *true*, but if you will try to replace the value *true* or *false* to other, for example, to *Motor,* an error message will be displayed.
- You will not be allowed to change the *Integer* value to *String* when it is a part of the value. For example, if you have the *120* value when trying to replace *20* with *AB*, an error message will be displayed.

Related concepts

Filtering

Searching

Related dialogs

Find dialog

Quick Find dialog

Find and Replace dialog

Filtering

The MagicDraw filtering mechanism allows for filtering within the types of the model elements in particular lists. For example, you can filter the types of the model elements that are displayed in the Model Browser, or you can select the types of the model elements when searching.

You can perform filtering in the following two cases, see the following procedures:

- To filter the types of the model elements that are displayed in the Containment tab
- To filter the types of the model elements when searching

To filter the types of the model elements that are displayed in the Containment tab

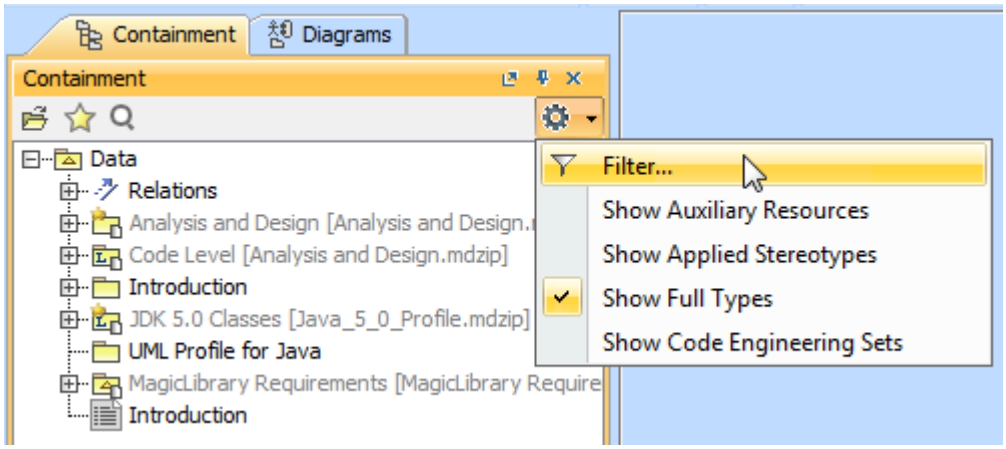
1. In the Model Browser, open the **Containment** tab (if it was closed or if another tab was open).



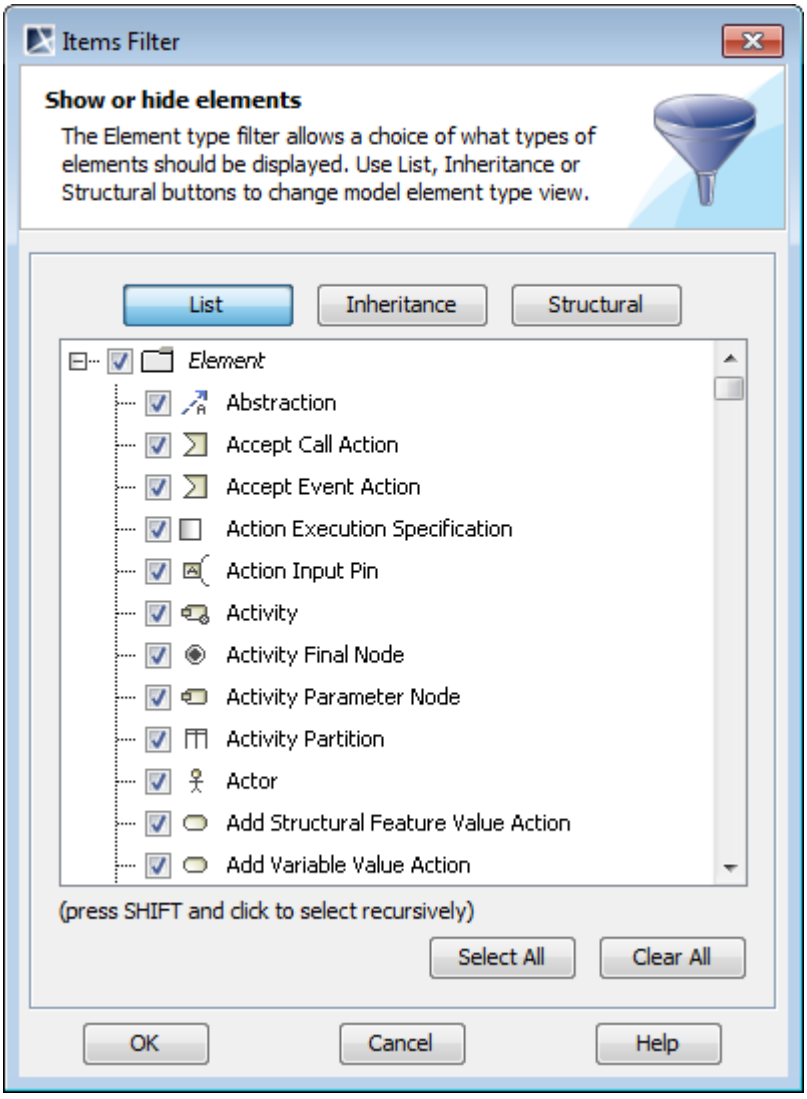
For more information about the **Containment** tab, see "Containment tab" on page 78.

Copyright © 1998-2015 No Magic, Inc.

2. Click and then select **Filter**.



The **Items Filter** dialog opens.



3. Do one of the following:
   - Click to clear the check box next to the model element type, which you do not want to be displayed in the **Containment** tab.
   - Click to select the check box next to the model element type, which you want to be displayed in the **Containment** tab.

Copyright © 1998-2015 No Magic, Inc.

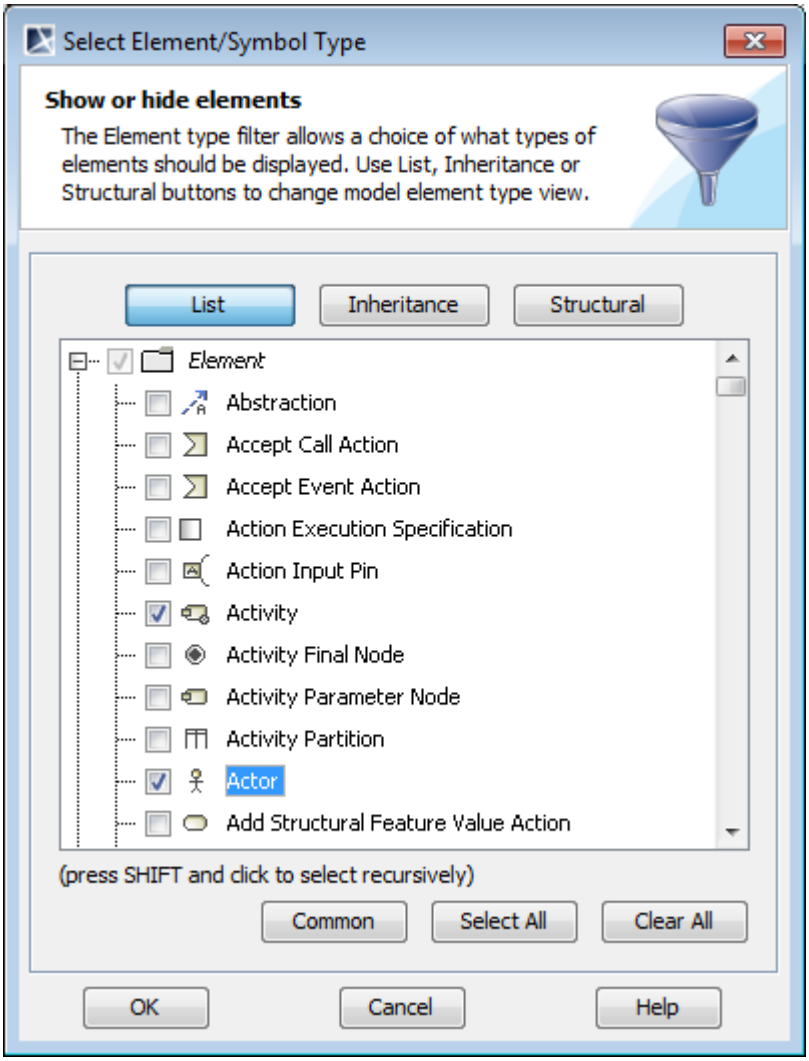
4. Click **OK**. Items in the **Containment** tab are filtered.

IMPORTANT!

The **Containment** tab all elements are displayed in hierarchy structure. That is, at the top of the hierarchy there is the root package *Data*. If you will clear the *Model* check box - no elements will be displayed in the **Containment** tab. The same is valid if you for all element types - if you will clear the check box next to type of element which has inner elements in the Model Browser, then you will not be able to see the inner elements.

To filter the types of the model elements when searching

1. From the **Edit** main menu, select the **Find** command. The **Find** dialog opens. For more information, see "Find dialog" on page 132.
2. Next to the **Type** box, click the **...** button. The **Select Element/Symbol Type** dialog opens.



3. Click to select the check box of the model element type, which you want to search for.
4. Click **OK**. The **Search Results** tab in the Model Browser opens with the search results in it. For more information about the **Search Results** tab, see "Search Results tab" on page 87.

Using the filtering dialog

In the filtering dialog you can select the check box next to the model element type, which you want to be filtered - displayed or searched for.

Copyright © 1998-2015 No Magic, Inc.

The same dialog opens when filtering items in the Containment tab and when searching for elements:

- The **Items Filter** dialog opens when filtering items in the **Containment** tab (see the following figure). The purpose of the **Items Filter** dialog is to give possibility to hide elements which you do not want to see in the Containment tab. For more information, see "To filter the types of the model elements that are displayed in the Containment tab" on page 143.
- The **Select Element / Symbol Type** dialog opens when searching for items. The purpose of the **Select Element / Symbol Type** dialog is to select the type of element for this search will be performed. For more information, see "To filter the types of the model elements when searching" on page 145.

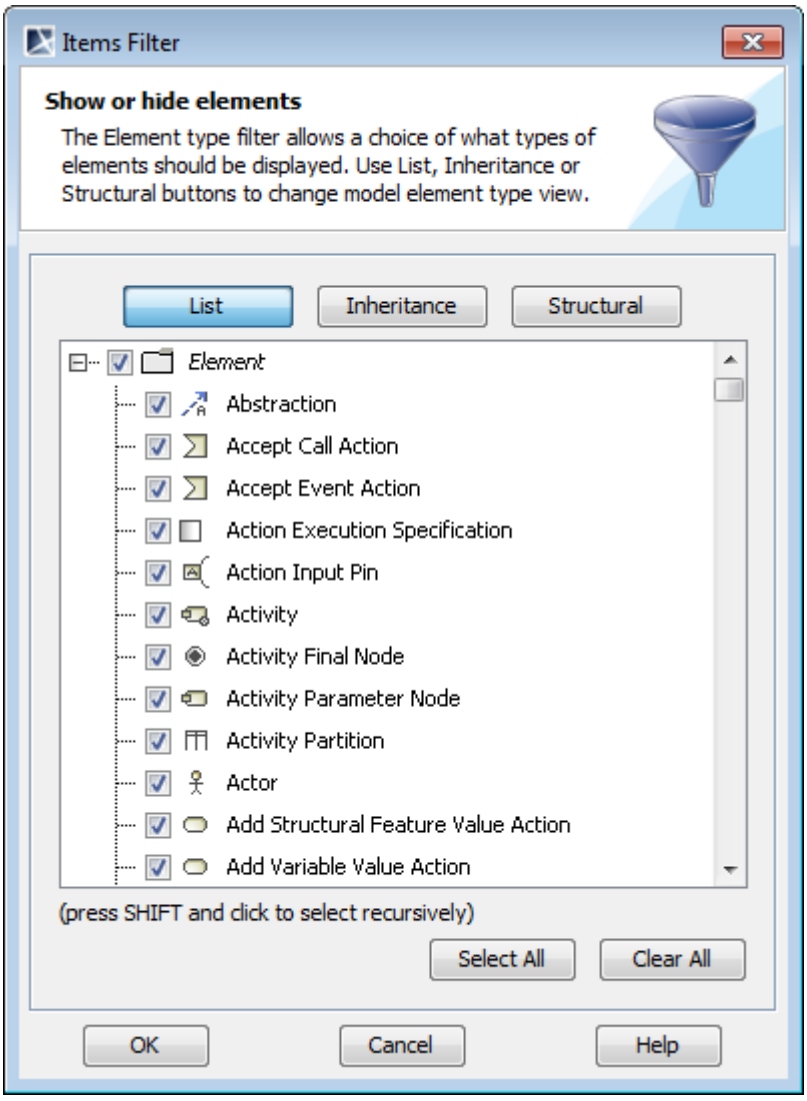


Figure 74 -- Example of filtering dialog

See the buttons of the filtering dialog described in the following table.

Button	Description
List	Click the **List** button to list the types of the elements in one list, which is listed alphabetically. There are the following two groups in the list: *Element* and *Symbol*.
Inheritance	Click the **Inheritance** button to group the types of the elements according to the inheritance to each other. For example, types of elements are grouped to the following inheritances: Named Element, Relations, and others. The Symbol group is listed also.

Copyright © 1998-2015 No Magic, Inc.

Structural	Click the **Inheritance** button to group the types of the elements according to the following groups: *DSL Customization*, *MagicDraw Profile*, *UML2 Metamodel*, *Symbol*.
Select All	Click the **Select All** button to select all types of elements in the list - all elements will be displayed when filtering will be performed or then search will be performed.
Clear All	Click the **Clear All** button to clear all types of elements in the list - no elements will be displayed when filtering will be performed or then search will be performed.

Related concepts

Searching

Project Partitioning



Shared Packages

In life, all large things are built from smaller parts. The same statement applies for large projects.

For large models having several weakly dependent parts, it is advisable to split them into several separate project files (or project resources in a collaborative environment), so called, used projects.

The project partitioning opens up the additional possibilities, such as:

- Reusing the same model part (for example, library) in several projects.
- Separate administration of different project parts. You can define different access rights in the collaborative environment for each used project. Separate (teams of) developers can be assigned for each resource, clearly establishing responsibility boundaries and preventing inadvertent modifications, when these projects are used in the read-only mode.
- Separate versioning. Each used project can have its own version history with its own tagging and branching, lifecycle of development, feature-freezes, and stable releases.
- Increased performance on very large projects. Whenever used projects are marked as loaded manually, user can selectively load just the necessary parts of the project. Or he can open the isolated used project separately.

A model can be decomposed to separate projects. A used project is a part that can be separated from the main project and can be used for a specific purpose.

MagicDraw supports the following used project accessibility modes:

- Read-only
- Read-write

The decision to use a project as read-only or read-write depends on the maturity of this project and the organization ownership or responsibility rules for the projects using other projects.

If the library in the used project is complete (changes to it are not expected / likely / possible) it should be used in read-only mode.

If a project is developed by a separate team and this team is responsible for this project and the project is reused in another project, it is recommended to use this project as read-only. This prevents inadvertent changes in the project by another team.

Copyright © 1998-2015 No Magic, Inc.

In case a project is actively developed and evolves together with the projects that are using it, this project can be used in read-write accessibility mode. In this case you must be careful and remember that your changes to the used project will be reflected in other projects. Usage of Teamwork Server might be advisable in this case. And, of course, there can be mixed usage situations - when a project is used as read-only in some projects and as read-write in others.

Related procedures

Decomposing Model
Composing Model
Managing Used Projects
Automated Module Usages
Maintaining Decomposed Model Integrity

Decomposing Model

Model decomposition has a package level granularity. Smaller elements cannot be split into separate projects. Basically, each project package could be partitioned into a separate project, however this is excessive.

The decision on how to split a model into parts should be made carefully. You should isolate model parts which form some cohesive, logically complete piece of a structure (for example, a subsystem, code library, or profile) and have light interdependencies.

When there are many one-way dependencies to some model part (for example, parts **A, B, C** depend on a part **D**, but part **D** does not depend on any of the parts **A, B, C**), this part is a good candidate for a placement into a separate project.

When one big project is used to store all the modeling information (for example, use case models, high level architectural models of the project, detailed implementation level class, sequence, state, and other information), it may be useful to partition the models according to the modeling domains (that is, use cases in one used project, architectural models in another, implementation level models in yet another). This allows unloading unnecessary used projects while working on one part or another (saving a computer and improving the performance), but still retain the relationships between domains and load used projects on demand.



Avoid decomposing a model into parts which have circular dependencies. That is, $A \leftrightarrow B$ or $A \rightarrow B \rightarrow C \rightarrow A$ situations.

Usually programmers are very adept at splitting large code bases into libraries. The very same criteria should be applied for splitting the large models into separate projects.

The MagicDraw project decomposition functionality allows two important possibilities:

- Working without all used projects loaded.
- Using the project in the read-write accessibility mode.

Used projects are often exploited for storing profiles, however a used project **is not** a profile, and it is important not to mix them. Any model part can be stored in a separate project.

Related procedures

MOF Support
Sharing project data
Indexing

Copyright © 1998-2015 No Magic, Inc.

Exporting packages to new projects

NOTE

This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

Using the **Export Packages to New Project** dialog, you can partition the model and save the content of a selected package as a separate project. Once exported, the package and its containing elements are read-only, and the project name is displayed in brackets next to the package name in the Model Browser.

To export a package as new project

1. Do one of the following:
   - From the **File** menu, select **Export To** > **Project Usage**.
   - In the Model Browser, select one or more packages you want to save as a separate project. Open its shortcut menu and click **Project Usage** > **Export Packages to New Project**.

   The **Export Packages to New Project** dialog opens.

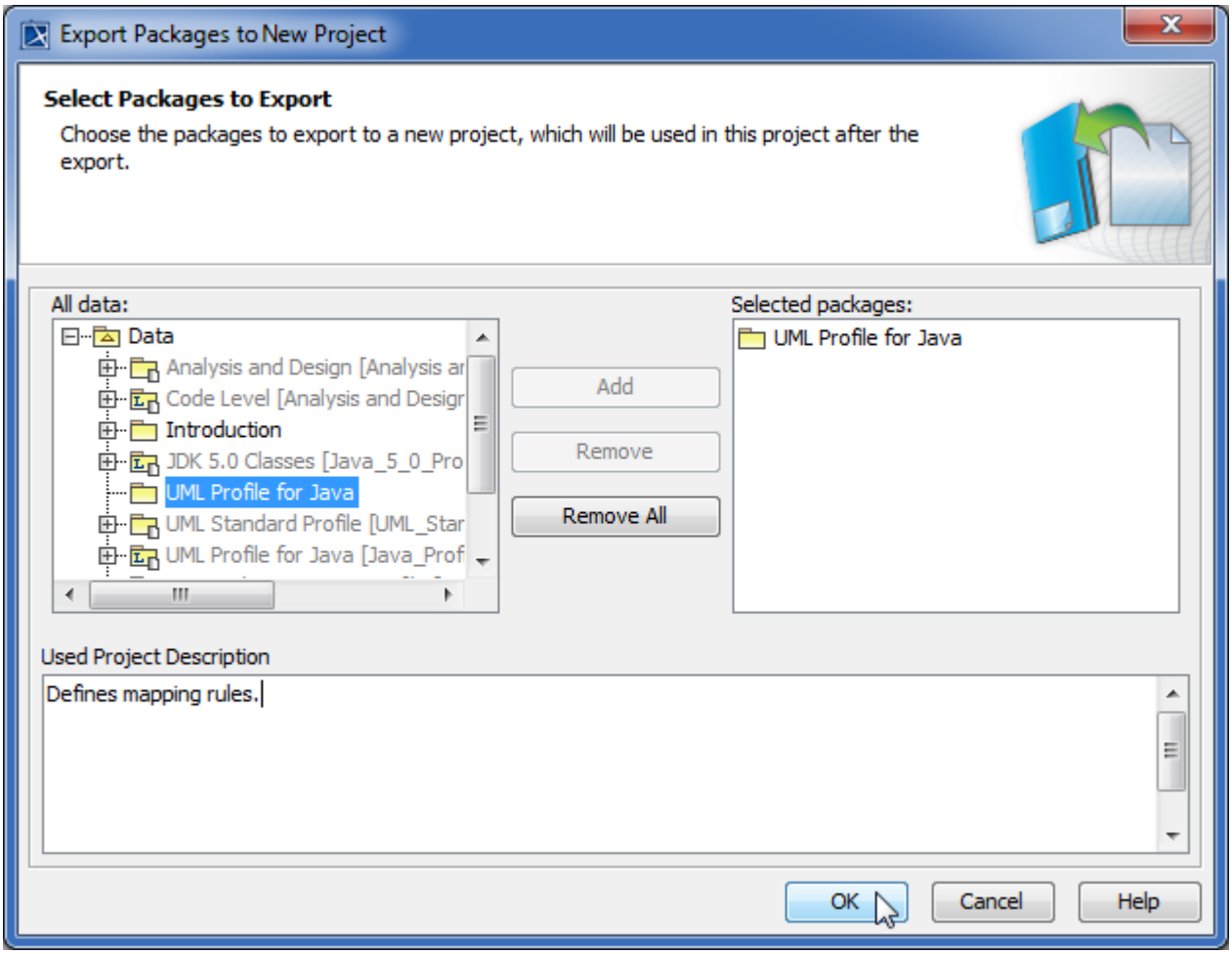


Figure 75 -- Export Packages to New Project dialog

2. In the **All data** list, select the package you want to save as a separate project and click **Add**. The package is added to the **Selected packages** list.
3. If needed, type a description of the exported packages in the **Used Project Description** box. This description is displayed in the documentation of the package.
4. Click **OK**.

Copyright © 1998-2015 No Magic, Inc.

Before exporting selected packages to a separate project, MagicDraw will check for dependencies between the model and the exported part of the model. If any error occur, you have to resolve them, otherwise recovered elements may appear in the new project. The dependency resolution process is the same as for resolution of dependencies between shared and not shared parts of the project.

When dependencies are resolved, MagicDraw will ask to specify the new project location and name.

This action can be thought as consisting of 3 elementary steps:

- Saving model elements into a separate file.
- Sharing the entire contents of this project.
- Using the project in the main project.

Alternatively, if you have several small, related projects, you can join them together into a larger, partitioned project to work with all the information in one place. This is achieved by using the **Use Project** button on the **Used Projects** dialog (to open the dialog, select **Options** > **Project Usages**).



Only packages can be exported to new projects. To export an element or diagram, you must move it to a package and only then export that package.

Sharing project data



This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

Not all contents of the used project are visible in the main project. The used project has a shared part and not shared part. Only contents of the shared part are visible in the main project. The concept “shared” is similar to the public/private attributes of a class in programming languages (for example, Java).

To designate packages of the project as shared

1. Do one of the following:
   - From the **File** menu, select **Share Packages**.
   - From the package shortcut menu, select **Project Usages** > **Share Packages**. The **Shared Packages** dialog opens.

   Use the **Add** button to select more packages for multiple simultaneous sharing, if needed.
2. Click **OK**.

The selected package including its owned elements is shared.

When the project is used in another project, all shared packages of the used project appear in that project. Moreover, each shared package can have a different location in the Containment tree. Any available package in a project can be used as a container for the shared package of a used project.

Additionally, each shared package can provide a “Preferred Path” that can be used as a hint where to mount a package when it is used in other project. No “Preferred Path” means that the package will be mounted directly under the root package *Data*. Profiles are typically mounted directly under the root package *Data*, however this can be changed.



Shared package “*util*” from the used project can be mounted on the “*com::company*” path in the main project in order to form the “*com::company::util*” path. The Preferred Path of the Shared Package can be used in this case to serve as a hint for the suggested location where to mount the package.

Copyright © 1998-2015 No Magic, Inc.

Used projects form a recursive data structure — the main project uses one or more other projects; these projects in turn can use other projects; those other projects can use another set of projects, and so on. All model pieces from these projects are gathered and connected into the integral model, which is shown in the Model Browser when the main project is opened.

Indexing

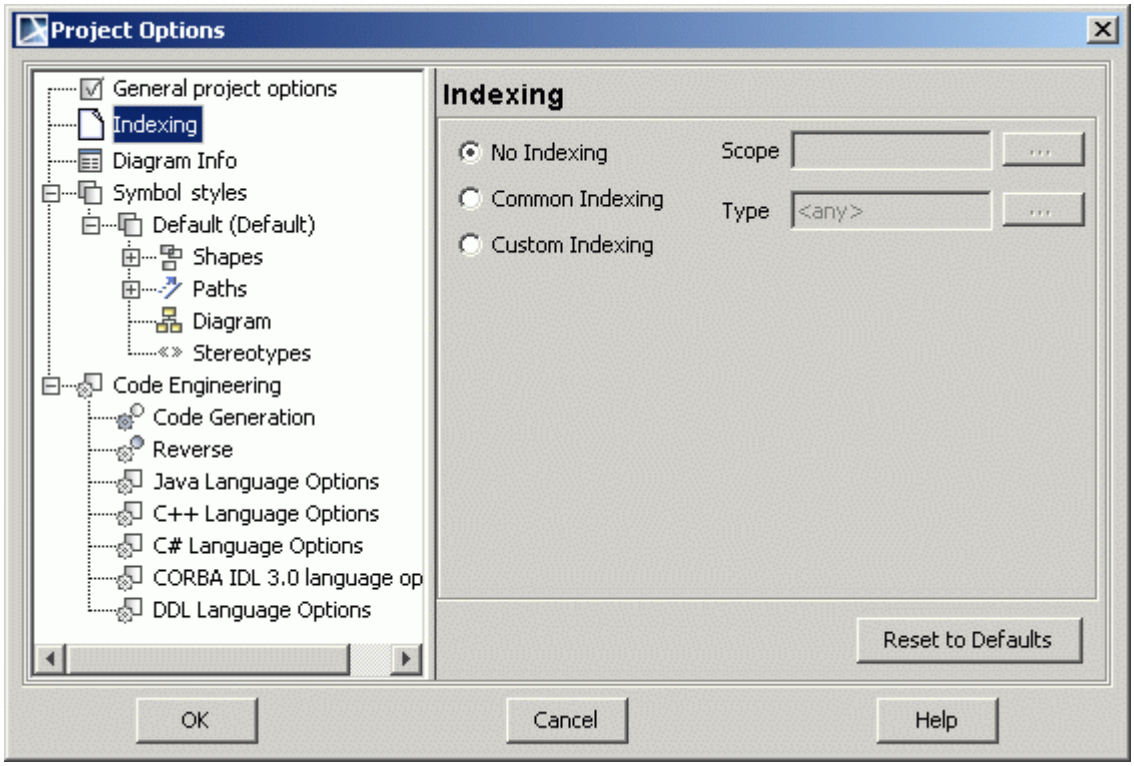
Indexing can be considered as an intermediate form of work, between working with a fully loaded used project and working with not loaded used project.

When a used project is unloaded/not loaded in the project, only necessary elements are shown in the place of the used project. However, there is a possibility to have more elements from the unloaded used project than is visible by default. Especially there is one case, where this functionality is very useful.

Consider the large software library in a project. Let's say, only the various classes are used in the main project - some library classes are set as types of properties in the model classes, some model classes inherit from the library classes, etc. In this case, structural information of the library classes (their properties and methods) is not important. If proxies of all classes could be retained when the used project is unloaded, this library could be used in the main project in the unloaded state (saving a considerable amount of computer resources). The indexing feature allows achieving this functionality.

To specify indexing scope

1. Open the used project separately from the project that uses it.
2. From the **Options** menu, select **Project**. The **Project Options** dialog opens.
3. Select the **Indexing** property group.



4. Select the **Common Indexing** option. This enables indexing of the used project and determines what information will be indexed.
5. When common indexing is chosen, classifiers and their inheritance relationships will be indexed. If you want more elements to be indexed, select the **Custom Indexing** option and fine-tune what element types (properties, methods, etc) should be indexed. The more elements you select, the more elements will be accessible in the project using them as proxies. However, your gains in performance from the used project unloading will also diminish. Hence, a balance

Copyright © 1998-2015 No Magic, Inc.

is needed when customizing the index. It is usually better to use the common indexing variant. Click **OK**.

6. In the project using the indexed project, go to the options of this used indexed project (from the main menu, select **Options** > **Project Usages**) and select the **Use Index** check box.

Such setup causes all the classes of the used project to be visible as proxies when this used project is not loaded (it is also advisable to change the loading mode of this used project to **Manual load**).

These proxies can be used as normal model elements in the project using them, without ever loading the used project. They can be set as types of properties of the classes in the main project, they can be set as an association ends, classes may be derived from them, etc. If you ever need more information from that used project, you can load it at any time to access the full data.

An example could be a project holding standard Java classes (rt.jar was reversed into it). This project is large, having all the details of standard Java classes. Many of these details are unused in the main project; frequently only class information is used in the project for modeling tasks.

Composing Model

You can compose your model by using parts of models stored in other projects.

Using other projects in a project

When a project is used in another project, shared elements of the used project become visible and accessible in the Containment tree of the project that uses them as if they were a part of the project.

To start using a project

1. From the **File** menu, select **Use Project.**
2. In the **Use Project** wizard, select a project you want to use in your project and then specify the project usage options.
3. Click **OK**.

After your project starts using another project, the elements of the used project are still stored separately:

- Used project elements — in the used project file.
- Project elements — in the project file.

The accessibility mode of the used project can be specified in the **Used Projects** dialog. By default, projects are used in the read-only mode.

To change the accessibility mode of a used project

1. From the **Options** menu, select **Project Usages**. The **Used Projects** dialog opens.
2. In the tree on the left side of the open dialog, select the used project and click the **Read-only** or **Read-write** option in the **Accessibility** area.
3. Click **OK**.

If you need to change the contents of a project used in the read-only mode, you can open it separately from the project that uses it. For this, click the **Open Project** button on the same dialog.

Related procedures

Analyzing Package Dependencies

Resolving unconfirmed module usages

Copyright © 1998-2015 No Magic, Inc.

Related references

Use Project wizard

Maintaining Decomposed Model Integrity

Related concepts

Project Partitioning

Use Project wizard

The **Use Project** wizard helps to start using one or more other projects in an open project. The wizard guides you through the following steps:

1. Step #1. Selecting a project
2. Step #2. Specifying project usage options

To open the **Use Project** wizard

Do one of the following:

- From the **File** menu, select **Use Project**.
- In the **Used Projects** dialog, click the **Use Project** button.

Copyright © 1998-2015 No Magic, Inc.

Step #1. Selecting a project

Select a project either from the file system or from the predefined location.

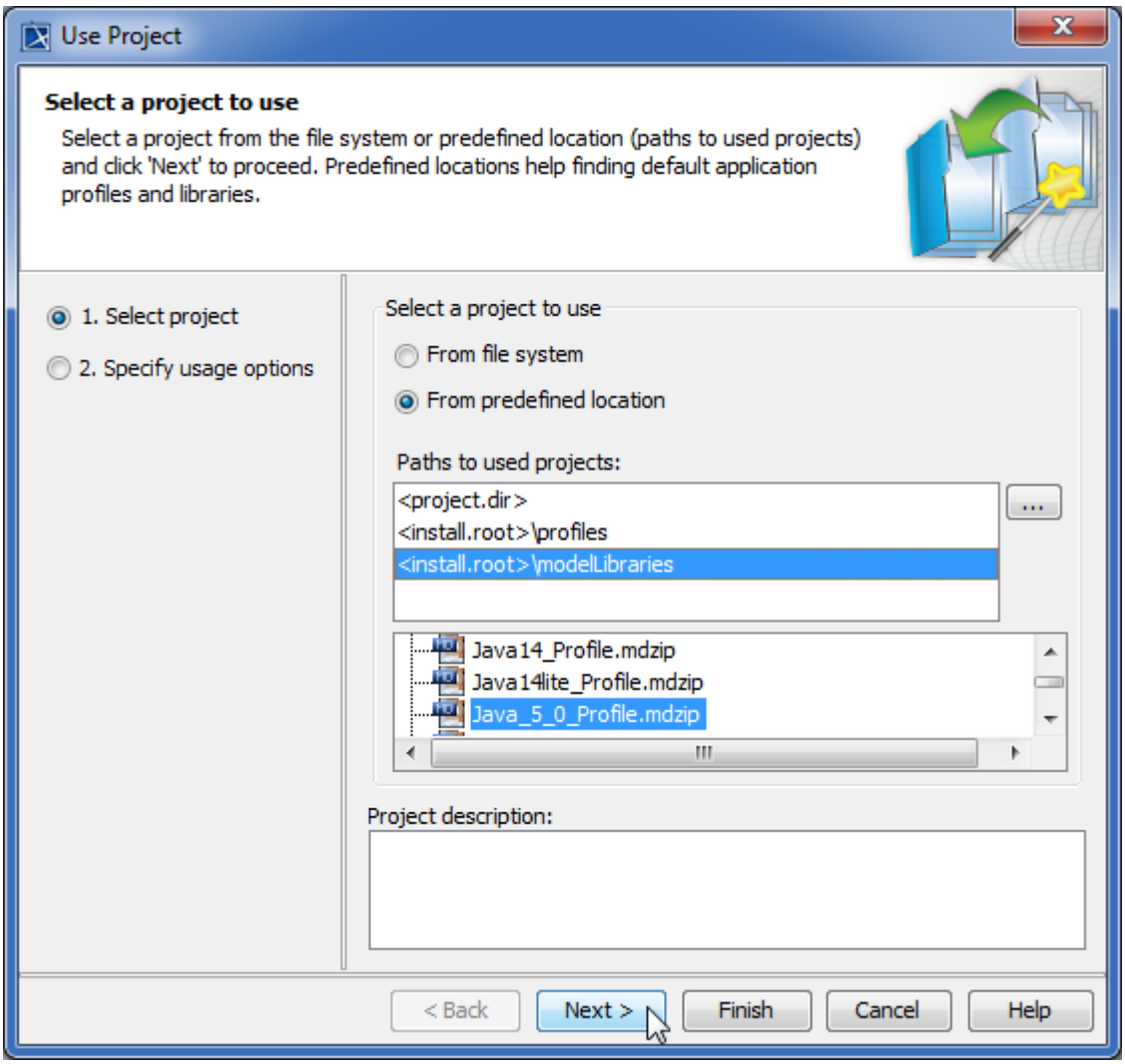


Figure 76 -- Use Project wizard. Selecting to use project from predefined location

Dialog element name	Description
From file system	Click to be able to select a project from your file system. To select the project to use, click the **...** button next to the **Project file** box.
From predefined location	Click to be able to select a project from predefined locations from the **Paths to used projects** list. Select a path from the list, and then select a project to use from the list below. To edit the list of predefined locations, click the **...** button next to the **Paths to used projects** list.
Project description	Description of the selected project.
Next	Click to open to the next step of the wizard to specify the selected project usage options.
Finish	Click to finish the wizard.
Cancel	Click to cancel the wizard without saving changes.
Help	Click to display MagicDraw Help.

Copyright © 1998-2015 No Magic, Inc.

Step #2. Specifying project usage options

Specify the selected project usage options, such as the accessibility mode, load mode, and mounting locations of shared packages.



For more information, see "Managing Used Projects" on page 155.

Related procedures

Using other projects in a project
Analyzing Package Dependencies
Resolving unconfirmed module usages

Related references

Managing Used Projects

Importing models from used projects

If you no longer need to have the set of elements in a separate project, you can import these elements into the main project.

To import a used project into the main project

1. In the Model Browser, select the package of this used project and open the shortcut menu.
2. Select **Project Usages** > **Import**.

All the elements of this used project are copied into the main project, and links to this used project are removed from the main project. The unlinked project and its elements continue to exist in the file system or server after the import.



- If a used project you want to import uses other projects, after the import these projects become directly used in the main project.
- The Standard/System Profiles cannot be imported according to our tool policy.

Importing models from other projects

Another way of composing your model, is to import a model from another project.

To import a model from another project

1. In the main menu, select **File** > **Import From** > **Another Project**.
2. Select a project you want to import.
3. Click **Import**. All elements from another project will be imported.



If a model that you are importing also uses projects, these projects will be reused directly in you project after the model import.

Managing Used Projects

You can manage usages of an open project in the **Used Projects** dialog.

To open the **Used Projects** dialog

Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

- From the **Options** menu, select **Project Usages**.
- Right-click a used project in the Containment tree and, from its shortcut menu, select **Project Usages** > **Options**.

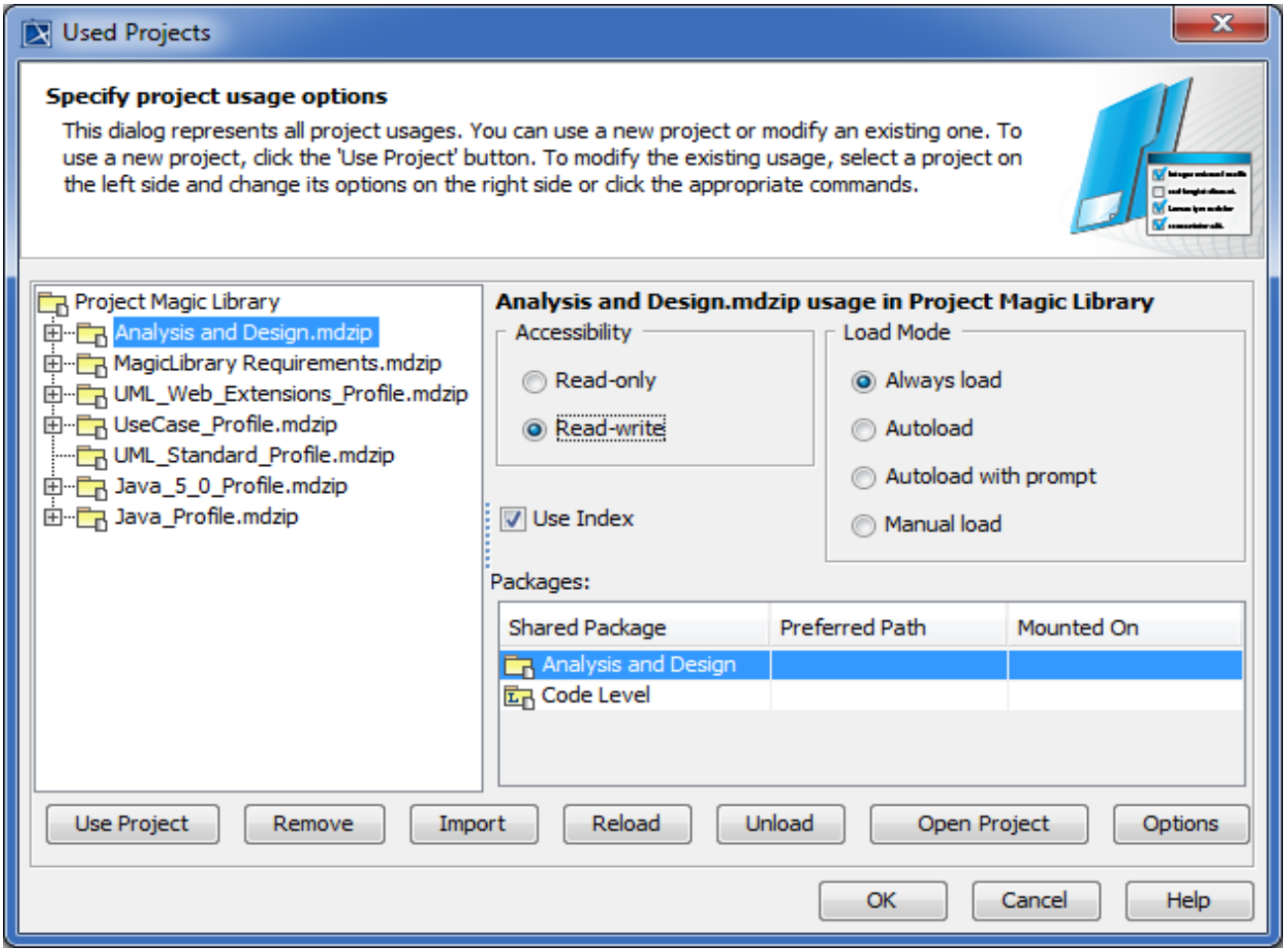


Figure 77 -- Used Projects dialog

Dialog element name	Description
Accessibility	Specifies the way a selected used project can be accessed in a project: • **Read-only**. The project is loaded for reading only and cannot be modified in the main project. Elements of this used project can be used and referenced in the main project. Usually, libraries and profiles are used in the read-only accessibility mode. • **Read-write**. The project can be edited directly in the main project. Elements of this used project can be used, referenced, and modified in the main project.

Copyright © 1998-2015 No Magic, Inc.

Dialog element name	Description
Load Mode	By selecting an appropriate load mode, you can save a memory used by a program and project: • **Always load** (default). A selected used project is always loaded when the main project is opened. • **Autoload**. A selected used project is not loaded when the main project is loaded. MagicDraw monitors user activities in the project and loads the particular used project on the demand by the project. • **Autoload with prompt**. This mode is similar to an autoload mode. The difference is that MagicDraw asks the user a confirmation before loading it. • **Manual load**. A selected used project is not loaded when the main project is loaded. The model integrity is not broken, as all required elements of the used project exists, just simplified versions (that is, loaded as proxies) of the elements are used in the project. This load mode is recommended for all used projects that are stable or rarely modified. Reload the main project after changing the load mode of the selected used project.
Use Index	Select this check box to load indexed elements from the not loaded used project. This is valid only when the used project is not loaded. Using indexed elements in the project increases the performance, as only simplified version (that is, proxies) of elements are loaded. For more information about indexing, see "Indexing" on page 151.
Packages	This table lists all shared packages of the used project and paths where these packages are mounted in the main project.
Shared Package	Name of a shared package.
Preferred Path	Suggestion where to mount the package in the main project. This location is used by default if not specified otherwise.
Mounted On	Real location where the package is currently mounted in the main project. To change a package location, click the **...** button.

Button	Description
Use Project	Click to select projects for usage in the open project and specifying the usage options. The **Use Project** wizard opens. For more information, see "Use Project wizard" on page 153.
Remove > Remove with References	Click to remove a selected used project from the project. After removal, all the elements of that used project disappear from the Containment tree, and all the references to these elements are cleared from other model places.

Copyright © 1998-2015 No Magic, Inc.

Button	Description
Remove > Remove, keep References	Click to remove a selected used project, but keep the model references across it in the model. There are two cases when you may need this kind of used project removal. **Case #1:** You may need to replace a used project with an equivalent project carrying the same elements (but perhaps with slightly different content!) and retaining all the references. In this case, removing the initially used project, but keeping references creates an automated project usage that temporarily carries the model-level references. This automated usage is invalid, since it is unconfirmed. You can confirm it by replacing the initially used project with another project (see "Resolving unconfirmed module usages" on page 165). **Case #2:** You may need to refactor projects that are created with MagicDraw earlier than 17.0.3 and prune the surplus project usages. These versions of MagicDraw have only one type of project usages, i.e., user-defined usages. Because of this, a project usually had direct user-defined usages to indirectly used projects besides the indirect usages through directly used projects. So both “project → directly used project → indirectly used project” and “project → indirectly used project” usage paths were present. In this case it may be advantageous to remove the surplus usage “project → indirectly used project” by clicking the **Remove, keep References** button.
Import	Click to import the contents of a selected used project to the main project. The **Import** button is available if the selected used project: • is not a MagicDraw native project • is locked (valid for server projects only) • is not locked by another user (valid for server projects only) The same used project can be imported only once. After the import, all the elements of the used project are copied to the main project. The inner structure of the used project is included into the project as a package. The imported project is not deleted from the file system. Standard and system profiles cannot be imported.
Reload	Click to reload the selected used project in the main project. This button is available only when the selected used project is loaded.
Reload Index	Click to reload indexed elements of the selected used project. This button is available when the selected used project is not loaded in the project.
Load	Click to load the selected used project. This button is available when the selected used project is not loaded.
Unload	Click to unload the selected used project from the main project leaving only used elements loaded. It is recommended to unload rarely used projects in order to reduce a memory usage while working with large projects. This button is available when a selected used project is loaded.
Open Project	Click to open the selected used project for editing.
Options	Click to open the **Project Options** dialog to manage styles and element display settings of the selected used project. It allows for changing element styles in the used project even without opening it.

Copyright © 1998-2015 No Magic, Inc.

Button	Description
Lock / Unlock	Click to locks the selected used project in order to modify its settings in the project or import it to the project. Lock the selected used project, if you need to modify: • Version that is used in the main project, as well as a baseline (a trunk or branch) • Accessibility mode (read-only or read-write) • Load mode (always load, autoload, autoload with prompt, or manual load) • Mount location of shared packages • Element styles Unlock the used project after you have finished editing its settings. This button is available when you are working with server projects. For more information about locking and unlocking, see "Locking model elements and diagrams for editing" on page 1055 and "Unlocking model elements and diagrams after editing" on page 1065.



For more information about working with server projects, see "Data Partitioning on Teamwork Server" on page 1100.

Related procedures

Indexing

Setting Project Options

Automated Module Usages

Related references

Use Project wizard

Reloading used projects in a project

The best way to access the latest changes in a used project is to reload it. All modifications made in the used project becomes visible after it is reloaded in the open project. Alternatively, used projects are reloaded automatically when the open project is reloaded, or the project is updated on the server.

To reload the used project

1. In the Containment tree, right-click the package of the used project.
2. From the shortcut menu, select **Project Usages** > **Reload**.



If you open the used project for editing, be sure to save any changes you have made. The modifications made appear after reloading the used project in the main project.

Copyright © 1998-2015 No Magic, Inc.

Working with partially loaded projects

To increase modeling efficiency when working with very large projects, MagicDraw allows you to work with partially loaded modules[1]. This may help to decrease the memory consumption.



For MagicDraw, “large” is tens of thousands of classes and other complex elements. If counting all small elements, such as properties, methods, method parameters, several hundreds of thousands of elements is considered a large project. Also diagrams are large elements. 1000 or more complex diagrams should be considered large.

You can unload a particular module at any moment, when working with a large partitioned project.

To unload a module from a project

- On the selected module shortcut menu, click **Modules** > **Unload Module**.

Not loaded (or unloaded) module can be loaded in a project any time. For more information about managing modules, see "Managing Used Projects" on page 155.

To load a module in a project

- From the unloaded module shortcut menu, select **Modules** and then **Load Module**.

When the module is not loaded or unloaded, there can be some model elements left from the module. These elements are not editable, and are characterized by a small M letter in the upper left corner of their icon (see the following figure).

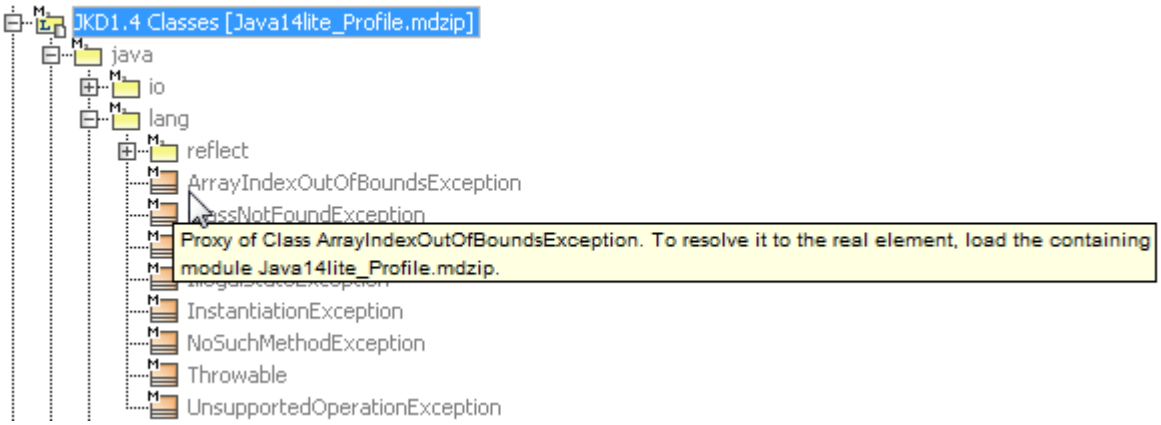


Figure 78 -- Notation of unloaded module elements (fragment of Containment tree)

These are the so-called “proxy” elements of the original elements from the module. Instead of the original model elements, the proxy is a lightweight surrogate that carries only the name and kind of the model element information. These proxies are left in the place of the original module elements, and are dedicated to maintain model integrity (so that there are no dangling ends of relationships, types of properties do not disappear, and so on.) while the module is not accessible or not loaded.

There are 4 module loading modes:

- **Always load** (default). In this mode, modules are always loaded when the project is opened. They can be unloaded if the user deems it necessary.

1. Starting from version 18.1, referred as “used project” in MagicDraw UI. This section mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

- **Autoload**. A module is not loaded when the project using it is loaded. However, MagicDraw monitors user activities in the project and tries to guess when the user might want to use the model piece from the unloaded module. For example, if the user does the search, finds usages/ dependencies, reports, metrics, transformations, or code engineering actions with a scope that touches the unloaded module, MagicDraw will load the module.
- **Autoload with prompt**. This mode differs from the *Autoload* mode in this way: MagicDraw asks the user before loading the module.
- **Manual load**. A module is not loaded when the project using it is loaded. It can be loaded, using the aforementioned **Load Module** command.

To change the module loading mode

1. From the **Options** menu, select **Modules.** The **Modules** dialog opens.
2. In the modules tree, select a module.
3. In the **Module Load Mode** area, select the appropriate loading mode.

Frequently used modules should be set in the *Always load* mode.

Occasionally used modules should be set in the *Autoload* mode (or *Autoload with prompt* if you like to have more control on the loading behavior).

Very rarely used modules, can be set in the *Manual load* mode. Another frequent case where modules can be set into *Manual load* mode is when modules represent some software library, which is not expected to change.

Related references

Maintaining Decomposed Model Integrity

Automated Module Usages

In the earlier versions of MagicDraw, there was just one type of module[2] usages. Let us call them *user-defined usages,* since they are explicitly created by the user and carry additional parameters, such as whether the module is used in the read-only or read / write mode, what are the module loading options, what module version is required, and so on.

When modifying projects with complex composition, always keep in mind the following condition:
Whenever there are model level references from elements in resource A, which can be either a module or the main project, to elements in resource B, there is a module usage A → B.

In earlier versions, you could see this rule in action when working with modules in the read / write mode. Whenever you had modified the contents of one module in such way that it started to refer to another module and then attempted to save it, MagicDraw would add an additional usage and inform you with the messages: “Module B was directly attached to A” or “Mounts were updated …” in even earlier versions.

However in some cases this system behavior caused undesirable consequences (see "Easier refactoring of composite modules" on page 162 and "Controlling dependency creation between modules" on page 163).

MagicDraw version 17.0.3 introduces a new type of module usages. Let us call them *automated module usages*. They are fully managed by MagicDraw, i.e., created, removed, and rearranged as necessary without user intervention. They are not shown in the **Modules** dialog, since only user-defined usages are shown there. The user does not have to worry about them. He / she should only care about the module usages that he / she has created explicitly himself / herself.

2. Starting from version 18.1, referred as “used project” in MagicDraw UI. This section (including subsections) mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Project Partitioning

Related procedures

Easier refactoring of composite modules

Controlling dependency creation between modules

Resolving unconfirmed module usages

Related references

Managing Used Projects

Easier refactoring of composite modules

One of the problems caused by the old approach that is described in "Automated Module Usages" on page 161 was in refactoring composite modules (modules composed of submodules). Whenever a composite module was re-arranged by removing and / or re-importing submodules, all the projects / modules using this module more often than not became affected.

In order to explain the problem explicitly, we will consider two examples.



Let's say, we have a module "ElectricComponents" with 3 submodules: "Capacitors", "Inductances", and "Resistances". We also have a project "RadioSet", which uses the module "ElectricComponents". So far so good. Now let's use several model elements from each submodule in the project "RadioSet".

Old problematic behavior
As a result, the three direct user-defined usages "RadioSet" → "Capacitors", "RadioSet" → "Inductances", "RadioSet" → "Resistances" were created, effectively "freezing" the contents of the module "ElectricComponents".

So if you tried to refactor the module "ElectricComponents" by removing its submodules and re-importing their contents, the project "RadioSet" would be affected and started complaining about the missing modules. No earlier version of MagicDraw could rearrange the user-defined usages for the fear of losing the user-specified module composition data.

New behavior
As of version 17.0.3, this is the case, when MagicDraw creates automated module usages instead of creating user-defined ones. So whenever the module "ElectricComponents" was refactored (keeping the same model elements but changing its structure), the automated usages in the project "RadioSet" would be automatically rearranged.

NOTE: Problems can now occur only in case the rearranged module does not contain all the same elements it had before the rearrangement. Thereby some automated usages may turn into the unconfirmed ones. For the information about unconfirmed usages, see "Controlling dependency creation between modules" on page 163.

Copyright © 1998-2015 No Magic, Inc.



Another related problem was “freezing” the used module versions. Let’s say, the project “RadioSet” uses the 100th version of the module “ElectricComponents”, which in turn uses 150th version of the submodule “Resistances”.

Old problematic behavior
Whenever the project “RadioSet” referenced the submodule “Resistances”, the direct usage would be created to the 150th version of it. So if the user wanted to migrate the project “RadioSet” to use the 200th version of module “ElectricComponents” which used the 300th version of the submodule “Resistances”, a version conflict would occur, since the direct usage would be still pinned to the 150th version.

New behavior
This is not a problem as of MagicDraw version 17.0.3, since automated usages does not carry information about version and does not preclude the version update through the user-defined usage paths.

NOTE: If you already have projects with complex composition (having modules/ submodules) that are prepared in any earlier version, you may want to refactor your module structure to reap the benefits of this feature.

Related concepts

Automated Module Usages

Controlling dependency creation between modules

In earlier versions of MagicDraw, you could accidentally create unneeded or incorrect (from the user standpoint) dependencies between modules in the read / write mode. The careless editing often caused cyclic module dependencies or inadvertent expansion of the scope of the other projects, when the same module was used in several projects.

With the introduction of automated module usages this problem is now addressed.

Every automatically created usage needs to be backed by some existing user-defined module usage. And if it is not so, the automated module usage needs to be explicitly confirmed by the user. While the usage is unconfirmed, it cannot cause undesired effects in other projects.

Automated module usage A → B is called *unconfirmed* when there is no user-defined usage path from A to B (neither direct user-defined usage A → B nor user-defined usage path of arbitrary length A → … → X → B). Note that user-defined usages are always confirmed.

During the ordinary course of the matters, automated usages are not visible to the user. When the user modifies the main project’s model and adds dependencies to modules / submodules, these usages are good. They are backed by the fact that the module / submodule is attached to the project in some way; hence the user-defined usage path refers from the main project to the module / submodule.

But there are situations when automated module usages are not backed and thus unconfirmed.

An unconfirmed automated module usage is considered to be invalid (not yet valid). It is just one of project composition errors, similar to recovered element errors, conflicting user-defined usage parameters, or cyclic dependencies.

Unconfirmed automated module usages are caught by special automatic validation rules and are displayed in validation results as warnings or errors depending on the project composition and prompting the user to remedy the situation: either confirm or reject the usage.

Unconfirmed usages can occur in several cases. They are described as follows.

Copyright © 1998-2015 No Magic, Inc.

Case #1: Editing a project that uses modules in read / write mode

Whenever a model part that belongs to a module with read / write permissions, is edited, an additional model-level reference can be created to some other parts of the project: either back to the project itself (cycle) or to some module.

These model-level usages may or may not be valid from the user point of view. Since the creation of usages is very easy, sometimes users create the undesired usages inadvertently. On the other hand, usage can be entirely benign. MagicDraw cannot decide whether the usage is good or bad. For example, creating references from the module “Implementation” to the module “Requirements” is valid, while creating references of the opposite direction may be invalid (since requirements usually are standalone, and implementation refers to them).

Case #2: Partial refactoring of a composite module

Reorganization or removal of submodules may affect the projects or other modules that are using the module. This happens only in case when new refactored module does not carry all the contents that the old module had.



Let’s say, we have a module “ElectricComponents” with 3 submodules: “Capacitors”, “Inductances”, and “Resistances”. We also have a project “RadioSet”, which uses the module “ElectricComponents”. So far so good. Now let’s use one model element from each submodule in the project “RadioSet”. As a result, the three automated usages “RadioSet” → “Capacitors”, “RadioSet” → “Inductances”, “RadioSet” → “Resistances” will be created.

If the module “ElectricComponents” was refactored by re-importing the submodules “Capacitors” and “Inductances” and completely separating the submodule “Resistances”, then on loading the project “RadioSet”, the automated usages “RadioSet” → “Capacitors” and “RadioSet” → “Inductances” will be automatically updated. However the usage “RadioSet” → “Resistors” will remain unaddressed. It will appear as an unconfirmed module usage, and the user will have to confirm it, i.e., explicitly attach the module “Resistances” to the project by creating the user-defined usage.

Related concepts

Automated Module Usages

Related procedures

Resolving unconfirmed module usages

Maintaining Decomposed Model Integrity

To maintain the model integrity, there are two types of elements appearing in the model:

- Proxies
- Recovered elements

A proxy is a lightweight surrogate that carries only the name and kind of the model element information. These proxies appear in the place of the original module[3] elements, and are dedicated to maintain model integrity (so that there are no dangling ends of relationships, types of properties do not disappear, and so on.) while the module is not accessible (for example, you do not have permissions) or is not loaded.

When you do not have granted “Read model” permission for a module, this module will not be loaded in a project where it is being used. To maintain the model integrity and keep model references that point to elements

3. Starting from version 18.1, referred as “used project” in MagicDraw UI. This section (including subsections) mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

of “not accessible” module, proxies are being displayed instead of referenced not accessible elements. That means that you will be able to see only the following information of these elements:

- Metaclasses of the element
- Names of the element
- Owners of the element

Recovered element is really an indication of the dangling reference. Appearance of these elements indicates that there are elements in the project (for example, elements in the main project or its used modules) that refer to the element in the module that was previously there but currently cannot be found. It may be that the element was unsafely deleted/ removed/somehow made unavailable in the module. In such case MagicDraw continue maintaining model integrity of unresolved reference by recovering missing elements - so called “Recovered element” appear in a place of the missing element where the original element existed in the past.

Related procedures

Working with partially loaded projects
Resolving unconfirmed module usages
Eliminating recovered elements from model
How to avoid triggering element recovery?

Resolving unconfirmed module usages

For the overview of automated module usages and situations when they can become unconfirmed, see "Controlling dependency creation between modules" on page 161.

Unconfirmed module usages are caught by special automatic validation rules and displayed as a warning or error validation result. The severity depends on the exact model composition situation, but the solution methods are the same in both cases.



Since problematic module usages are not model elements, they cannot be visible in the Containment tree. So when the model contains unconfirmed usages, the root package *Data* is highlighted as incorrect (regardless of which usage is actually problematic).



When there is an error level unconfirmed usage of some module, then all the elements of the module, that are referenced form other model places are shown as Recovered Elements. These recovered elements are in turn flagged as errors. Hence single unconfirmed usage can cause a lot of error validation results: one for the unconfirmed usage itself (rule abbreviation - NCAMU) and one for each recovered element (rule abbreviation - REF). In this situation the unconfirmed usage result should be examined first, because solving it may automatically solve all other results.

Copyright © 1998-2015 No Magic, Inc.

There are two ways for resolving an unconfirmed module usage. The usage can be either confirmed or rejected.

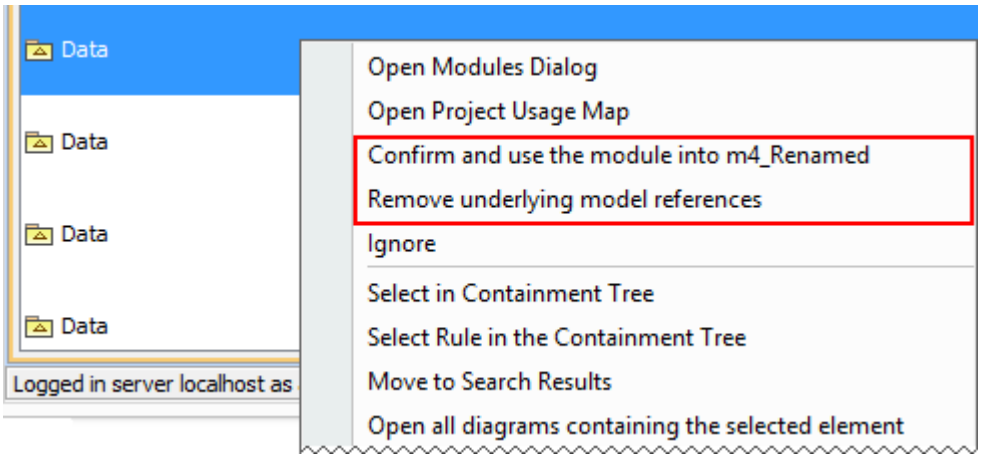


Figure 79 -- Two ways for resolving unconfirmed module usages

Confirming usages

If the module usage A → B is good and necessary according to the user policy, it can be confirmed, i.e., the user-defined usage is to be created instead of the current unconfirmed automated module usage.

You can confirm an automated usage

- By using specific command
- Manually

To confirm an automated usage by using the specific command

1. Right-click the automated usage in the validation results.
2. Select **Confirm and use the module into <module_name>** from the shortcut menu (see the preceeding figure). The **Use Module** wizard opens. The required module is already selected in the first step of the wizard.
3. Click **Finish**. The necessary user-defined module usage will be created.

To confirm an automated usage manually

NOTE

In this case you can better control the usage creation process. So if you need, for example, to use not only a required module, but some other one that brings the required module as a part of it, be advised to confirm the usage manually.

For example, let's say there are three unconfirmed usages from the project "RadioSet" to modules "Capacitors", "Inductances", and "Resistances". Instead of confirming each usage, you can create just one user-defined usage "RadioSet" → "ElectricComponents", since the module "ElectricComponents" brings in "Capacitors", "Inductances", and "Resistances" as submodules.

IMPORTANT!

However this is the case, when you must be sure that the user-defined usage will be created in the right place.

1. Click **Options** > **Modules** on the main menu. The **Modules** dialog opens.
2. Select a module that is the usage target resource.
3. Click the **Use Module** button. The **Use Module** wizard opens.
4. Follow the steps of the usual procedure (select a required module and finish or continue to the second wizard step to provide more configuration options for module usage and then finish).

Copyright © 1998-2015 No Magic, Inc.

Rejecting usages

If the usage A → B is not good according to the user policy, it must be rejected and removed.

You can reject an automated usage

- By using specific command (for removing underlying model level references)
- Manually (by removing or redirecting underlying model level references to different elements)

To reject an automated usage by using the specific command



Use this way to reject a model-level usage that, for example, was created inadvertently.

1. Right-click the automated usage in the validation results.
2. Select **Remove underlying model references** from the shortcut menu (see the preceding figure). All underlying model-level references (the ones that cause this automated usage) will be removed.



This command is very similar to the **Clear Recovered Element Usages** command.

To remove an automated usage manually

In more complex cases you may want to address each usage individually. Use MagicDraw dependency analysis means to examine what dependencies are there and review them on a case-by-case basis (see "Analyzing Usages and Dependencies" on page 509, "Analyzing Package Dependencies" on page 604, "Displaying Related Elements" on page 508, "Dependency Matrix" on page 764, and so forth).

Useful commands for preparing to resolve unconfirmed module usages

When you need to explore your project composition in detail before attempting to resolve unconfirmed usages, use the following commands for navigating:

- **Open Modules Dialog** (see "Using other projects in a project" on page 152)
- **Open Project Usage Map** for collaboration projects (see "Project Usage Map" on page 1104)

Related concepts

Controlling dependency creation between modules

Related procedures

Analyzing Usages and Dependencies

Analyzing Package Dependencies

Related references

Using other projects in a project

Use Project wizard

Project Usage Map

Recovered elements representation in projects

Recovered element icons and symbols differ from regular element icons and symbols. The top-left corner of a recovered element icon as well as the same corner of its symbol is decorated with “R”. A ToolTip, which

Copyright © 1998-2015 No Magic, Inc.

appears on the screen while pointing to a recovered element icon or symbol, uncovers the possible reasons of element recovery.

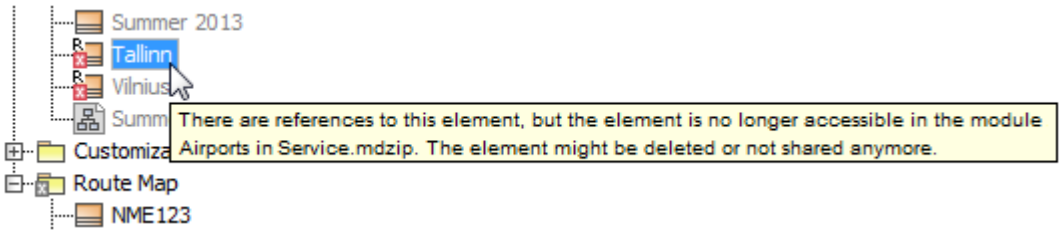


Figure 80 -- Recovered element representation in Model Browser

As recovered elements are captured by MagicDraw composition inspection, they are also decorated as composition inspection violations (see a small cross sign at the bottom-left corner of a recovered element icon in the preceding figure).

TIP!

For more about marking active validation violations, see

- "Marking invalid elements in the Model Browser" on page 633
- "Highlighting invalid elements on the diagram" on page 634

In the **Active Validation Results** panel, recovered elements are grouped under the **Composition Integrity** package. The text in the **Message** column explains the reason of each element recovery (this is the same information as is shown in the Tooltip that was described earlier in this section).

IMPORTANT!

The abbreviation of recovered element violation is REF.

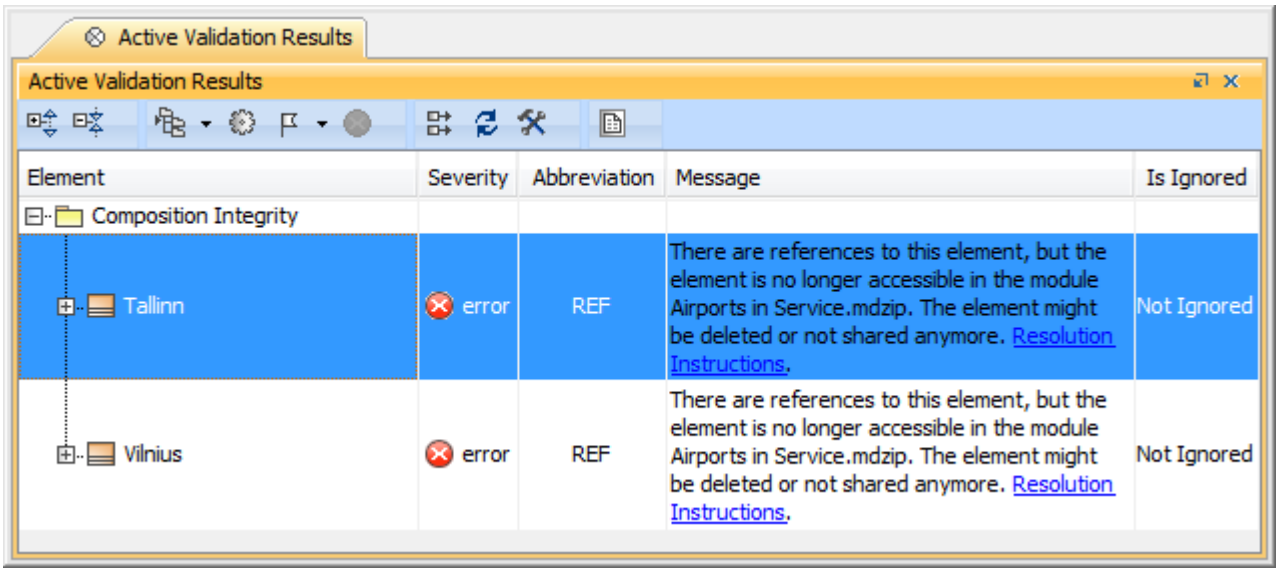


Figure 81 -- List of recovered elements in Active Validation Results panel

To open the **Active Validation Results** panel

1. Right-click a recovered element in the Model Browser.
2. From its shortcut menu, select **Validation** > **Recovered Element** > **Select in Validation Results**.

TIP!

For more about opening the **Active Validation Results** panel, see "Active Validation" on page 632.

Copyright © 1998-2015 No Magic, Inc.

Eliminating recovered elements from model

Recovered elements indicate that there are unresolved references in your project.

An unresolved reference means that your project contains an element that continues referring to some module element, which became inaccessible, because, for example, the element is no longer shared, the element was removed or moved to a non-shared location, and so on.

To maintain your model integrity and project composition, when element is inaccessible, MagicDraw recovers the inaccessible elements. This allows for maintaining in your project references that are indicated as unresolved.

It is highly recommended to eliminate recovered elements from your model. This can be done in one of the following ways:

- **By eliminating the reasons of the element recovery.** This is the case when recovered elements still exist. Find the reason of the element recovery and eliminate it, for example, share the package that contains the recovered element, move back the element from a non-shared location to the shared one, and so on.
- **By restoring recovered elements.** This is the case when recovered elements no longer exist, but can be restored from the bits of information MagicDraw still has. The restored element will have only ID, name, and type.



For more information, refer to "Restoring recovered elements" on page 169.

- **By managing the unresolved references that refer to the appropriate recovered elements.** This is the case, when recovered elements no longer exist and cannot be restored.



For more information, refer to "Managing unresolved references" on page 170.

Restoring recovered elements

To restore a recovered element

Do either:

- In the Model Browser, right-click the recovered element and from its shortcut menu select **Validation** > **Recovered Element** > **Restore Element**.
- In the **Active Validation Results** panel, right-click the recovered element and from its shortcut menu select **Restore Element**.

MagicDraw will restore the missing element from the bits of information it still has, that is, ID, name, and type of the element.



If you do not see the **Restore Element** command on the recovered element's shortcut menu, please check the accessibility mode of the module wherein the recovered element is stored. It might be that the module is being used in the read-only mode. Change the mode to read-write in this case.

Copyright © 1998-2015 No Magic, Inc.

Managing unresolved references

You can manage the unresolved references either in the Model Browser or in the **Active Validation Results** panel wherein they are grouped by recovered elements they refer to.

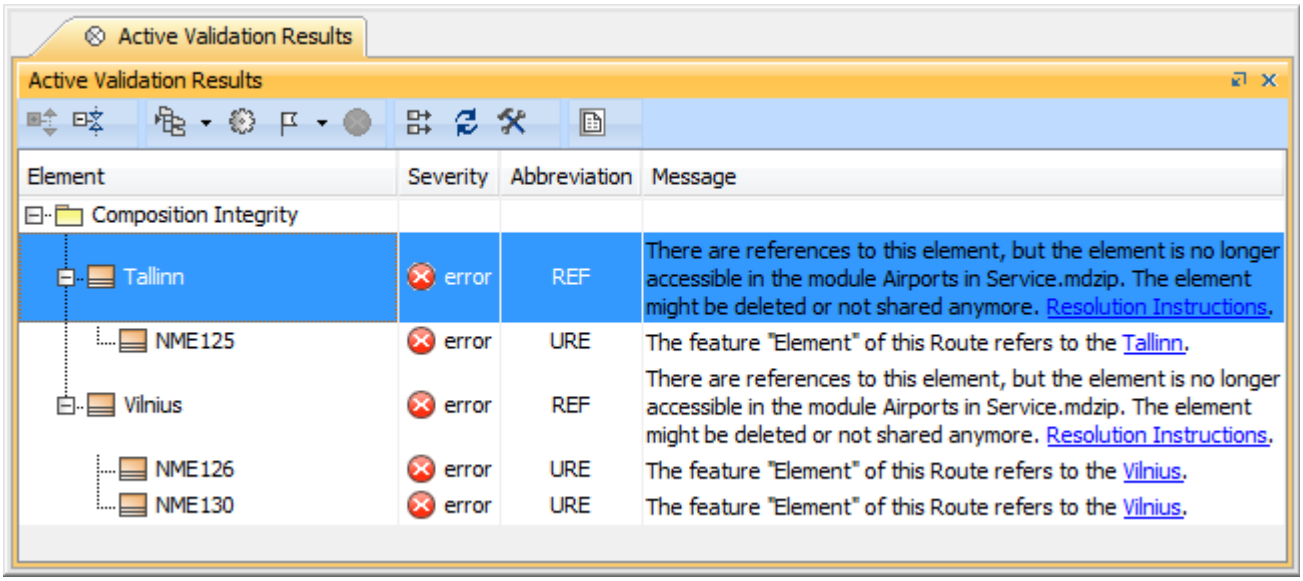


Figure 82 -- Representation of unresolved references grouped by recovered elements in Active Validation Results panel

These are the ways of managing unresolved references by using specific commands:

- Replacing recovered element with another element (described on page 170)
- Removing unresolved reference (described on page 171)
- Removing recovered element's symbol (described on page 171)

Replacing recovered element with another element

This is the case, when the user decides to refer another regular element instead of the recovered one in the model.

To replace a reference from a recovered element to another one

1. Do either:
   - In the Model Browser, right-click an element that refers to the recovered element and from its shortcut menu select **Validation** > **Usage of Recovered Element** > **Use Another Element**.
   - In the **Active Validation Results** panel, right-click an element that refers to the recovered element and whose abbreviation is URE and from its shortcut menu select **Use Another Element**.
2. In the element Selection dialog, select an element you want to use instead of the recovered element.
3. Click **OK**. The reference is now pointing to the selected element. And it is no longer unresolved.



You can use the command for more than one element at a time.



If some of the unresolved references are read-only, belong to the read-only module, are locked by another user, or still not locked, the command will skip these elements.

Alternatively you can replace all references from a particular recovered element to another one using one operation.

Copyright © 1998-2015 No Magic, Inc.

To replace a recovered element with another one in all references

1. Do either:
   - In the Model Browser, right-click the recovered element and from its shortcut menu select **Validation** > **Usage of Recovered Element** > **Change Usages To**.
   - In the **Active Validation Results** panel, right-click the recovered element and from its shortcut menu select **Change Usages To**.
2. In the element Selection dialog, select an element you want to use instead of the recovered element.
3. Click **OK**. All the references are now pointing to another element. And they are no longer decorated as unresolved.

Removing unresolved reference

This is the case, when a reference to some recovered element is no longer needed and can be removed.

To remove a reference to the recovered element

Do either:

- In the Model Browser, right-click an element that refers to the recovered element and from its shortcut menu select **Validation** > **Usage of Recovered Element** > **Clean Usage of Recovered Element**.
- In the **Active Validation Results** panel, right-click an element that refers to the recovered element and whose abbreviation is URE and from its shortcut menu select **Clean Usage of Recovered Element**.

The reference to the particular recovered element is removed.



You can use the command for more than one element at a time.



If some of the referring elements are non-editable, belong to the read-only module, are locked by another user, or still not locked, the command will skip these elements.

You can remove all unresolved references (including all used modules) of a recovered element in one operation.

To remove all references to a recovered element

Do either:

- In the Model Browser, right-click the recovered element and from its shortcut menu select **Validation** > **Usage of Recovered Element** > **Clean Usage of Recovered Element**.
- In the **Active Validation Results** panel, right-click the recovered element and from its shortcut menu select **Clean Usage of Recovered Element**.

All unresolved references to the particular recovered element are removed. As the unresolved references are cleared, the recovered element is removed as well.



Unresolved references will not be removed from the referring elements, if they are stored in a read-only module, are locked by another user, or still not locked.

Removing recovered element's symbol

This is the case, when a recovered element is depicted on diagrams of your project.

Copyright © 1998-2015 No Magic, Inc.

To remove a recovered element's symbol

Do either:

- On a diagram pane, select the symbol, then on its smart manipulator toolbar click  and then select **Symbol of Recovered Element** > **Remove Symbol** as is shown in the following figure.
- On the diagram pane, select the symbol and press DELETE.
- In the **Active Validation Results** panel, right-click the symbol and select **Remove Symbol** from its shortcut menu.

The symbol of the recovered element is removed from the diagram.



The same diagram can have many symbols of the recovered element. To remove all of them, repeat the same procedure for each symbol.



You can use the **Remove Symbol** command for more than one symbol at a time.

If the diagram with recovered element symbols is non-editable, belong to the read-only module, is locked by another user, or still not locked, the command will not be applied.

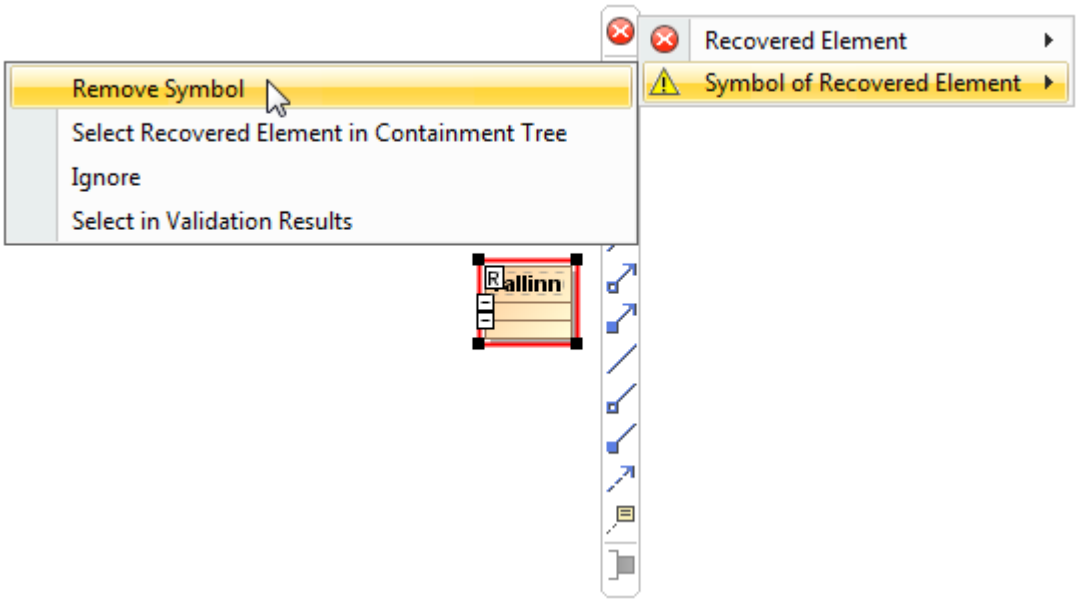


Figure 83 -- Using smart manipulator toolbar to remove recovered element's symbol

How to avoid triggering element recovery?

To learn how to avoid triggering element recovery in your project, read the following recommendations:

1. Avoid unexpected modifications of shared elements (for example, removing a shared element or moving the element to a non-shared location, etc.).
2. Configure MagicDraw so that it prompts your confirmation to proceed a potentially dangerous action (for example, attempting to stop sharing a package, removing the content of a shared package, or moving elements from one module to another, when both modules are used in the read-write mode as is shown in the following figure).
3. Select the relevant level of model integrity and project composition inspection.

Copyright © 1998-2015 No Magic, Inc.

4. Manage unresolved references as soon as possible, since they might produce another unresolved references. Remember that the less recovered elements your model has, the more valid it is.

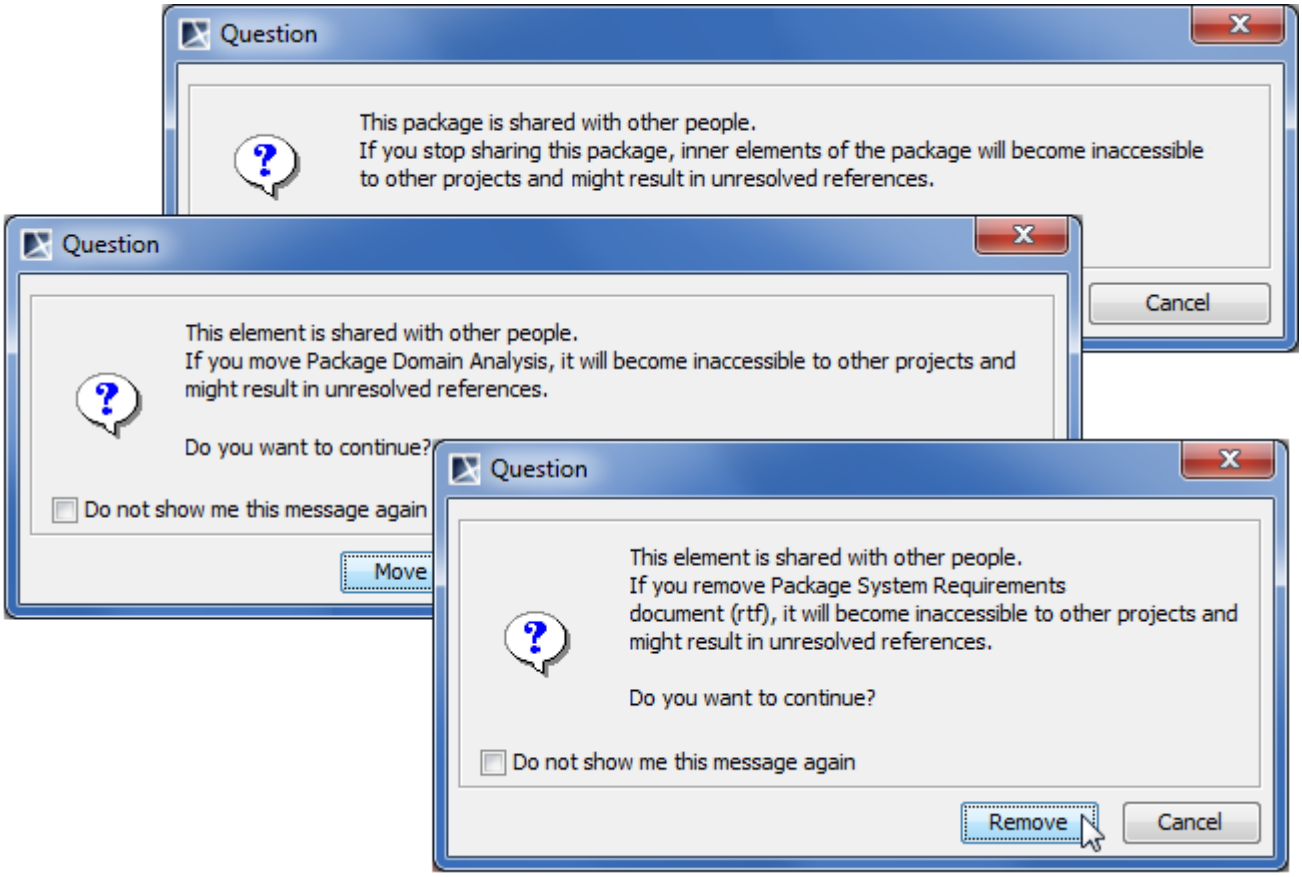


Figure 84 -- Examples of messages prompting to confirm potentially unsafe actions

To define a set of actions for MagicDraw supervision to prevent recovered elements creation

1. Select **Options** > **Environment** to open the **Environment Options** dialog.
2. Select the **Composition Inspection** options group on the left.
3. To define which actions should be supervised, select appropriate check boxes.

Copyright © 1998-2015 No Magic, Inc.

4. Click **OK** when you are done.

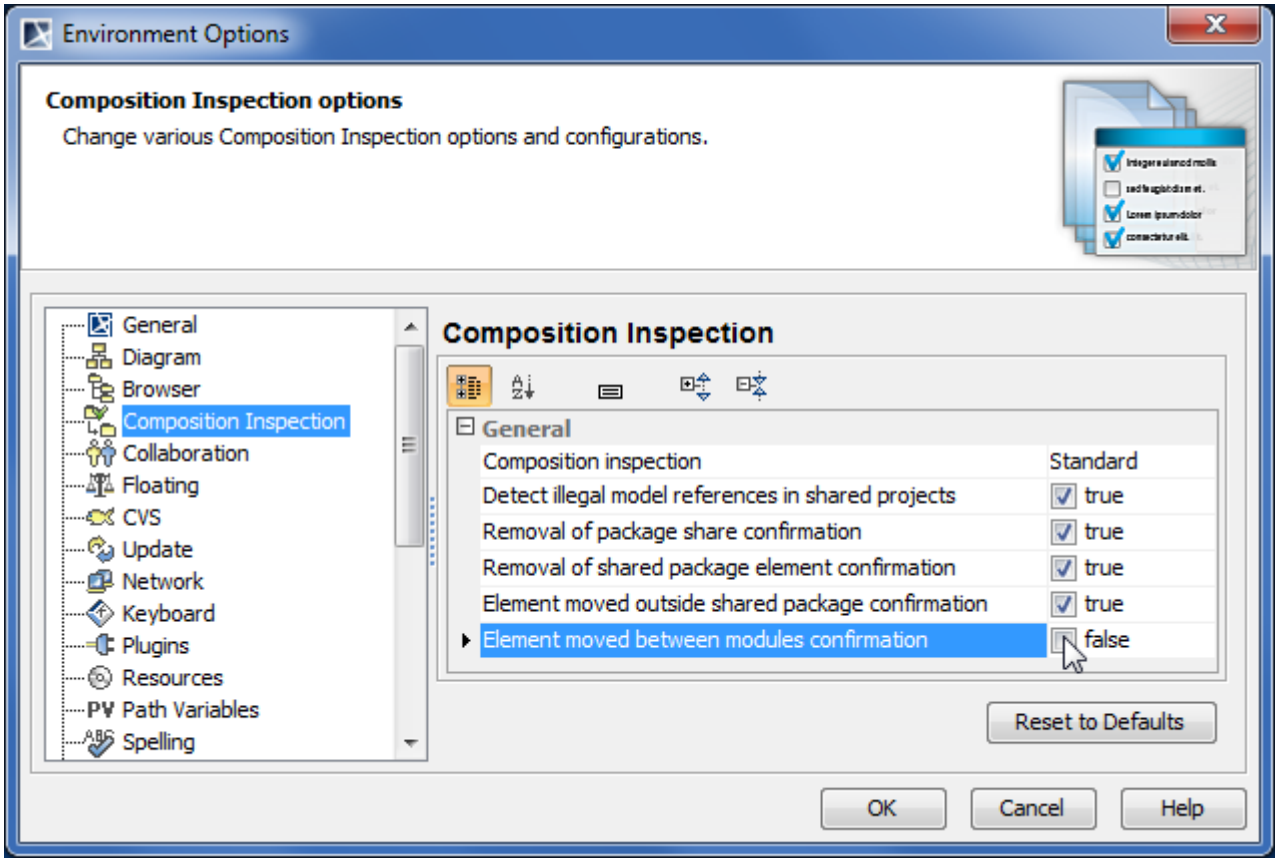


Figure 85 -- Selecting actions for MagicDraw supervision

To select the relevant level of project composition inspection

1. Select **Options** > **Environment** to open the **Environment Options** dialog.
2. Select the **Composition Inspection** options group on the left.
3. Specify the **Composition inspection** option value:
   - Select **Standard**, if you need the active validation to search for issues only in the project, but not in the modules it uses.
   - Select **Advanced**, if you need the active validation to search for issues in the project as well as in the modules it uses.
4. Click **OK** when you are done.

Copyright © 1998-2015 No Magic, Inc.

Validating Project Integrity

Use the Project Integrity Correctness suite to find integrity issues in your project composition.

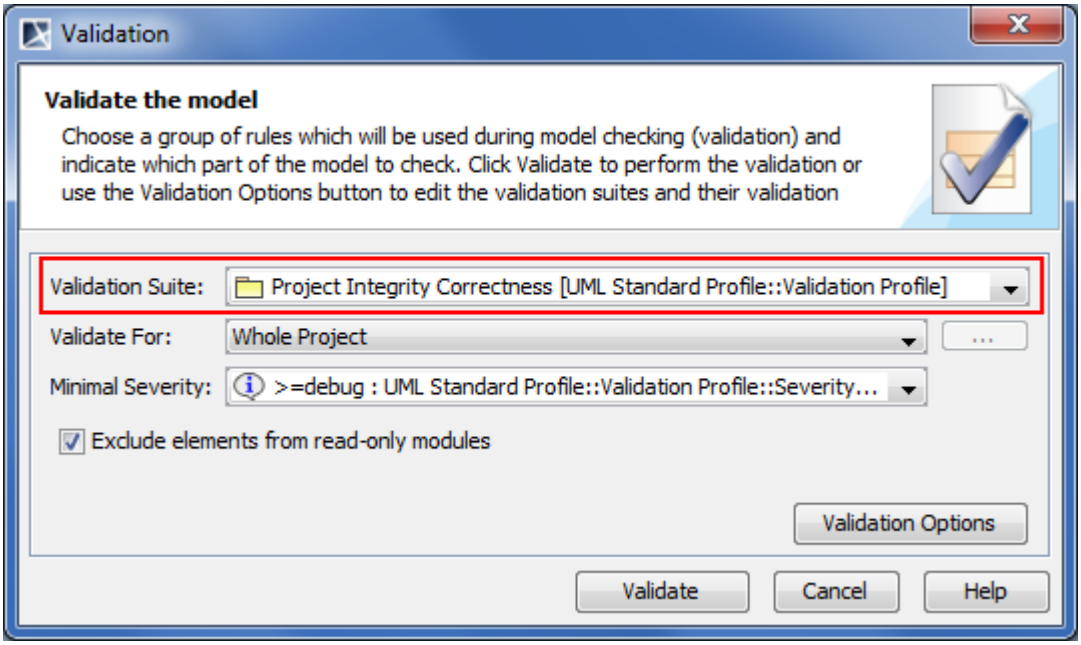


Figure 86 -- Project Integrity Correctness suite

Violations of your project integrity will be displayed in the **Validation Results** panel.

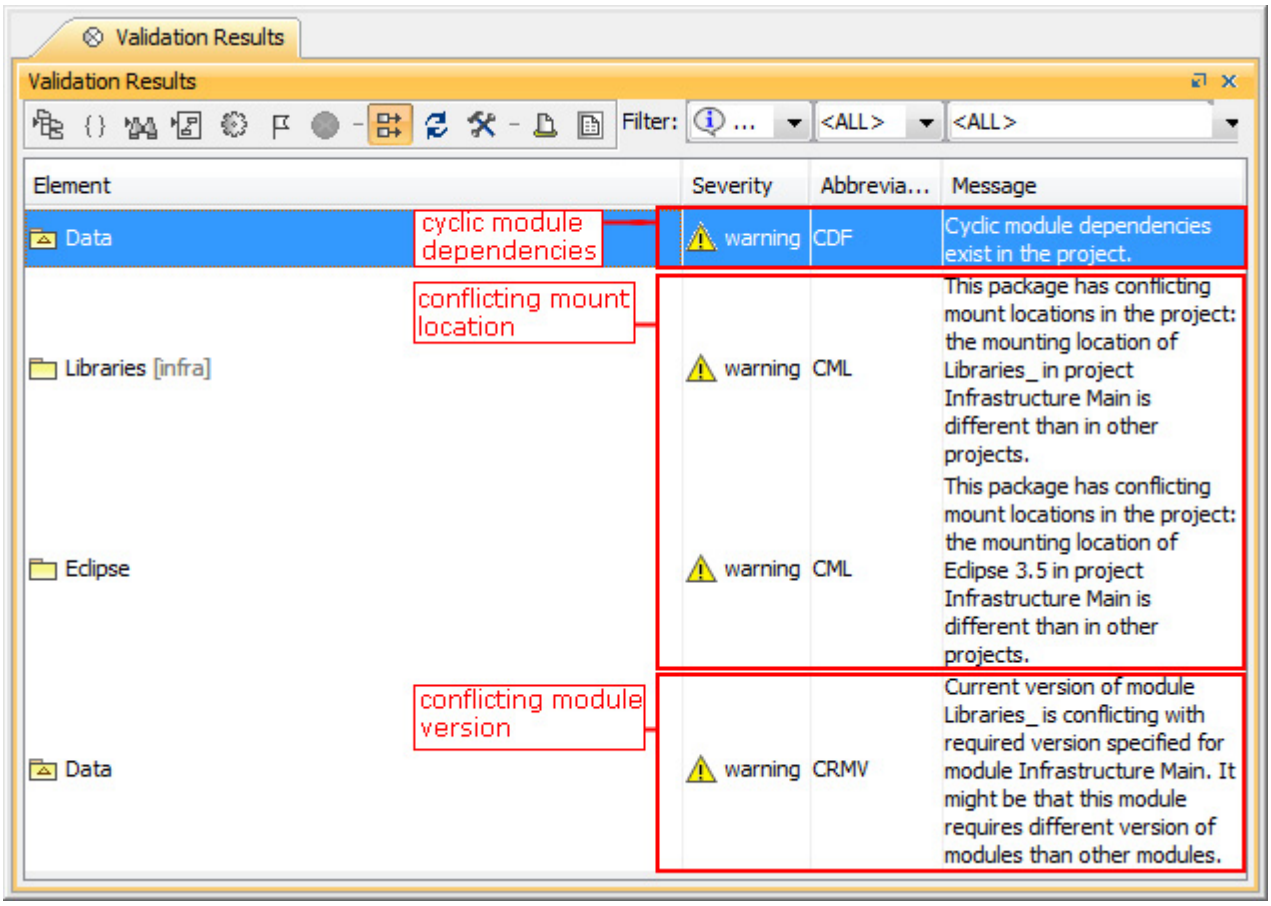


Figure 87 -- Example of violations detected while checking project integrity correctness

Copyright © 1998-2015 No Magic, Inc.



- For more information on how to run the model validation, see "Validating" on page 614.
- For more information about the **Validation Results** panel, see "Validation Results Panel" on page 616.

The Project Integrity Correctness suite includes the rules for **detecting** the following project integrity violations:

- Cyclic dependencies among modules
- Mount location inconsistencies in used modules
- Required version inconsistencies in used modules

Cyclic dependencies among modules

You can resolve the conflict either in the **Modules**[4] dialog (see "Managing Used Projects" on page 155) or by using the project usage map (see "Project Usage Map" on page 1104).

Mount location inconsistencies in used modules

This is the case, when packages of the module have different mount locations defined in separate projects.



For example, the module “Licensing_” has mounted a package of the module “Libraries” under the package “Infra”, while the project “Project Infrastructure Main” has mounted the same package “Libraries” under the root package *Data*. The package “Libraries” belongs to the module “Libraries_”.

As it is shown in the following figure, the mount location conflict occurs because the project “Project Infrastructure Main” is using “Licensing_” and “Libraries_” modules at the same time.

4. Starting from version 18.1, referred as “used project” in MagicDraw UI. This section mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

You can solve the conflict in the **Modules** dialog by unifying the **Mounted On** value in both projects (see "Managing Used Projects" on page 155).

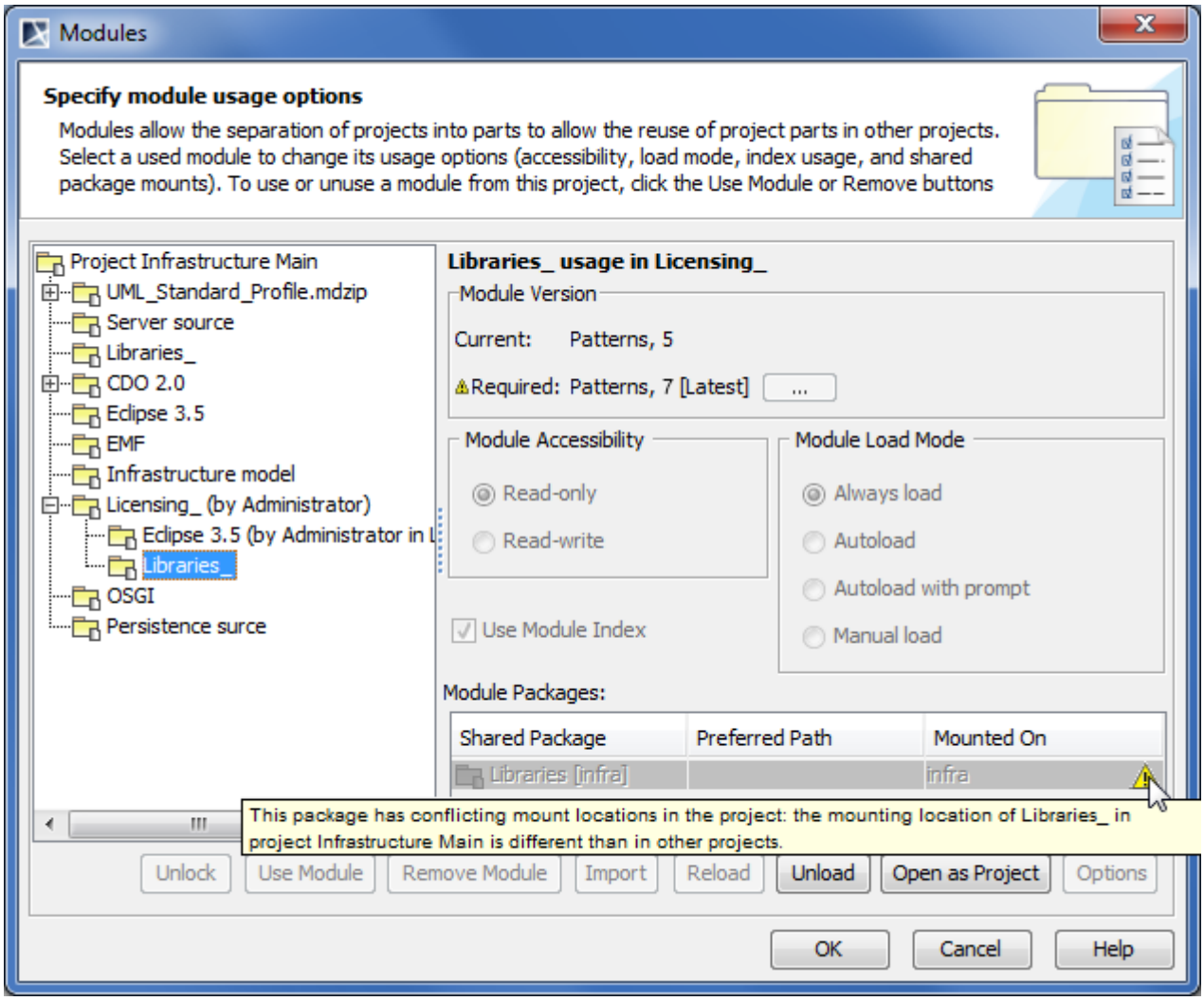


Figure 88 -- Example of conflicting mount locations warning displayed in Modules dialog

Required version inconsistencies in used modules

This is the case, when a project uses (both directly and indirectly) several different versions of the same module at the same time.

EXAMPLE

For example, the module “Licensing_” is using the 7th version of the module “Libraries_” and the project “Infrastructure Main” is using the 5th version of the module “Libraries_”.

If the project “Infrastructure Main” additionally used the module “Licensing_”, a module version conflict would occur (see the following figure).

Since the project used the 5th version of the module directly and the 7th version of the module via the module “Licencing_”, two different versions of the module would be required at the same time.

Copyright © 1998-2015 No Magic, Inc.

You can solve the conflict in the **Modules** dialog by unifying the required versions in both projects (see "Managing Used Projects" on page 155).

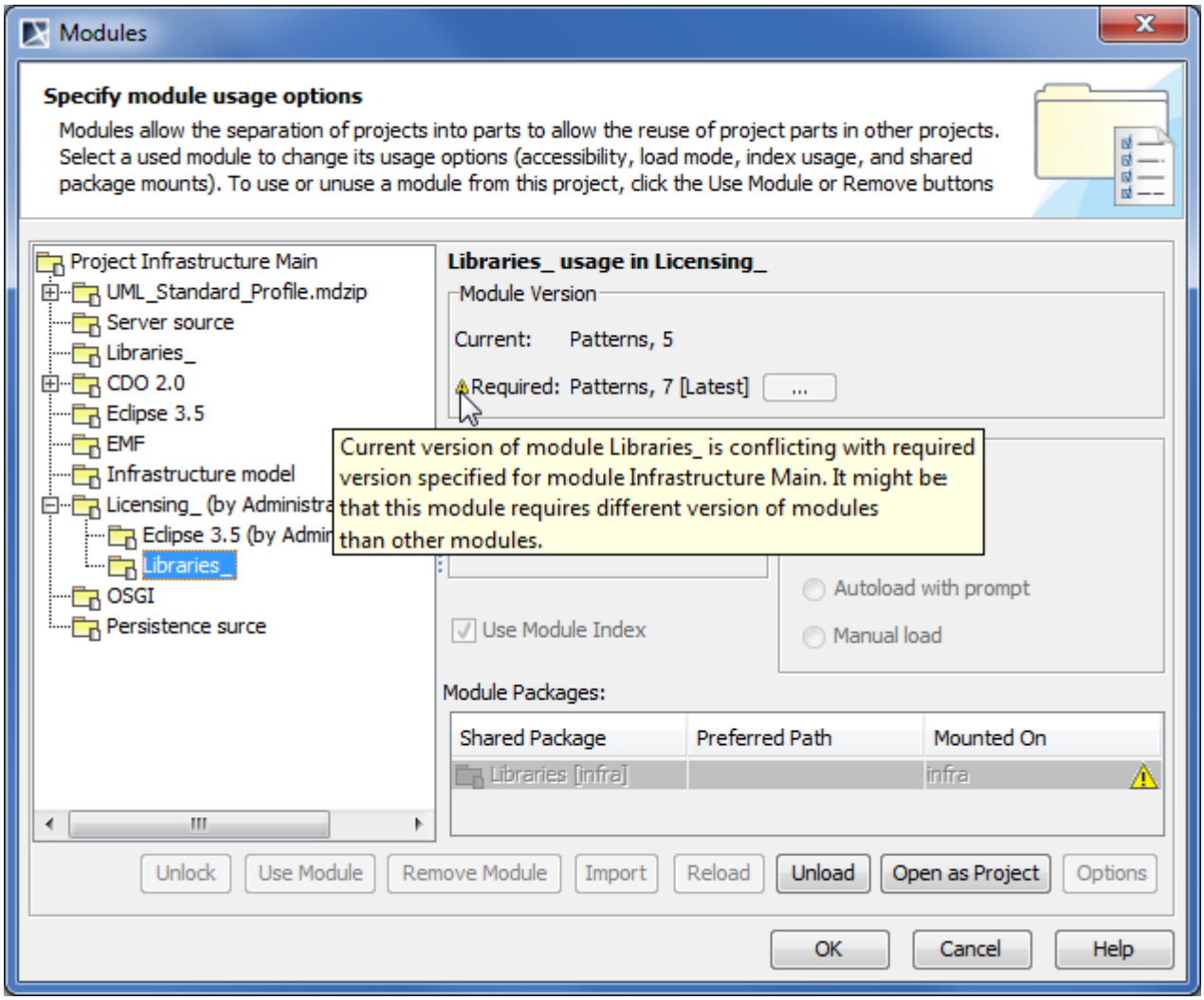


Figure 89 -- Example of conflicting versions warning displayed in Modules dialog

Related references

Validation
Validation Results Panel
Project Usage Map
Managing Used Projects

MOF Support

MagicDraw is able to export / import the UML model into / from the MOF (both CMOF and EMOF) XMI file. MOF 2.0 and 2.4 (MOF 2.4.1 to be more specific) are supported.

For more information about the MOF domain model, see Meta Object Facility (MOF) Core Specification at http://www.omg.org/spec/MOF/2.0/PDF/ and http://www.omg.org/spec/MOF/2.4.1/PDF/.

This section contains the following subsections:

- "Exporting projects to MOF files" on page 179.

Copyright © 1998-2015 No Magic, Inc.

- "Importing projects from MOF files" on page 180.

Exporting projects to MOF files

MagicDraw allows for exporting a selected project either to the EMOF or the CMOF package. You can choose to export either the whole project or selected packages only. The model, all except auxiliary resources (for example, the UML Standard Profile package), will be exported to the selected MOF file.

IMPORTANT! Diagram data and model features available only in UML (behavioral models in particular) cannot be exported.

To export a project to a MOF file

1. From the main menu, select **File** > **Export To** > **MOF XMI File** > **MOF Whole Model**. The **Select Target File** dialog opens (see the following figure).
2. Type a file name and select a location for the exported model.
3. On the right side of the dialog under **MOF Kind**, select a MOF kind. The file type of the exported model changes according to your selection.
4. Click to select or to clear the **Validate** check box. For more information about validating refer to "Exported elements validation" on page 180.
5. Click the **Export** button when you are finished.

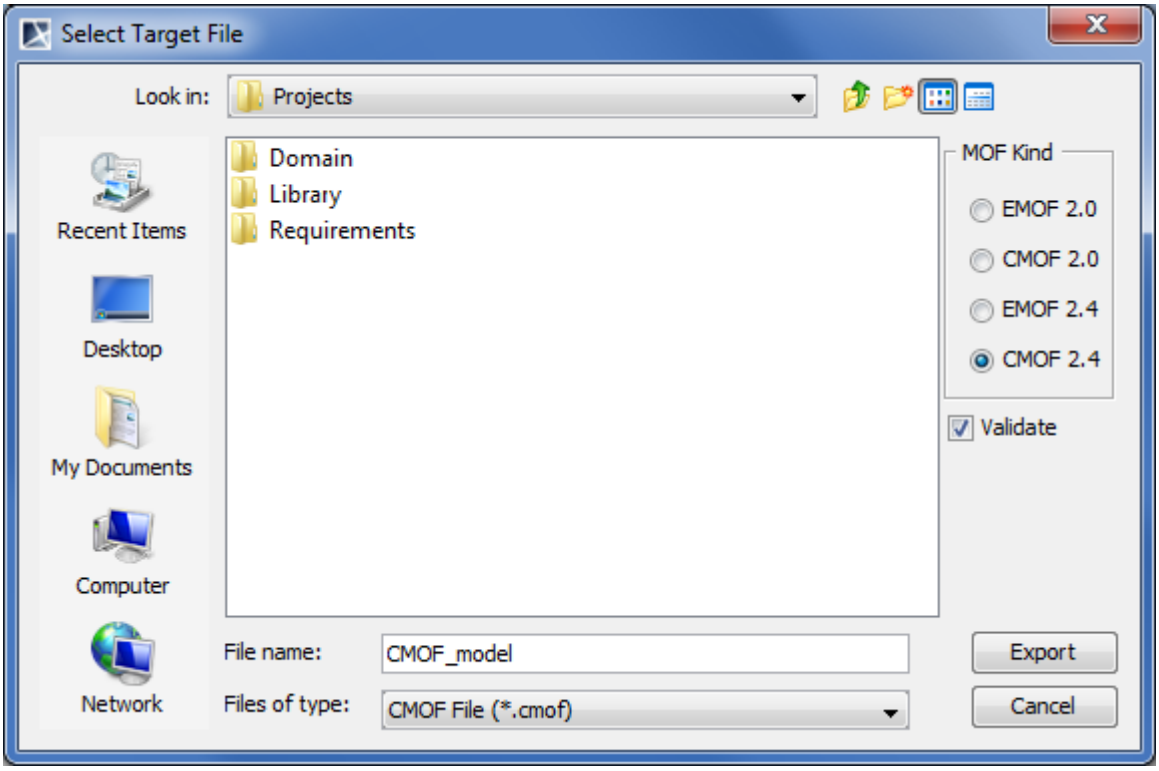


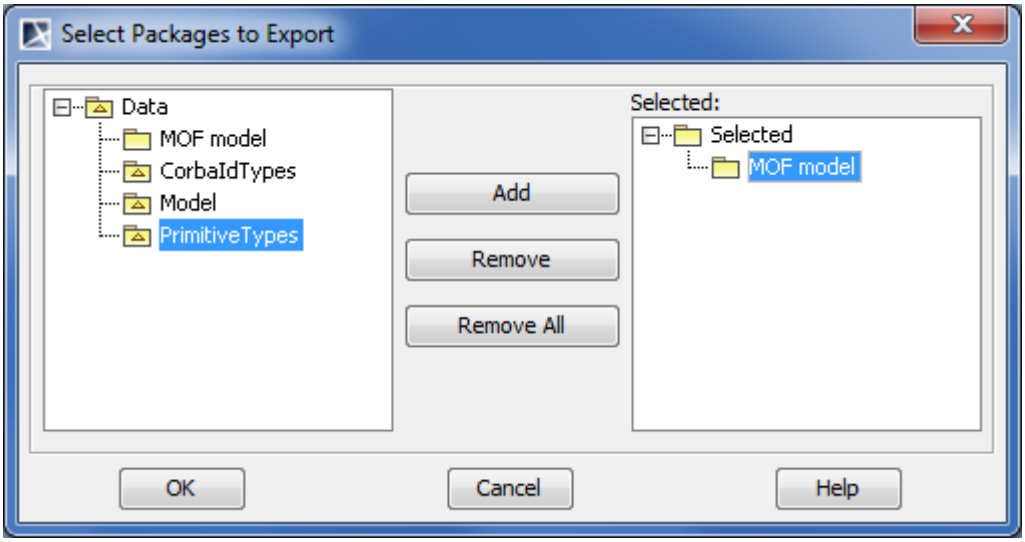
Figure 90 -- Select Target File dialog. Exporting project to CMOF 2.4 file

To export selected packages to a MOF file

1. From the main menu, select **File** > **Export To** > **MOF XMI File** > **MOF Selection**.

Copyright © 1998-2015 No Magic, Inc.

2. In the **Select Packages to Export** dialog, select packages you want to export and click **Add** to move them to the **Selected** list. Click **OK** when you are done.



3. Perform the procedure "To export a project to a MOF file" starting from the step #2.

Exported elements validation

MagicDraw provides two validation suites (one for CMOF and one for EMOF) for validating a model that is being exported to a MOF file. These suites contain batches of rules to check exported elements. Warnings about not exported elements (for example, diagrams, behavioral elements, or other) are displayed after the validation process is completed.

The validation process does not preclude the model from being exported. Unsuitable elements are simply skipped.

You can choose whether to run the validation or not before exporting the model (see the procedure "To export a project to a MOF file" on page 179).

Importing projects from MOF files

To import a project from a MOF file

1. From the main menu, select **File** > **Import From** > **MOF XMI File**.
2. In the opened dialog, select the EMOF or CMOF file you want to import and click **Open**. The file is imported as a separate project.

Ecore Support

The Ecore model is a model type supported by Eclipse Modeling Framework (EMF). This model type can be colloquially called the EMF model (even though EMF supports many types of models, e.g., the UML model).

The Ecore model can be used for various purposes. Several of them are as follows:

- Metamodeling purposes, where its expressive power is roughly similar to EMOF (and even slightly higher than EMOF).
- Simple class modeling purposes, where the Ecore model is used as a subset of UML.

Copyright © 1998-2015 No Magic, Inc.

Ecore models, prepared with MagicDraw, can be exported as Ecore models for the further processing (generating model repositories, code or XML parsing and storing, etc.) with other EMF tools.

This section contains the following subsections:

- "Creating projects for Ecore modeling" on page 179.
- "Ecore modeling" on page 184.
- "Exporting projects to Ecore files" on page 192.
- "Importing projects from Ecore files" on page 196.

Creating projects for Ecore modeling

You can use the following ways to create a project for Ecore modeling:

- To create a new project from the Ecore template (page 181).
- To apply the Ecore profile to a project (page 182).
- To use the standard Ecore library in a project (page 183).

To create a new project from the Ecore template

1. From the main menu, select **File** > **New Project**. The **New Project** dialog opens.
2. In the **Other** domain, select **Project from Template**.
3. Type a project name.
4. Specify the project location.
5. Under **Select Template**, expand **Metamodeling** and then select **Ecore Template**.

Copyright © 1998-2015 No Magic, Inc.

6. Click **OK** when you are done.

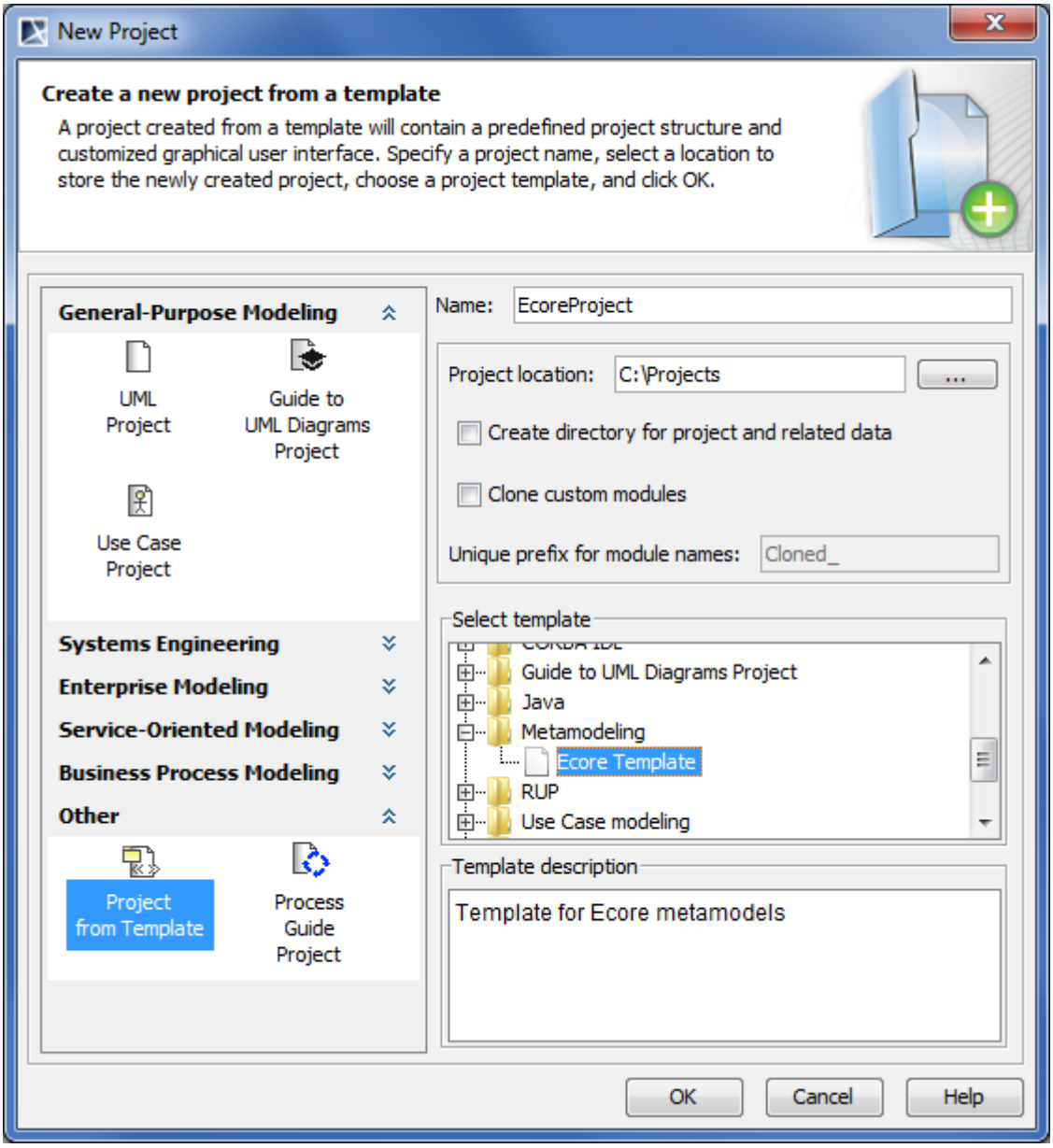


Figure 91 -- Creating project from Ecore template

For more information about creating a project from a template refer to the procedure "To create a new project from a template" on page 122.

To apply the Ecore profile to a project

NOTE The Ecore profile can be applied only to existing UML or CMOF / EMOF projects.

1. From the main menu, select **File** > **Use Module**[5]. The **Use Module** wizard opens.
2. Under **Select module file**, click **From predefined location**. The **Project modules paths** list appears.
3. Select **<install root>\profiles**. The content of the <install root>\profiles folder is listed bellow.
4. Select **Ecore_Profile.xml** in the list.

5. Starting from version 18.1, referred as “used project” in MagicDraw UI. This section mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

5. Click **Next**, if you want to change module usage settings.
6. Click **Finish**.

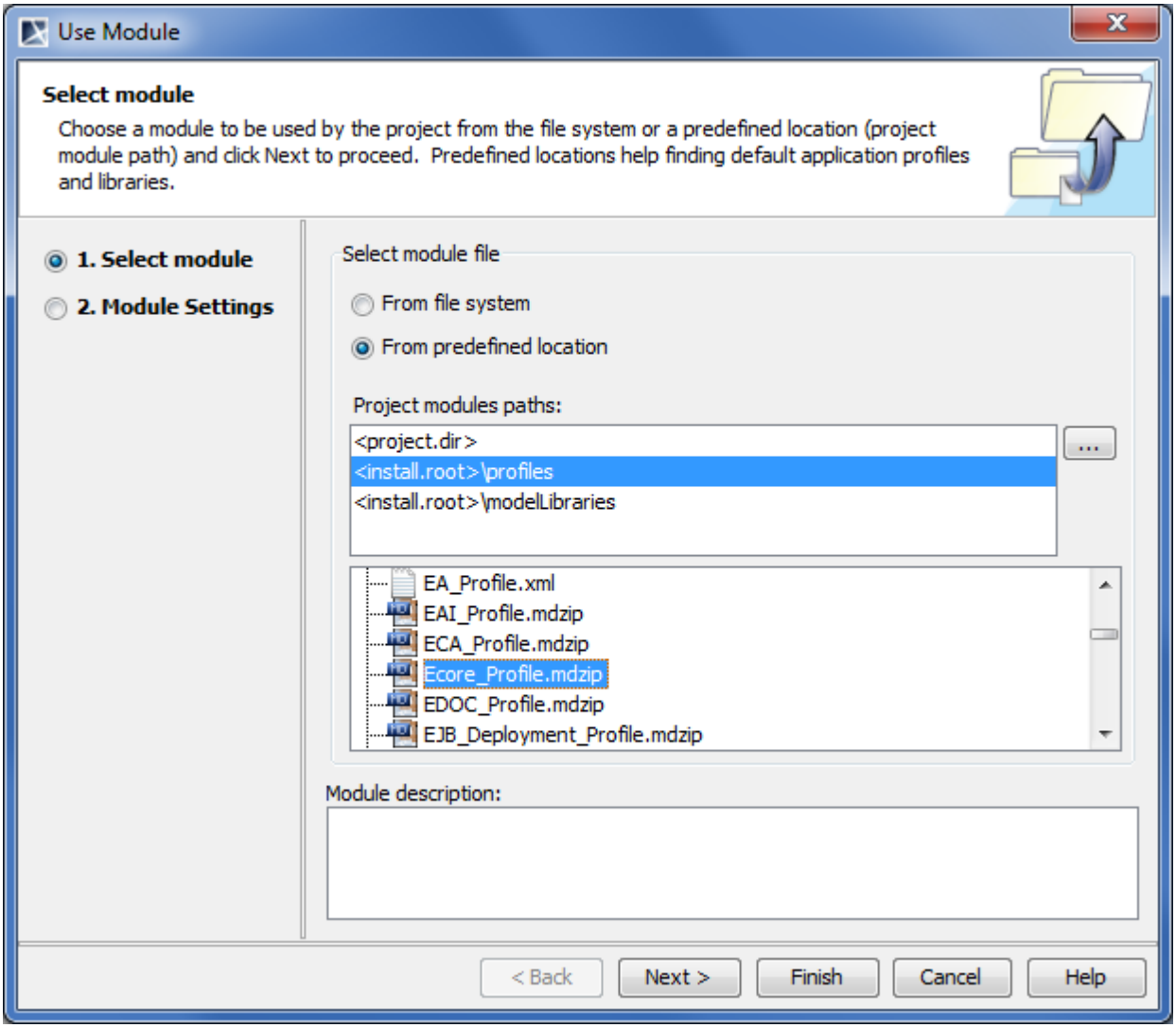


Figure 92 -- Selecting Ecore profile

For more information, see "Use Project wizard" on page 153.

If your Ecore model references some standard Ecore elements (such as standard data types (for example, EShort) or standard metaclasses (for example, EStructuralFeature), you need to use the standard Ecore library in your project.

To use the standard Ecore library in a project

1. From the main menu, select **File** > **Use Module**.The **Use Module** wizard opens.
2. Under **Select module file**, click **From predefined location**. The **Project modules paths** list appears.
3. Select **<install root>\modelLibraries**. The content of the <install root>\modelLibraries folder is listed bellow.
4. Select **Ecore.mdzip** in the list.
5. Click **Next**, if you want to change module settings.

Copyright © 1998-2015 No Magic, Inc.

6. Click **Finish**.

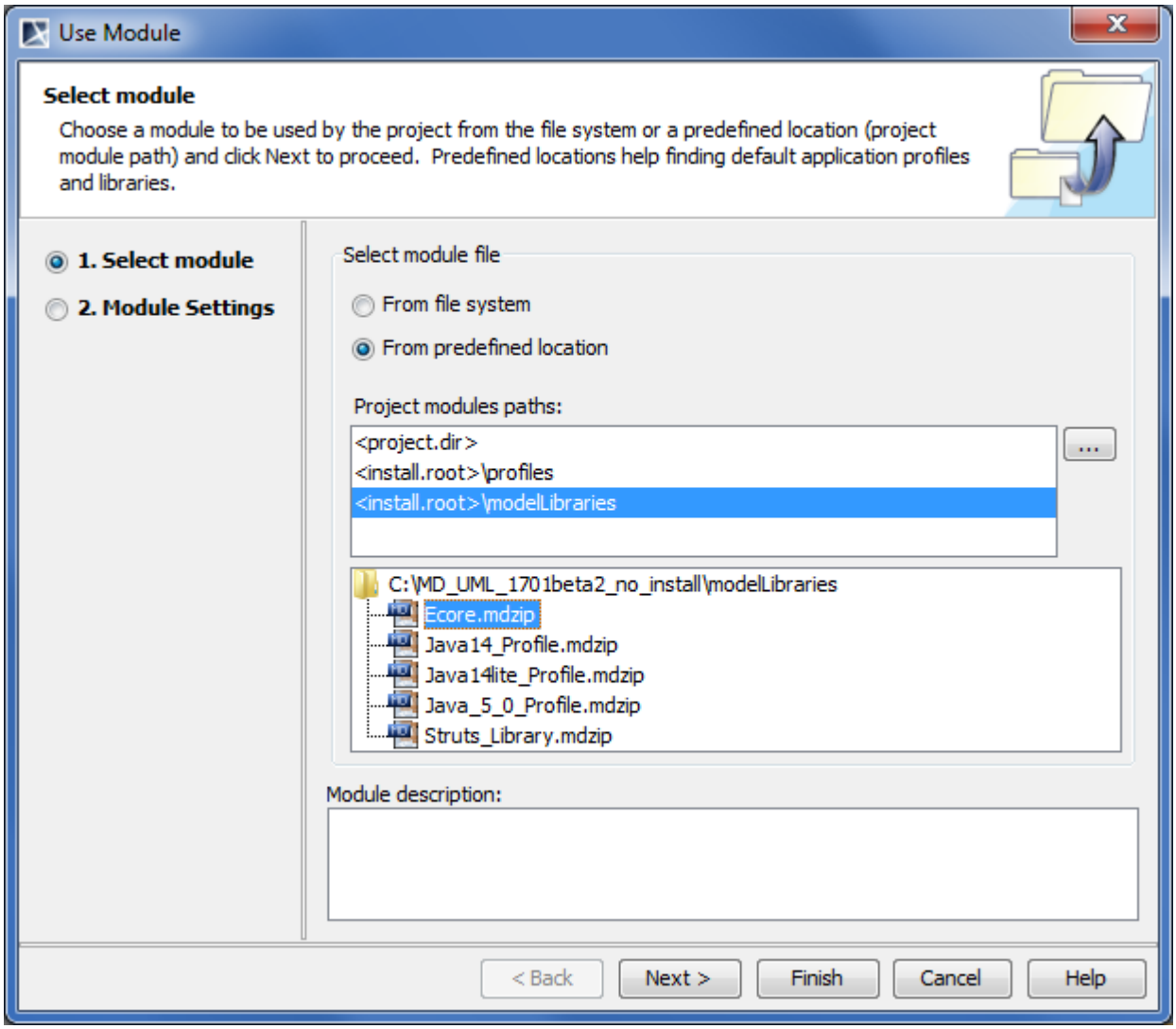


Figure 93 -- Selecting Ecore library

For more information, see "Use Project wizard" on page 153.

Ecore modeling

Learn about Ecore modeling in the following subsections:

- "Diagrams and elements in Ecore models" on page 184.
- "Element properties in Ecore models" on page 185.
- "Annotation modeling" on page 188.

Diagrams and elements in Ecore models

There are no specific diagrams for editing Ecore models in MagicDraw. You can use the same Class diagrams as you use for your UML models. Since Ecore is almost a subset of UML (with a few additions), familiar UML elements are used for the modeling. You can also develop Ecore models without using the Ecore profile at all. If your Ecore model uses only UML-specific information, you can develop it using plain UML and export it to Ecore without any problem.

Ecore is even more similar to EMOF. You can export the same model to both Ecore and EMOF.

Copyright © 1998-2015 No Magic, Inc.

Class, DataType, Enumeration, Package, Operation, Parameter have a direct one-to-one correspondence between UML and Ecore.

Ecore has two flavors of structural features, EAttribute and EReference, while UML has just one - Property. Fortunately a differentiation between an attribute and a reference is unambiguous and automatically resolved: the property, whose type is a data type is treated as EAttribute; the property, whose type is a class is treated as EReference. Hence the user does not need to worry about this - he/she can simply use properties.

There are no standalone Association and Generalization model elements in Ecore, but there is analogous information in Ecore: two EReferences, pointing to each other by their **opposite** property is equivalent to the association; the EClass::**eSuperTypes** property is equivalent to a generalization. Hence it is possible and meaningful to draw associations and generalizations in your model for exporting this information to Ecore.

Ecore generics (templates) are also supported. You can use the UML template support to model Ecore generics. While the modeling is not trivial (and not one-to-one due to weak semantics of Ecore's EGenericType), it is possible to model all cases of template types, even ones with complexly nested type bounds like, for example, **SortedList<T extends Comparable<? super T>>**.

Your models can also contain any other UML elements, which are not present in Ecore. These elements are simply skipped during the export to Ecore. A warning is given about these elements (see "Exported data type mappings" on page 194).

Element properties in Ecore models

There are few Ecore-specific properties, which are brought in when the Ecore profile is used. These properties are used to capture Ecore specific information, not existing in UML. MOF-specific properties are also relevant for Ecore. These special properties are as follows:

Copyright © 1998-2015 No Magic, Inc.

- The Ecore package has the additional properties: nsPrefix (**Namespace Prefix**) and nsURI (corresponds to the **URI** property specified in the UML v2.4.1).

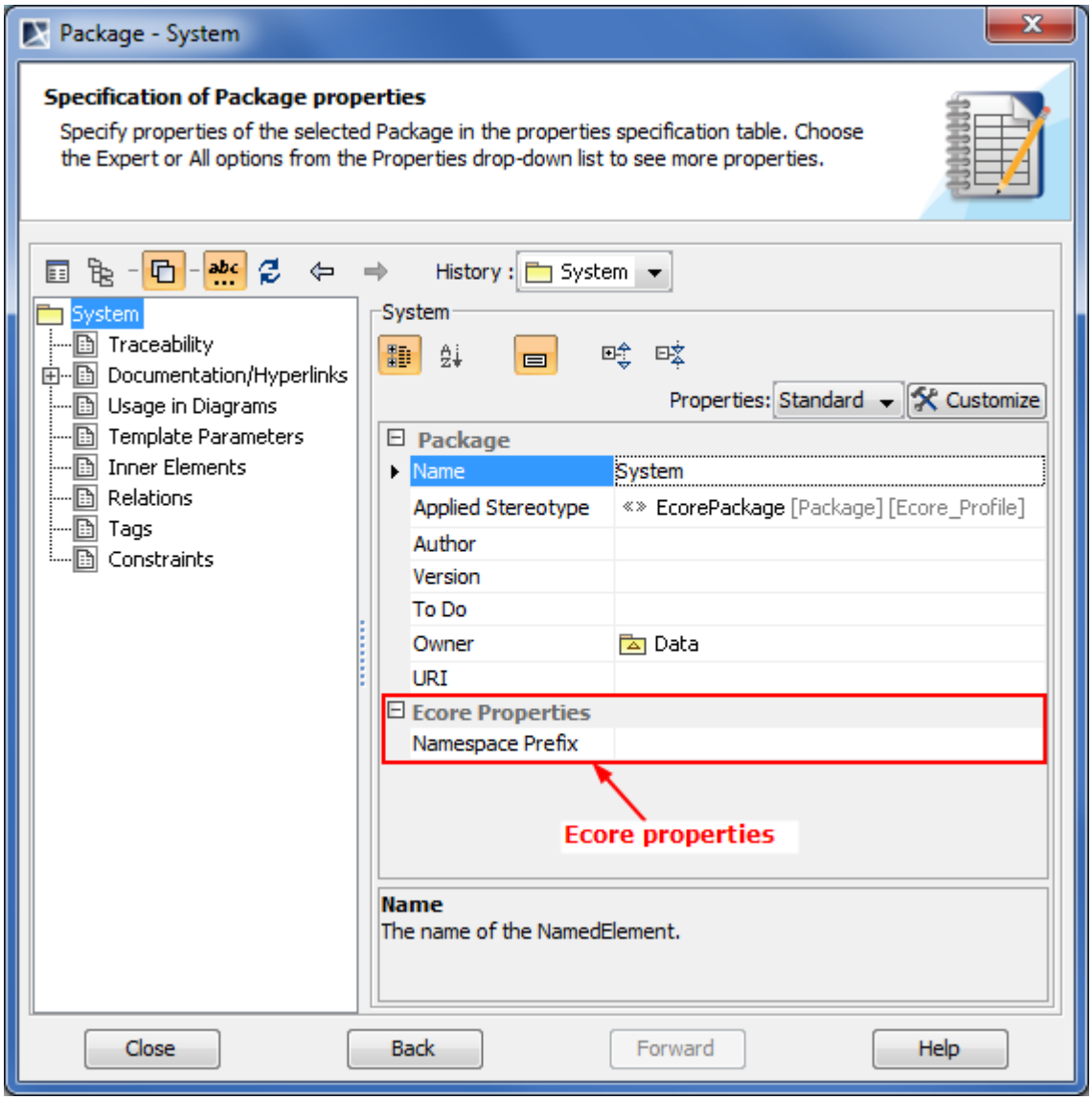


Figure 94 -- Additional properties of Ecore package

Copyright © 1998-2015 No Magic, Inc.

- The Ecore classifier (class, data type, enumeration) has the additional properties: instanceClassName (**Instance Class Name**) and instanceTypeName (**Instance Type Name**).

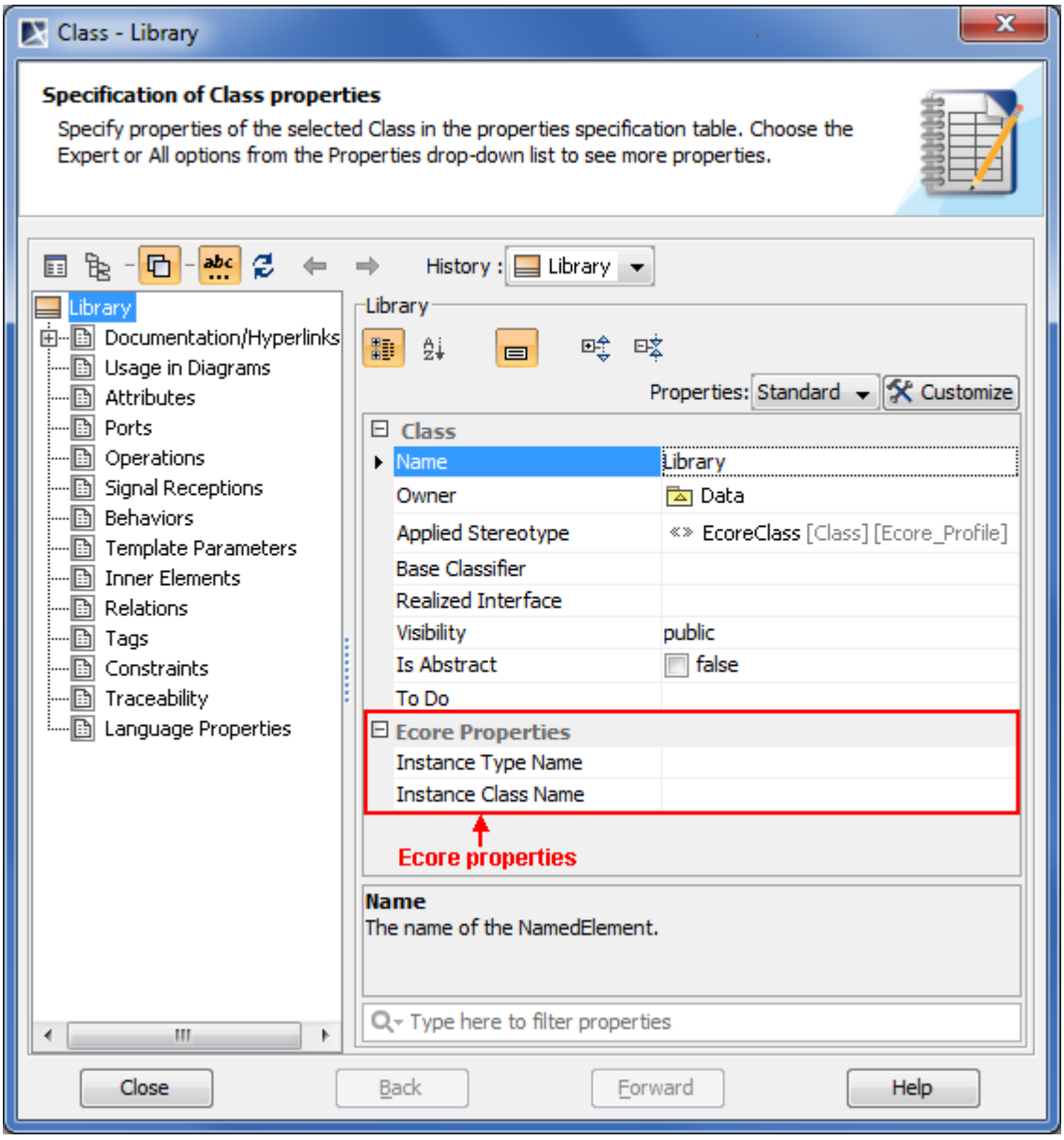


Figure 95 -- Additional properties of Ecore classifier

Copyright © 1998-2015 No Magic, Inc.

- Ecore attributes and references (modeled as the UML property) have the additional properties: volatile (**Volatile**), transient (**Transient**), unsettable (**Unsettable**), and resolveProxies (**Resolve Proxies**, used for references only).

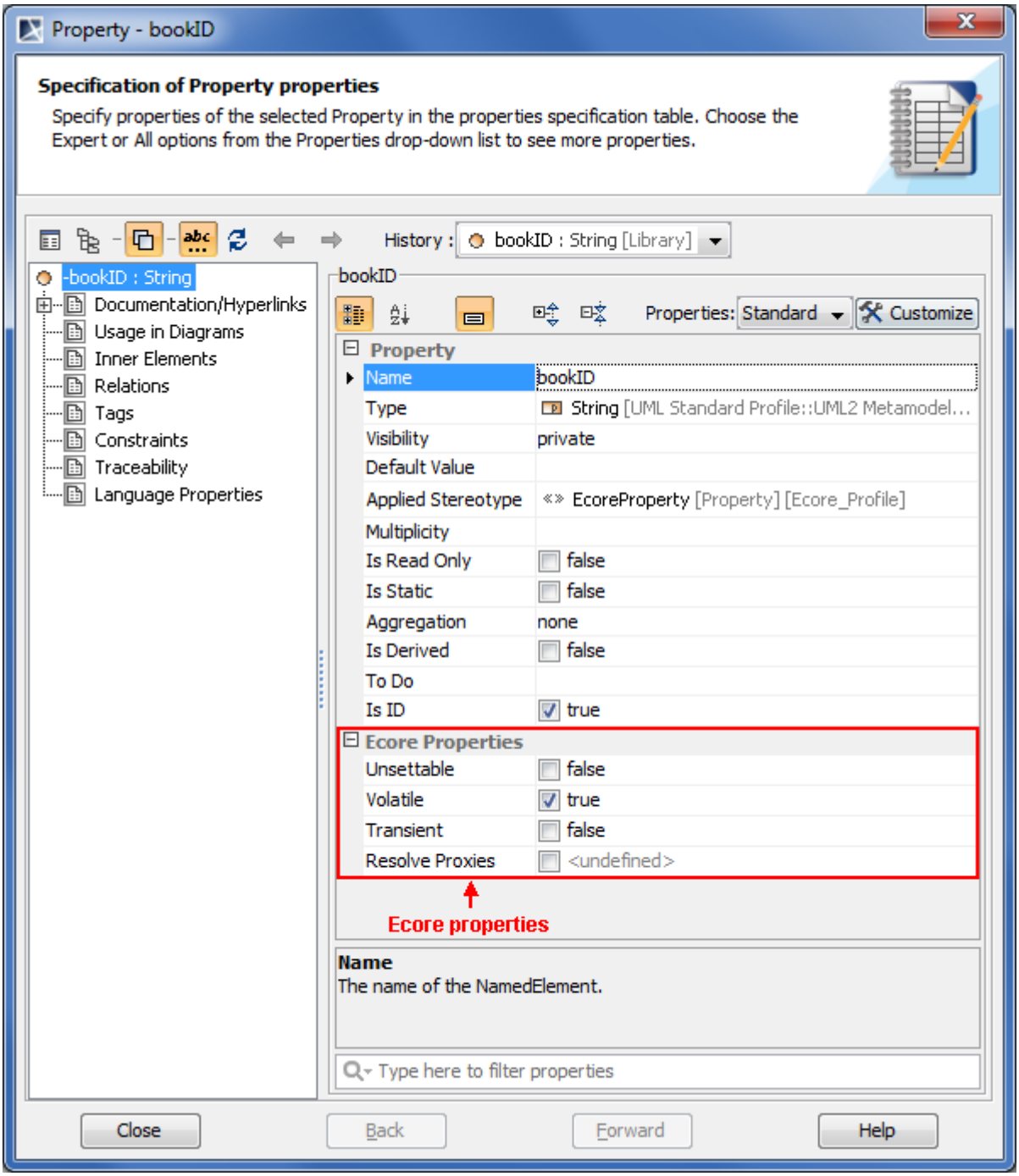


Figure 96 -- Additional properties of Ecore attribute

Annotation modeling

Ecore annotations are modeled as UML comments. For simple annotations no additional actions are necessary.

However Ecore annotations have more powerful semantic than UML comments - they can have an internal substructure. In particular they can have an additional key-value map. For this additional information, there is a special «EcoreAnnotation» stereotype, that can be applied on an annotating comment. After applying the

Copyright © 1998-2015 No Magic, Inc.

stereotype, the key-value map can be entered in a separate node of the annotating comment Specification window. Key-value pairs are stored as internal subcomment elements of the annotation.

To create an Ecore annotation

1. In the Ecore model, create a Comment element.
2. Apply the «EcoreAnnotation» stereotype to the element. For the instructions how to apply a stereotype refer to "Changing the stereotype display mode" on page 988.

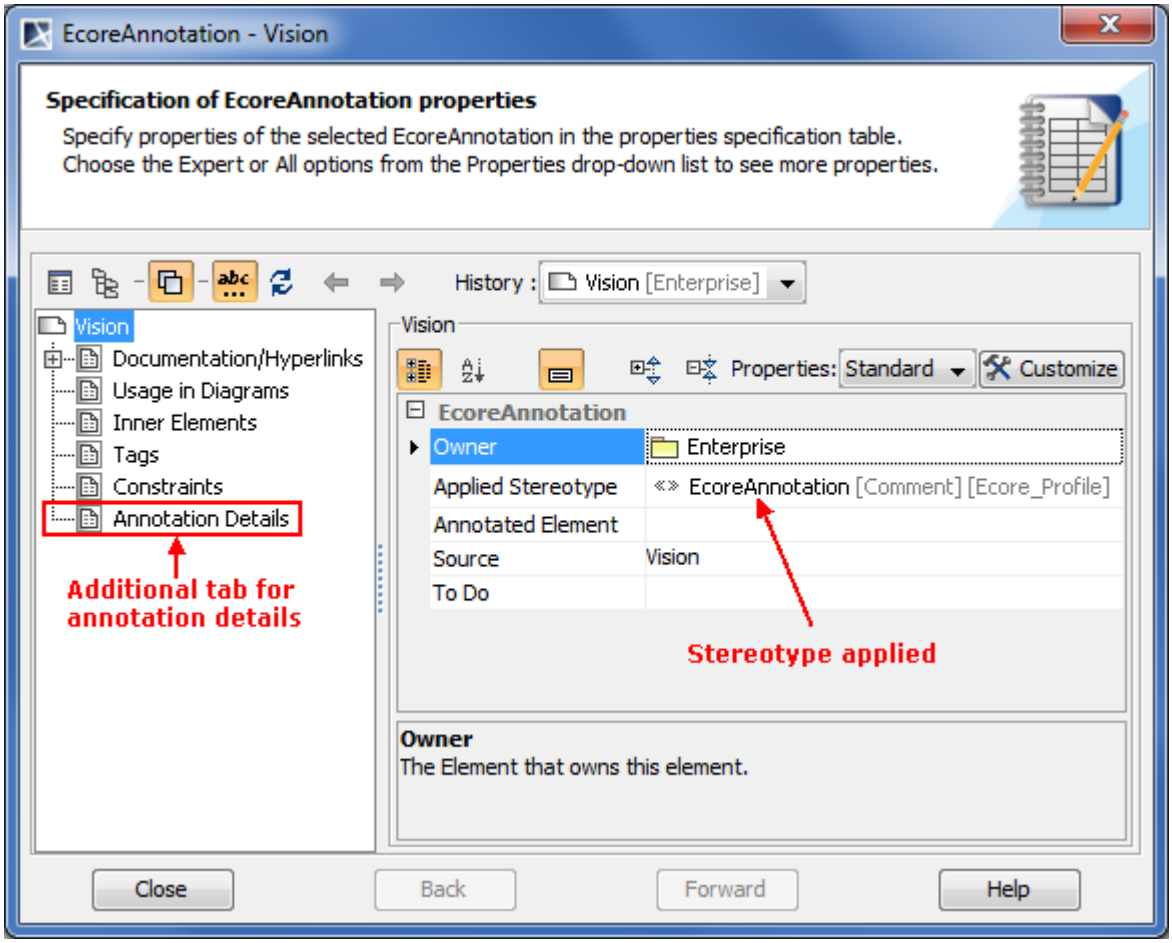


Figure 97 -- Ecore annotation's Specification window

Copyright © 1998-2015 No Magic, Inc.

To create a key-value pair

1. In the Ecore annotation's Specification window, select the **Annotation Details** property group.

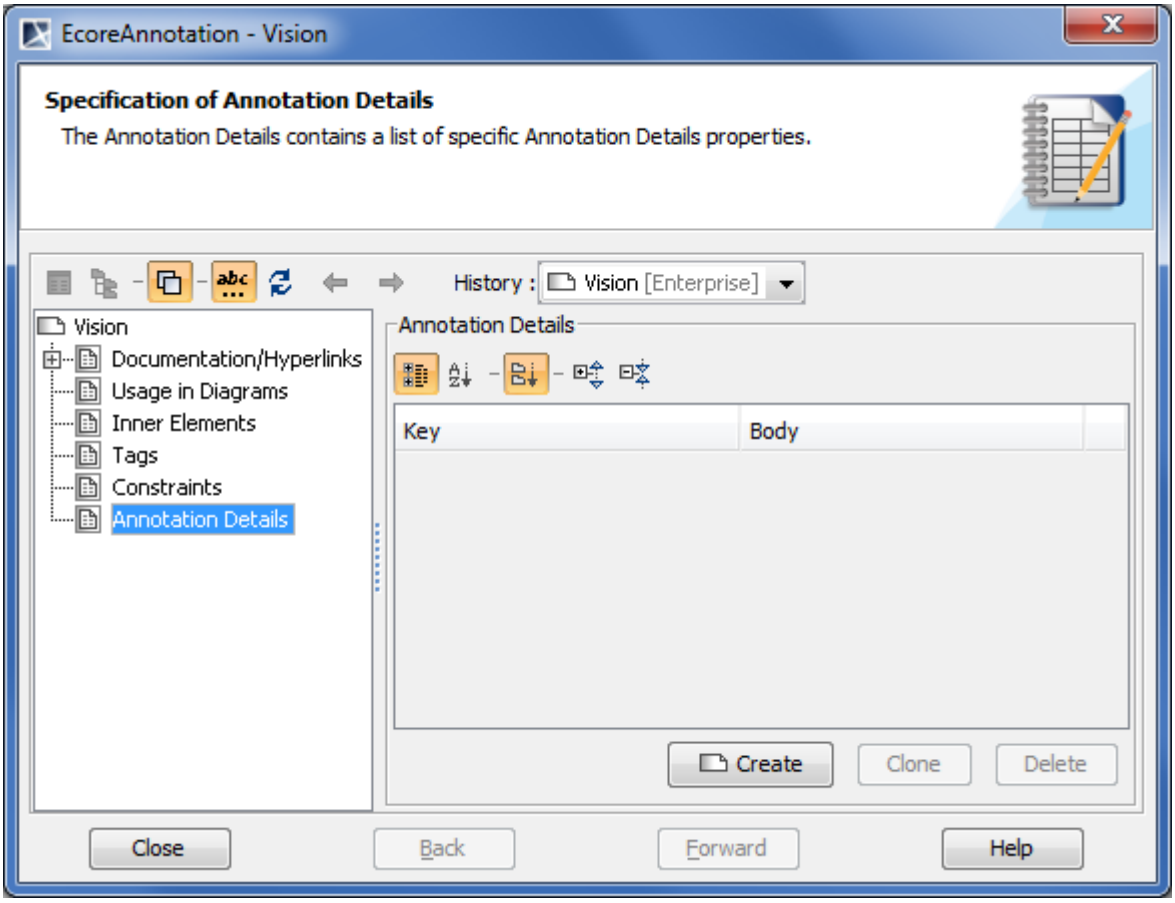


Figure 98 -- Annotation Details property group in Ecore annotation's Specification window

Copyright © 1998-2015 No Magic, Inc.

2. Click the **Create** button. The key-value pair's Specification window opens.

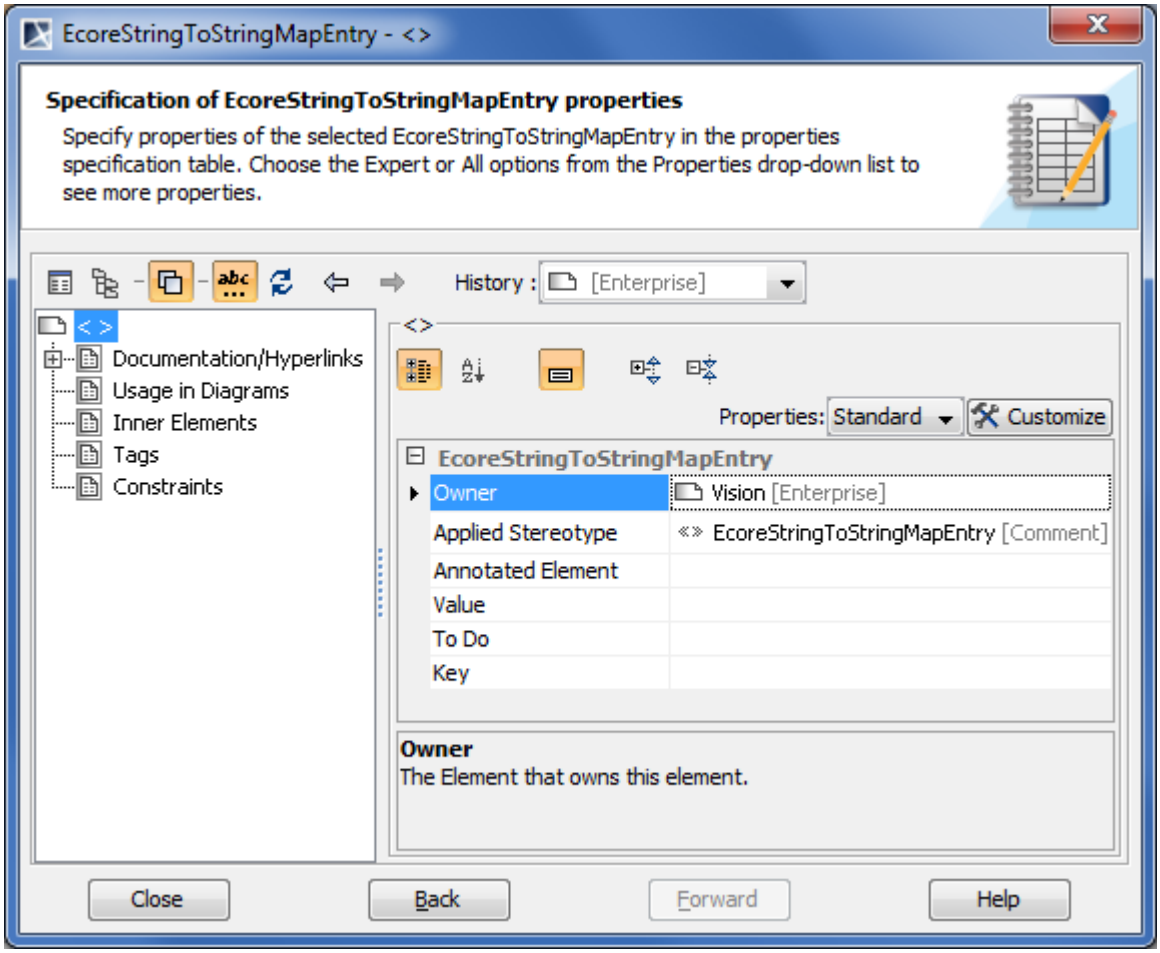


Figure 99 -- Kay-value pair's Specification window

3. Enter values for both the **Value** and **Key** properties.

Copyright © 1998-2015 No Magic, Inc.

4. Click the **Back** button when you are done. You will see the key-value pair created.

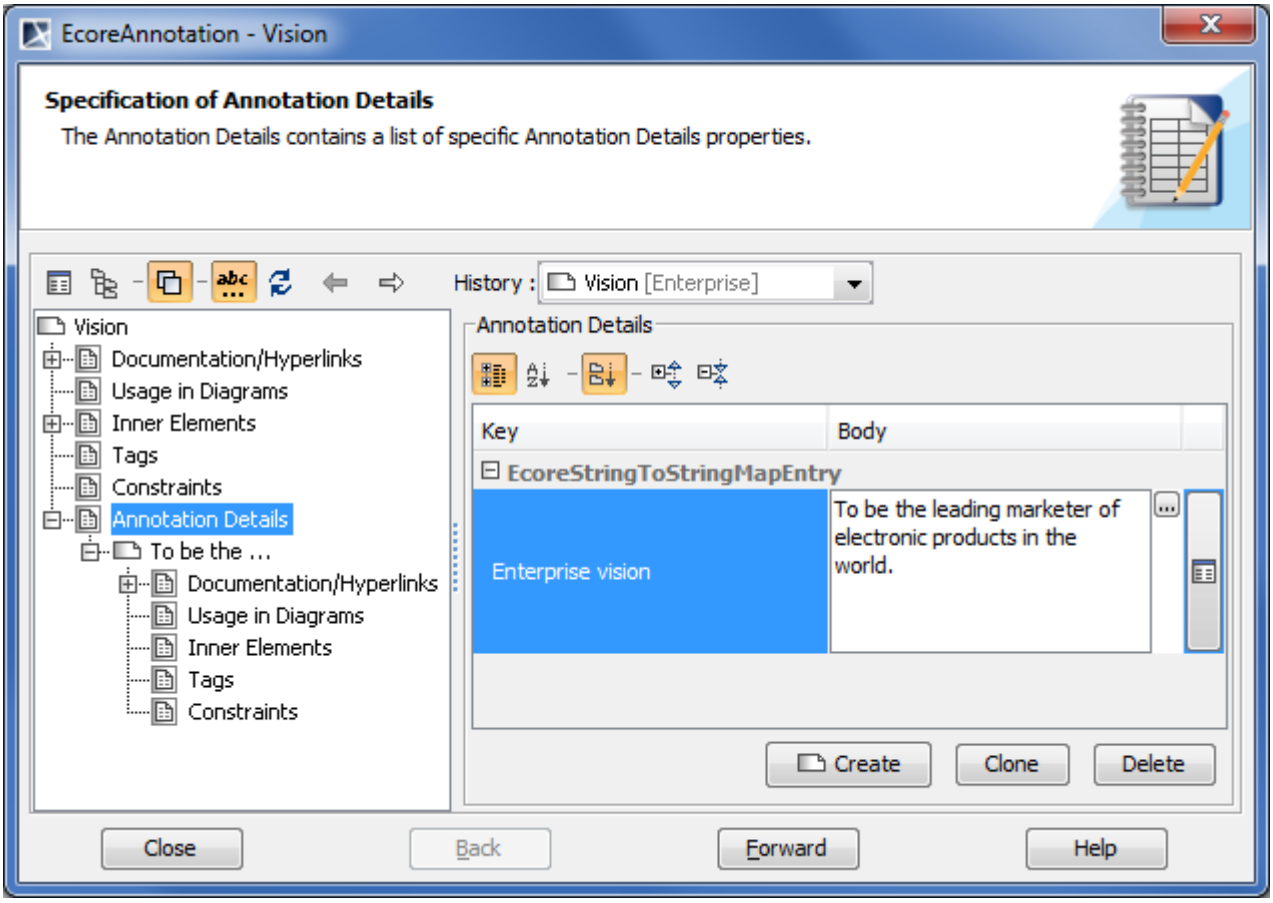


Figure 100 -- Key-Value pair created

Exporting projects to Ecore files

The Ecore model exporting is very similar to the EMOF / CMOF model exporting. After the Ecore model is created, you can export either the whole project or selected packages to an .ecore file.

This section contains the following subsections:

- "Exporting simple projects" on page 192.
- "Exporting project with modules" on page 193.
- "Exported data type mappings" on page 194.
- "Ecore elements validation" on page 195.

Exporting simple projects

To export a project to an Ecore file

1. From the main menu, select **File** > **Export To** > **EMF Ecore File** > **Ecore Whole Model**. The **Select Target File** dialog opens (see the following figure).
2. Type a file name and select a location for the exported model.
3. Click to select or to clear the **Validate** check box. For more information about validating refer to "Ecore elements validation" on page 195.

Copyright © 1998-2015 No Magic, Inc.

4. Click the **Export** button when you are finished.

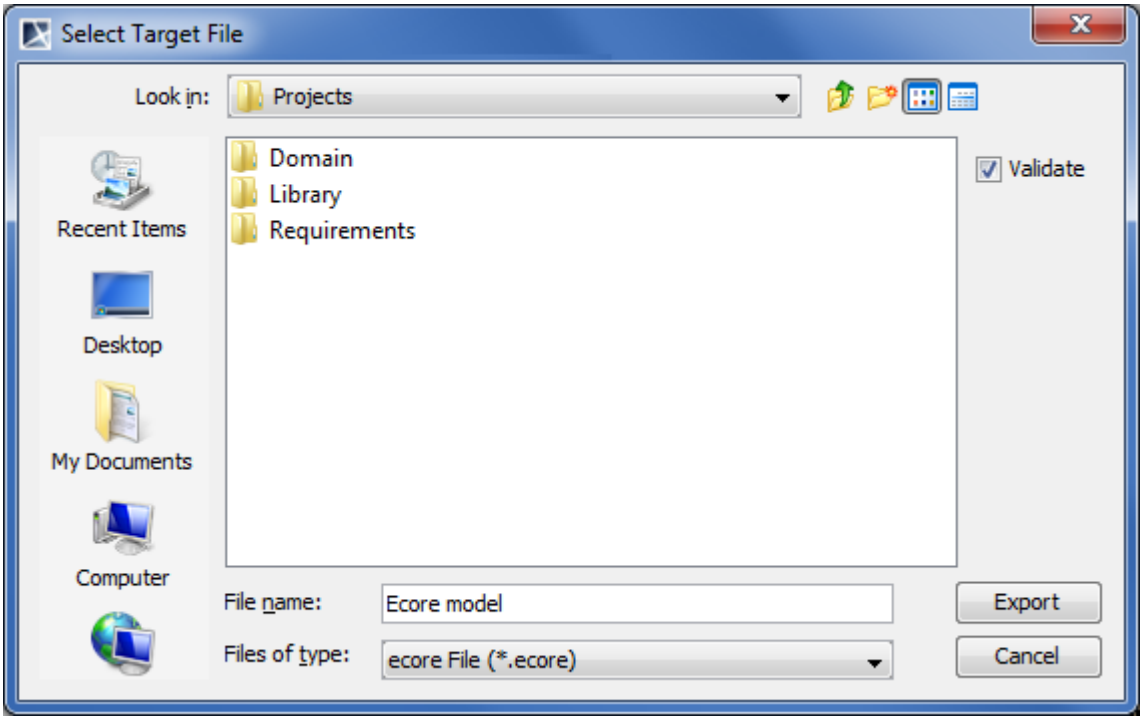
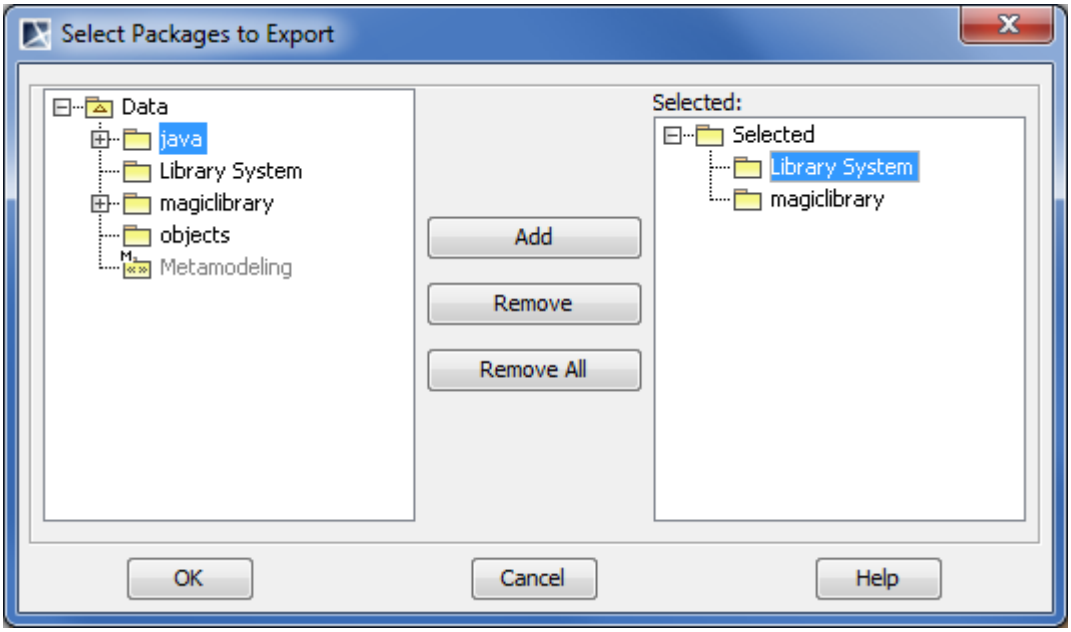


Figure 101 -- Select Target File dialog. Exporting project to Ecore file

To export selected packages to Ecore file

1. From the main menu, select **File** > **Export To** > **EMF Ecore File** > **Ecore Selection**.
2. In the **Select Packages to Export** dialog, select packages you want to export and click **OK** when you are done.



3. Perform the procedure "To export a project to an Ecore file" starting from the step #2.

Exporting project with modules

It is important to notice that any modules[6] used by MagicDraw project are not exported together with the project on its export to an Ecore file. Only references to modules are exported. Therefore the output of the project

Copyright © 1998-2015 No Magic, Inc.

export is an Ecore file (model) containing the direct content of the MagicDraw project and referencing to other Ecore files (modules) that are used in the MagicDraw project. For referencing to modules, i.e., other Ecore files, Ecore references are used.

Each module used by the project must be exported to an Ecore file individually. For this you have to open each module as a project and then export it to an Ecore file (see "Exporting projects to Ecore files" on page 192).

It is strongly recommended to read the following paragraph before your very first attempt to export a MagicDraw project that uses modules. Getting familiar with this information may help you to escape a serious problem that arises because of the nature of Ecore references.

Ecore references, crossing a resource boundary (when the element in one file references the element in another file), *are qualified-name-based, but not id-based* as in case of CMOF, EMOF, or UML. Hence when exporting a project with references to elements in other projects (modules), the export tool must know the full path of elements in the module's Ecore file. This information cannot be determined from UML model without additional information. For this the following approach is adopted:

1. **Save the module after the export.** After a project is exported to an Ecore file, the qualified names (paths) of the shared elements are recorded in special stereotypes («EcoreExportServiceInformation») / tags (ecoreExportPath) of the project's shared packages. The side effect is that the project is modified during the export. To preserve this information for later usage, you need to save the project after the export.
2. **Export modules before exporting the main project.** When a project that references to elements in one or more modules is exported to an Ecore file, the export tool needs to know these elements' paths, saved on the appropriate module export to Ecore. If this information is missing (e.g., in case the project is being exported before modules are exported), the export tool tries to guess the correct path of each element in the module and gives warnings about this. That is why modules should be exported to Ecore before exporting the main project that uses these modules.

Exported data type mappings

Standard UML data types are exported as standard Ecore data types. The following table shows which Ecore data type corresponds to which UML data type.

UML data type	Ecore data type
String	EString
Boolean	EBoolean
Integer	EInt
UnlimitedNatural	EInt
Real	EReal

Standard data types from the MagicDraw profile are exported as Ecore types. The following table shows which Ecore data type corresponds which UML data type.

Data types in MagicDraw profile	Ecore data type
boolean	EBoolean
byte	EByte
char	EChar
date	EDate

6. Starting from version 18.1, referred as "used project" in MagicDraw UI. This section mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

Data types in MagicDraw profile	Ecore data type
double	EDouble
float	EFloat
int	EInt
real	EReal
long	ELong
short	EShort



The void data type is exported as an absence of the type.

References to Ecore model elements (standard datatypes like EInt, metaclasses like EStructuralFeature), defined in the standard Ecore library are exported as standardized Ecore references to Ecore metamodel elements (the resource identifier part of the Href is *http://www.eclipse.org/emf/2002/Ecore*).

Ecore elements validation

MagicDraw provides a validation suite for validating a model that is being exported to an Ecore file. This suite contain batches of rules to check exported elements. Warnings about not exported elements (for example, diagrams, behavioral elements, or other) are displayed after the validation process is completed.

The validation process does not preclude the model from being exported. Unsuitable elements are simply skipped.

You can run the Ecore validation on a model (project or module) export to Ecore. All UML elements that are not suitable for the Ecore, will be highlighted.

You can also run this validation suite at any time while you are developing an Ecore model.

Copyright © 1998-2015 No Magic, Inc.

To run the Ecore validation

1. From the main menu, select **Analyze** > **Validation** > **Validate**. The **Validation** dialog opens.

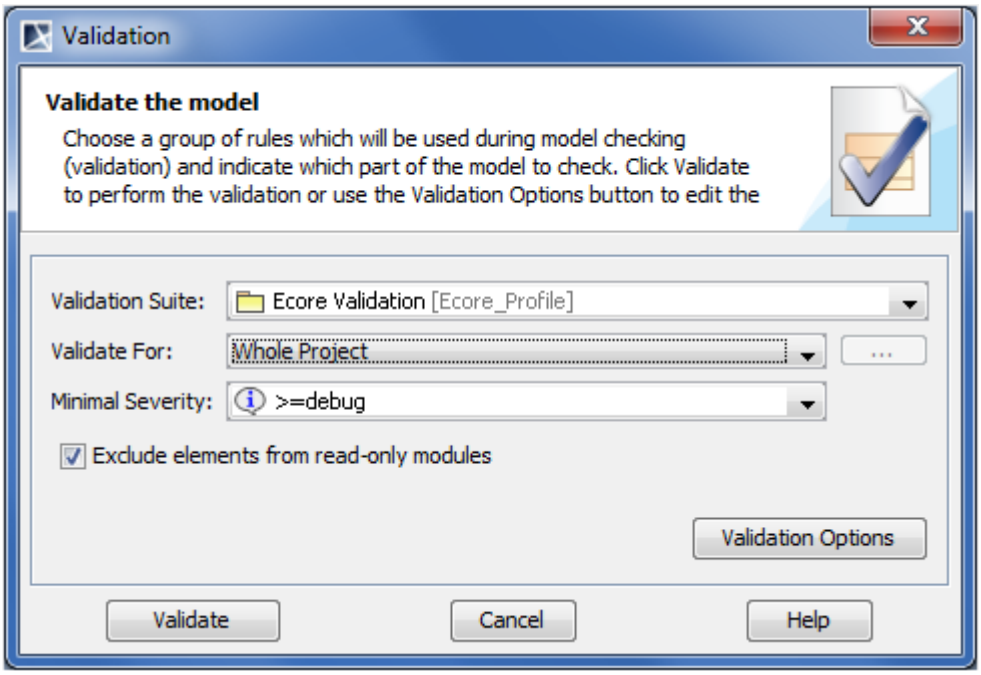


Figure 102 -- Validation dialog

2. In the **Validation Suite** drop-down list, select **Ecore Validation**.
3. In the **Validate For** drop-down list, select the scope of the validation.
4. In the **Minimal Severity** drop-down list, select the level of severity.
5. Click **Validate**.

Importing projects from Ecore files

There is no possibility to import Ecore files directly. Ecore files could be imported using the EMOF import feature.

To import an Ecore file

1. In the Eclipse environment, open the *.ecore file and save it as *.emof.
2. Use the procedure "To import a project from a MOF file" on page 180.

IMPORTANT! This indirect way looses some Ecore-specific model details that are not existent in EMOF.

Working with Standard Profiles

Standard Profiles as System Resources

All profiles and libraries, which are bundled with MagicDraw family products are considered as standard/system resources, which are non-modifiable and are essential for the correct tool behavior.

Copyright © 1998-2015 No Magic, Inc.

We highly recommend not to modify our provided standard profiles and libraries as it could cause problems on version updates, plugins, core malfunctions, and model corruptions.

Users will be warned on any intentional or unintentional attempt to modify profiles in the following ways:

- Open a profile as a project.
- Use in the read-write mode.
- Import into a project.
- Merge projects.
- Any other cases.

Plugin and Profile Versions

Standard profiles are usually upgraded to support the latest versions of the specification of standards they represent (for example, SysML 1.2 or UML 2.4) in the every MagicDraw release cycle. The MagicDraw application, the plugins code, and the behavior are modified accordingly to reflect these changes.

There is a very high probability that the latest version of MagicDraw or plugins cannot work with earlier or later profile versions and may cause an unpredictable behavior or even model distortions. For example, SysML plugin 16.5 requires to use SysML Profile 16.5, as it could malfunction when using SysML Profile 16.0 or 16.6.

To protect the user from such cases, every MagicDraw project knows which profiles or plugins versions were used to create it and are required to load data correctly.

Every standard profile has the version number. Normally it is the same as the MagicDraw (or a plugin) version number.

MagicDraw requires to use the corresponding version of the profile with the corresponding software version. You will get a warning, if your used plugins or profiles are obsolete or you miss some plugins or profiles.

Resource Manager with selected missing resources will be launched automatically, so you will be able to install missing plugins/profiles in few clicks.

If new versions of plugins are not purchased or you simply do not want to install it, but need to take a look at the project content, warnings may be ignored and the project may be loaded. In this case, proxy elements for missing profile elements will be created to retain missing references. Missing custom diagrams will be loaded as regular UML diagrams or will be restricted for a review. Do not save such project! Use it for the preview only.

Old projects will be loaded/converted without any warnings, if you have newest versions of corresponding plugins and profiles as MagicDraw is always backward compatible.

Standard Profiles in Teamwork Server

Standard/system profiles are not added into Teamwork Server, because every user has recent versions installed locally. As profiles are non-modifiable, the version control is not needed. It solves multiple profiles usage / modification / update issues in the teamwork and at the same time increases the teamwork performance, as standard profiles will not be transferred via networks.

Information about an updating UML Standard Profile due to the migration to UML 2 is presented in the Teamwork Server readme file. You can find this file in <*MagicDraw Teamwork Server installation directory*>.

Copyright © 1998-2015 No Magic, Inc.

5 DIAGRAMMING

This chapter offers an overview of working with diagrams and symbols. In general, the topics discussed apply to all supported diagram types.

When working with diagrams it is helpful to keep in mind the following concepts:

- A **shape** refers to a notation of a model element, such as a package, class, state, use case, object, etc.
- A **path** refers to the notation for the various kinds of relationships such as associations, aggregations, dependency, message, and links.
- Both paths and shapes are defined as **symbols**.

In the chapter, you will find the following sections:

1. "Working with Diagrams" on page 198
2. "Working with Symbols of Model Elements" on page 224
3. "Working with Paths and Relationships" on page 227
4. "Smart Manipulation" on page 233
5. "Compartments" on page 236
6. "Smart Shape Sizing" on page 239
7. "Selection and Multiple Selections" on page 240
8. "Copying Text or Images to Diagrams" on page 242
9. "Nesting Image Shapes" on page 243
10. "Dragging, Copying, Cutting, and Pasting" on page 244
11. "Zooming" on page 254
12. "Using the Grid" on page 255
13. "Layout" on page 256
14. "Pusher and Magnet" on page 259
15. "Saving as Image" on page 261
16. "Printing" on page 264

Working with Diagrams



Basics

Creating Diagrams

To create a diagram

Do one of the following:

- From the toolbar:

  Click the  **Create Diagram** button. The Create Diagram dialog opens.

Copyright © 1998-2015 No Magic, Inc.

- From the **Diagrams** menu:
  On the main menu point to **Diagrams** and then click the **Create Diagram** command. The Create Diagram dialog opens.
- From the Model Browser:
  Right-click the desired model element in which you would like to create a diagram, on the shortcut menu, point to the **Create Diagram** command, and then on the submenu, click the diagram type that you want to create.
- Using shortcut key:
  Press Ctrl+N. The Create Diagram dialog opens.
- From the model element's Specification window:
  Open the **Inner Elements** property group in the Package's, Profile's, Model's, or other element's Specification window. Click **Create**. On the shortcut menu click **Create Diagram** and then select the diagram type. Define the diagram and click **Close**.
- Using Content diagram:
  On the Content diagram pallet, click the **Create Diagram** button. The Create Diagram dialog opens.
- Using a diagram creation wizard:
  On the main menu, click **Diagrams** > **Diagram Wizards** and then select the appropriate wizard depending on what diagram you want to create. Wizards for creating a Class, Generic Table, Package Dependency, Package Overview, Hierarchy, Activity Decomposition Hierarchy, Realization, Sequence Diagram from Java Source, Content diagram are available.
- Using a diagram creation wizard (from the **Model Visualizer** dialog):
  On the main menu click **Analyze** > **Model Visualizer**. The **Model Visualizer** dialog with the list of available diagram creation wizards will open. Select one and click the **Start** button to open the appropriate diagram creation wizard.



- Right-click the diagram tab, to open the shortcut menu.
- The open diagram has the diagram pallet. Right-click the empty area on the diagram pallet, to open the diagram pallet shortcut menu in which you can change the settings of the diagram pallet.

Copyright © 1998-2015 No Magic, Inc.

Create Diagram dialog

The Create Diagram dialog is designed to minimize the time needed to create any diagram type.

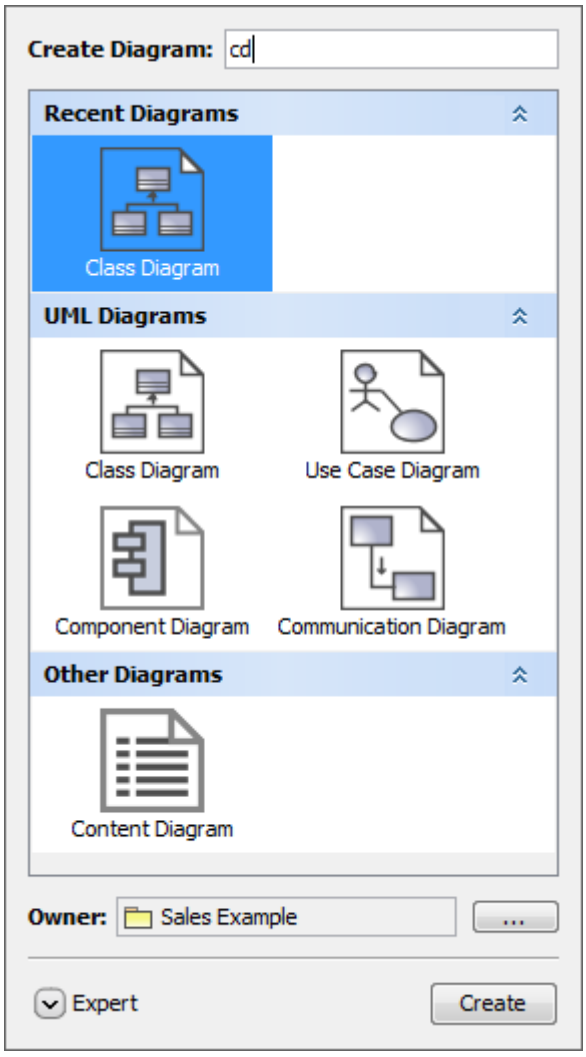


Figure 103 -- Create Diagram dialog

To create a diagram by using the Create Diagram dialog

1. In the Model Browser, select the owner for the diagram.
2. Do one of the following:
   - On the main toolbar, click the **Create Diagram** button.
   - On the **Diagrams** menu, click the **Create Diagram** command.
   - On the Content diagram pallet, click the **Create Diagram** button.
   - Press Ctrl+N.
3. In the **Create Diagram** box, type the diagram type you are searching for.
4. Do one of the following:
   - Double-click the diagram.
   - Press **Enter** or click to select the diagram if more than one diagram is listed.

Copyright © 1998-2015 No Magic, Inc.

- Select the diagram and click **Create**.



Use the Camel Case, when searching for diagrams:

- type the first word letters - "*CD*" to find "*Class Diagram*"
- search using non-capital letters – "cd" instead of "CD"
- skip the spacers – "cv3" works as well as "CV-3" or "cv-3"
- perform partial search – "ibd" finds SysML Internal Block diagram

To close the Create Diagram dialog

Do one of the following:

- Click outside the Create Diagram dialog.
- Press Esc.
- Create the diagram and the dialog will close automatically.



The **Create Diagram** dialog closes when switching between the projects or when switching between the programs.

The following table describes the components of the Create Diagram dialog.

Element Name	Element Type	Description
Create Diagram	Text box	Type the diagram type you are searching for.
Recent Diagrams	List of diagrams	Select the diagram to create from the recently created diagrams list.
Owner	Text box with 	Click the **...** button to select the package or the other owner, which contain the created diagram. Note that you can select the owner for the diagram before opening the Create Diagram dialog.
Expert	Button	Click to turn on the expert mode in the Create Diagram dialog, that is, expand lists of diagrams for expert user.
Standard	Button	Click to return on the standard mode in the Create Diagram dialog.
Create	Button	Click the button to create the selected diagram. To create a diagram, you can also double-click the diagram or select the diagram and then press Enter.



For more information on how to customize the perspective in the Create Diagram dialog, see "Customizing and Selecting Perspective" on page 65.

Opening Diagrams

You can choose to open diagrams either in the same tab or in a new tab – just like in the most popular Internet browsers.

Here are the benefits of using the tabbed browsing:

- Different aspects of the system (for example, requirements, architecture, test cases, domain models) can be analyzed in separate tabs. Both backward and forward navigations are supported for each aspect (tab) separately. Navigation does not open new tabs by default, which greatly helps in limiting the number of open tabs.

Copyright © 1998-2015 No Magic, Inc.

- Ability to compare separate views of the system side-by-side. For example, business and IT architectures can be displayed in two tabs side-by-side and analyzed by drilling down in their hierarchies. Navigation deeper in the structures will not open new tabs, which allows the smooth analysis of the modeled system.
- Smooth work with multiple screens. Browsing in a tab of the second screen never opens diagrams in the first screen, which means you stay in the same screen. New tabs open in the same screen, to the right of the active one.

To open a diagram

- From the Model Browser,
  Do one of the following:
  - Double-click the diagram.
  - Select the diagram and press Enter.
  - Right-click the diagram and then click **Open in New Tab**.
  - Select the diagram and press Ctrl+Enter. The diagram opens on the top of the active tab.
  - Right-click the diagram and then click **Open**. The diagram opens on the top of the active tab.
- From the diagram pane:
  Double-click a model element, to which the diagram is assigned. The new diagram opens in the same diagram tab.
- From the Content Diagram (available in the Standard, Professional, Architect, and Enterprise editions),
  if the diagram is added to the table of contents or a shape of the diagram is drawn on the diagram pane, do one of the following:
  - Double-click the diagram shape. The diagram opens in the same diagram tab.
  - On the diagram pane, right-click the diagram shape, and then click **Open**. The diagram opens in the same diagram tab.
  - On the diagram pane, right-click the diagram shape and then click **Open in New Tab**.
- From the **Diagrams** menu,
  Do one of the following:
  - On the main menu point to **Diagrams** and then click **History**. The **Diagram History** dialog opens. Double-click the diagram that you want to open or press Enter.
  - Press F12. The **Diagram History** dialog opens. Double-click the diagram you want to open or press Enter.



- You can change the default diagram opening behavior in the **Environment Options** dialog, the **General** options group, under the **Navigation** category.
- Press the Ctrl key when opening diagram, to turn on the opposite diagram opening mode. For example, if by default opening in the same tab is active, by using the Ctrl key, the diagram will open in the new tab and vice versa.

Copyright © 1998-2015 No Magic, Inc.

TIP!

- To load all diagrams that have been created in the project, from the **Diagrams** menu, select **Load All Diagrams**.
- In the **General** pane of the **Environment Options** dialog box, you can select a method for loading diagrams while opening a project. Three options are available:
  - **Load all Diagrams** – loads all diagrams that exist in the project.
  - **Load Only Open Diagrams** – loads only diagrams that were not closed in earlier usages of the project.
  - **Do not Load Diagrams** – all diagrams are not loaded and closed after opening a project.

To navigate backward and forward between diagrams in the same diagram tab

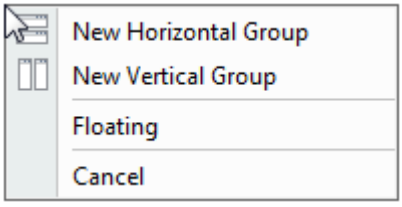
Do one of the following:

- On the diagram toolbar, click the *Backward* or *Forward* button.
- Press Alt+Left Arrow or Alt+Right Arrow.

Using Diagram Tabs

To split diagrams in new horizontal or vertical group

1. Do one of the following:
   - Right-click the diagram tab. The shortcut menu opens.
   - Select the open diagram tab, then drag and drop it to the diagram pane. The shortcut menu opens.



2. Select **New Horizontal Group** or **New Vertical Group** to split diagram pane and have more than one diagram opened at the same time.

NOTE

To split diagrams, more than one diagram should be open.

To turn the diagram window to floating

- Right-click the diagram tab and click **Floating**.

To show the diagram owner on the diagram tab

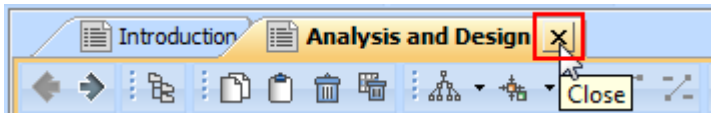
1. In the package, create diagram.
2. Select the open diagram tab and right-click to open the shortcut menu.
3. Select the **Show Owner** check box. The package name appears on the diagram tab.

To close a diagram

Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

- On the diagram tab, next to the diagram name, click the *Close* button.



- Right-click the empty area on the diagram pane, and then from the shortcut menu, select **Close Tab**.
- Press Ctrl+F4.

Showing Diagrams in Full Screen

If you want to see your diagrams in full screen and work exclusively from the diagram, use the **Show Tabs in Full Screen**) functionality. In a full screen mode, all necessary modeling commands are visible with an option to hide, and the Browser is in an auto hide mode. You can also manage the interface components to be displayed or hidden.

To turn on diagram full screen mode

Do either:

- Double-click the diagram tab.
- From the diagram shortcut menu, select the **Show Tabs in Full Screen** command.
- From the **View** menu, select the **Show Tabs in Full Screen** command.
- Press the F11 key.



You can change the **Show Tabs in Full Screen** shortcut key in the **Environment Options** dialog, **Keyboard** pane. For more information, see "Assigning Shortcut Keys" on page 104.

You can turn off the diagram full screen mode in the same way as turning it on.

Managing Interface Components in Diagram Full Screen Mode

The Model Browser is in an auto hide mode when the diagram full screen mode is turned on. Model Browser tabs are located in the left side of the program window. To display the browser, move the mouse cursor over the desired browser window tab (for example, on the Containment tab) and the tab opens. For more information about working with windows in the auto hide mode, see "Model Browser" on page 75.

The main toolbar is hidden when the diagram full screen mode is turned on. To display the main toolbar, set the **Hide Toolbars in the Full Screen Mode** property to *false* in the **Environment Options** dialog, **General** pane, **Display** group.

The diagram toolbar is displayed when the diagram full screen mode is turned on. Right-click the diagram toolbar to manage it.

Copyright © 1998-2015 No Magic, Inc.

Specifying Diagram Properties

You can specify diagram properties in the diagram Specification window.

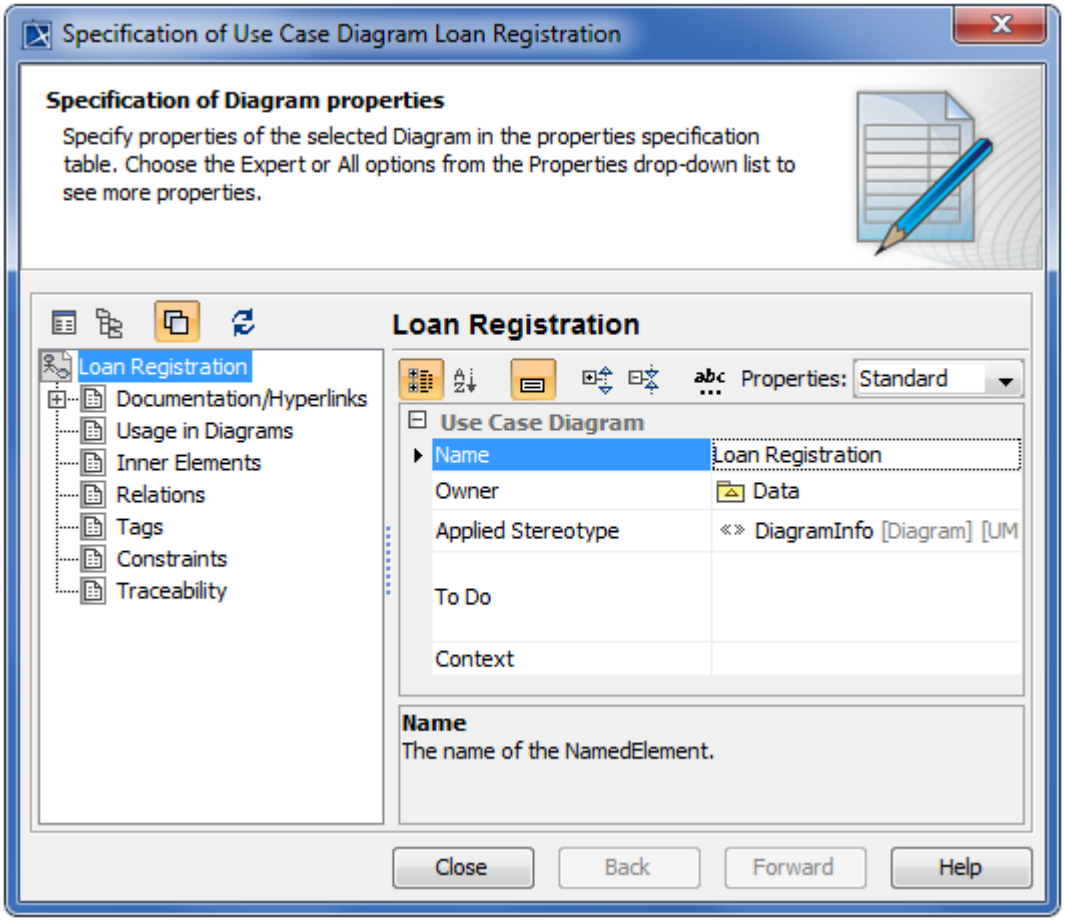


Figure 104 -- Diagram Specification window

TIP!

For more information on how to work with Specification windows, see "Specification Window" on page 273.

Specifying Diagram Style Properties

Customize the diagram style (color, grid) in the **Diagram Properties** dialog.

To open the **Diagram Properties** dialog

- Select **Diagram Properties** from the diagram shortcut menu.
- On the **Edit** menu, point to **Symbol**, and then click **Diagram Properties**.
- Press SHIFT+ENTER.

Copyright © 1998-2015 No Magic, Inc.

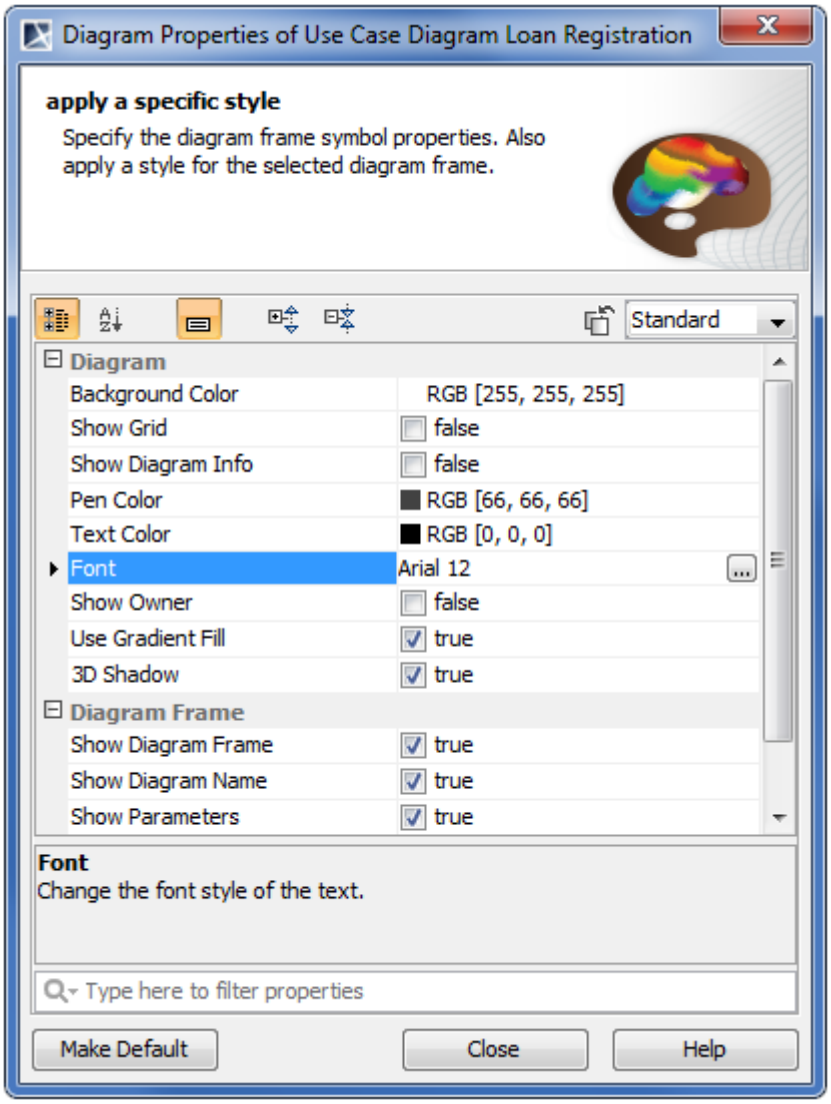


Figure 105 -- Diagram Properties dialog

Diagram Name and its Context Name Synchronization

The diagram name and its context name are synchronized automatically.

EXAMPLE

Create an Activity diagram. Type a name for the Activity diagram, for example, Receive. The name of the Activity automatically changes to Receive. And conversely - change the name of the Activity and the Activity diagram name will be changed automatically. This is synchronization of a diagram name and its context name:

Synchronization works in the following cases:

- Activity and Activity diagram inside.
- Interaction and Communication or Sequence diagram inside.
- (Protocol) State Machine and (Protocol) State Machine diagram inside.
- Class and all available inner diagrams inside.

Copyright © 1998-2015 No Magic, Inc.

To turn off the synchronization

Clear the **Synchronize the diagram name with it's context name** check box in the **Environment Options** dialog box, **General** branch, and **Editing** group.

NOTE If the second diagram will be created in the branch, diagram names will not be synchronized.

Diagram Frame

All UML diagrams have the content area that is restricted by the diagram frame as it is stated in UML specification. The diagram frame is primarily used in cases where the diagrammed element has graphical border elements (like ports for classes and components, entry/exit points on statemachines).

By default the diagram frame is displayed on the new diagram pane. The frame is a rectangle in all diagrams, but state machine and activity. State machine and activity diagram frames have rounded corners.

The frame can be resized manually by dragging its corners or borders.

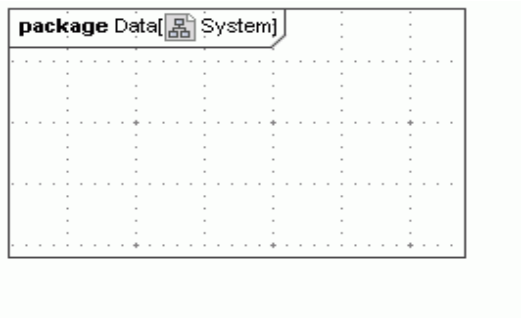


Figure 106 -- Diagram frame

To hide the diagram frame

Do either:

- **NEW!** Right-click the diagram and on the shortcut menu, click to clear the **Show Diagram Frame** check box.
- Open the **Diagram Properties** dialog and set the **Show Diagram Frame** property to *false*.

To change the diagram frame properties

Do either:

1. From the **Options** menu, choose **Project**. The **Project Options** dialog opens.
2. In the options tree, expand **Symbol styles** and then click **Diagram**. The **Diagram** options are displayed on the **Diagram** pane.
3. In the **Diagram Frame** options subgroup, change the desired options.

Or:

1. Open the **Diagram Properties** dialog (the ways for opening the dialog are described in Section "Specifying Diagram Style Properties" on page 205).
2. Change the properties in the **Diagram Frame** property group.

NOTE You can see the selected option description in the description area of the **Project Options** dialog. For more information about the P**roject Options** dialog, see "Using Project Options dialog" on page page 130.

Copyright © 1998-2015 No Magic, Inc.

To hide the icon on the diagram frame

1. Open the **Diagram Properties** dialog (the ways for opening the dialog are described in Section "Specifying Diagram Style Properties" on page 205).
2. Change the **Show Stereotypes** property value to **Text**.

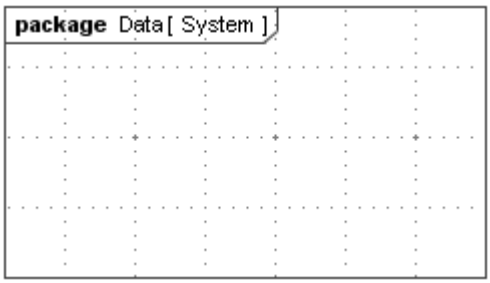


Figure 107 -- Diagram Frame with hidden diagram icon

When a new diagram is created, the diagram stereotype icon is displayed on the diagram frame header by default, though you can change it to the context stereotype icon.

To display the context stereotype icon instead of the diagram stereotype icon on the diagram frame

1. Open the **Diagram Properties** dialog (the ways for opening the dialog are described in Section "Specifying Diagram Style Properties" on page 205).
2. In the **Diagram Frame** property group, change the **Use Stereotype** property value to **Context**. The context stereotype icon will be displayed instead of the diagram stereotype icon in the diagram frame header.



The **Use Stereotype** property takes effect only in case the **Show Stereotypes** property value is **Text and Icon** or **Icon**.

There is a possibility to show the abbreviation of a diagram type instead of the full diagram type in the diagram frame header.

To display the abbreviated diagram type

1. Open the **Diagram Properties** dialog (the ways for opening the dialog are described in Section "Specifying Diagram Style Properties" on page 205).
2. In the **Diagram Frame** property group, select the **Show Abbreviated Type** check box.

The abbreviated diagram types are listed in the following table.

Diagram name	Abbreviation
Use Case	uc
Communication	comm
Sequence	sd
State Machine	stm
Activity	sct
Composite Structure	cs

Copyright © 1998-2015 No Magic, Inc.

Drawing Diagram Shapes

You can draw diagram shapes in any type of diagram.



For more information about working with shapes,
see "Working with Symbols of Model Elements" on page 224.

To display a diagram shape on a diagram pane

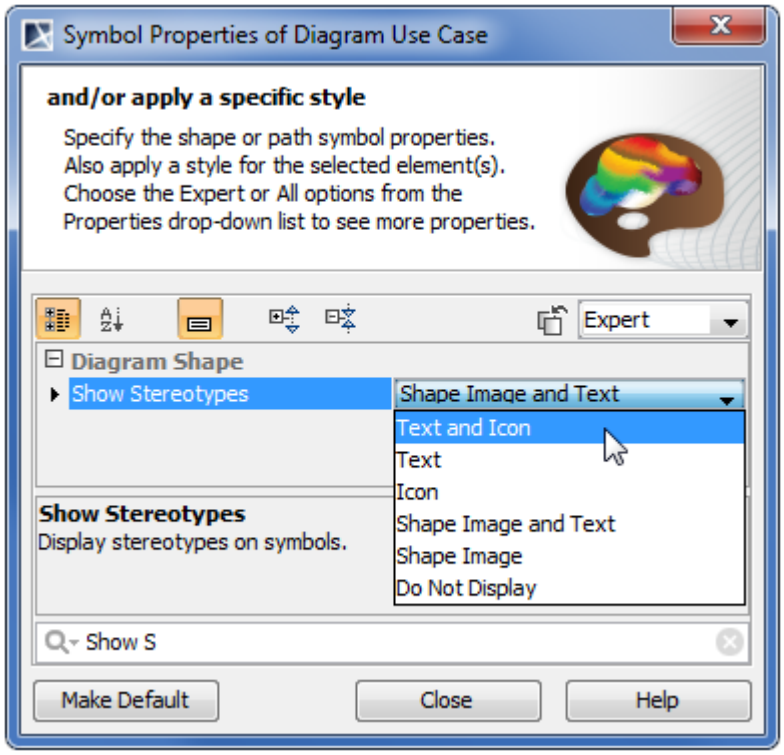
1. Select the diagram in the Model Browser.
2. Drag the diagram to the diagram pane. The diagram shape will be drawn on the diagram pane (see the example in the following picture).



You can change the representation of the diagram shape by changing the diagram stereotype display mode.

To change the representation of the diagram shape

1. Right-click the shape and from the shortcut menu select **Symbol Properties**.
2. In the **Symbol Properties** dialog select a new value for the **Show Stereotypes** property (see the following picture).



You can also select showing or hiding diagram constraints or tagged values on the diagram shape.

To show diagram stereotype constraints and tagged values on the diagram shape

1. Right-click the diagram shape and from the shortcut menu select **Symbol Properties**.
2. In the **Symbol Properties** dialog select the **Show Constraints** and **Show Tagged Values** check boxes.

Copyright © 1998-2015 No Magic, Inc.

To display the context stereotype icon instead of the diagram stereotype icon on the diagram shape

1. Right-click the diagram shape and from the shortcut menu select **Symbol Properties**.
2. In the **Symbol Properties** dialog, change the **Use Stereotype** property value to **Context**. The context stereotype icon will be displayed instead of the diagram stereotype icon on the diagram shape.



The **Use Stereotype** property takes effect only in case the **Show Stereotypes** property value is **Text and Icon** or **Icon**.

There is a possibility to show the abbreviation of a diagram type instead of the full diagram type on the diagram shape.

To display the abbreviated diagram type

- From the diagram shape shortcut menu, open the **Symbol Properties** dialog box and select the **Show Abbreviated Type** check box.

The abbreviated diagram types are listed at the end of Section "Diagram Frame" on page 207.

Overviewing Other Diagrams



This feature is available in Standard, Professional, Architect, and Enterprise editions.



You can also learn about overviewing diagrams while analyzing the Diagram overview sample.

To open the sample, do any of the following:

- On the Welcome screen, select **Samples** and then in the **Product Features** section click **Diagram overview**.
- Go to *<MagicDraw installation directory>\samples\product features* and open the *diagram overview.mdzip* file.

As of version 17.0 you can overview other diagrams, including dependency matrices, tables, and relation maps, on a diagram pane. For this the diagram overview shape can be used.

Read and learn how to use the diagram overview shape in the following sections:

- "Creating a diagram overview shape" on page 210.
- "Modifying the content of a diagram overview shape" on page 214.
- "Diagram overview shape environment" on page 217.
- "How to..." on page 221.

Creating a diagram overview shape

You can use one of the following ways to create a diagram overview shape:

- **Drag the diagram shape from the Model Browser and change it into the diagram overview shape afterwards.** This way created diagram overview shape *shows the full content* of the corresponding diagram (see Figure 108 on page 214) and is updated automatically according to *all changes* made in this diagram. For detailed description of the procedure see section "To create a diagram overview shape by using a drag-and-drop operation" on page 211.

Copyright © 1998-2015 No Magic, Inc.

- **Use the**  **Diagram Overview button that is located in the diagram pallet.** This way created diagram overview shape shows *the full content* of the corresponding diagram (see Figure 108 on page 214) and is updated automatically according to *all changes* made in this diagram. For detailed description of the procedure see section "To create a diagram overview shape by using the diagram pallet" on page 212.
- **Paste a copied part of the diagram's content using a special command from the main menu.** This way created diagram overview shape *shows any copied part* of the corresponding diagram's content (see Figure 109 on page 214) and is updated automatically according only to *the changes made in this copied part*. Note that this way cannot be used for dependency matrixes, tables, and relation maps, as it is not allowed to copy and paste different parts of them. For detailed description of the procedure see section "To create a diagram overview shape by pasting any part of diagram's content" on page 213.

To create a diagram overview shape by using a drag-and-drop operation

1. Open a diagram wherein you want to create the diagram overview shape.
2. In the Model Browser select a diagram you want to overview.
3. Drag the diagram shape to the opened diagram pane. The diagram shape will be drawn on the diagram pane (see the example in the picture below).



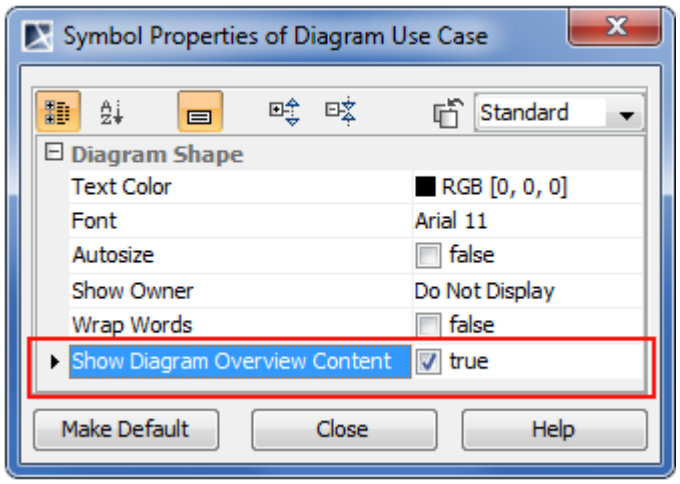


If you cannot drag the diagram shape to the opened diagram pane within a teamwork project, make sure you have the right to edit model of this project and then try to lock for edit this diagram.



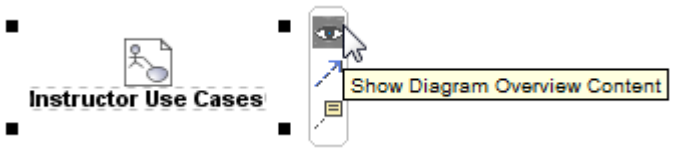
For more information about locking elements, see "Locking model elements and diagrams for editing" on page 1055.

4. Make the diagram shape show the full content of the corresponding diagram using any of the following GUI features:
   - Right-click the diagram shape and from the shortcut menu select **Symbol Properties**. In the opened dialog, set the **Show Diagram Overview Content** property value to *true*.



Copyright © 1998-2015 No Magic, Inc.

- Click the diagram shape and on the smart manipulator toolbar select the button.



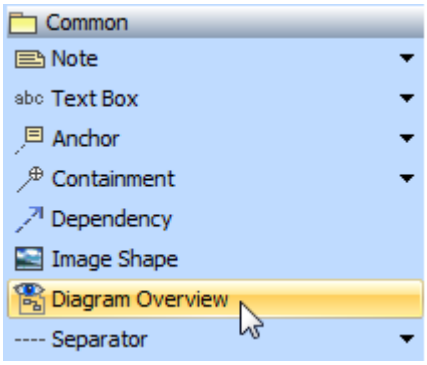
The diagram shape will be changed to the diagram overview shape that shows the full content of the corresponding diagram (see Figure 108 on page 214).

NOTE The content of this way created diagram overview shape will be updated automatically according to *all changes* made in the corresponding diagram.

TIP! Double-click the diagram overview shape to open the corresponding diagram.

To create a diagram overview shape by using the diagram pallet

1. Open a diagram wherein you want to create the diagram overview shape.
2. On the diagram pallet click the **Diagram Overview** button that is located within the **Common** buttons group.

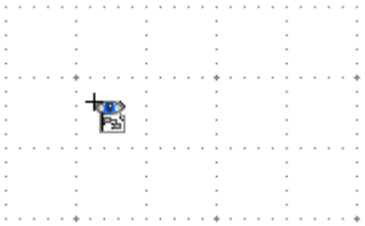


NOTE If the diagram pallet is inactive within a teamwork project, make sure you have the right to edit model of this project and then try to lock for edit this diagram.

TIP! For more information about locking elements, see “Locking Model Elements and Diagrams for Editing” in MagicDraw Teamwork UserGuide.pdf.

The **Select Diagram** dialog will open (to learn more about using the dialog please refer to section "Selecting an Element" on page 352).

3. Select a diagram for which you want create a diagram overview shape and click **OK**.
4. Click a free space of the diagram pane.



The created diagram overview shape will show the full content of the corresponding diagram (see Figure 108 on page 214)

Copyright © 1998-2015 No Magic, Inc.

.

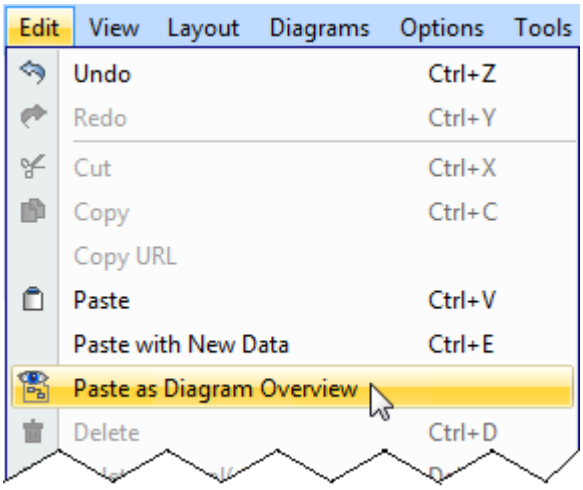
NOTE The content of this way created diagram overview shape will be updated automatically according to *all changes* made in the corresponding diagram.

TIP! Double-click the diagram overview shape to open the corresponding diagram.

To create a diagram overview shape by pasting any part of diagram's content

NOTE This way cannot be used for dependency matrixes, tables, and relation maps, as it is not allowed to copy and paste different parts of them.

1. Open a diagram for that you want to create the diagram overview shape.
2. Select the element shapes that you want to appear in the diagram overview shape and then copy them.
3. Open the diagram wherein you want to create the diagram overview shape.
4. On the **Edit** menu, click **Paste as Diagram Overview**.



NOTE If the command is inactive within a teamwork project, make sure you have the right to edit model of this project and then try to lock for edit the diagram wherein you want to create the diagram overview shape.

TIP! For more information about locking elements, see "Locking Model Elements and Diagrams for Editing" in MagicDraw Teamwork UserGuide.pdf.

The created diagram overview shape will show only the copied part of the corresponding diagram's content (see Figure 109 on page 214)

NOTE The content of this way created diagram overview shape will be updated automatically according only to *the changes made in the copied part* of the corresponding diagram.

TIP! Double-click the diagram overview shape to open the corresponding diagram.

Copyright © 1998-2015 No Magic, Inc.

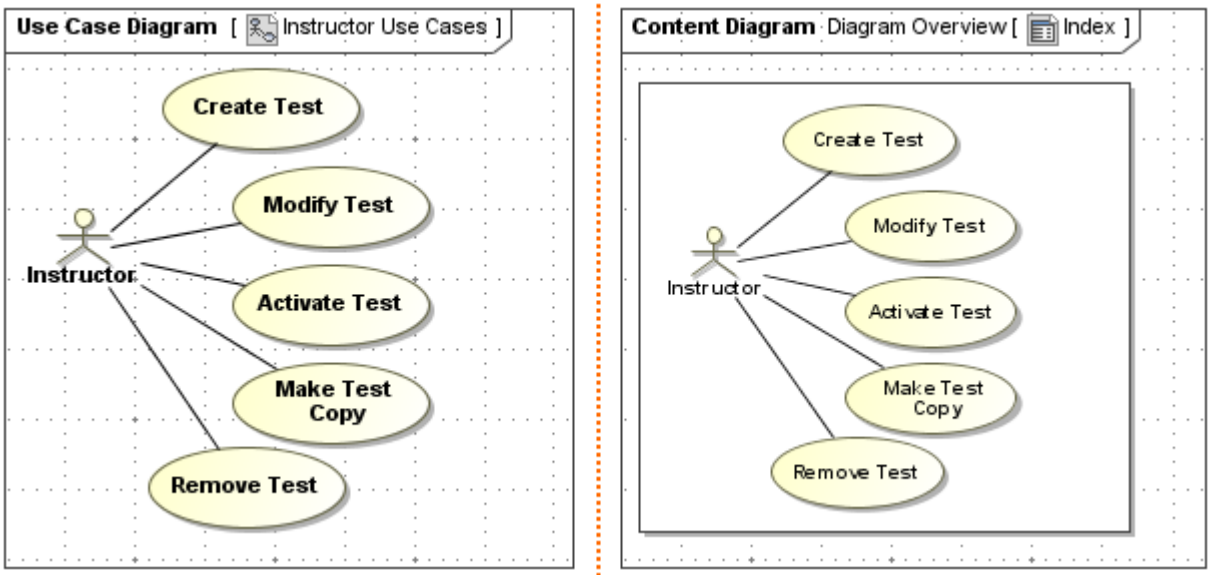


Figure 108 -- An example of diagram and its diagram overview shape showing the full content of the diagram

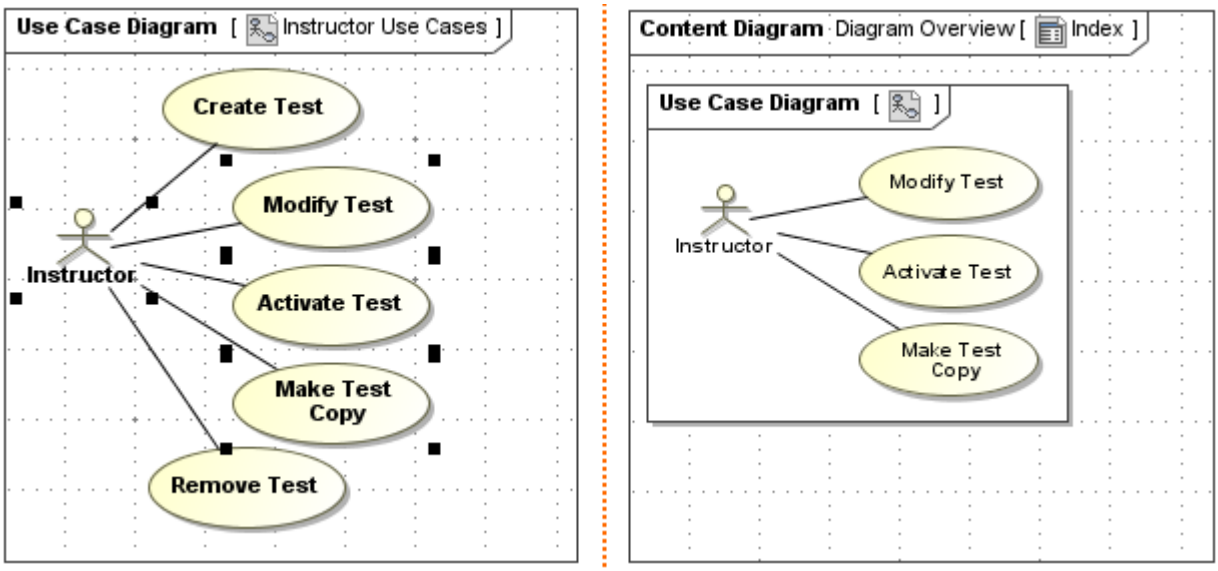


Figure 109 -- An example of diagram and its diagram overview shape showing only the part of the diagram content

Modifying the content of a diagram overview shape

Once the diagram overview shape is created, you can modify its content: add more element shapes or remove no more needed ones by using the **Compartment Edit** dialog.

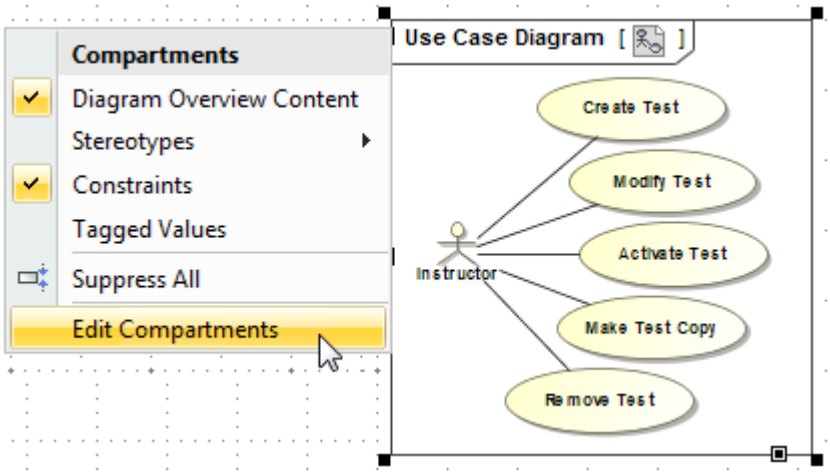
To open the **Compartment Edit** dialog for modifying the content of a diagram overview shape

Do any of the following:

- Right-click the diagram overview shape, on the shortcut menu point to **Edit Compartments**.

Copyright © 1998-2015 No Magic, Inc.

- Click the diagram overview shape and on the shape area click the "..." *Compartments* smart manipulator. Then on the menu, click **Edit Compartments**.



There are two lists in the **Diagram Overview Content** tab:

Hidden	• If the **Hidden** list is *not empty*, lists element shapes that are not shown in the diagram overview shape. • If the **Selected** list is *empty*, lists all element shapes meaning that all of them are shown in the diagram overview shape. In this case the diagram overview shape will reflect any changes in the corresponding diagram.
Selected	Lists element shapes that are shown in the diagram overview shape. In this case the diagram overview shape will reflect only changes of the selected shapes in the corresponding diagram.

Copyright © 1998-2015 No Magic, Inc.

If the diagram overview shape is created by using instructions given either in the first or the second procedure in section "Creating a diagram overview shape" on page 210, then you will see that all element shapes are listed in the **Hidden** list within the **Compartment Edit** dialog.

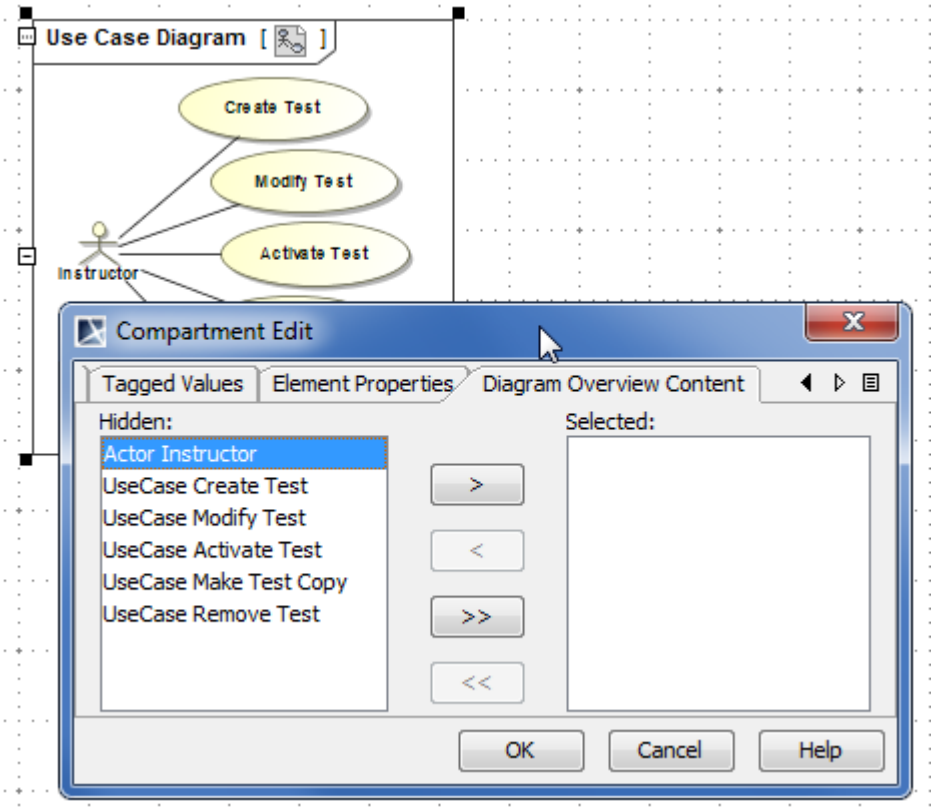


Figure 110 -- An example of the Compartment Edit dialog in case of showing the full content of the diagram in the diagram overview shape

Copyright © 1998-2015 No Magic, Inc.

If the diagram overview shape s created by using instructions given in the third procedure in section "Creating a diagram overview shape" on page 210, then you will see that only the element shapes which are displayed in the diagram overview shape are listed in the **Selected** list (even in case of copying all shapes).

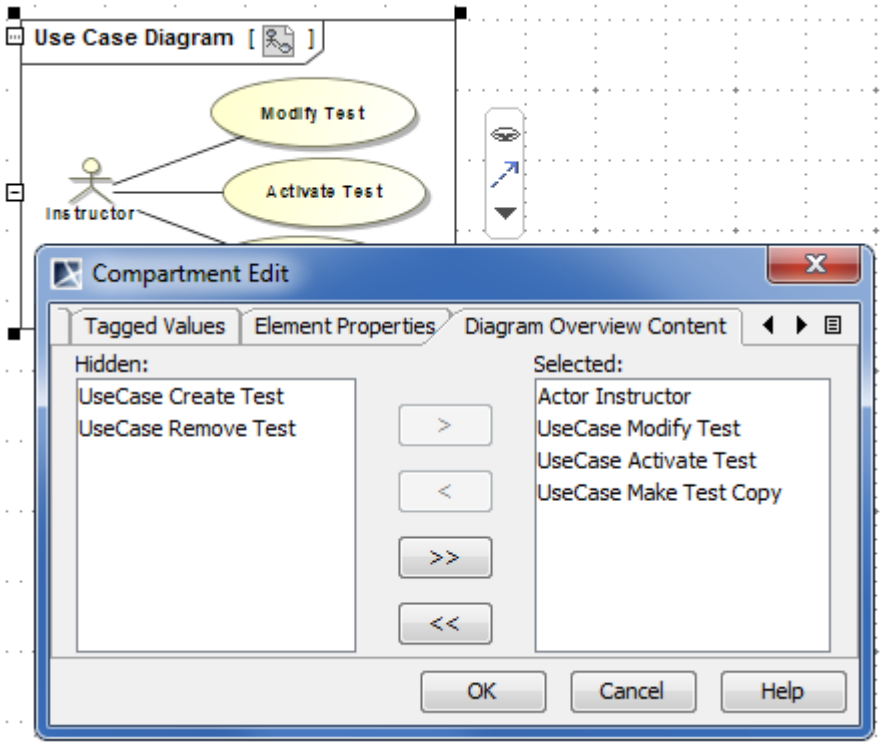


Figure 111 -- An example of the Compartment Edit dialog in case of showing only the part of the diagram content in the diagram overview shape

To modify the content of the diagram over view shape

1. Open the **Compartment Edit** dialog using one of the ways described in the procedure above.
2. Use the **>**, **>>**, **<**, and **<<** buttons to manage item moving between the lists. Remember the rules of showing and not showing shapes included in each list.
3. Click **OK** when you are finished.

Diagram overview shape environment

This section describes GUI features that can be used for handling the diagram overview shape:

- "Shortcut menu" on page 218.
- "Symbol Properties dialog" on page 219.
- "Smart manipulator toolbar" on page 220.
- "Shape area buttons" on page 221.

Copyright © 1998-2015 No Magic, Inc.

Shortcut menu

This section introduces diagram overview shape-specific commands, but not all commands from the diagram overview shape shortcut menu.

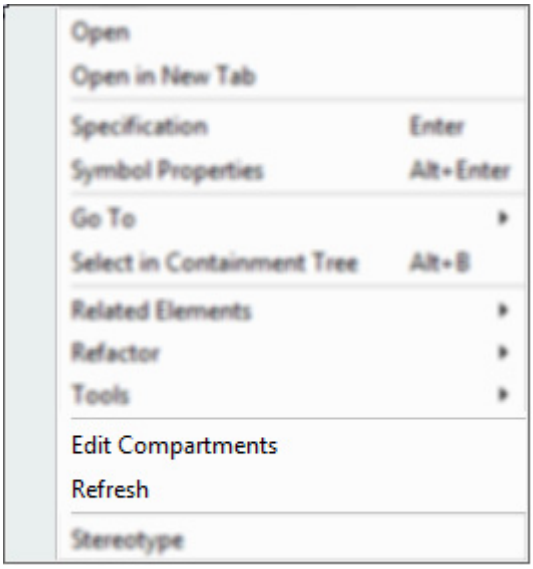


Figure 112 -- Diagram overview shape-specific commands in the shortcut menu

Command	Description
Edit Compartments	Opens the **Compartment Edit** dialog. **TIP!** You can also open this dialog by clicking the **...** *Compartments* smart manipulator and then **Edit Compartments**. The **...** *Compartments* smart manipulator is on the frame of a diagram overview shape area (to see the smart manipulator, click the diagram overview shape).
Refresh	Updates the content of the diagram overview shape according to changes in the model. **NOTE:** This button is available in the shortcut menu only if a dependency matrix, table, or relation map is displayed in the diagram overview shape. **TIP!** You can also update the content of the diagram overview shape by clicking the smart manipulator that is on the frame of a diagram overview shape area (to see the smart manipulator click the diagram overview shape).

Copyright © 1998-2015 No Magic, Inc.

Symbol Properties dialog

This section introduces diagram overview shape-specific properties, but not all properties in the **Symbol Properties** dialog of a diagram overview shape.

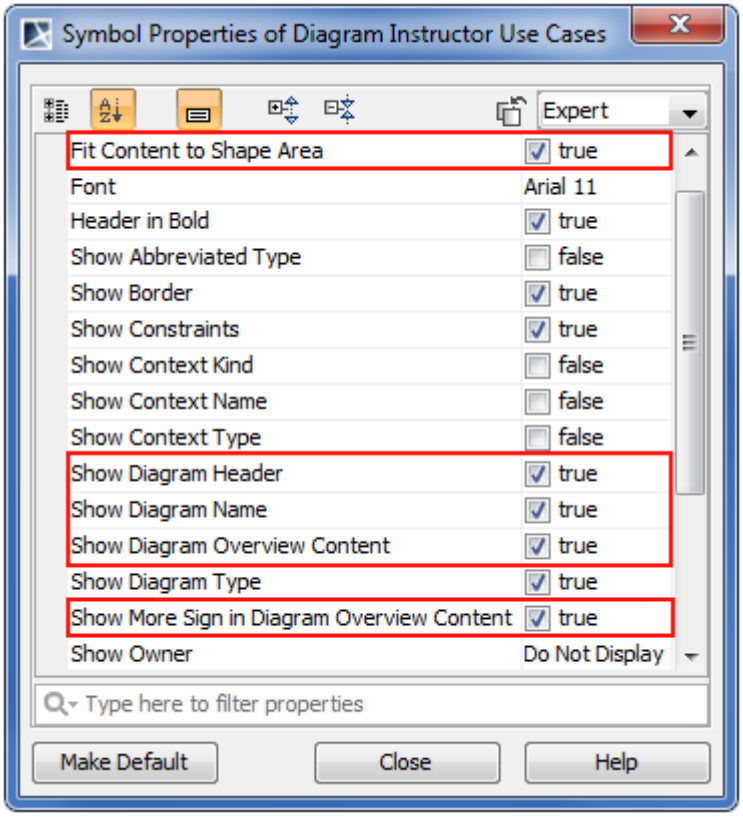


Figure 113 -- Diagram overview shape-specific properties in the Symbol Properties dialog

Property	Description
Fit Content to Shape Area	Select to adjust the zoom ratio of the diagram content so that it fits the size of the diagram overview shape area. Unselect to fix the current zoom ratio of the diagram content. It will persist in any size change. **NOTE:** Any value setting for this property can take effect on the diagram overview shape in case the **Autosize** property value is *false*. If you need to reset the diagram content ratio to 1:1, set the **Autosize** property value to *true*.
Show Diagram Header	Select to show the diagram header in the diagram overview shape.
Show Diagram Name	Select to show the diagram icon and name in the diagram header of the diagram overview shape, in case the diagram header is displayed, i.e., the **Show Diagram Header** property value is *true*.
Show Diagram Overview Content	Select to change the diagram shape into the diagram overview shape that shows the content or a part of the content of a corresponding diagram. Unselect to change the diagram overview shape into the diagram shape. For more information about using this command refer to step #4 of the procedure "To create a diagram overview shape by using a drag-and-drop operation" on page 211.

Copyright © 1998-2015 No Magic, Inc.

Property	Description
Show More Sign in Diagram Overview Content	Select to display the more sign at the bottom of the diagram overview shape area, when only a part of the diagram content is displayed the diagram overview shape (see the following figure).

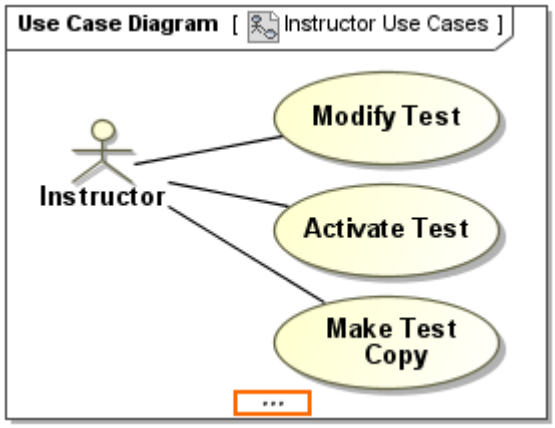


Figure 114 -- An example of the more sign on diagram overview shape area

Smart manipulator toolbar

In the smart manipulator of the diagram (overview) shape there are two buttons that need to be described.

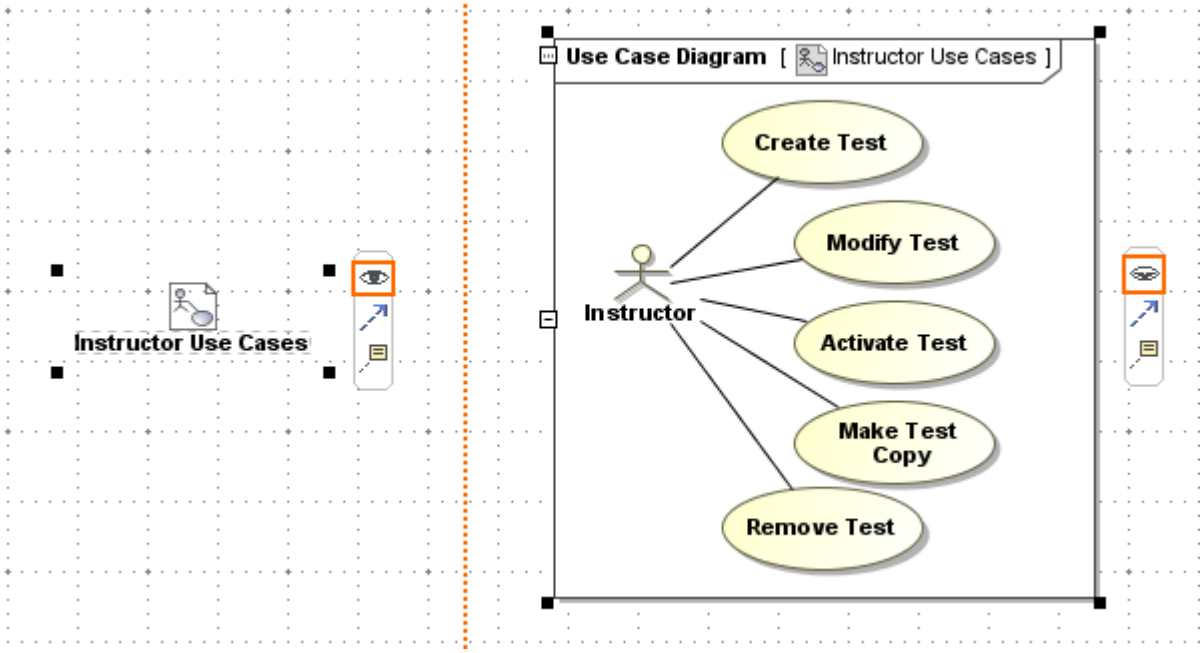


Figure 115 -- Buttons for showing (on the left) and hiding (on the right) the diagram overview content in the smart manipulator toolbar

Button	Description
	Changes the diagram shape to the diagram overview shape that shows in its content the selected-to-overview shapes of the corresponding diagram. For more information about using this command refer to step #4 of the procedure "To create a diagram overview shape by using a drag-and-drop operation" on page 211.

Copyright © 1998-2015 No Magic, Inc.

Button	Description
	Changes the diagram overview shape to the diagram shape.

Shape area buttons

There are different sets of buttons on the diagram overview shape area showing a diagram and the one showing a dependency matrix, table, or relation map.

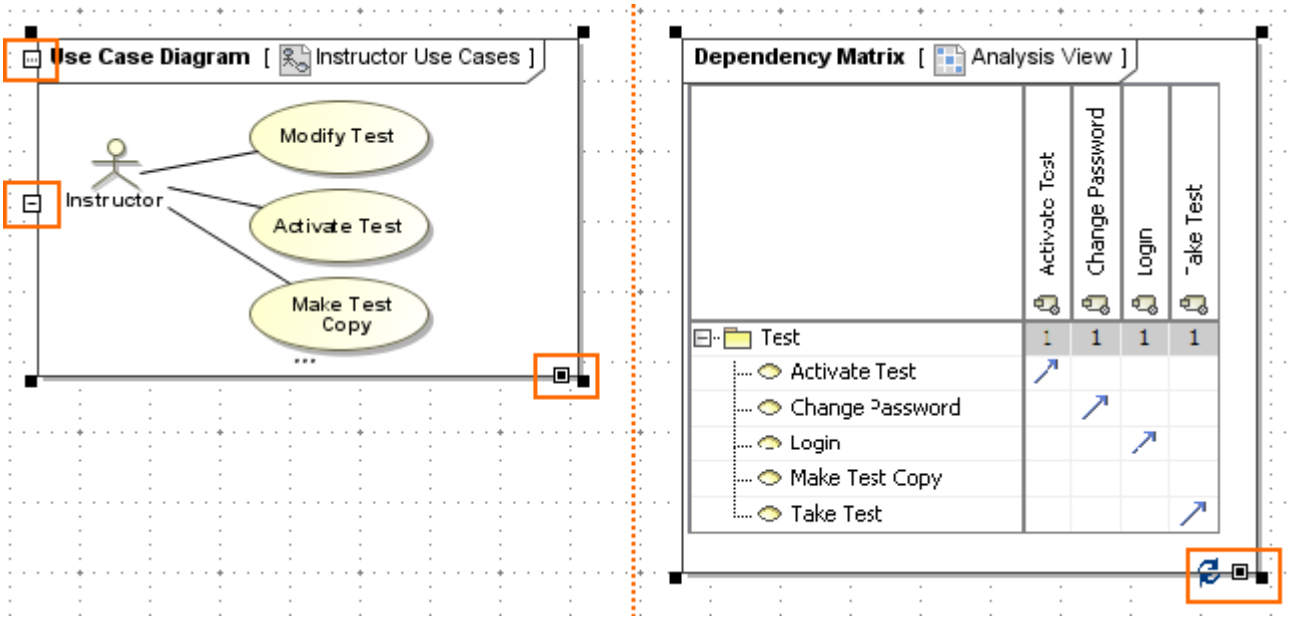


Figure 116 -- Diagram overview shape area buttons in case of overviewing diagram (on the left) and dependency matrix (on the right)

Button	Description
	Opens the *Compartments* menu. For more information, see "Smart Manipulators" on page 235.
	Suppresses the diagram overview content.
	Updates the content of the diagram overview shape according to changes in the model. **NOTE:** This button is available on the diagram overview shape area only if a dependency matrix, table, or relation map is displayed in the diagram overview shape.
	Resets the diagram overview shape's content ratio to 1:1.

How to...

This section gives some remarks that can be useful when struggling to arrange sizes of your diagram overview shapes. The solutions we offer are as follows:

- "How to make a couple of diagram overview shapes to be of the same size?" on page 222.
- "How to set desired shape sizes in the a diagram overview shape?" on page 222.

Copyright © 1998-2015 No Magic, Inc.

How to make a couple of diagram overview shapes to be of the same size?

Let's say you have two diagram overview shapes. You want their zoom ratio to be 1:1 and shape areas to be of the same height (or width) at the same time.

Solution

1. For both diagram overview shapes set the **Fit Content to the Shape Area** property value to *false*.
2. For both diagram overview shapes set the **Autosize** property value to *false*.
3. Make both diagram overview shapes of preferred size.
4. Make both diagram overview shapes of the same height (or width) by dragging corners of the shape areas.

How to set desired shape sizes in the a diagram overview shape?

Solution

1. Set the **Fit Content to the Shape Area** property value to *true* for this diagram overview shape.
2. Set the desired shape content size by dragging corners of the shape area.
3. Set the **Fit Content to the Shape Area** property value to *false* for this diagram overview shape.

In case the diagram overview shape shows the full content of the corresponding diagram, all new shapes added to the diagram pane in the diagram overview shape will be of the previously set size.

If you want to reset the shape sizes to ratio zoom 1:1, do any of the following:

- Set for the diagram overview shape the **Autosize** property value to *true*.
- Make the diagram overview shape of the preferred size.

Table with Diagram Information

On the diagram, you can display a table containing various diagram details: its name, author, status, the dates it was created and modified, etc. By default, the Diagram info table is displayed at the right top corner of the diagram frame, but you can drag and drop it to any other position on the diagram.

The table includes the following fields:

- Diagram name
- Author
- Creation date
- Modification date
- Last Modified by
- Other available tag definitions

To show the table containing the diagram information

1. From the diagram shortcut menu, select **Show Diagram Info**.

Copyright © 1998-2015 No Magic, Inc.

2. The table with the predefined information will be displayed on the diagram.

Diagram name	Requirements
Author	andre
Creation date	5/18/07 9:02 AM
Modification date	2/26/10 11:34 AM
Last modified by	martin

To define information that will be included in the table

1. From the diagram info table shortcut menu, select **Customize**, or from the **Options** menu, select **Project**. The **Project Options** dialog box opens.
2. Open the **Diagram Info** pane.
3. In the **Source** pane, select the type of information you want to include in the table: **Standard Mode** or **Custom Mode**.
4. **Standard Mode** contains the following fields that will be shown in the table: Author, Creation date, Modification date, and all other tag definitions that can be assigned to the diagram. In the **Custom Mode** field, you can create your own table or any other object in HTML.
5. Preview the selected table or other created object in the **Preview** pane.

Changing the Diagram Type

The diagram type can be changed to another type of diagram if both diagram types are compatible. Note: diagram elements are not converted.

Changing the diagram type is usable:

- To migrate with existing project to the a diagram type which was not available till then. For example, to migrate Class diagram to the SysML Block Definition diagram.
- To migrate with existing project from the diagram type, which will be dropped from your project. For example, if user has decided to drop plug-in, and needs to convert plugin specific diagram to standard MagicDraw diagrams.

Diagram conversion scenarios:

- Any static diagram can be converted to another type of static diagram.
- Any dynamic diagram can be converted to another diagram, if both diagrams are based on the same diagram type and diagrams are compatible.

To change the diagram type:

1. Select one or more the same type diagrams in the Browser (Containment or Diagrams tree).
2. From the shortcut menu, choose command **Refactor** > **Change Type To** and select desired diagram type from the list (see Figure 117 on page 224).

Copyright © 1998-2015 No Magic, Inc.

You can also convert multiple the same type diagrams – select them in the Browser and choose command **Refactor** > **Change Type To**.

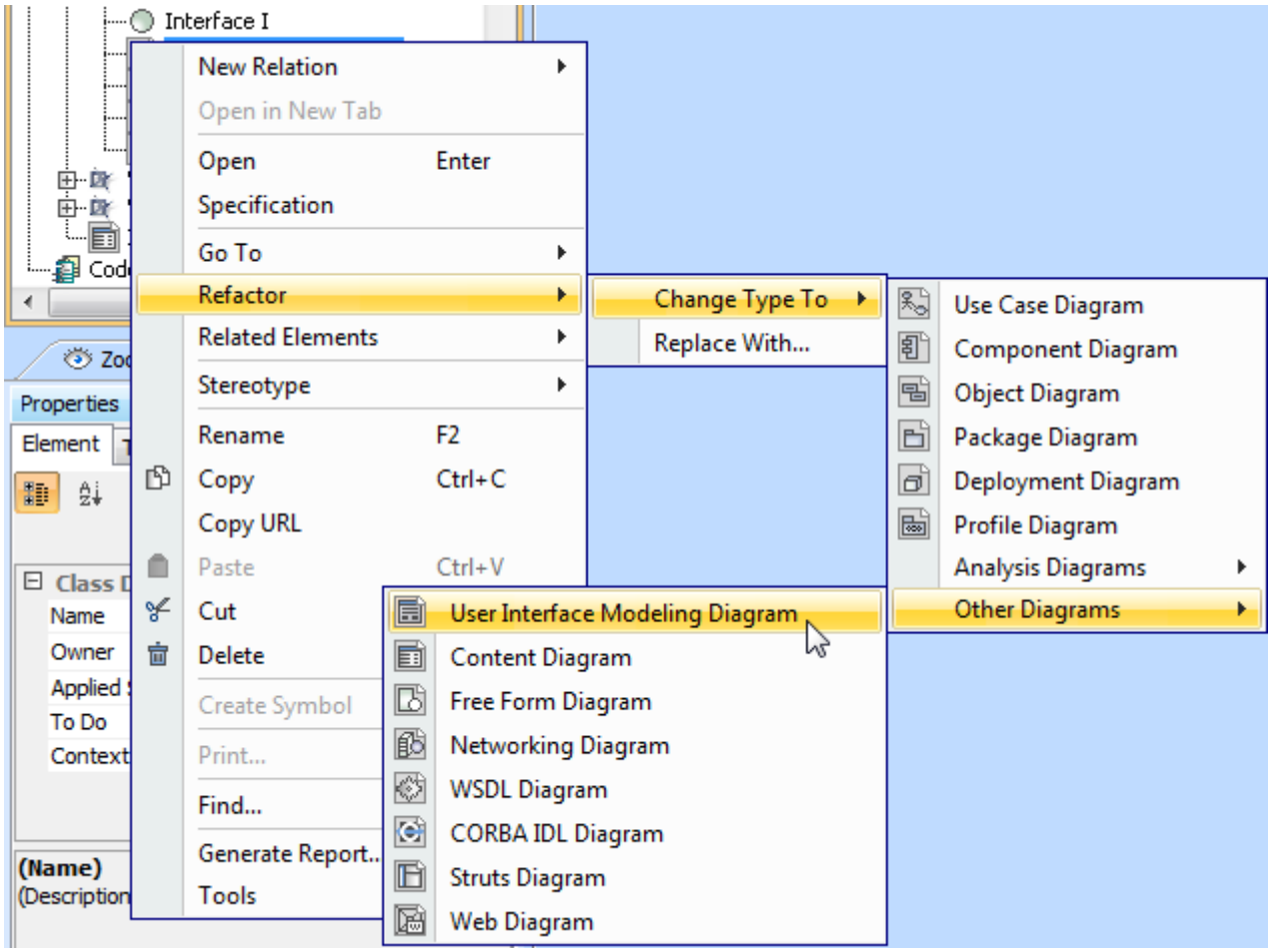


Figure 117 -- Changing the diagram type

Working with Symbols of Model Elements

DEMO Basics

To create an element on the diagram pane

1. Click the element button on the diagram pallet, or press the appropriate shortcut key for the element you wish to draw (the button remains pressed).
2. Click the desired location on the diagram pane. The new symbol is created on the diagram pane at the point you click.

TIP! For a detailed description of MagicDraw user interface, see "Understanding MagicDraw User Interface (UI)" on page 70.

To create several elements of the same type on the diagram pane

Do one of the following:

- Use the Shift key:

Copyright © 1998-2015 No Magic, Inc.

1. Press and hold the Shift key.
2. Click the element button on the diagram pallet.
3. Click the diagram pane. With each click, the new element of the same type will be created.



After releasing the Shift key, you can create elements as usually.

- Use the *Sticky* button:
  1. On the diagram pallet, under the **Tools** category, click the *Sticky* button or press Z.
  2. On the diagram pallet click the element button.
  3. Click the diagram pane. With each click, the new element of the same type will be created.



To turn the sticky mode off, click the *Sticky* button or press Z.

To create a symbol of the existing element

1. Activate a diagram on which you wish to draw a symbol.
2. From the Model Browser, select an item you wish to draw.
3. Drag and drop the selected model element onto the diagram pane.

To specify the name of the element

Do one of the following:

- Specify the name in the Specification window:
  1. Double-click the symbol or select **Specification** from the symbol shortcut menu. The corresponding **Specification** window opens.
  2. Type the element name in the **Name** text box and click **Close**.
- Type the symbol name directly on the selected symbol on the diagram pane.
- Type the element name after slowly double-clicking the shape in the Model Browser.

To create several symbols for one element

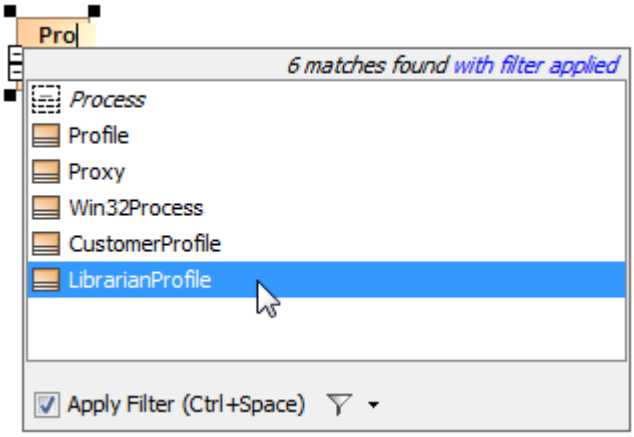
Do one of the following:

- In the Model Browser, select an element and drop it on the diagram pane. Repeat it if needed.
- On the diagram pane, select a symbol, copy and then paste by using Ctrl+C and then Ctrl+V.
- Open the name autocompletion list and then select the name of the already existing element. For more information, see the next topic.

Copyright © 1998-2015 No Magic, Inc.

To open the name autocompletion list when typing the element name

1. On the diagram pane, click the element shape and then click the element name area. The auto completion list with names of already created elements appears.



2. Type the beginning of the searched element name and the list is reduced according to the letters you typed. For example, if you are searching for *Profile* class, type the *Pro* letters and all classes, which begins with *Pro* will be shown in the list.

3. To narrow the search scope, you can customize scope filter options. For this, click  below the autocompletion list. Be sure the **Apply Filter** check box is selected! For more information about the commands for the search scope filter customization, see "Searching for Elements in Element Selection Dialog" on page 355.



These shapes will contain identical data.



- If you want to create the new element with the same name as the already existing element name, on the diagram pane type the name and press Enter. Note that two elements with the same names could not be created in the same owner, that is, owners of these elements should be different.
- If you attempt to enter an existing name in the corresponding **Specification** window, an error message alerts you to the existence of the current name of the symbol. You may not specify a name for a new symbol if another symbol of the same name and kind is already present in the package.

To resize selected shapes

- Drag the corner of the shape to the desired size.

To automatically resize selected shapes to a preferred size

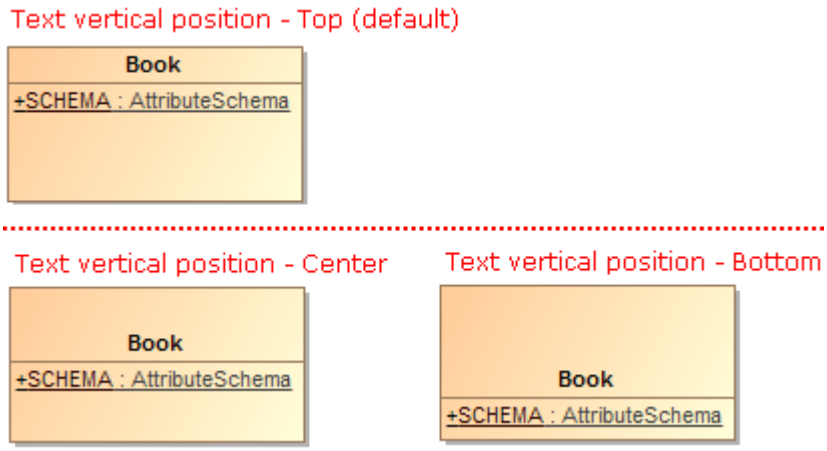
- Click the shape and then click the ▣ *Make Preferred Size* smart manipulator. Note that the smart manipulator is always displayed on any of the visible borders. Just select the shape, and the manipulator appears at the right of the shape's horizontal border or at the bottom of the vertical shape's border.

NEW! To change text and compartments vertical position on shapes

1. Right-click the symbol on the diagram, and then click **Symbol Properties**.

Copyright © 1998-2015 No Magic, Inc.

2. Click the **Text Vertical Position** value cell to open the list of available positions, and click the **Top**, **Center** or **Bottom** position.



To delete selected model elements

Do one of the following:

- From the **Edit** menu, select **Delete** (both data and symbol are deleted.)
- On the diagram toolbar, click the *Delete* button.
- Press Ctrl+D.
- In the Activity and Sequence diagrams, press the Delete key.

To delete selected symbols

Do one of the following:

- From the **Edit** menu, select **Remove from Diagram**.
- On the diagram toolbar, click the *Remove from Diagram* button.
- Press the Delete key.

IMPORTANT!

In the Activity and Sequence diagrams, all the steps described above, deletes not only symbols, but model elements too.

Working with Paths and Relationships

DEMO

Basics

To create a path between shapes

1. Click the appropriate path button on the diagram toolbar for the relationship you wish to draw.

TIP!

For a detailed description of the diagram toolbars, see "Toolbars" on page 72.

2. Click the first (source) shape of the path.

Copyright © 1998-2015 No Magic, Inc.

3. Drag the path to the second (target) shape of the path and drop it there.



When drawing a new path MagicDraw automatically chooses the optimal route for the path (that is, path avoids other shapes).



- Every diagram has the Manipulation Highlighting feature. When drawing a path between two model elements, you will see that those shapes are bordered with a red or blue rectangle. The red color indicates that the path can not be drawn between these shapes. Blue rectangle allows a path to be drawn.
- You can remove the manipulation highlighting in the **Environment Options** dialog box, **Diagram** section, **Edit** group. For more information, see "Customizing Environment Options" on page 96.
- For drawing a path, you can use smart manipulator toolbar, which appears near the element symbol. Select a path and drag it to the target shape. For more information, see "Smart Manipulator Toolbar" on page 235.

To remove the selected path between shapes

- To remove the selected path from the diagram, press Delete. After the deletion, the relationship will no longer be referenced by the deleted path (symbol), but still can be referenced by other appropriated symbols.
- To remove the selected relationship from the model, press Ctrl+D.

To change a path style

Do one of the following:

- On the diagram, right-click a path(s), and on the shortcut menu, click **Reset Labels Positions** or **Remove Break Points**.
- On the diagram, select a path(s), on the diagram toolbar, click the **Remove Break Points**, **Rectilinear,** or **Reset Labels Positions**.
- On the diagram, select a path(s), on the **Edit** menu, click **Paths**, and then click one of the following:
  - **Path Style** to select one of the path styles - rectilinear, oblique, or bezier.
  - **Change Path Style** (shortcut keys Ctrl+L) to change the path style to rectilinear, oblique, or bezier.
  - **Reset Labels Positions**.
  - **Remove Break Points** to remove all angles of the path.
- On the diagram, right-click a path(s), and on the shortcut menu, click **Symbol Properties**. Then in the **Symbol Properties** dialog, change the path style or other path's properties.

To route paths by using the Route Paths Rectilinearly button

Do one of the following:

- On the diagram, select a path, then on the smart manipulator toolbar, click the *Route Paths Rectilinearly* button. The selected path is routed rectilinearly.
- On the diagram, select a shape, then on the smart manipulator toolbar, click the *Route Paths Rectilinearly* button. All the paths that are connected to the shape are routed rectilinearly.

Copyright © 1998-2015 No Magic, Inc.

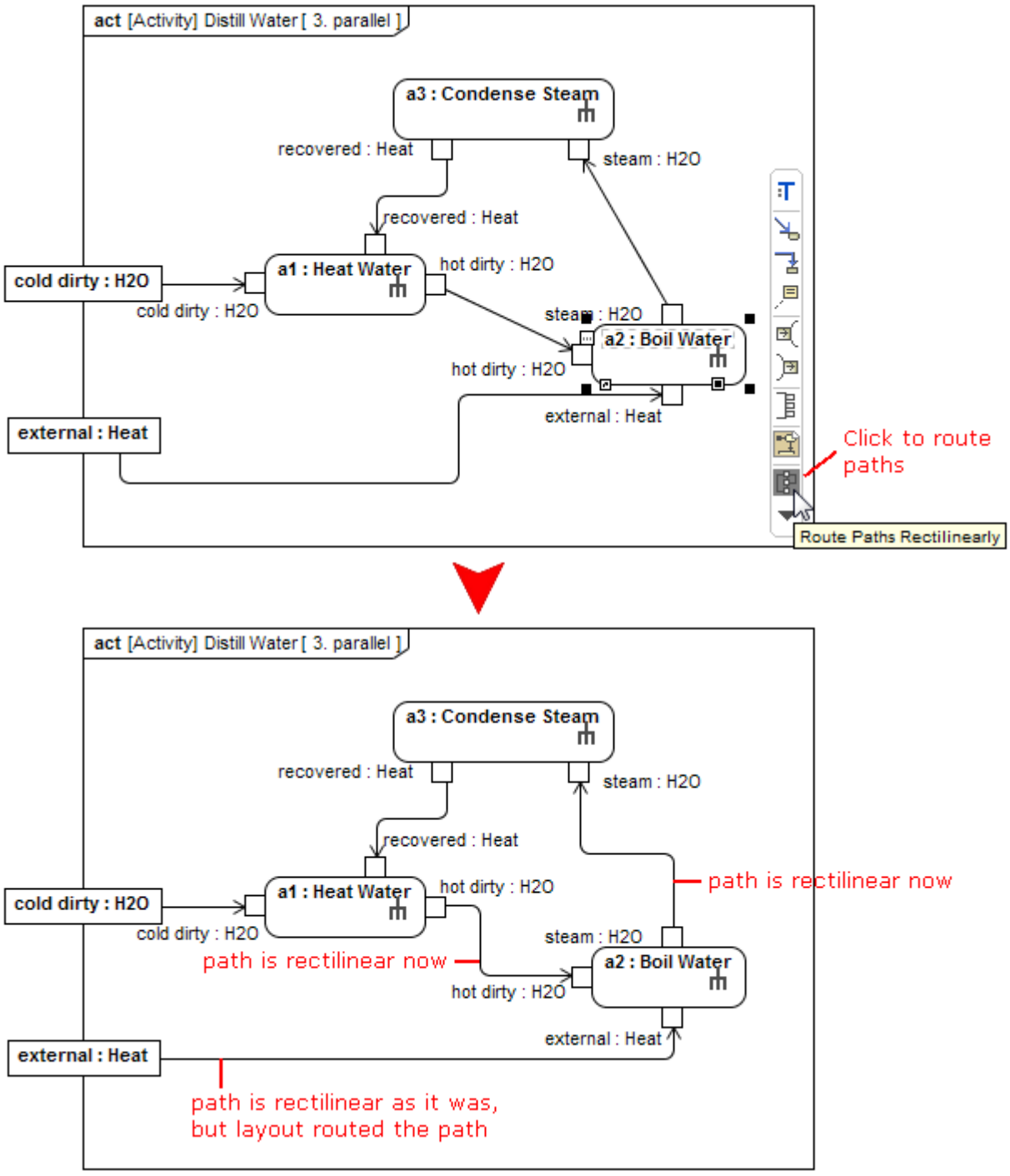


Figure 118 -- Example of routing paths rectilinearly

TIP!

For more information about path layout, see "Layout" on page 256.

To make the path corners rounded

- Right-click the path and click **Symbol Properties**. Then in the Symbol **Properties** dialog, click to select the **Rounded Corners** property.

Copyright © 1998-2015 No Magic, Inc.

To create line jumps

Line jumps represent an intersection of lines. If you have a large diagram with lots of intersecting paths, line jumps make the diagram easier to understand.

By default line jumps are not displayed.You can configure a diagram to display line jumps in the following way:

1. From the diagram shortcut menu, choose **Diagram Properties**. The **Diagram Properties** dialog box appears.
2. Change the **Add Line Jumps To** property.

The **Add Line Jumps To** property has the following options:

- *None*. Line jumps are not displayed. (Default value)

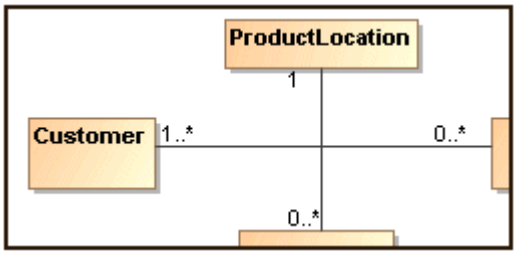


Figure 119 -- Diagram with no line jumps

- *Horizontal Line*. Line jumps are displayed on horizontal lines.

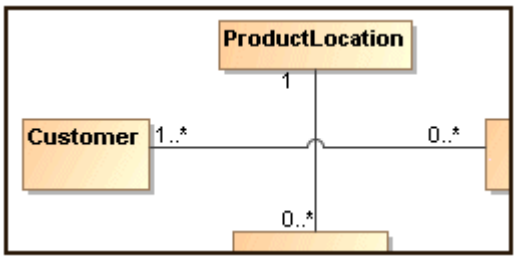


Figure 120 -- Diagram with horizontal line jumps

- *Vertical Line*. Line jumps are displayed on vertical lines.

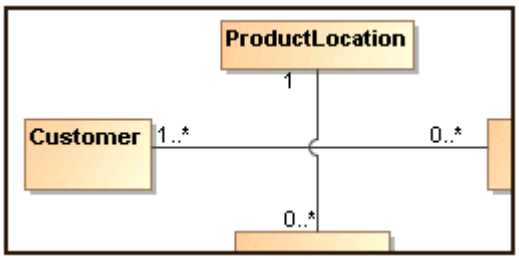


Figure 121 -- Diagram with vertical line jumps

Changing the **Add Line Jumps To** property for a particular diagram from within the diagram **Properties** dialog box, will change line jumps for the current diagram only.



For more information about how to change line jumps for the whole project, see "Style Engine" on page 324.

Copyright © 1998-2015 No Magic, Inc.

Inserting a Shape on the Path



This functionality is available in the State and Activity diagrams.

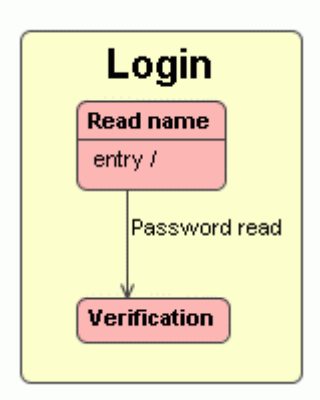
In the State and Activity diagrams you can split a path into two paths, by drawing a symbol on it. This is valid for Transition/Control Flow/Object Flow relationships and allowed to connect with these path elements.

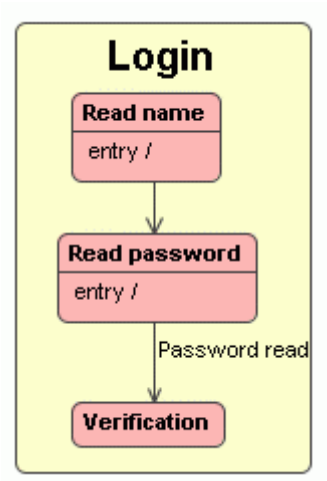
To insert a new shape splitting path on the diagram pane

1. Select the symbol you want to insert or click the diagram toolbar button to create a new one.
2. Drag it on the path. The path is highlighted in blue.
3. Drop the symbol. A Message dialog box appears asking if you want to insert the symbol on the path.

Possible solutions:

- **Before <path type>.** Symbol is inserted before the path. It means a new path is created, then the dropped element symbol is drawn and then the existing path is drawn. For example: *Password read* transition is drawn from *Read Name* state to *Verification* state. If you want to insert *the Read password* state before the *Password read* transition, drop the *Read password* state on the transition and in the open dialog, click the **Before Transition** button.

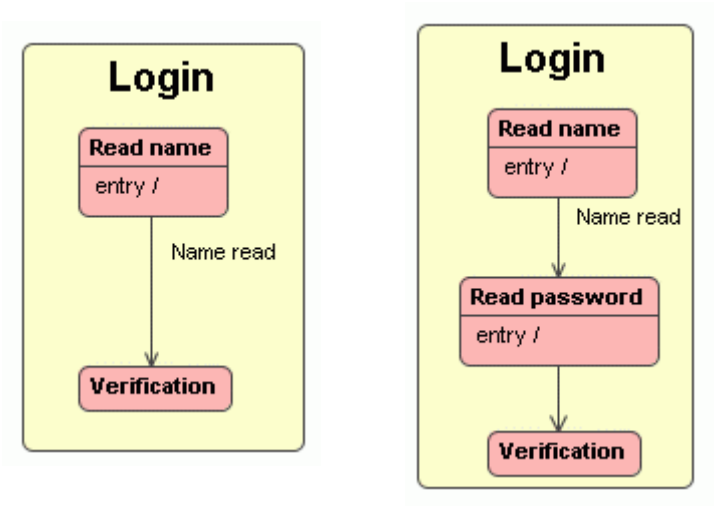




- **After <path type>.** Symbol is inserted before path. It means, the existing path is created, then the dropped element symbol is drawn, and then a new path is drawn. For example: *Name read* transition is drawn from *Read Name* state to *Verification* state. If you want to insert the *Read*

Copyright © 1998-2015 No Magic, Inc.

password state after the *Name read* transition, drop the *Read password* state on the transition and in the open dialog, click the **After Transition** button.



- **Do not insert.** Action is cancelled and the dialog is closed.

Select the Remember my choice check box and the next time an element will be inserted before or after the path, depending on your selection made this time.

Creating Relations from the Model

The main purpose of this functionality is to allow connecting and create traceability, according to UML, among elements, which are not from the same diagram. In other words, to link elements from a model without the need to place them in the same diagram.

Advantages of this implementation:

- Working time is saved on creating a diagram just to link elements for traceability.
- Some elements cannot be added to the same diagram and linked (elements from Behavior diagrams - Actions, States, Lifelines cannot be added to Static diagrams), this feature will allow the linking of such elements.
- Capability to relate multiple elements to a single element quickly. This is the usual case when a single element is represented with many elements in different abstraction levels or domains.
- The allocation relationship can provide an effective means for navigating the model by establishing cross relationships and ensuring the various parts of the model are properly integrated. For example, activity allocation to a block in SysML.
- Another example is creating an abstraction relationship with a stereotype «trace» between the model elements or sets of model elements that represent the same concept in different models.
- Smart manipulator toolbar allow connecting to any existing element quickly. Also, any Relation used with an element has a smart manipulator toolbar. Draw any relation from any element and you will be able to select the existing target element from the browser.

To create a new relation for an element

1. From an element shortcut menu in the browser, select **Create Relation** and then select the desired link from the group of **Outgoing** or **Incoming** relations. The **Create New <relation name> To (From)** dialog box opens.
2. In the model element tree, select an element to (from) which you want to create a relation. Click **OK.** The link will appear in the Browser. Type the name or leave it unnamed.

-or-

Copyright © 1998-2015 No Magic, Inc.

1. In the element **Specification** dialog box, select the **Relations** group.
2. Click the **Outgoing** or **Incoming** button and then select the desired link from the list. The **Create New <relation name> To (From)** dialog box opens.
3. In the model element tree, select an element to (from) which you want to create a relation. Click **OK.** The link will appear in the **Relations** group.

-or-

1. On the diagram pane, select an element and then from the smart manipulator toolbar, select the desired link.
2. Right-click to open the target element list and select the **Select From Model** command. The **Create New <relation name> To** dialog box opens.
3. In the model element tree, select an element to which you want to create a relation. Click **OK.** The link will be drawn on the diagram pane.

Smart Manipulation



Smart Manipulation

Smart manipulation is a feature designed to make modeling with MagicDraw diagrams easier. Use Smart manipulation to display or suppress compartments, create elements, set an auto-size option, reset a label position on a path, and draw relationships with most commonly used elements. MagicDraw offers varying smart mechanisms depending on the shapes involved.

There are the following types of smart manipulation:

- Smart Manipulators
- Smart Manipulator Toolbar

Smart Manipulators

Small buttons are displayed within the symbol on the diagram pane.

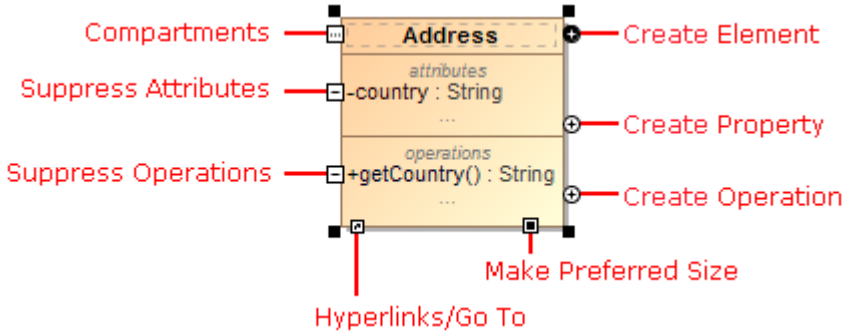


Figure 122 -- Example of smart manipulators on class shape

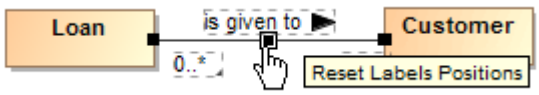
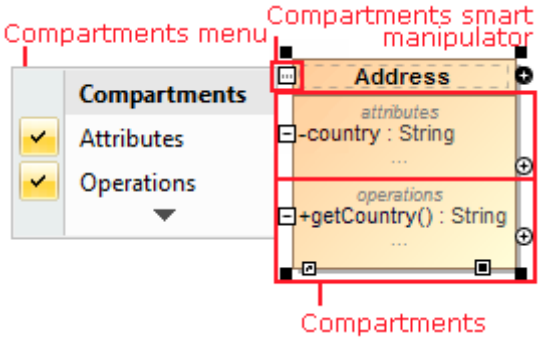
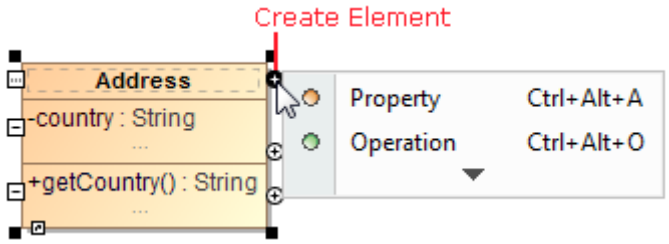


Figure 123 -- Example of smart manipulator on association

Copyright © 1998-2015 No Magic, Inc.

See the description of smart manipulators in the following table.

Smart Manipulators	Tool-tip	Description
▭	Compartments	Opens the *Compartments* menu, where you may display or suppress compartments and display or hide elements in compartments on shapes. For more information about working with compartments, see "Compartments" on page 236.  Note that the *Compartments* smart manipulator is displayed only on shapes that have compartment boxes and is not displayed on shapes that have only label compartments (for example, pins, paths, and others).
⊟	Suppress Attributes, Suppress Operations, others	Hides the particular compartment. Note that the ⊟ *Suppress <element name>* smart manipulator exists only next to the compartment that is represented as a box on a shape. For example, Tagged Values do not have compartment and thus do not have the suppress compartment smart manipulator. Note that where are exceptions and the following compartments do not have the ⊟ smart manipulator: Region compartment on State, Interaction Operands compartment on Combined Fragment, and Clause compartment on Conditional Node.
⊕	Create Element	Opens the menu by using which you can create the element into the compartment. 
⊕	Create Property, Create Operation, others	Creates, for example, the property, the operation, or other particular element into the compartment. Note that the ⊕ *Create <element name>* smart manipulator exists only next to the compartment that is represented as a box on a shape. For example, Tagged Values do not have compartment and thus do not have ⊕ *Create <element name>* smart manipulator. If shapes have only one compartment, the ⊕ smart manipulator is displayed at the right-top of the shape, instead of the ⊕ *Create Element* smart manipulator, for example, on the Signal shape.

Copyright © 1998-2015 No Magic, Inc.

Smart Manipulators	Tool-tip	Description
⊕	Create Element	Opens the menu where you can select to create an element into the compartment. The ⊕ *Create Element* smart manipulator is displayed at the right side of the shape, next to the compartment. If into one compartment you can create two or more types of elements (this is possible by using DSL), the ⊕ *Create Element* smart manipulator is displayed instead of the ⊕ *Create <element name>* smart manipulator. After clicking the smart manipulator, the menu with the available to create elements appears.
⊡	Hyperlinks/Go To	Opens the menu, where you can: • navigate to the associated diagram or elements • create a new or manage the existing hyperlink. For more information, see "Defining Hyperlinks" on page 340.
▣	Make Preferred Size	Resizes the shape so that it uses minimum space. The ▣ *Make Preferred Size* smart manipulator is displayed only if the shape is not autosized. Note that the ▣ *Make Preferred Size* smart manipulator exists on shapes, on diagram frames, but do not exists on paths. The ▣ *Make Preferred Size* smart manipulator is always visible on the screen as it moves dynamically to the right and to the top on the diagram frame or large shape border.
▣	Reset Labels Positions	Resets labels position to the default one. Note that this smart manipulator exists only on paths.
⊡	Edit Element Properties	Opens the **Compartment Edit** dialog. Note that the ⊡ *Edit Element Properties* smart manipulator exists only on Note shapes and the Note has to be connected with an element. For more information about the **Compartment Edit** dialog, see "Displaying or Hiding Elements in Compartments on Shapes" on page 238. For more information about the Note element, see "Note" on page 943.

Smart Manipulator Toolbar

Smart manipulator toolbar appears when symbols are selected on the diagram pane.

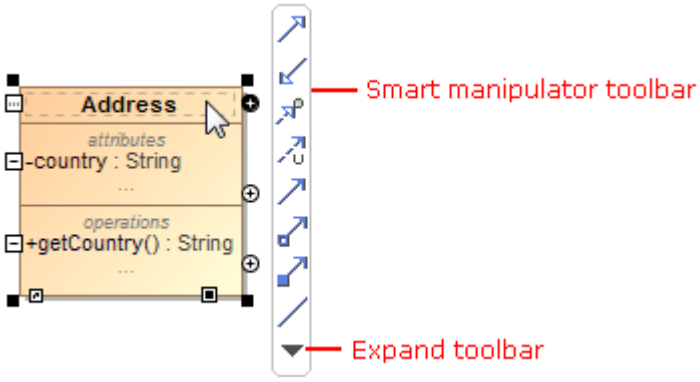


Figure 124 -- Example of smart manipulator toolbar

Copyright © 1998-2015 No Magic, Inc.

In the smart manipulator toolbar, buttons are divided into *standard* and *extra* modes. You can toggle between these two modes by clicking the **Expand** button - the arrow symbol - located at the bottom of every smart manipulator toolbar. The program remembers your mode choice and displays it for all elements.

Use the smart manipulator toolbar to quickly perform simple actions and create new elements.

To create a new element connected to a particular element

1. Select a symbol on the diagram pane. The smart manipulator toolbar appears. In the toolbar, select the relationship that you want to draw. The drawing of the selected relationship is initiated and the mouse cursor displays the new element which will be created.

EXAMPLE

Create a class symbol. In the class smart manipulator toolbar, select the Directed Association relationship for drawing. The drawing of this Directed Association relationship is initiated and the mouse cursor displays a class icon. Click the left mouse button. The element displayed on the mouse cursor is created together with the relationship.

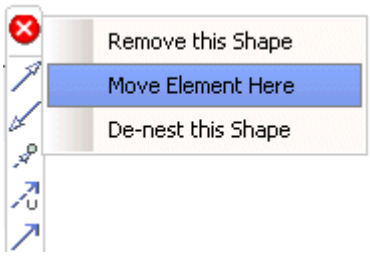
2. Use the smart manipulator toolbar to select which element you want to draw at the other end of relationship. In the toolbar, select the relationship and then click the right mouse button. The list of elements available for creating appears. Select the element from the list and it will be created.

TIP!

- To create a path breakpoint use the following keyboard combination: Ctrl key + Mouse click.
- To cancel the drawing of an element, press **Esc**.

To solve the detected symbol ownership problem

1. Select the element from the diagram pane, which is drawn on an incorrect ownership (which is highlighted in red). The smart manipulator toolbar appears.
2. Click the red button, which is at the top of the smart manipulator toolbar. The menu with the possible problem solving solutions appears.



TIP!

For more information, see "Active Validation" on page 632.

To hide smart manipulators and the smart manipulator toolbar

1. From the **Options** menu, select **Environment**. The **Environment Options** dialog opens.
2. In the **Diagram** options group, **Smart Manipulators** category, click to clear the **Show Smart Manipulators** check box and click **OK**.

Compartments

The compartment is area on the shape or next to the shape that is dedicated for the specific type of information.

Copyright © 1998-2015 No Magic, Inc.

There are the following types of compartments:

- Compartment boxes. For example, the class shape has compartments that separate, for example, attributes, operations to the boxes.

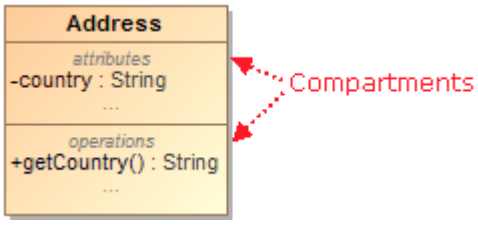


Figure 125 -- Example of attributes and operations compartments on class shape

TIP!

- You can hide compartments by suppressing them or expand the hidden compartments to see their data.
- Compartment names are displayed by default. However, you can hide them by changing their visibility settings either in the Symbol Properties dialog or the Project Options dialog. Each compartment has its own property for visibility settings (for example, **Show Attributes Compartment Name**).

- Compartments that are displayed next to the element name on the shape. For example, a shape has a compartment stereotypes, constraints, and others.

NOTE

- Stereotypes (or stereotypes compartment) are displayed above the element name.
- Constraints (or constraints compartment) are displayed under the element name.
- By default, tagged values are displayed under the element name or you can select to display tagged values in the compartment box,
- Element Properties are displayed in the compartment box.

- Compartments that are displayed next to the element name, next to the shape. For example, an Actor has a compartment for element properties, a compartment for stereotypes, behaviors, and others.
- Compartments on paths that are displayed next to the path label. For example, a compartment for element properties, a compartment for stereotypes, and others.

Creating Elements in Compartments on Shapes

To quickly create elements in compartments on shapes

Do one of the following:

- On the diagram, click the shape and click the *Create Element* smart manipulator. Then on the menu click the element that you want to create.
- On the diagram, click the shape. Then next to the particular compartment, click the *Create <element name>* smart manipulator.

TIP!

For more information about smart manipulators, see "Smart Manipulation" on page 233.

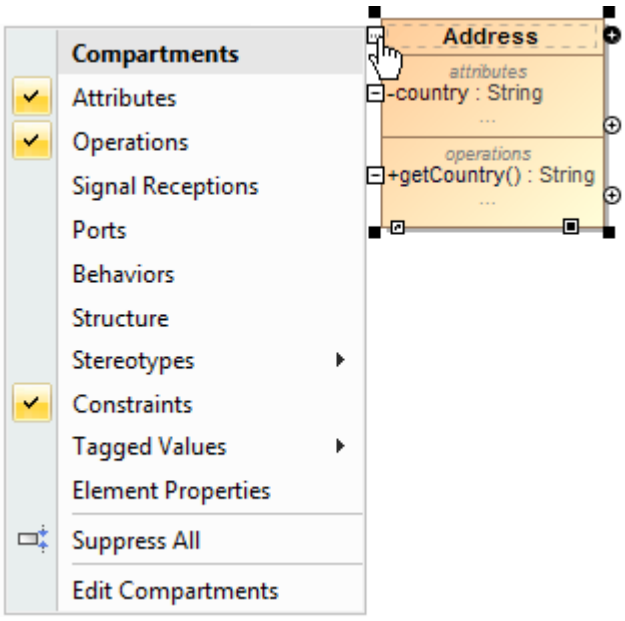
Copyright © 1998-2015 No Magic, Inc.

Displaying and Suppressing Compartments on Shapes

To display or suppress compartments

Do one of the following:

- On the diagram, click the shape and click the *Compartments* smart manipulator. Then click to select or clear the particular compartment, for example, **Attributes**, **Operations**, or others.



NOTE

The *Compartments* smart manipulator exists on shapes that have the compartment boxes. For example, pins, initial nodes, and others do not have the *Compartments* smart manipulator. For more information about smart manipulators, see "Smart Manipulation" on page 233.

- On the diagram, click the shape, and then click the *Suppress <compartment name>* smart manipulator.
- On the diagram, right-click the shape, and click **Symbol Properties**. Then clear or select the *Suppress <compartment name>* check box, for example, **Suppress Attributes**, **Suppress Operations**, or others.

Displaying or Hiding Elements in Compartments on Shapes

By using the **Compartment Edit** dialog you may control the list of elements and properties that are visible in compartments on shapes. You can display or hide stereotypes, constraints, tagged values, attributes, operations, and others. If there are hidden properties on the boxable compartment, the ... is displayed.

In addition, in the **Compartment Edit** dialog, under the **Element Properties** tab, you can display or hide the element specification properties, such as, *Name*, *Is Active*, *Owner*, *To Do*, and others.

Copyright © 1998-2015 No Magic, Inc.

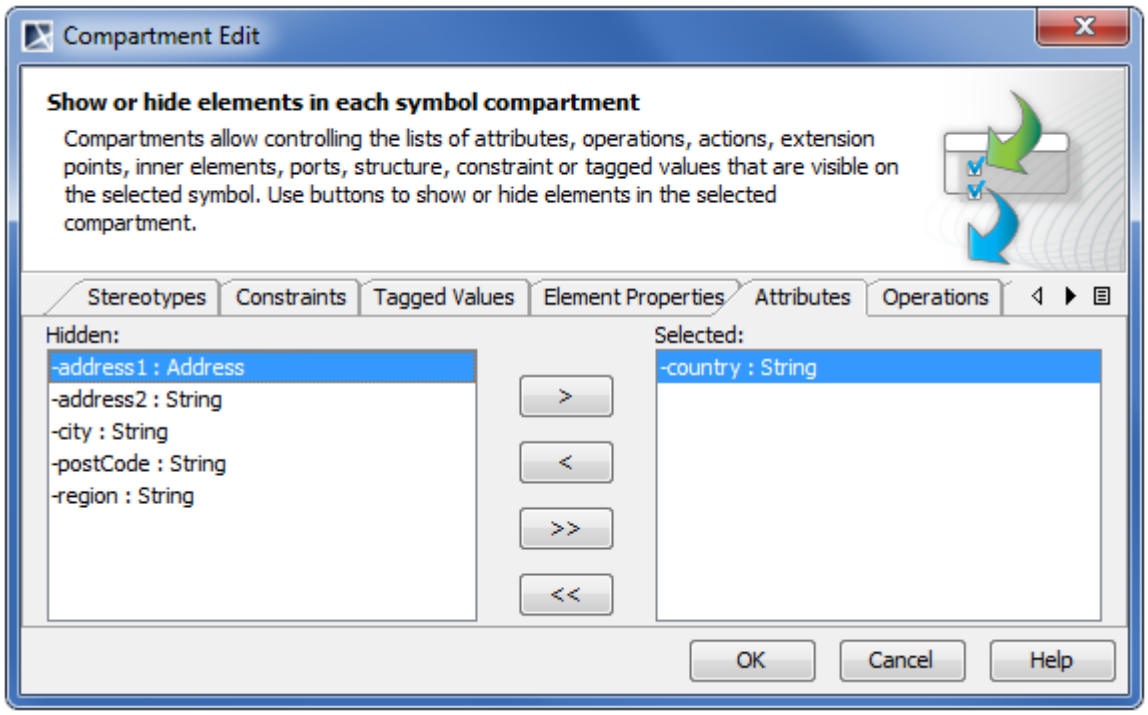


Figure 126 -- Compartment Edit dialog

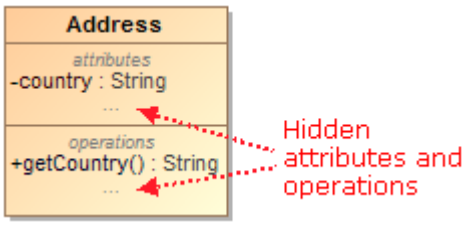


Figure 127 -- Example of hidden elements in the compartments

To open the **Compartment Edit** dialog

Do one of the following:

- On the diagram, click the shape and then click the *Compartments* smart manipulator. On the menu, click the arrow to expand the menu, and then click the **Edit Compartments** command.

  TIP! For more information about smart manipulators, see "Smart Manipulation" on page 233.

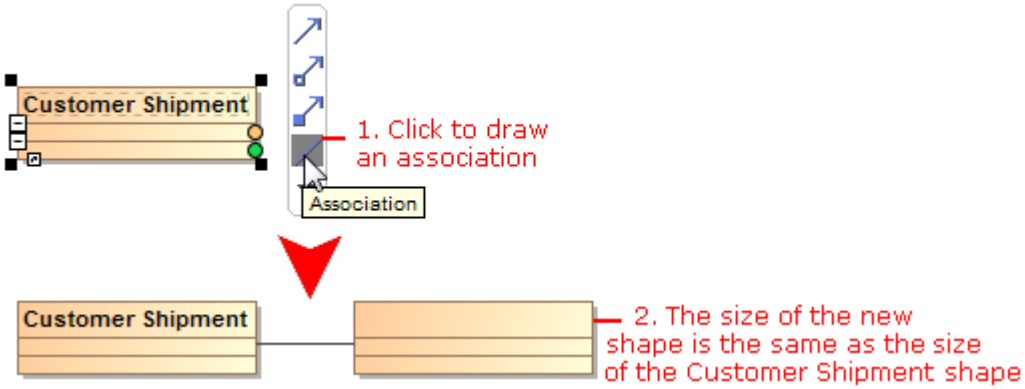
- On the diagram, right-click the shape and from the shortcut menu, select **Edit Compartments**.
- On the diagram, on the Note shape, click the *Edit Element Properties* smart manipulator. For more information, see "Note" on page 943.

Smart Shape Sizing

MagicDraw introduces the smart shape sizing when drawing new elements from other elements via paths. The size of the newly created shape is kept the same as the size of the shape from which the path is drawn:

Copyright © 1998-2015 No Magic, Inc.

- Creating a shape on the diagram by using smart manipulator while drawing a path. The size of the new shape is the same as the size of the source shape. Note that this is valid if the source shape is not large, for example, if it is not more than 3 times larger than the new shape.



- Inserting a new shape into the flow in the activity or state diagram. The size of the newly inserted shape is the same as the size of the source shape.

Selection and Multiple Selections

To select a shape

- Click the desired shape on the Diagram pane.

To deselect the selected shape

- Click outside the shape on the Diagram pane.

To select all shapes of the same element type

- Press ALT and click the shape. All shapes of the same type are selected.

To select all shapes on the diagram, do one of the following

- From the **Edit** menu, select **Select All.**
- Press CTRL+A.

To make multiple selections, do either

- Do the following:
    1. Click the shape on the diagram pane.
    2. Hold down SHIFT and click another shape. Repeat until you select the desired number of shapes.
- Drag the cursor diagonally across the area you wish to select. All shapes in the selected area will be selected.

Copyright © 1998-2015 No Magic, Inc.

To select a group of shapes

To select a group of shapes drag the cursor diagonally across the area you wish to select. This is a simple and fast way to select a group of shapes on the diagram.

See the sample of the rectangular selection in Figure 1.

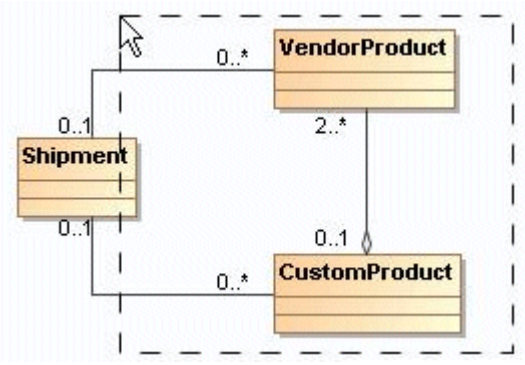


Figure 128 -- Rectangular selection

After the selection process represented in the Figure 1, the following shapes are selected, as shown in Figure 2.

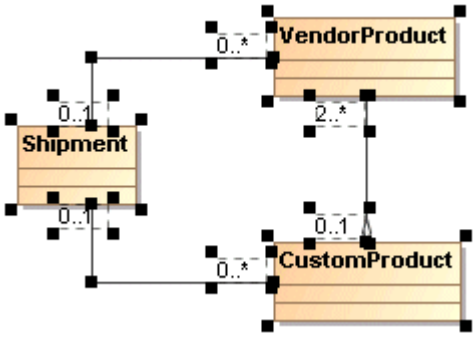


Figure 129 -- Rectangular selection result with partial selection coverage mode

The following rectangular selection modes are available:

1. *Partial coverage*. Symbols which are only partly covered with the rectangular selector are selected. See Figure 2. After the selection process represented in Figure 1, the class *Shipment* is selected by the rectangular selector, even though it was only partially covered.
2. *Complete coverage*. Only those symbols that are fully covered by the rectangular selection process will be selected. See Figure 3. For example, after the selection process shown in the Figure 1, the class *Shipment* and the associations are not selected because these symbols were not fully covered.

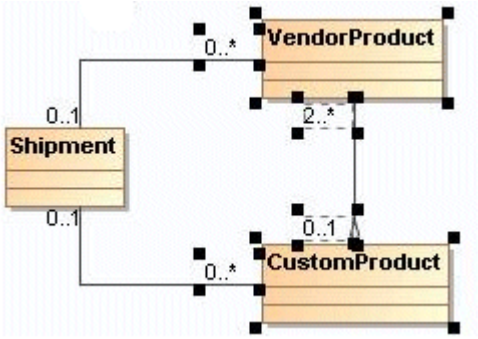


Figure 130 -- Rectangular selection result with complete coverage selection mode

Copyright © 1998-2015 No Magic, Inc.

Default selection mode is Partial coverage.

To quickly change the group selection mode from Partial coverage to Complete coverage mode or conversely

- Press the **Ctrl** key and then drag the cursor diagonally across the area you want to select.

To change the group selection mode for the whole project, do either

- In the main diagram toolbar press the **Complete coverage mode for group selection** button.

-or-

- Do the following:
  1. From the **Options** main menu, select **Environment**. The **Environment Options** dialog box appears.
  2. In the **Diagram** branch, **Symbols Manipulation** group, change the property of the **Group selection mode** option.

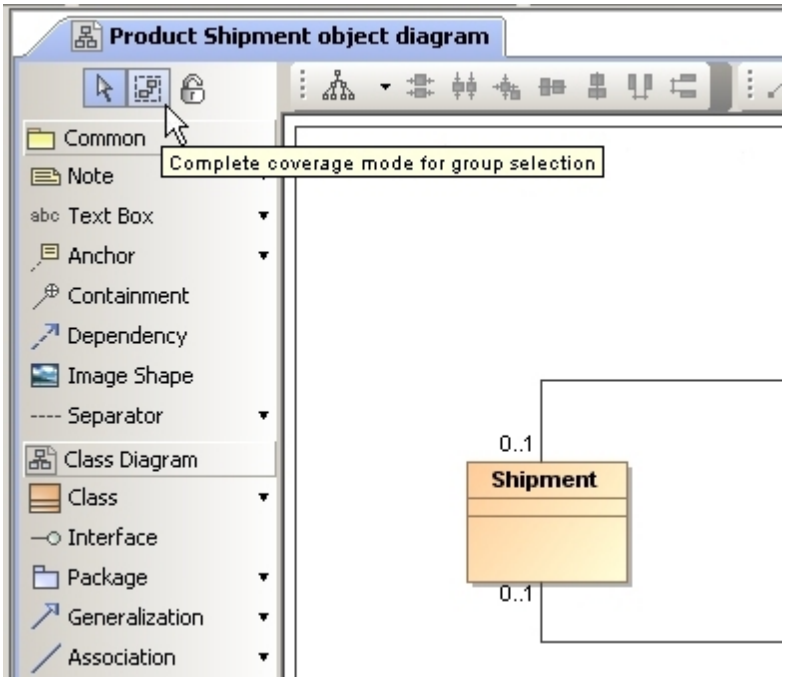


Figure 131 -- Complete coverage mode for group selection button

Copying Text or Images to Diagrams

It is possible to copy the text or images to a diagram. The copied text or image is pasted into a text box or an image shape. MagicDraw supports the HTML and plain text, and images of gif, jpg, svg, and png image file types. This copy and paste functionality allows you to drag and drop data from other applications, for example, web browsers.

To copy the text or image to a diagram

1. Select the text or image.
2. Press Ctrl+C.
3. Open the diagram.
4. Press Ctrl+V. The **Paste Special** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

5. Select Image, Plain Text, or HTML Text accordingly to your what you want to paste on the diagram.

See the text or image copied on the diagram pane.

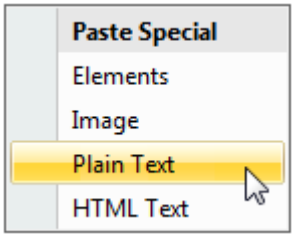


Figure 132 -- Paste Special dialog

TIP!

You can use copy-and-paste operation for element creation in the model. For more information, "NEW! Creation of Elements from other Resources" on page 349.

Nesting Image Shapes

You can now drag an image to any elements in a diagram as nested a element.

To drag an image to an element

1. Select an image in the diagram pane.
2. Drag it to the image shape.

Dragged images will be nested by the following elements: Package, Model, Subsystem, Instance, Node, Part, Combined Fragment, Composite State (State diagram), Interruptible Activity Region, Structured Activity Node, Expansion Region, and Conditional Node (Activity diagram).



Figure 133 -- Samples of Images Nested to the Products Package and Server Component

Copyright © 1998-2015 No Magic, Inc.

Dragging, Copying, Cutting, and Pasting

Dragging Objects

You can move a shape to another location on the diagram pane by dragging and dropping.

To drag multiple selected symbols

- Select the symbols and drag them to the desired area on the diagram pane.

To copy a shape using a drag-and-drop operation

- Hold down the CTRL key while dragging the selected shape to the area where you wish to make a copy.

To drag and drop items from the Model Browser to the diagram pane

1. In the Model Browser, select the created model element you wish to draw.
2. Drag the selected model element to the desired location on the diagram pane and drop it there.



- You can select several model elements and draw them on the diagram pane.
- If the selected model elements are not compatible with the open diagram, you will not be allowed to draw those model elements.

Dragging Files

The drag-and-drop capability allows you to drag any files from your file system to any element in the Model Browser or to a symbol on a diagram. A hyperlink is automatically created for the element or symbol to which the file is dragged. Double-click the element or symbol to open the dropped file.

Copyright © 1998-2015 No Magic, Inc.

To drag a file onto an element

1. Select a file.

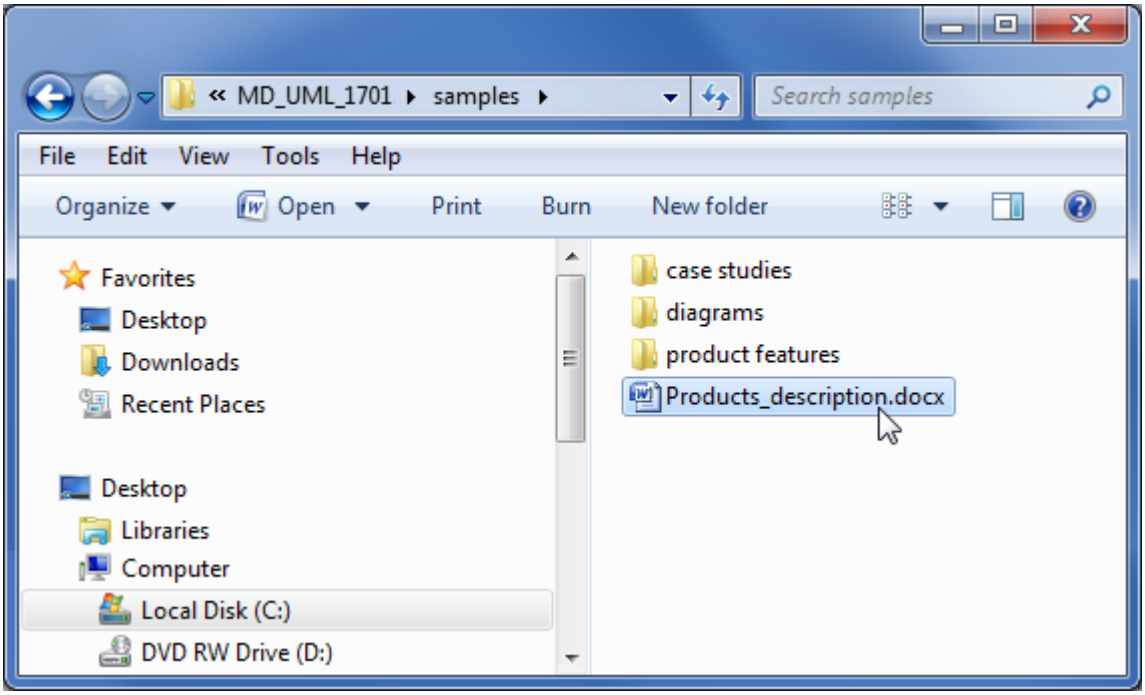


Figure 134 -- Selecting file

Copyright © 1998-2015 No Magic, Inc.

2. Drag the selected file onto the element in the Model Browser or onto a symbol on a diagram. A hyperlink to the file is created.

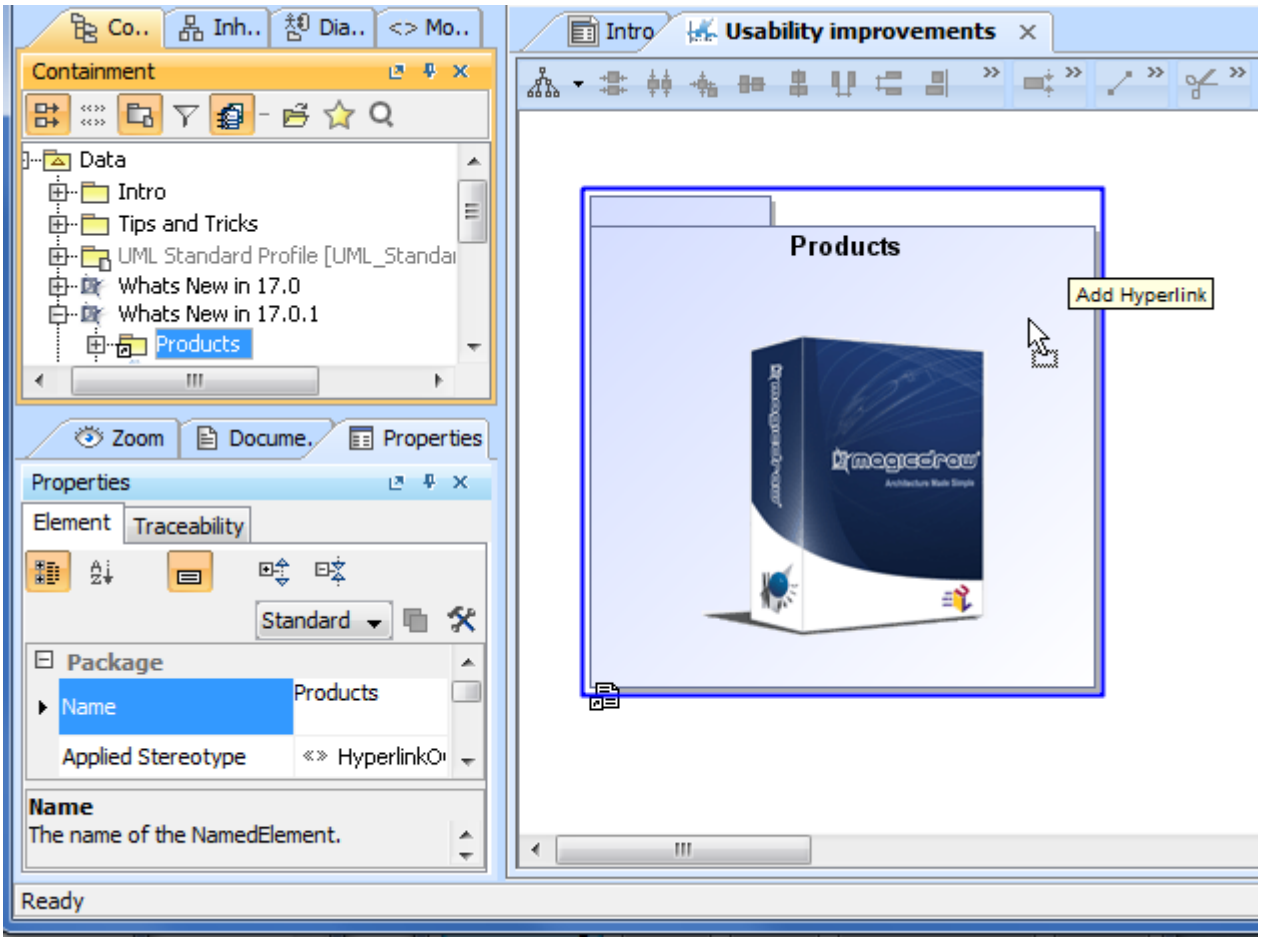


Figure 135 -- Dragging file onto the symbol and creating hyperlink

To drag a MagicDraw project file to a diagram to open the project

1. In your file system, select a MagicDraw project file.
2. Drag this file to a diagram pane. The selected MagicDraw project opens.

TIP! You can drag source code files from your file system to a MagicDraw Code Engineering Set.

Dragging Images

You can drag an image file from your file system to an element in the Model Browser or a symbol on a diagram pane. The dragged images are set as the Image property value of the element.

If the image is dragged to a Stereotype, this image is set as a Stereotype icon.

To drag the image to an element or symbol

1. In your file system, select an image file.

Copyright © 1998-2015 No Magic, Inc.

2. Drag this file to the element in the Model Browser or the symbol on the diagram.



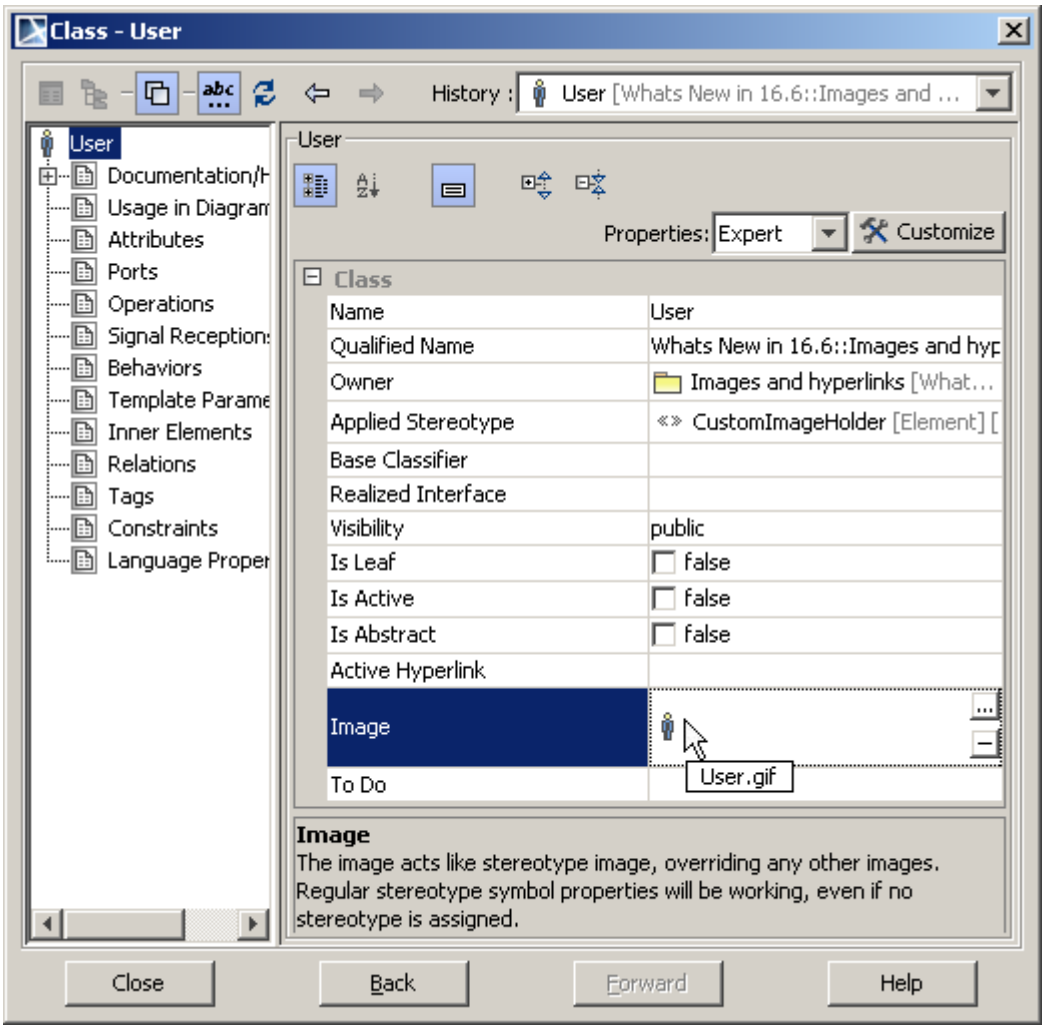


Figure 136 -- Representation of image dragged to class shape and as Image value in Specification window

Dragging Elements in the Specification Window

You can drag any elements to any properties in the Specification window. For example, you can drag a Class element from the Model Browser to the **Type** property in the **Operation** Specification window. That Class element is assigned as the Operation type in the Specification window.

Let's describe the element dragging to the Specification window in the following example:

To drag elements to the Specification window

1. Open the sample project *class.diagram.mdzip*. This sample can be found in *<MagicDraw installation directory>\sample\diagrams*.
2. Open a diagram *domain.User*.
3. Open the Specification window of the class Customer. In the **Operations** group, select the *getProfile* operation.

Copyright © 1998-2015 No Magic, Inc.

4. In the Model Browser, select the *CustomerProfile* class and drag it to the **Type** property value cell in the opened **Class - Customer** Specification window. The *CustomerProfile* class is assigned as the value of the *getProfile* operation type.

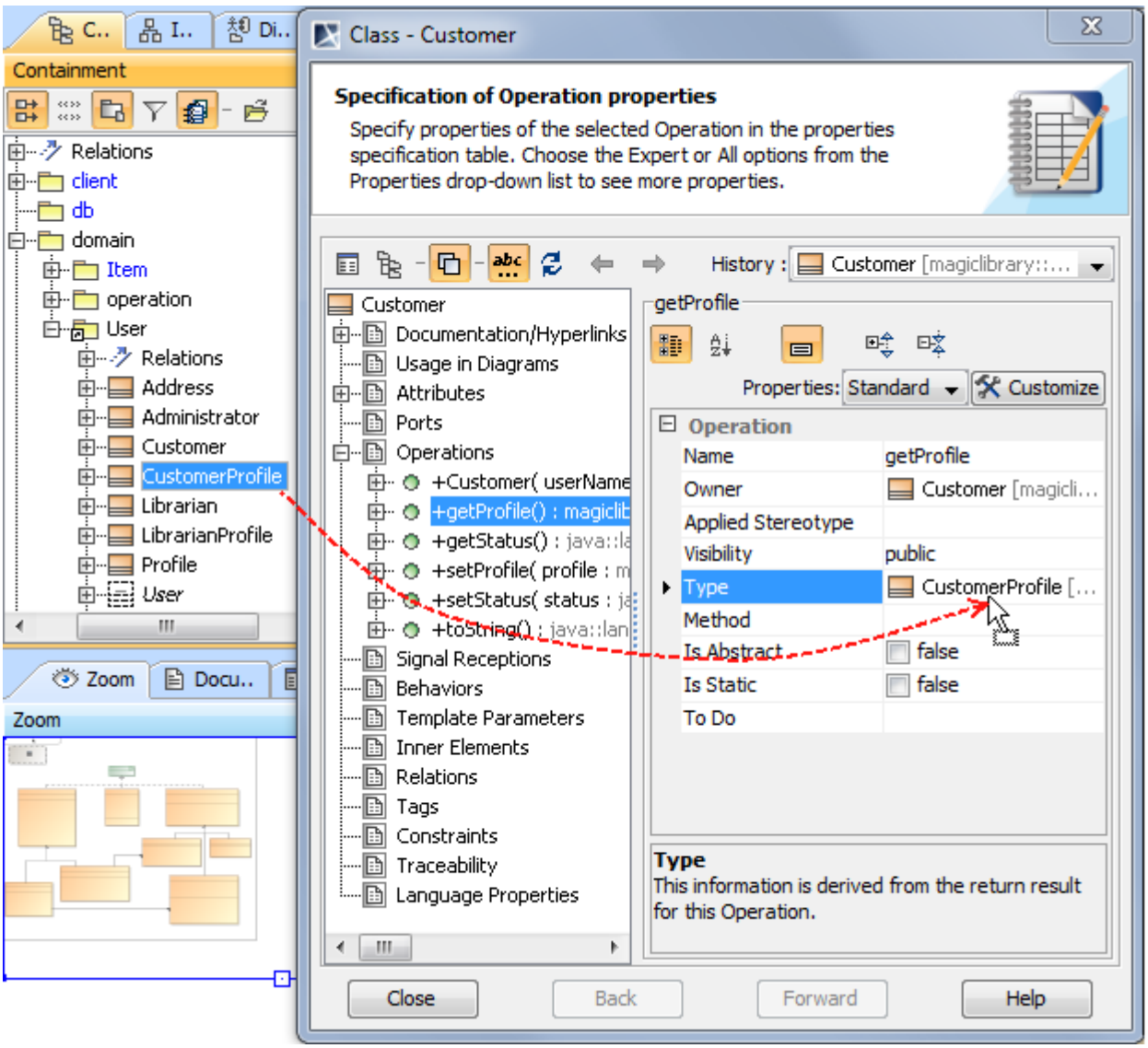


Figure 137 -- Dragging Class to the Property in the Specification window

You can drag any elements from the Specification window to a diagram or to the browser. Let's describe this case in the following example:

To drag elements from the Specification window

1. Open the sample project *class.diagram.mdzip*. This sample can be found in *<MagicDraw installation directory>\sample\diagrams*.
2. Select the class *Customer* and open its Specification window.
3. In the **Operations** group, select the *getProfile* operation. The operation properties opens.
4. In the **Type** property value cell, select the *CustomProfile* and drag it to the one of the following:
   - An empty place on a diagram. In this case, the class symbol is created on the diagram (depicted in the Figure 138 on page 249).
   - A particular shape on a diagram. In this case, a new attribute is created.

Copyright © 1998-2015 No Magic, Inc.

- A particular element in the Model Browser. In this case, a new model element is created.

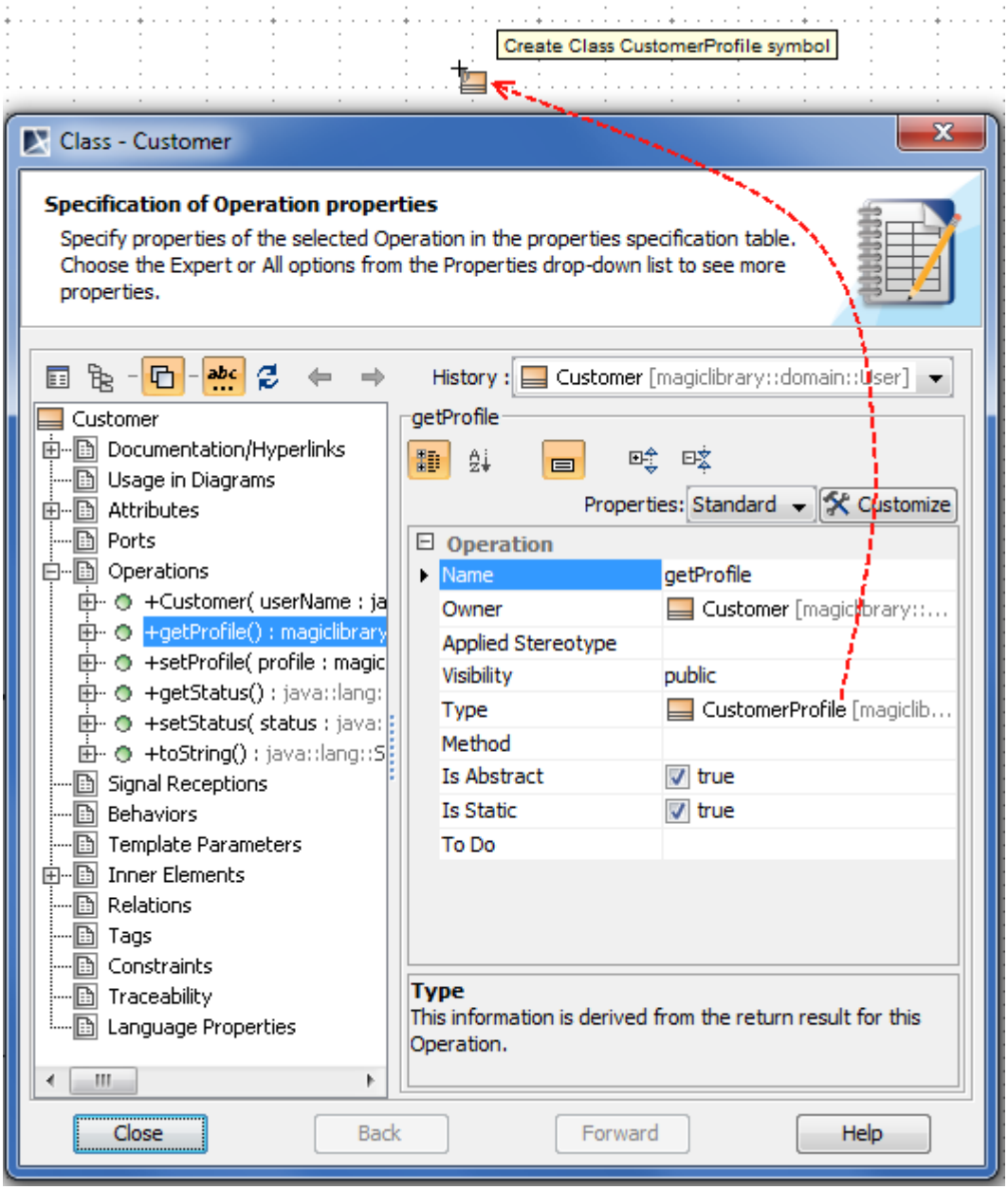


Figure 138 -- Dragging from the Specification window to diagram pane

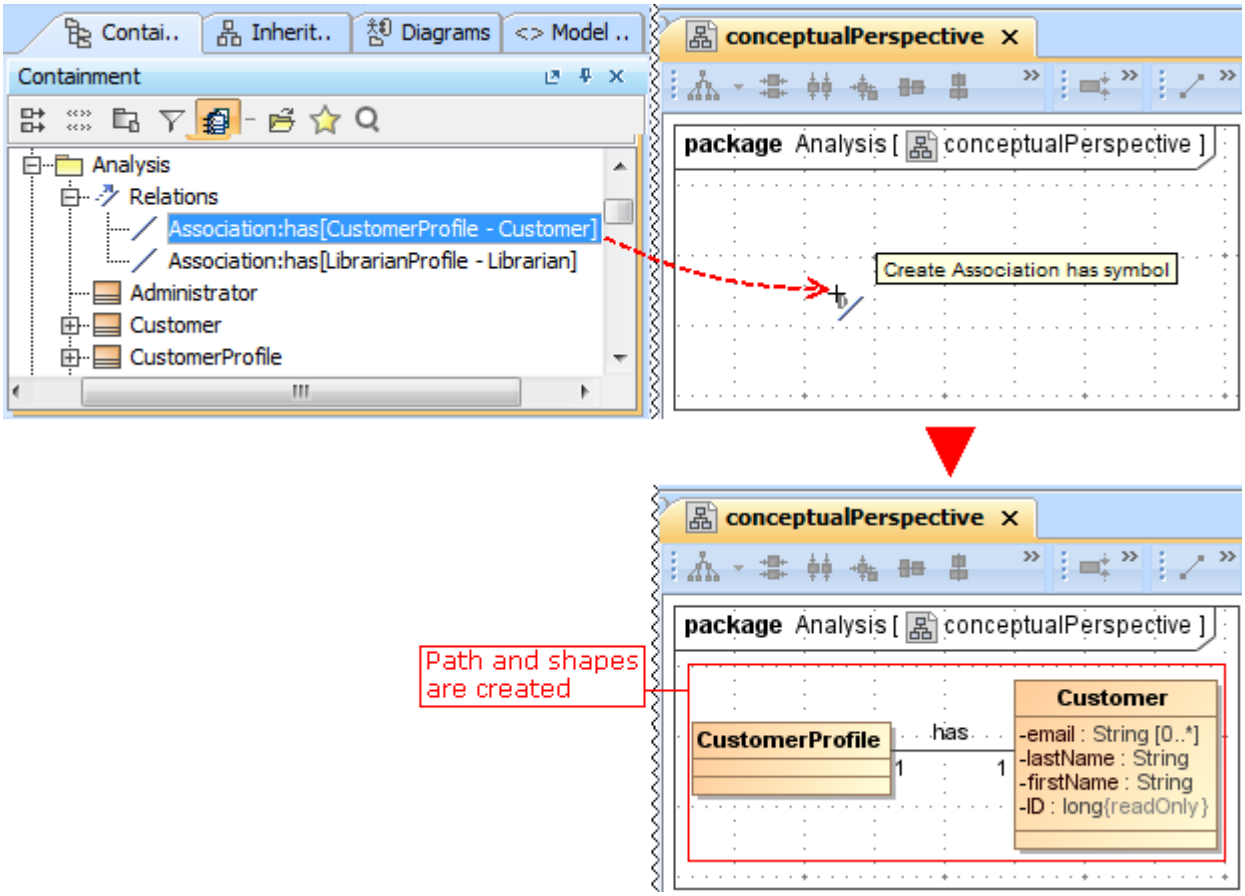
Dragging in Diagrams

Dragging relationships to a diagram

When dragging-and-dropping relationships to a diagram the following symbols will be created:

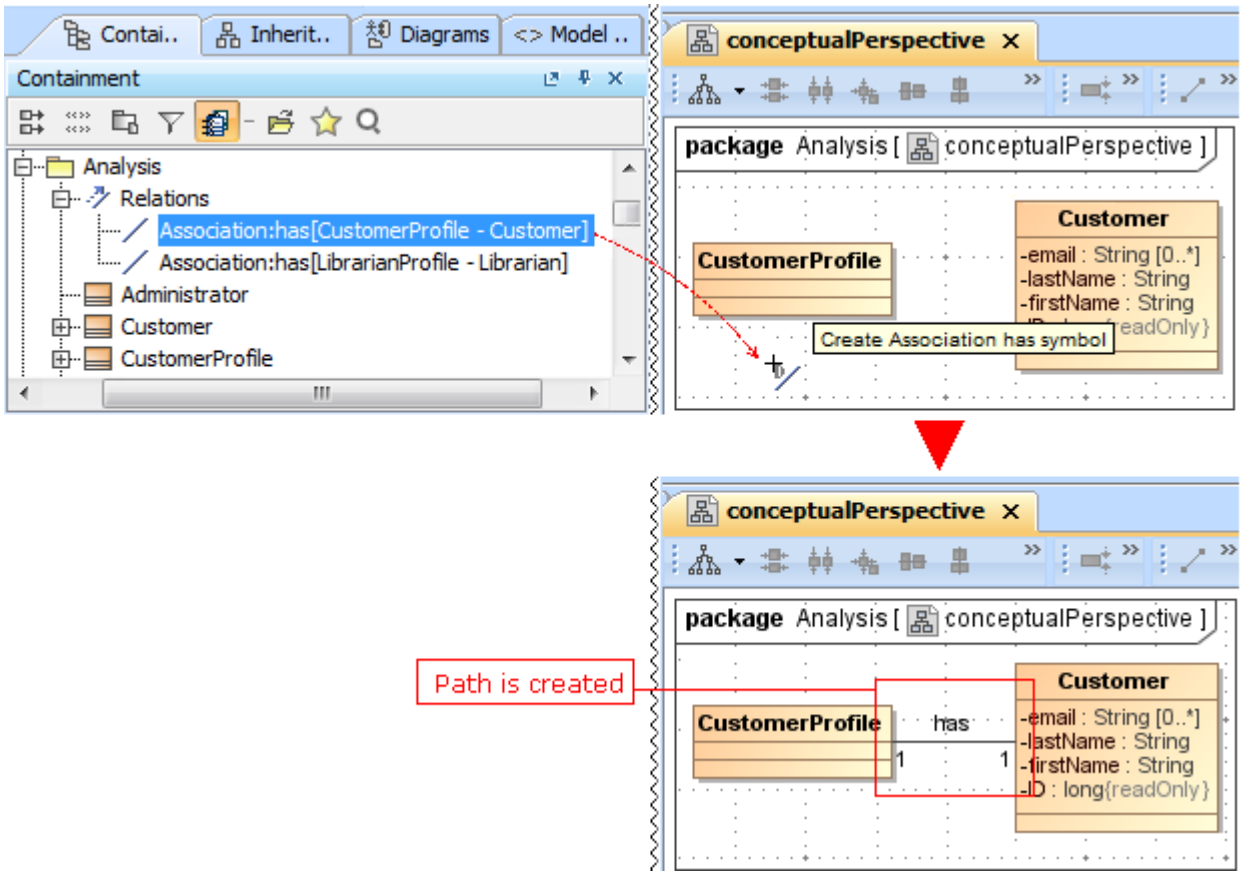
Copyright © 1998-2015 No Magic, Inc.

- The path for the relationship will be created along with the shapes at the end of the path.



Copyright © 1998-2015 No Magic, Inc.

- The path will be created between the shapes (if shapes at the end of the path already were represented on the diagram).



Dragging on a note and a text box symbol in diagrams

You can add a hyperlink to an element on a note and a text box symbols by simply dragging and dropping model element on a note or text box symbol in diagram.

TIP!

For more information, see "To add a hyperlink to an element on a note symbol" on page 945, and "To add a hyperlink to an element on a text box" on page 1034.

Dragging in Sequence diagram

In the Sequence diagram, you can:

- Drag an Operation from Model Browser to a Message. The Message becomes a Call Message once the operation is assigned.

NOTE

The Lifeline type must have or inherit this operation.

- Drag an Interaction to an Interaction Use. The Interaction will be set as **Refers To** element.
- Dragging a Signal to a Message converts the Message into a Send Signal Message and assigns the Signal to the Message.

Dragging in State Machine diagram

You can drag an Event element to a Transition element in a State Machine diagram. A Trigger with this Event is created for the Transition element.

Copyright © 1998-2015 No Magic, Inc.

Dragging in Activity diagram

In the Activity diagram, you can:

- Drag an Activity, Interaction, or State Machine to an Activity diagram to create a Call Behavior Action.
- Drag a Signal to an Activity diagram to create a Send Signal Action.
- Drag a Signal to a Send Signal Action to set or change the Signal.
- Drag a Signal to an Accept Event Action to set or change the Signal.



Once the Signal is assigned to the Accept Event Action, a Signal Event for the diagram and a Trigger for the action will be created automatically.

- Drag an Event to an Activity diagram to create an Accept Event Action.
- Drag an Event to an Accept Event Action to set the Event.
- Drag a Signal Event to an Activity diagram to create an Accept Event Action.
- Drag a Property, Actor, Class or Instance Specification to an Activity diagram to create a Swimlane. Elements will be set as representative elements.
- Drag a Property, Actor, Class or Instance Specification to an Activity diagram by using the right mouse button. From the shortcut menu, select to create Swimlane, Central Buffer Node, or available Actions.

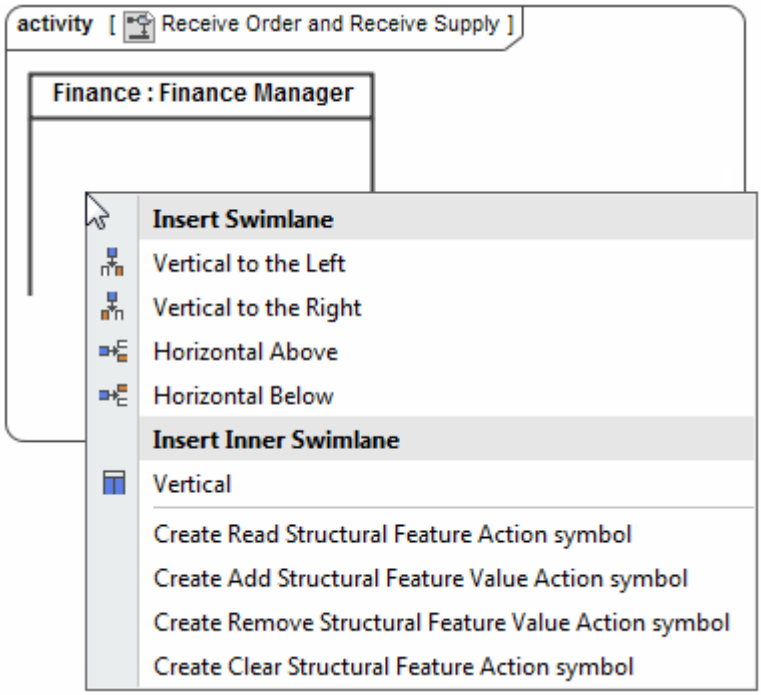


Figure 139 -- Example of shortcut menu displayed when dragging to Activity diagram

Other Dragging Actions

To apply a stereotype using drag-and-drop operation

1. Select a stereotype.
2. Drag it to a model element or symbol on a diagram pane to apply the stereotype.

Copyright © 1998-2015 No Magic, Inc.

Copying and Cutting Objects

To copy and paste the selected shape on a diagram

1. Select one or more shapes on a diagram.
2. From the **Edit** menu, select **Copy** (shortcut keys CTRL+C).
3. Click the diagram.
4. From the **Edit** menu, select **Paste** (shortcut keys CTRL+V).



Not all elements can be copied and pasted.

To cut and paste the selected shape on a diagram

1. Select one or more shapes on a diagram.
2. From the **Edit** menu, select **Cut** (shortcut keys CTRL+X).
3. Click the diagram.
4. From the **Edit** menu, select **Paste** (shortcut keys CTRL+V).



Not all elements can be cut and pasted.

To paste one or more copied model elements by creating new model elements

1. Select one or more shapes on a diagram.
2. From the **Edit** menu, select **Copy** (shortcut keys CTRL+C).
3. Click the diagram.
4. From the **Edit** menu, select **Paste with New Data** (shortcut keys CTRL+E).



Use the **Paste with New Data** command when copying or cutting owned symbols from one shape to other in a diagram.

To copy the whole diagram and paste it to Microsoft Office or other application

1. Select all model elements on the diagram.
2. From the **Edit** menu, select one of the following:
   - **Copy as BMP Image** (shortcut keys CTRL+SHIFT+B).
   - **Copy as EMF Image** (shortcut keys CTRL+SHIFT+E).
   - **Copy as JPG Image** (shortcut keys CTRL+SHIFT+J).
   - **Copy as PNG Image** (shortcut keys CTRL+SHIFT+P).
3. Open the desired application and paste the copied diagram.

To copy the selected model elements and paste them to Microsoft Office or other application

1. Select the desired model elements on the diagram pane.
2. From the **Edit** menu, select one of the following:
   - **Copy as BMP Image** (shortcut keys Ctrl+Shift+B). Note that Copy as BMP Image is available only on Windows OS.
   - **Copy as EMF Image** (shortcut keys Ctrl+Shift+E).
   - **Copy as JPG Image** (shortcut keys Ctrl+Shift+J).
   - **Copy as PNG Image** (shortcut keys Ctrl+Shift+P).

Copyright © 1998-2015 No Magic, Inc.

3. Open the desired application and paste the copied model elements.

NOTE

You can copy or cut and paste the text only when using the shortcut keys Ctrl+C or Ctrl+X and Ctrl+V. When you use the buttons or commands, the whole element is copied/cut and pasted.

Zooming

Zooming allows you to select a particular part of a diagram, zoom into it, and make changes while working with a finer level of detail. You can also gain an overview of a diagram by zooming out from it.

To fit the current diagram in the window

- From the **View** menu or from the diagram shortcut menu, select **Fit in Window** (shortcut keys CTRL+W.)
- In the Browser **Zoom** tab, click **Fit in Window** .

To zoom into the current diagram

- From the **View** menu or from the diagram shortcut menu, select **Zoom In** (shortcut keys CTRL+NumPad PLUS SIGN (+) or scroll.)
- Click the **Zoom In** toolbar button .

To zoom out from the current diagram

- From the **View** menu or from the diagram shortcut menu, select **Zoom Out** (shortcut keys CTRL+NumPad MINUS SIGN (-) or scroll.)
- Click the **Zoom Out** toolbar button .

TIP!

You can zoom in or zoom out using the CTRL+wheel keys.

To restore the diagram to the original size

- From the **View** menu or from the diagram shortcut menu, select **Zoom 1:1** (shortcut keys CTRL+NumPad SLASH MARK (/).)
- Click the **Zoom 1:1** toolbar button .

To view the selected shapes at maximum size

Select the shapes and then from either the **View** menu or the diagram shortcut menu, select **Zoom to Selection** (shortcut keys CTRL+NumPad ASTERICS MARK (*).)

To select the zoom settings

1. From the **Options** menu, select **Environment.** The **Environment Options** dialog box opens.

Copyright © 1998-2015 No Magic, Inc.

2. Open the **Diagram** pane and change the **Zoom Step Size** property. The maximum number is 1.0 (you can zoom a diagram twice.)



You can also zoom in or out of the diagram using the zoom panel in the Browser window. For the detailed description, see "Zoom tab" on page 93.

Using the Grid

The grid helps to arrange diagram symbols on the diagram pane. By default, the grid is not visible.

To show the grid on the diagram

Do either:

- From the main menu, select **View** > **Grid** > **Show Grid**.
- **NEW!** Right-click the diagram and from the shortcut menu, select **Show Grid**.
- Open the **Diagram Properties** dialog and set the **Show Grid** property to *true*.

To pull a path with the intersection of gridlines

From either the **View** menu or from the diagram shortcut menu, select **Grid** and then select/clear the **Snap Paths to Grid** check box.

To pull a shape with the intersection of gridlines

From either the **View** menu or from the diagram shortcut menu, select **Grid** and then select/clear the **Snap Shapes to Grid** check box.

To change the grid size

1. From either the **View** menu or from the diagram shortcut menu, select **Grid** and then select **Grid Size**.
2. The **Grid Size** dialog box opens.
3. Enter a grid size between 2 and 30 (default is 10).
4. Click **OK**.

To change the grid style

1. From the **Options** menu, select **Environment**. The **Environment Options** dialog box opens.
2. From the **Grid Style** drop-down list, select one of the following styles:
    - Dense
    - Sparse (default)

Copyright © 1998-2015 No Magic, Inc.

Layout



The diagram layout engine is available in Standard, Professional, Architect, and Enterprise editions.

It is easy to manage simple or complex diagrams using the automated layout features that optimize diagram layout for viewing.

MagicDraw automatically layouts symbols on diagrams when displaying related elements, creating diagrams by using wizards, when creating related diagrams.

You can layout diagrams quickly by using the Quick Diagram layout. The quick layout is a recommended type of layout, and it is specific for each diagram.

The layout mechanism is built on various layout tools. All layout tools could be separated into 2 different groups: general layout tools and specific diagram layout tools. These are the general layout tools:

- Orthogonal Layout
- Hierarchic Layout
- Organic Layout
- Tree Layout
- Circular Layout
- Oblique Path Router
- Rectilinear Path Router

These layout tools are provided by *yFiles* layout tool component. You can arrange each diagram (except Sequence diagram) by using any of the 5 general layout tools.

The specific diagram layout tools are:

- Class Diagram Layout
- Composite Structure Diagram Layout
- Activity Diagram Layout
- State Machine Diagram Layout
- Business Process Diagram Layout

To layout diagrams by using the quick diagram layout

Do one of the following:

- Press Ctrl+Q.
- On the diagram toolbar, click the *Quick Diagram Layout* button.
- On the **Layout** menu, click the **Quick Diagram Layout** command.



Sequence diagrams, because of its specific representation, has no layout mechanism.

To layout diagrams by using layout tools

Do one of the following:

- On the **Layout** menu, click one of the layout tools.
- On the diagram toolbar, click the small arrow next to the *Quick Diagram Layout* button, and select one of the layout tools.

Copyright © 1998-2015 No Magic, Inc.

- Right-click the diagram, on the shortcut menu, point to **Layout**, and then select one of the layout tools.

TIP!

You can also layout the selected symbols. Just select symbols or part of the diagram and then use one of the layout tools.

To change the diagram layout options

1. On the **Layout** menu, click **Layout Options**.
2. In the **Diagram Layout Options** dialog, specify the layout options and click **OK**.

TIP!

- For more information on how to resize shapes, see "To resize selected shapes" on page 226.
- For more information on how to route paths, see "To change a path style" on page 228 and "To route paths by using the Route Paths Rectilinearly button" on page 228.

Label layout in the diagram

Default label positions

Default label positions leaves after moving a path, shape, or related element if it is semantically logical decision. See an example below, there association multiplicities leaves at their default positions after class is moved.

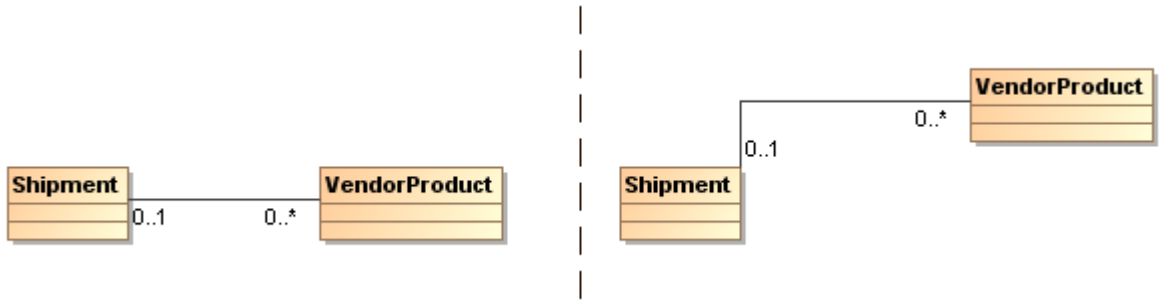


Figure 140 -- Default label position example

NOTE

If labels are at their default position, reset labels position functionality is disabled.

Labels positions after moving a path, shape or related element

After moving a path, shape or related element default label positions leaves if it is semantically logical decision.

For nicer representation of labels in diagram in the following cases labels positions are reseted to their default position automatically:

3. Symbol properties edit. When symbol properties edit causes label text box addition or removal from diagram pane labels positions are reset.
4. Path, path end or port properties edit. When path, path end or port data edit causes label text box addition or removal from diagram pane labels positions are recalculated. See an example when qualifier is added in Figure 142 on page 258.

Copyright © 1998-2015 No Magic, Inc.

5. Path, shape or related element movement. See an example, when related element is moved Figure 143 on page 258.

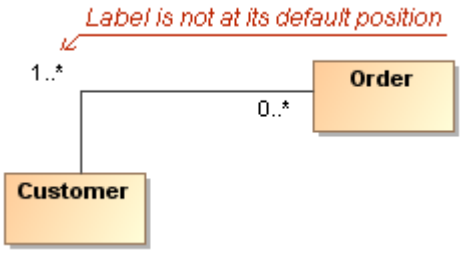


Figure 141 -- Label position before changes

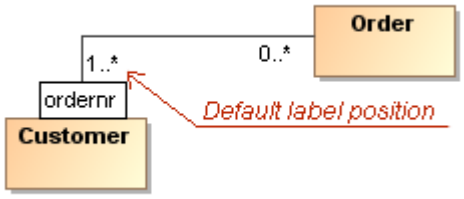


Figure 142 -- Label position is reset to its default position after qualifier is added

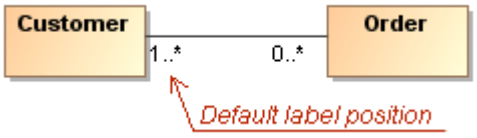


Figure 143 -- Label position is reset to its default position after Order class movement

Displaying label deviation from default position

While moving text box from default position, dotted line shows deviation from default position. This helps to see the current labels owner (see the following figure).



Figure 144 -- Dotted line, which shows deviation from the default position

Indicating label if it is not at its default position

If label is not at its default labels position, label right bottom corner is marked after label owner (path or shape) selection on diagram pane (see the following figure).

Copyright © 1998-2015 No Magic, Inc.



Figure 145 -- Marking the label when it is not at its default position

Pusher and Magnet

This feature allows you to easily rearrange element symbols on a diagram. Pusher moves a group of symbols towards the pushing direction thus creating an empty space for adding new symbols to a diagram.

Magnet works in the opposite way. It moves the selected group of symbols to override an empty space on a diagram.

Copyright © 1998-2015 No Magic, Inc.

Now you do not need to select separate symbols on a diagram to move them to another location. Just click the **Pusher** or **Magnet** button on the diagram palette and move the whole group of symbols toward the direction you need.

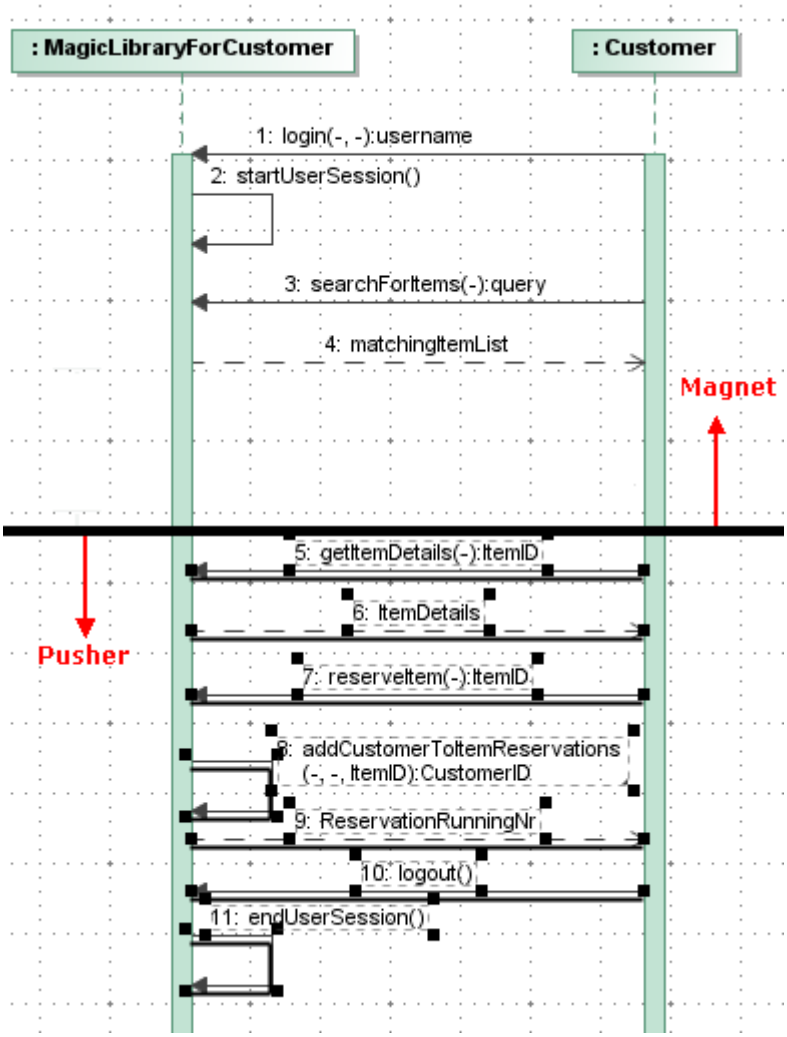


Figure 146 -- Example of horizontal pusher and magnet direction

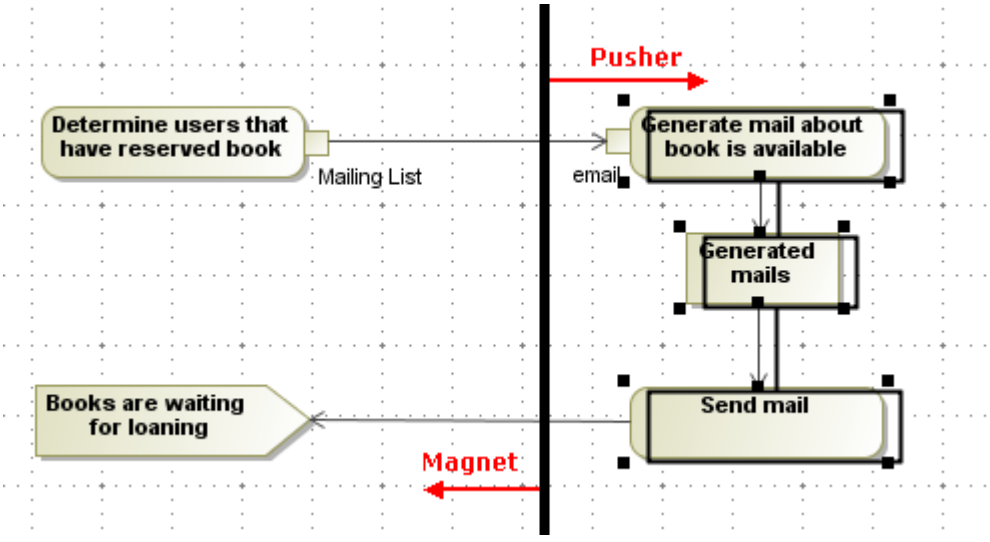


Figure 147 -- Example of vertical pusher and magnet direction

Copyright © 1998-2015 No Magic, Inc.

To push or lift the diagram fragment

1. On the diagram pallet, click the **Pusher**  button to push or the **Magnet**  button to lift the diagram fragment.
2. On the diagram pane, click the place whereon you need to make or remove an empty place and hold the right mouse button pressed.
3. Move the mouse in the desired direction, and the whole group of symbols will move toward this direction. The manipulation bar represented as a bold black line is displayed while moving the group of symbols.
4. Release the right mouse button after movement is finished.

Related references

Layout

Saving as Image

You can save as image an active diagram, selected symbols, or selected diagrams.

Diagrams and symbols that were created in the model can be saved as an image in the following formats:

- Enhanced Metafile Format (*.emf) (supports language specific symbols)
- Encapsulated PostScript (*.eps)
- Joint Photographic Experts Group (*.jpg, *.jpeg)
- Portable Network Graphics (*.png)
- Scalable Vector Graphics (*.svg)
- Tagged Image File Format (*.tif, *.tiff)
- Windows Metafile Format (*.wmf)

Copyright © 1998-2015 No Magic, Inc.

Save As Image Dialog

To open the **Save As Image** dialog

- From the **File** menu, select **Save as Image**.

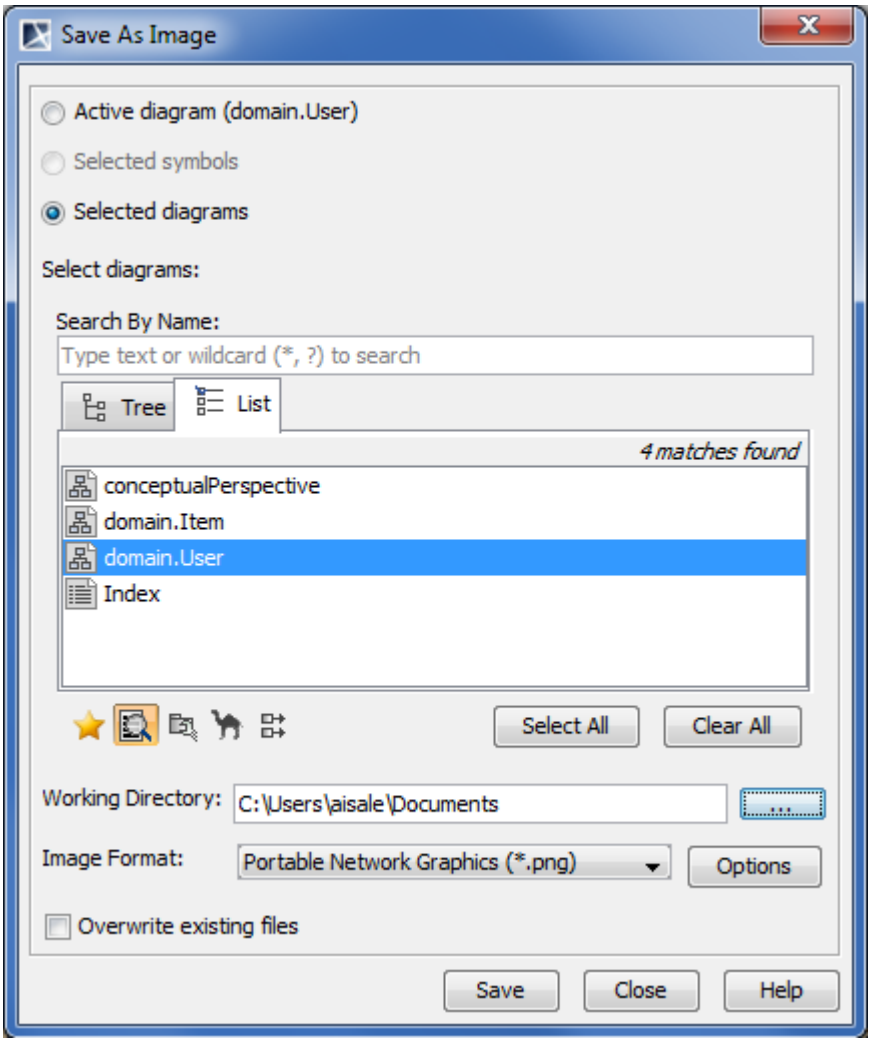


Figure 148 -- Save As Image dialog

The following table describe dialog elements and buttons that are used in this dialog.

Element name	Type	Description
Active diagram	Option button	Select to save as image the currently active diagram. The active diagram name is displayed in the brackets next to the option button name.
Selected symbols	Option button	Save as image the particular symbols that are selected on the diagram. The desired symbols should be selected on the diagram pane to activate this option button.
Selected diagrams	Option button	Select the option in order to save as image the selected diagrams in the model tree or diagram list.

Copyright © 1998-2015 No Magic, Inc.

Element name	Type	Description
Search By Name		The searching for diagrams is available only when the **Selected diagrams** option is selected. In the **Search By Name** box, type the name or a part of name of the diagram you want to search for. Search results are displayed in the model tree and diagram list. Click the **Tree** or **List** tab and, in search results, select the diagram(s) that you want to save as image. To optimize the search, you can specify the search modes and scope filters. The appropriate buttons are displayed below the search results. For more information about searching, refer to "Searching for Elements in Element Selection Dialog" on page 355.
Image File / Working Directory	Text Box	Select the working directory where the image file will be saved.
Image Format	Drop-down list	Click to select the image format.
Options	Button	Click to open the saving as image properties dialog. For more information, see "Properties Dialog" on page 263.
Overwrite existing files	Check box	Select to overwrite already saved image files.
Save	Button	Click to save the selected diagram(s) according to specified properties for saving as image.
Close	Button	Click to close the dialog.
Help	Button	Click to open MagicDraw Help.

Properties Dialog

In the **Properties** dialog, you can change image size, resolution (DPI), and specify other image properties that are specific to the selected image format.

To specify image options

1. From the **File** menu, select the **Save As Image** command. The **Save As Image** dialog opens.
2. Select the file format in the **Image Format** drop down list.

Copyright © 1998-2015 No Magic, Inc.

3. Click the **Options** button near the **Image Format** drop down list. The **Properties** dialog opens.

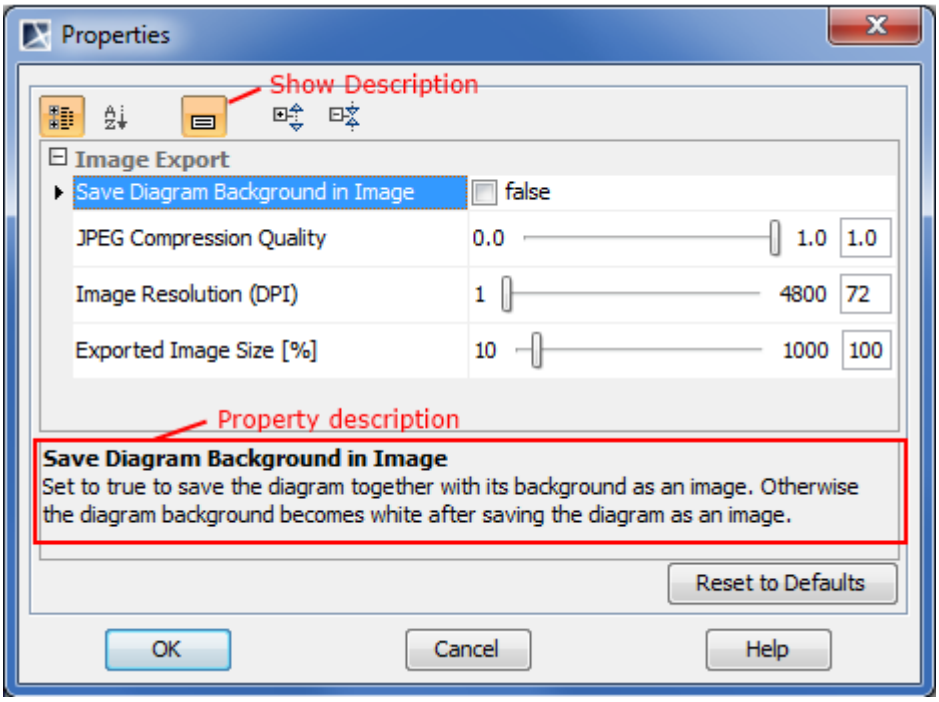


Figure 149 -- Properties dialog

NOTE

You can also define image saving options in the **Environment Options** window. For more information, see "Customizing Environment Options" on page 96.

Printing

With MagicDraw you are able:

- To print an active diagram
- To print a any diagram(s)
- To print all opened diagrams
- To print selected symbols of active diagram

You can specify the printing options or preview the pages by using the following dialogs:

- Print Dialog
- Print Options Dialog
- Print Preview Window

To print an active diagram

1. Open a diagram that you want to print.
2. Do one of the following:
   - In the main toolbar, click the **Print Active Diagram** button.
   - From the diagram pane, invoke the diagram shortcut menu, and then select the **Print Active Diagram** command.
   - From the **File** menu, select **Print**. On the **Print** dialog, under **Print what**, select **Active diagram**, and then click the **Print** button.

Copyright © 1998-2015 No Magic, Inc.

To print a any diagram(s)

1. From the **File** main menu, select **Print**. The **Print** dialog opens.
2. Under **Print what**, select **Selected diagram(s)**. The **Select Diagram** dialog opens.
3. Select diagram(s) that you want to print. For more information about working with the **Select Diagram** dialog, see "Selecting an Element" on page 352.
4. Click **OK**.
5. Click **Print**.



The **Print** button is unavailable if there are no selected diagrams.

To print all opened diagrams

1. Open all the diagrams that you want to print.
2. From the **File** main menu, select the **Print** command. The **Print** dialog opens.
3. Under **Print what**, select **All opened diagram(s)**.
4. Click **Print**.



The **Print** button is unavailable if there are no opened diagrams.

To print selected symbols of active diagram

1. Open a diagram.
2. On the diagram, select a symbol or symbols that you want to print.
3. Do one of the following:
   - On one of the selected symbols, make right-mouse click. In shortcut menu, select **Print Selected**.



The **Print Selected** command in the diagram shortcut menu exists only if on the diagram pane there is selected more than one symbol.

   - From the **File** menu, select **Print**. On the **Print** dialog, under **Print what**, select **Selected symbols of active diagram**, and then click the **Print** button.



The **Print** button is unavailable if there are no selected symbols in the active diagram.

Related references

Print Dialog
Page Setup Dialog
Header/Footer Dialog
Print Options Dialog
Print Preview Window

Print Dialog

To open the **Print** dialog, do one of the following

- From the **File** menu, select **Print**.

Copyright © 1998-2015 No Magic, Inc.

- Press CTRL+P.

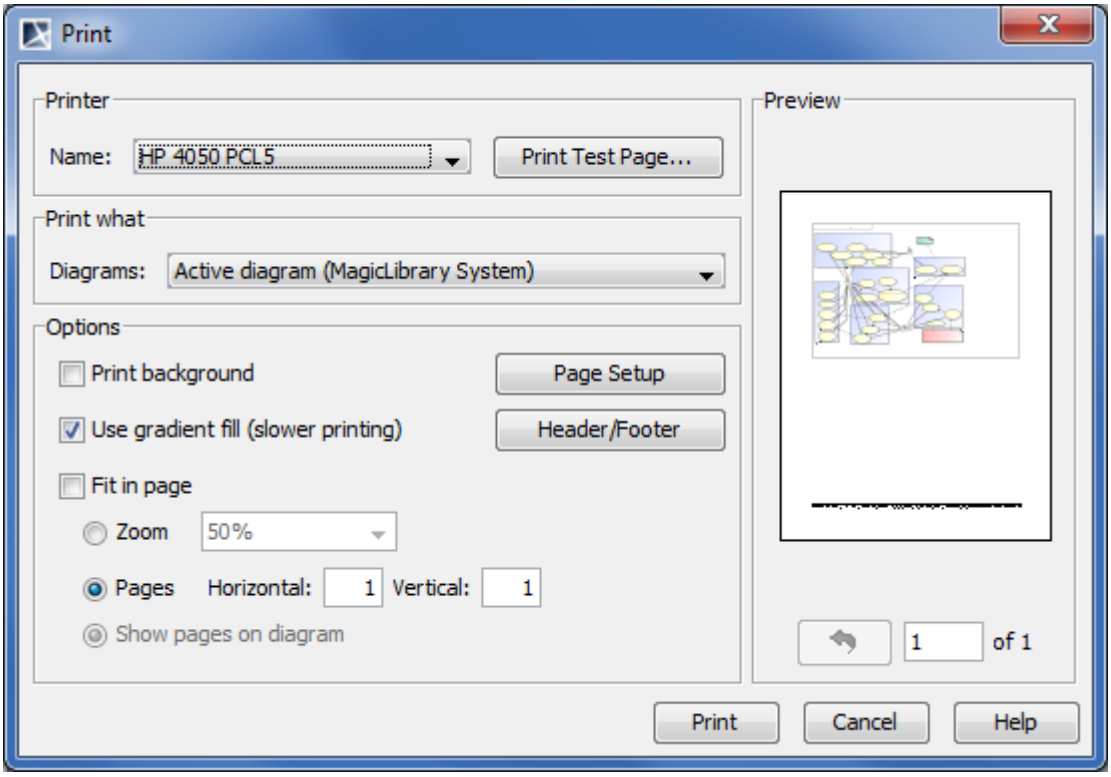


Figure 150 -- Print dialog

The following table describes elements and buttons that are used in the dialog.

Group name	Element name	Type	Description
Printer	**Name**	Drop-down box	Select the available printer.
	Print Test Page	Button	Print the test page to check page properties. The simplified **Print** dialog opens.
Print what	**Diagrams**	Drop-down box	Select one of the available options: • Active diagram (default). Select to print the currently active diagram. The active diagram name is displayed in the brackets, for example: *Active diagram (MagicLibrary System)*. • Selected diagram(s). You can use this option to print more than one diagram. The **Select Diagram** dialog opens. Select to print the diagrams. For more information, see "Selecting an Element" on page 352. **NOTE:** The number of the selected diagrams is displayed in the brackets near the **Selected diagrams** item. • All opened diagrams. Select to print all diagrams that are opened in the current project. • Selected symbols of active diagram. Select to print the symbols that are selected on the active diagram.
Options	**Print background**	Check box	Select to print the background color of your diagrams.

Copyright © 1998-2015 No Magic, Inc.

Group name	Element name	Type	Description
	Use gradient fill	Check box	Select this option to enable the diagram symbols gradient fill in printing. NOTE: Printing becomes slower when the **Use gradient fill** check box is selected.
	Page Setup	Button	Click to open the **Page Setup** dialog and to customize page properties. For more information, see "Page Setup Dialog" on page 268.
	Header/Footer	Button	Click to open the **Header/Footer** dialog. For more information, see "Header/Footer Dialog" on page 269.
	Fit in page	Check box	Select if you want to fit the selected diagram into one page.
	Zoom	Option button	Zoom the selected diagram to the size you want for printing (in percents). **NOTE**: You are not allowed to zoom a diagram if the **Fit in page** check box, and the **Pages** option button are selected.
	Pages	Option button	Set the number of pages to print. • Horizontal. The number of horizontal pages. • Vertical. The number of vertical pages. **NOTES**: • The **Pages** option is unavailable if the **Fit in page** check box is selected. • When the **Pages** option button is selected, the **Zoom,** and **Show pages** on diagram option buttons are unavailable.
	Show pages on diagram	Check box	Select to display the page boundaries and number of pages on the diagram pane. **NOTE**: The **Show pages on diagram** option button is unavailable if the **Fit in page** check box, and the **Pages** option button are selected.
Preview		Preview area	Click the previous and next buttons for previewing the diagram(s)' content arrangement on page(s). For more details about the print preview that allows to view the detailed print preview, see "Print Preview Window" on page 271.
		Button	Preview the previous page. **NOTE**: The button is available if there is more than one page.
		Button	Preview the next page. **NOTE**: The button is available if there is more than one page.
	Print	Button	Click to print the selected diagram(s) according to the specified properties for printing. The simplified **Print** dialog opens.
	Close	Button	Click to cancel printing and to leave the **Print** dialog.
	Help	Button	Click to open UML Help.

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Printing

Related references

Page Setup Dialog
Header/Footer Dialog
Print Options Dialog
Print Preview Window

Page Setup Dialog

In the **Page Setup** dialog you can modify the page format, orientation, and margins.

To open the **Page Setup** dialog, do one of the following

- Open the **Print** dialog, and then click the **Page Setup** button.
- Open the **Print Options** dialog, and then click the **Page Setup** button.

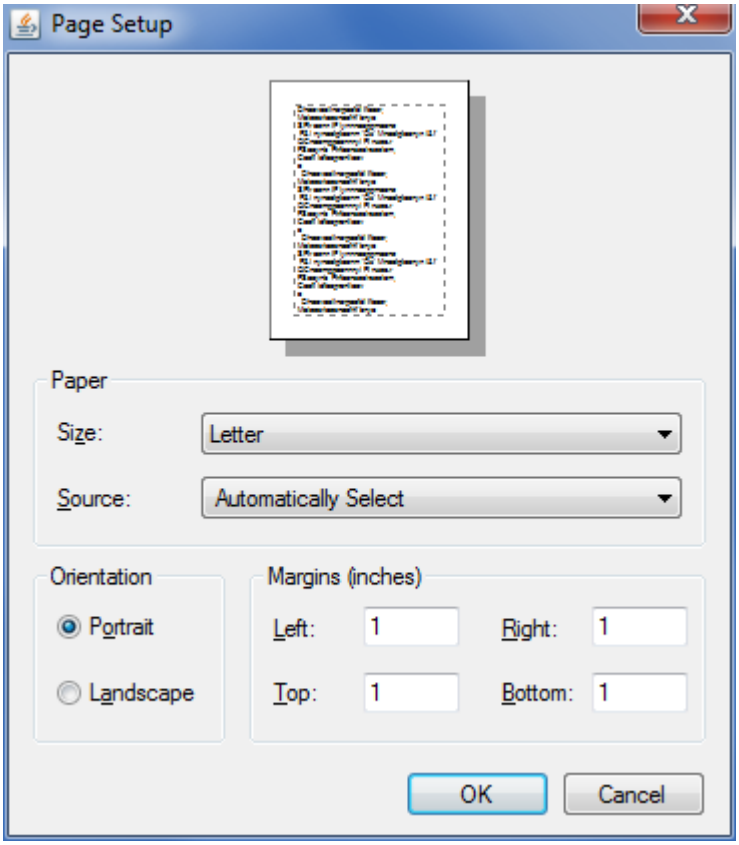


Figure 151 -- Page Setup dialog

Related concepts

Printing

Related references

Print Dialog
Print Options Dialog

Copyright © 1998-2015 No Magic, Inc.

Header/Footer Dialog

In the **Header/Footer** dialog you can customize the header and footer for the pages.

To open the **Print Header/Footer** dialog, do one of the following

- Open the **Print** dialog, and then click the **Header/Footer** button.
- Open the **Print Options** dialog, and then click the **Header/Footer** button.

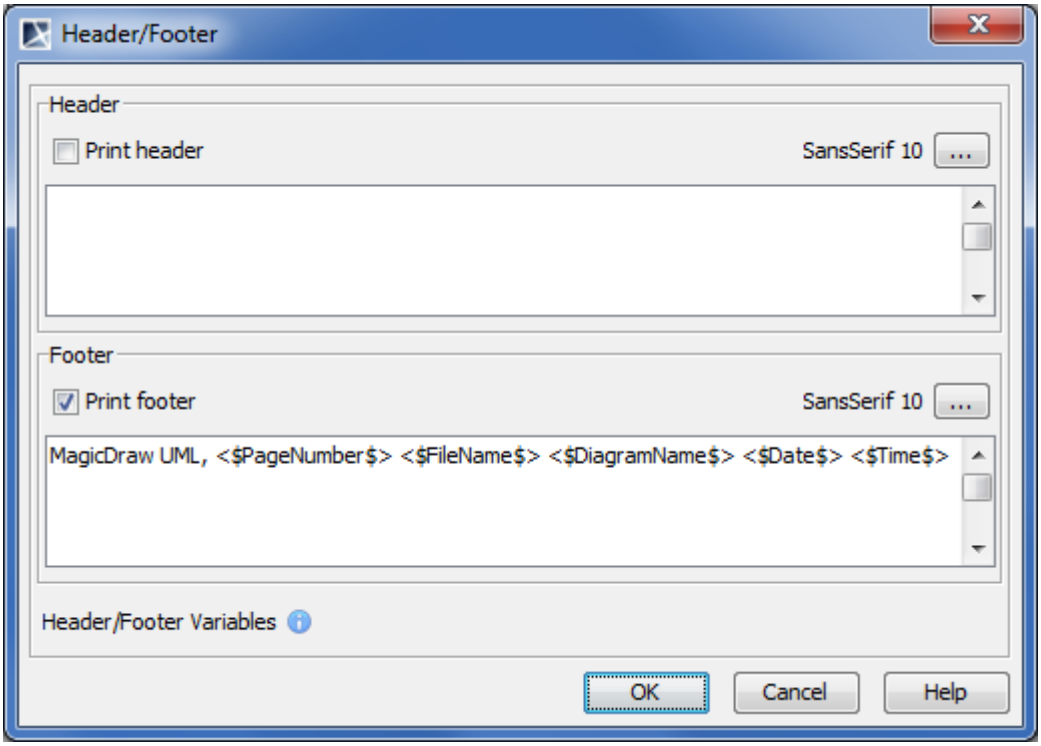


Figure 152 -- Header/Footer dialog

The following table describes elements and buttons that are used in the dialog.

Group name	Element name	Type	Description
Header	**Print header**	Check box	Select or clear the **Print header** check box to include or not the specified header in the printed pages. The **Print header** check box is cleared by default.
		Font and... button	Click the **...** button to select the desired font.
		Text box	Type the text to display it in the page footer.
Footer	**Print footer**	Check box	Select or clear the **Print footer** check box to include or not the specified footer in the printed pages. The **Print footer** check box is selected by default.
		Font and... button	Click the **...** button to select the desired font.

Copyright © 1998-2015 No Magic, Inc.

Group name	Element name	Type	Description
		Text box	Type the particular parameters to create the page footer. By default the following parameter values are added to the page footer: *UML, <$PageNumber$> <$FileName$> <$DiagramName$> <$Date$> <$Time$>*
Header/ Footer Variables		Information tip	Move the mouse cursor over the button and hold it there for a few seconds. The list with described variables opens.

Related concepts

Printing

Related references

Print Dialog

Print Options Dialog

Print Options Dialog

In the **Print Options** dialog, you can specify a printer, printing options, and preview the pages.

To invoke the Print Options dialog

- From the **File** menu, select the **Print Options** command.

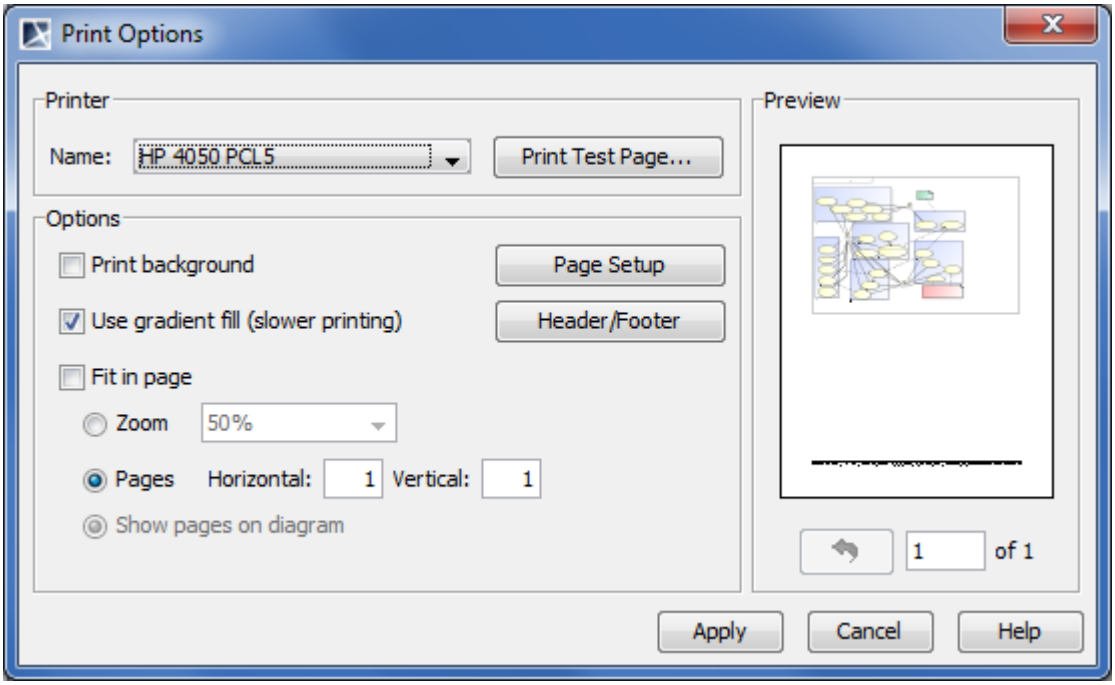


Figure 153 -- Print Options dialog

NOTE

The **Print Options** dialog has the same components as the **Print** dialog, except the **Print what** group. For more information about the **Print** dialog, see "Print Dialog" on page 265.

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Printing

Related references

Print Dialog
Page Setup Dialog
Header/Footer Dialog
Print Preview Window

Print Preview Window

In the **Print Preview** window, you can select the diagrams for printing, and preview the pages before printing.

To open the Print Preview window

- From the **File** menu, select the **Print Preview** command.
- In the main toolbar, click the **Print Preview** button.

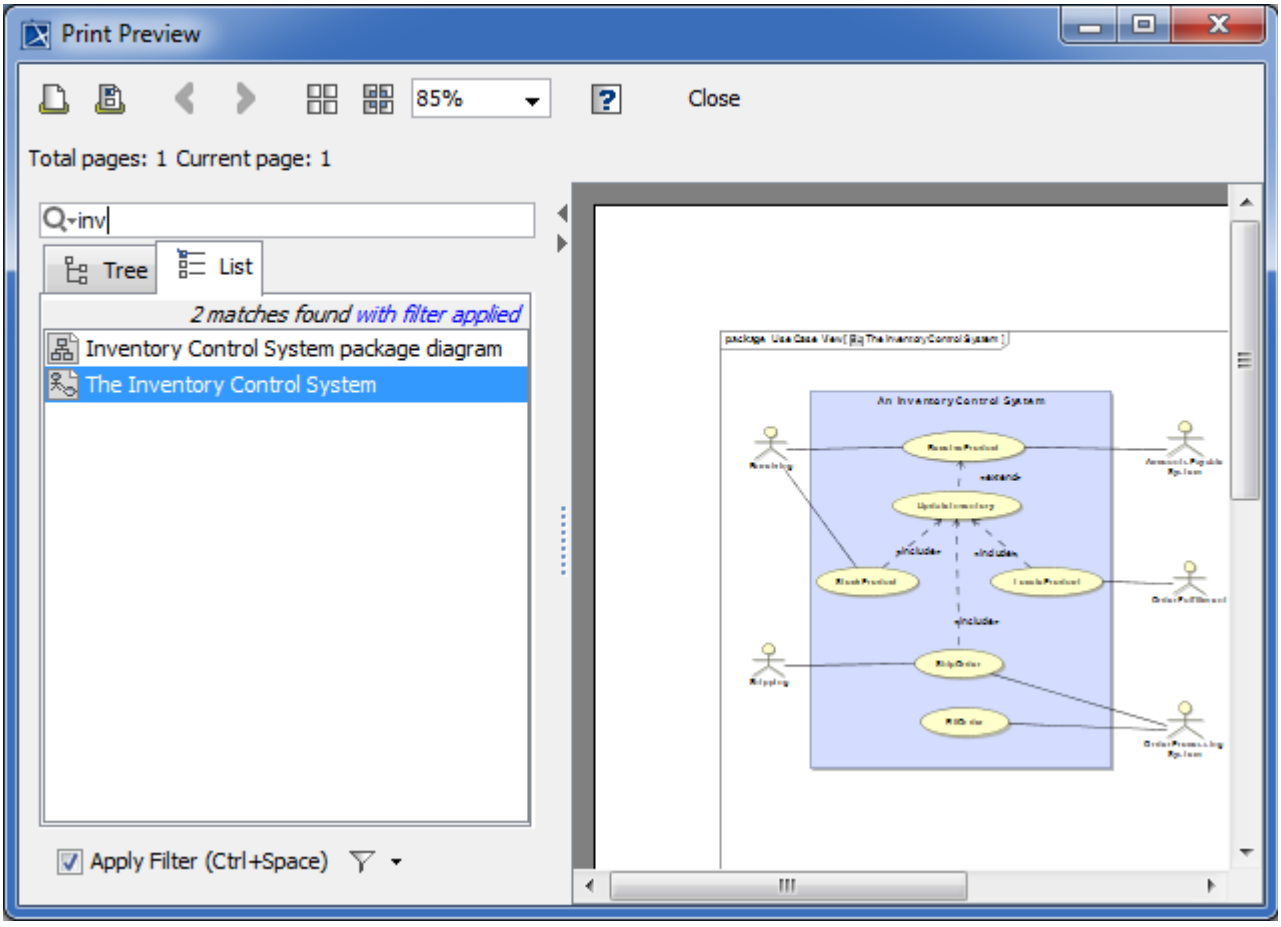


Figure 154 -- Print Preview window

The following table describes dialog elements and buttons that are used in the dialog.

Element name	Type	Description
Print	Button	Click to open the **Print** dialog for selecting a printer and its properties.

Copyright © 1998-2015 No Magic, Inc.

Element name	Type	Description
Print Options	Button	Click to open the **Print Options** dialog. For more information, see "Print Options Dialog" on page 270.
Previous Page	Button	Click to display the previous page. **NOTE**: The button is available when there are selected two or more pages.
Next Page	Button	Click to display the next page of the selected diagram. **NOTE**: The button is available when there are selected two or more pages.
Page Select	Button	Click the button to open the menu there you can set the number and layout of pages to display at one time. It is convenient to use the button when the diagram(s) do not fit in one page.
Actual Layout	Button	Click to display all pages of the selected diagram in the actual layout.
Preview Zoom	Drop-down box	Click to select the preview zoom size. **NOTE**: You can change the zoom size by one step, by clicking the diagram preview with the left-mouse button.
Help	Button	Click to display UML Help. **NOTE**: You can also open Help by pressing F1.
Close	Button	Click to close the dialog.
Search By Name	Text box	Type the name of the diagram you are searching for.
NEW! 	Menu button	Click the button to customize search options. For more information, see "Quick Find dialog" on page 137.
Tree	Tab	Click the **Tree** tab to display the hierarchical structure of the diagrams and their owners. For more information, see "Element Selection Views" on page 353.
List	Tab	Click the **List** tab to display the diagrams listed in the alphabetical order. For more information, see "Element Selection Views" on page 353.
NEW! **Apply Filter (Ctrl+Space)**	Check box	Click to clear the check box, if you need to remove the search scope filter.
NEW! 	Menu button	Click the button and customize search scope filter options. Be sure the **Apply Filter** check box is selected! For more information, see "Quick Find dialog" on page 137.

Related concepts

Printing

Related references

Print Dialog
Page Setup Dialog
Header/Footer Dialog
Print Options Dialog

Copyright © 1998-2015 No Magic, Inc.

6 WORKING WITH MODEL ELEMENTS

The chapter includes the following sections:

- "Specification Window" on page 273
- "Default Property Values" on page 296
- "Editing Property Values" on page 298
- "Formatting Symbols" on page 317
- "Style Engine" on page 324
- "Customizing Lists of Properties and Commands" on page 336
- "Defining Hyperlinks" on page 340
- "Owner of the Model Element" on page 346
- "NEW! Creation of Elements from other Resources" on page 349
- "Selecting an Element" on page 352
- "Selecting Port and Part Type" on page 357
- "Refactoring" on page 359
- "Extracting" on page 368
- "Generic Numbering Mechanism"on page 394
- "Smart Packages" on page 402
- "Favorites" on page 416
- "HTML Editor" on page 422
- "Copying/Opening Element URLs" on page 429
- "Parameters Synchronization" on page 430

Specification Window



Specification Windows

You can define all model elements in the Specification window.

MagicDraw shortcut menus, toolbars, and browser help ease the task of editing model elements.



MagicDraw also allows the editing of model elements and symbol properties directly in the **Properties** tab, which is located at the bottom of the Model Browser. For more information, see “Properties tab” on page 95.

Copyright © 1998-2015 No Magic, Inc.

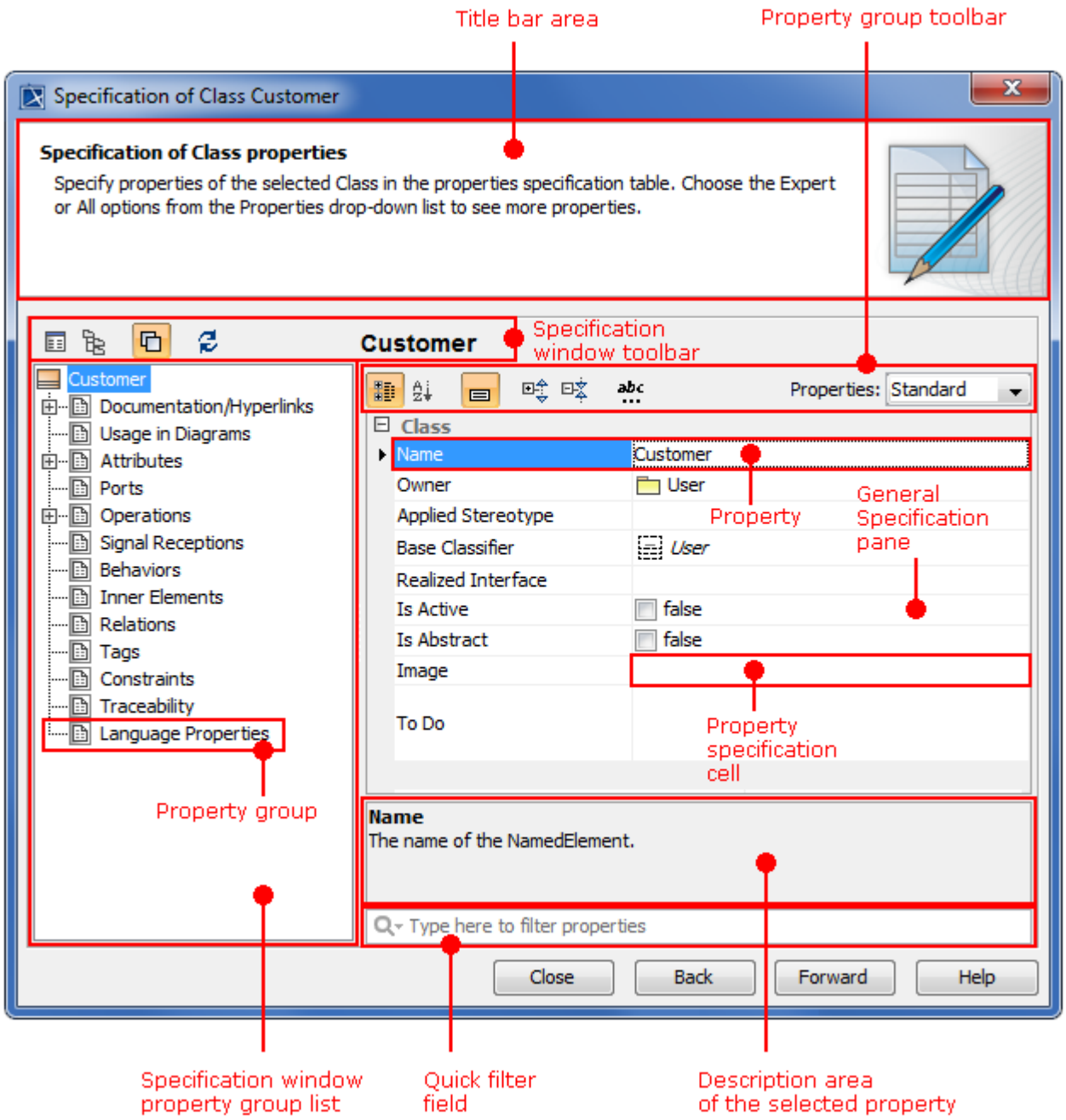


Figure 155 -- Specification window structure. General specification property group

The Specification window is used to define UML model elements such as class, package, activity, and others. Specification window is a non-modal window in which you may edit model element properties and work with a model simultaneously. For more information on how to edit property values, see "Editing Property Values" on page 298.

Copyright © 1998-2015 No Magic, Inc.

To open the element Specification window

Do one of the following:

- From the selected symbol shortcut menu, select **Specification**.
- Double-click a symbol on the Diagram pane or in the Model Browser.
- Select a symbol on the Diagram pane and press the ENTER key.
- The element Specification window opens when you add a model element to an owning model element in its Specification window. The second Specification window opens on top of the first. Use the **Back to** or **Forward to** arrow buttons for switching between windows.

Related references

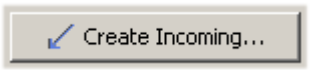
Functions of Specification Window
Specification Window Toolbar
Property Group Toolbar
Quick Filter
Specification Window Property Groups
Buttons Available in Specification Window

Functions of Specification Window

Using Specification window you may perform various actions that are necessary when working with model elements. Every model element has its own specification. In the following table, you will find described only common functions that Specification window is used for.

Function	How to
Add/modify model element properties and inner elements.	Choose the desired property group and specify its properties in the properties specification pane.
Open referenced elements specifications and work with them in the same window.	Click  or 
Navigate between specifications that were opened	Click  or 

Copyright © 1998-2015 No Magic, Inc.

Function	How to
Manage relations	●View all relationships in which the element participates. ●Modify the name of the relationship. ●View/change the direction of the relationship. ●Modify the target element. ●Create new outgoing  or incoming  relationships.
Add/edit the element documentation	Add or edit a documentation of the element. Documentation also can be written in HTML. More information about working with HTML text, see in “HTML Editor” on page 422.
Manage element hyperlinks	Open, edit, add, or remove hyperlinks from the selected model element to a file, address, other element/symbol, or requirements. More information about working with hyperlinks, see “Defining Hyperlinks” on page 340.
Manage element tags and their values	More information about working with tags you may find in "Editing tagged value" on page 992.
Manage element constraints	More information about working with constraints you may find in "Working with Constraints" on page 866.
Track the element’s symbol usage in diagrams and open these diagrams.	Find out in which diagrams the symbol is used.
Select the model element in the Containment tree	Click  or 
Track elements traceability	More information about the traceability you may find in Section “Traceability” on page 514.



Right-click the specification property and select the action you want to perform.

Related concepts

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Specification Window Toolbar

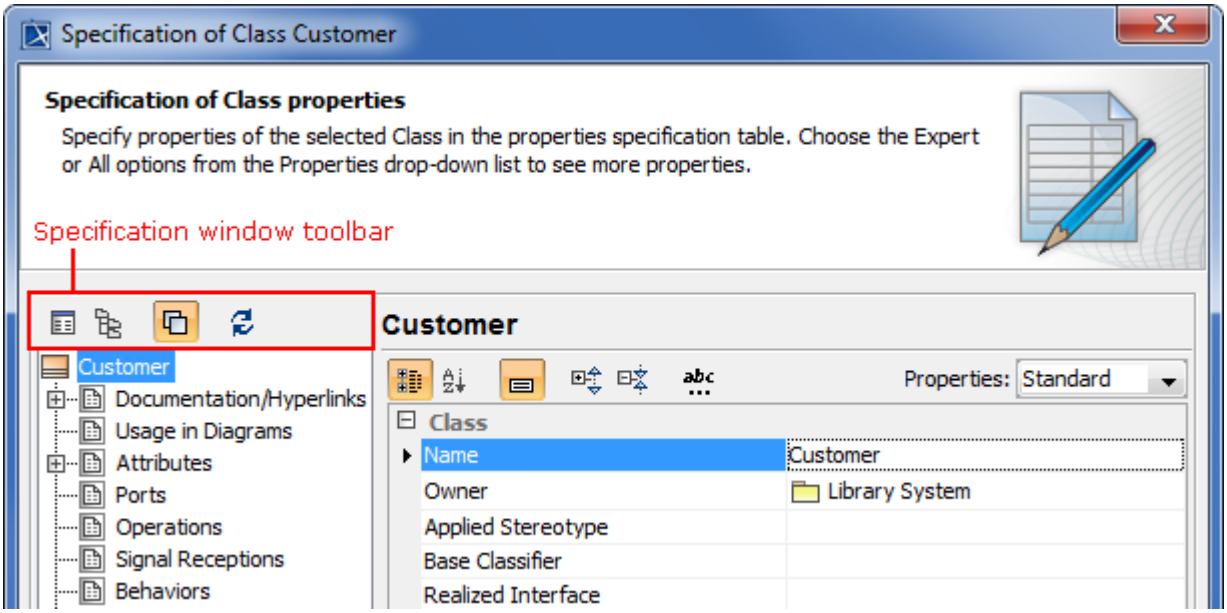


Figure 156 -- Specification window toolbar

Icon	Name	Function
	Open Specification	Opens Specification of the selected referenced element in the same window.
	Select in Containment Tree	Selects the selected element in the Containment tree of the Model Browser.
	Specification opening mode	If selected, the referenced Specification windows are opened in the same window.
	Refresh	Refresh data of the Specification window.

Related concepts

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Property Group Toolbar

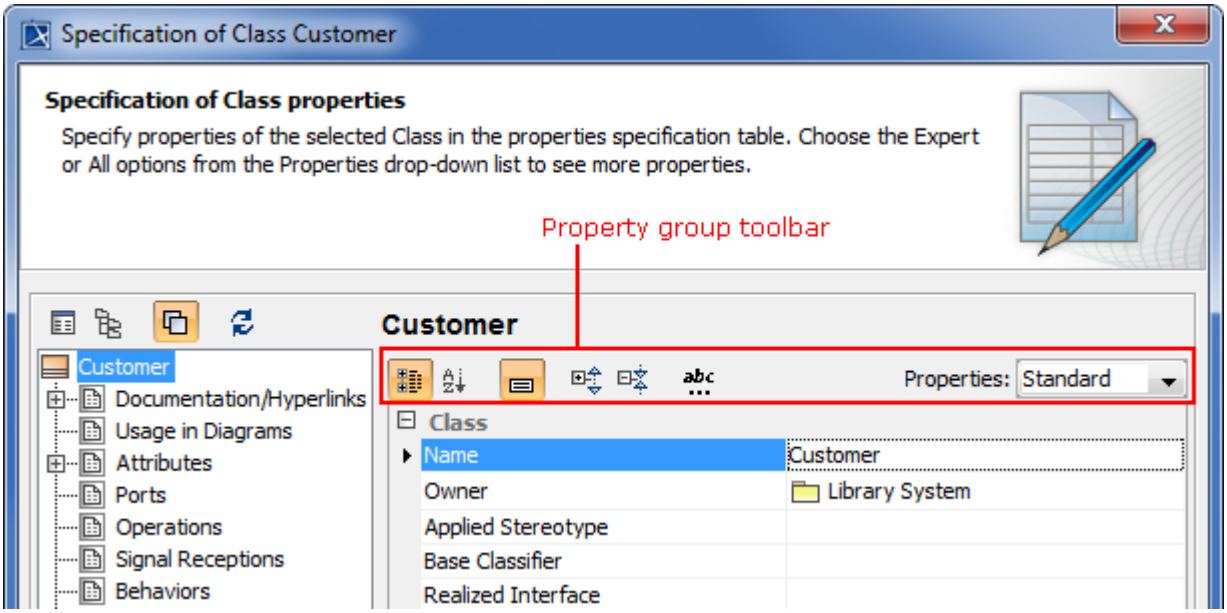
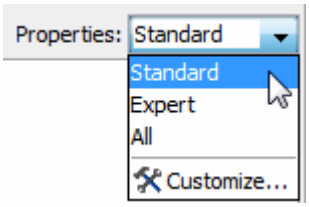


Figure 157 -- Property group toolbar in Specification window

Icon	Name	Function
	Categorized View	Groups properties by the categories.
	Alphabetical View	Lists properties alphabetically.
	Show Description	Turns on or off the description area wherein the description of the selected property is displayed.
	Expand	If element properties are categorized and those categories are collapsed, expands all categories.
	Collapse	If element properties are categorized and those categories are expanded, collapses all categories.
	Strip Multiline Text	If selected, the property's text (e.g., ToDo), which covers more than five rows, is striped, not showing all of it, by adding three dots at the end of the text.
	Properties	Displays the properties display mode: **Standard**, **Expert**, or **All**. Select **Customize**, to customize displayed properties. For more information, see "Customizing Lists of Properties and Commands" on page 336.

Copyright © 1998-2015 No Magic, Inc.



The property group toolbar which is described above is displayed only if the general element properties are selected in the property group list. For example, see in the figure above, in the property group list the top branch is selected, that is, the *Customer* class is selected. If, for example, the **Documentation/Hyperlinks** property group would be selected, other property group toolbar would be displayed.

Related concepts

Specification Window

Quick Filter



The quick filter field is displayed only then more than ten properties are displayed in the property list. For more information, see "Property Group Toolbar" on page 278.

Using the Quick filter box you may quickly find the required property in the property list. This is especially handy when the properties list is rather long. Properties can be filtered by the text entered in this box.

Click the Filter settings button to select filter options.

Copyright © 1998-2015 No Magic, Inc.

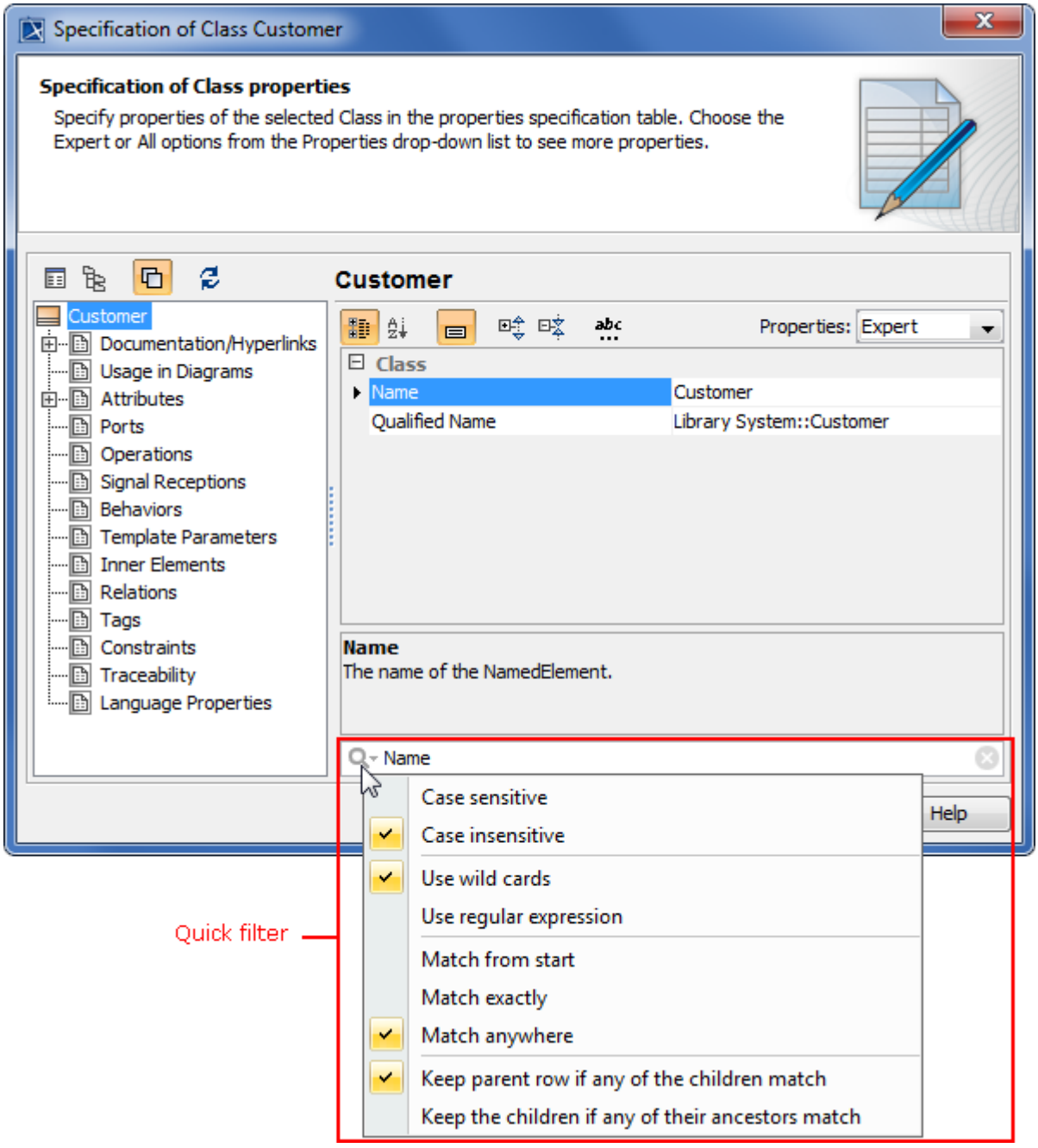


Figure 158 -- Quick filter box with its options.

Option name	Function
Case sensitive	Words differs in meaning based on differing use of uppercase and lowercase letters.
Case insensitive	Words do not differ in meaning based on differing use of uppercase and lowercase letters.
Use wild cards	Increase the flexibility and efficiency of a quick filter search by using wildcard characters that substitute any of a class of characters in a search.
Match from start	The search will be performed according to the first letters of the property.

Copyright © 1998-2015 No Magic, Inc.

Option name	Function
Match exactly	The search will be performed according to the exact name of the property.
Match anywhere	The search will be performed according to any of the letter of the property.
Keep parent row if any of the children match	Show the category name of the found property.
Keep the children if any of their ancestors match	Show all other properties that are in the same category as the found property.

Related concepts

Specification Window

Specification Window Property Groups

Model elements may have the following property groups:

- General property group.
- Documentation/Hyperlinks property group.
- Usage in Diagrams property group.
- NEW! Usage In property group
- Attributes property group.
- Operations property group.
- Inner Elements property group.
- Relations property group.
- Connectors property group
- Tags property group
- Constraints property group
- NEW! Instances property group

Related concepts

Specification Window

General property group

Name text box



Type or view the model element name. If you enter the name of an existing model element, an error message opens.

For some model elements (attribute, operation, and so forth), the default name *Untitled1* is set. You can change this name to a preferred name.

Copyright © 1998-2015 No Magic, Inc.

Is Active, or Is Abstract check boxes

Is Active	false
Is Abstract	false

When one of these check boxes is selected, the model element is correspondingly set as an active or abstract model element.

Applied Stereotype

Applied Stereotype	

Click the "..." button to open the list of all available applied stereotypes, select the check box for the chosen stereotype and click **Apply**.

Visibility

Visibility	public

To define an element access level, use the drop down list to set its visibility. There are four levels of access:

- **Public**. The element can be accessed by any outside object.
- **Package**. The element can be accessed by any classifier declared in the same package (or a nested subpackage, to any level).
- **Private**. The element can be accessed only from inside the current class.
- **Protected.** The element can be accessed from inside the current class and classes derived from that class.

ToDo

To Do	

Type or view information about an element. The **To Do** property is used for keeping special information, exclusive cases, or additional records.

Image

Click the **...** button to assign the image to the element. Assigned image can be displayed on the shape or instead of the shape.



For more information, about changing the image display mode, see "Displaying icon or image" on page 319.

Related concepts

Specification Window

Documentation/Hyperlinks property group

Use the **Documentation/Hyperlinks** property group to add comments to the selected element and to assign hyperlinks. The hyperlink can direct the user to a model element, address, or file.

Copyright © 1998-2015 No Magic, Inc.

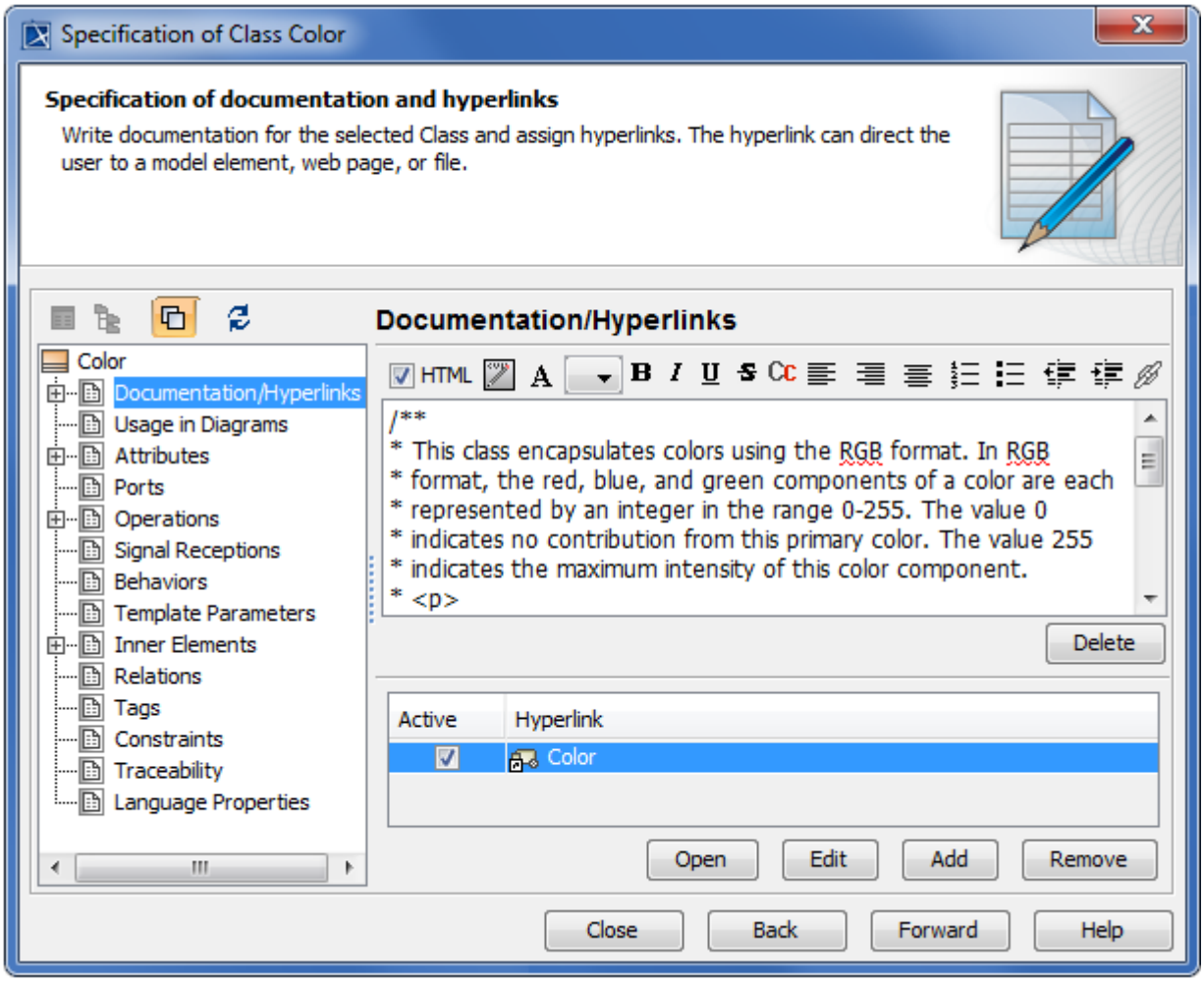


Figure 159 -- Specification window. Documentation/Hyperlinks property group

Writing HTML documentation

To write documentation in HTML format, simply select the HTML check box to display a menu with the available text formatting options.

TIP!

For more information about the HTML editor toolbar, see "HTML editor toolbar" on page 427.

Adding Hyperlinks

In the **Hyperlinks** group, manage the hyperlinks you want to add to the model element

Active	If selected, the hyperlink is activated and will work when double-clicking the model element.
Hyperlink	Display information about the hyperlink: a diagram or element name, file path, or URL name.
Open	Opens the previously assigned hyperlink.
Edit	The **Insert Hyperlink** dialog opens. Edit the selected hyperlink.
Add	The **Insert Hyperlink** dialog opens. Select the hyperlink you want to add to the model element.
Remove	Remove the selected hyperlink from the model element.

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Specification Window

Usage in Diagrams property group

The **Usage in Diagrams** property group lists the diagrams in which is drawn the symbol of the current element.



For more information about searching for symbol usage in diagrams from the **Usage In Diagrams** property group, see "Symbol Usage in Diagrams" on page 576.

Related concepts

Specification Window

NEW! Usage In property group

You can trace the Class and the Signal when they are used in other diagrams.

Class can be used as:

- Representer of the partition
- Part
- Representer of the lifeline
- Type for instances
- State activity
- Action

Signal can be used:

- In Trigger on Transitions or Accept Event Actions
- As Item Property type
- As Flow Property type
- In Send Signal Action or Broadcast Action
- In Sequence message

Copyright © 1998-2015 No Magic, Inc.

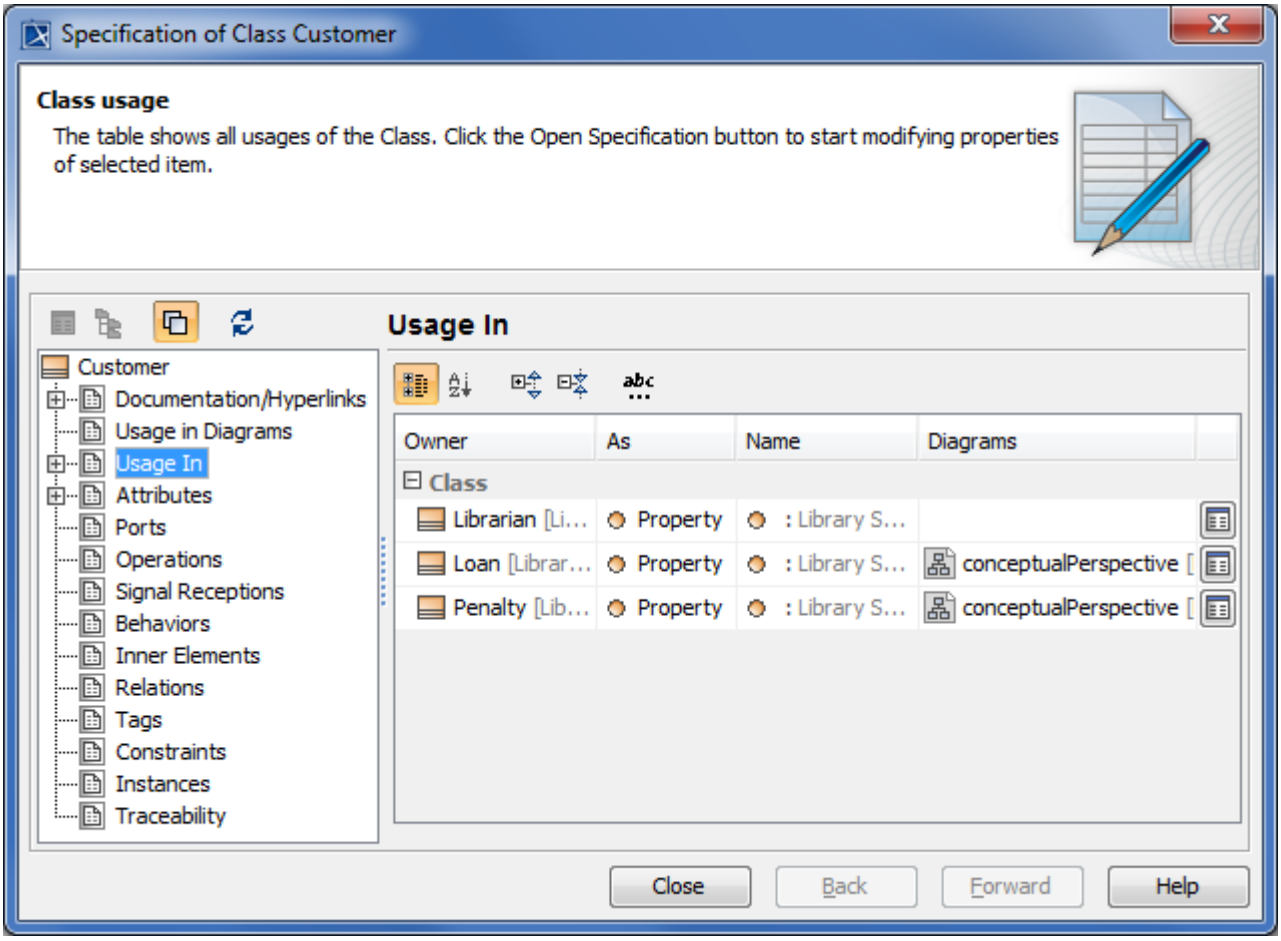


Figure 160 -- Specification window. Usage In property group

Column name	Description
Owner	The owner of the element type for which the class or signal is assigned.
As	Element type for which the class or signal is assigned.
Name	Name of the element type.
Diagrams	Name of diagram that displays the usage of the selected class or signal.

Button name	Description
	Opens the Specification window of the selected item.

Related concepts

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Attributes property group

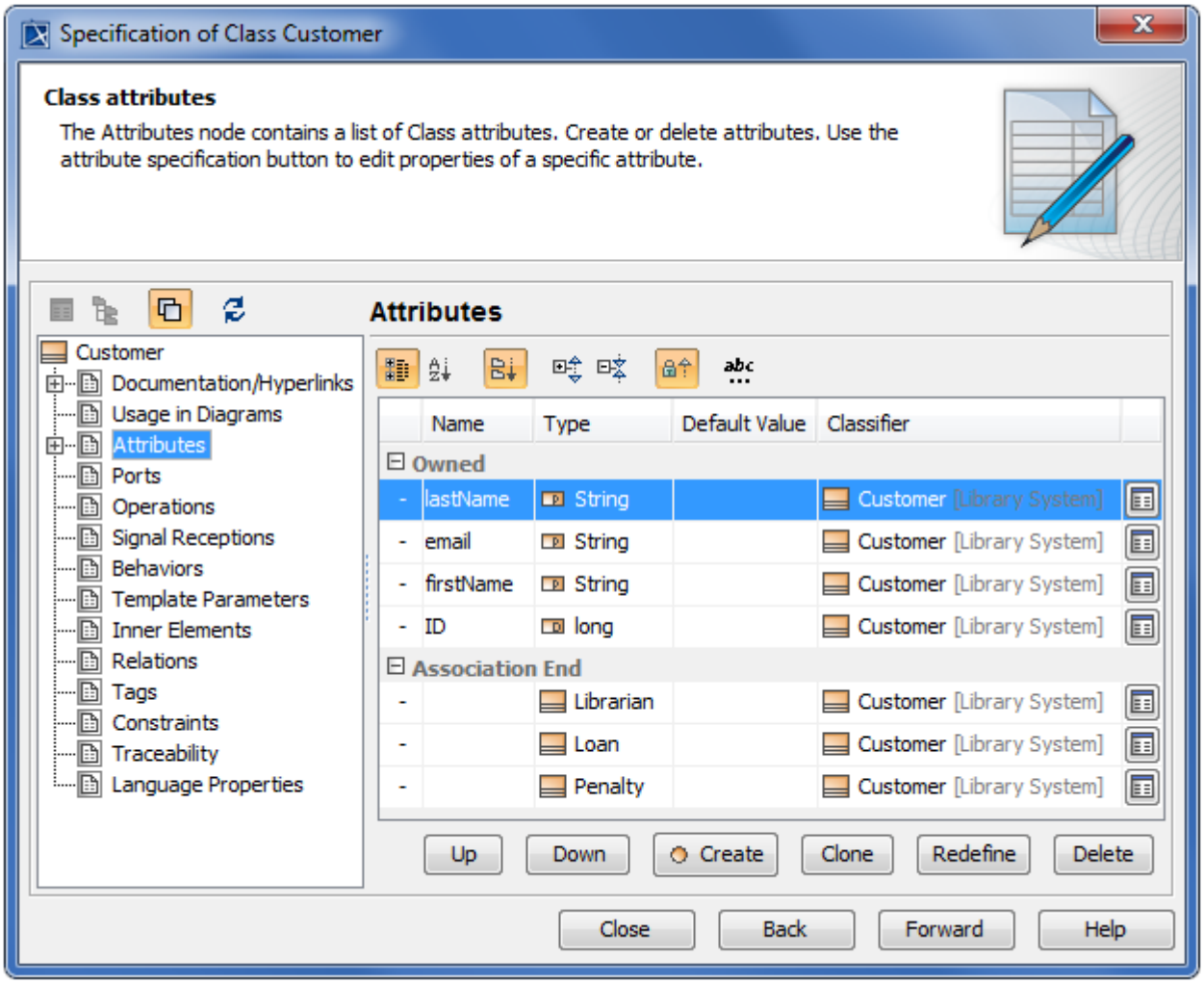
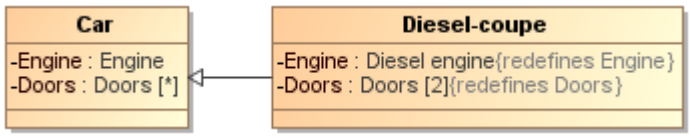


Figure 161 -- Attributes property group

The **Attributes** property group contains the model element attributes list and buttons for editing the attributes list.

Name	Attribute name.
Type	Attribute type. It can be a primitive type or another class.
Default Value	Attribute default value.
Classifier	Class name that contains the current attribute.
Up	Move item to upper position in the list.
Down	Move item to lower position in the list.
Create	Add a new attribute to the class. The attribute Specification window opens. **TIP!** You may also add a new attribute from the classifier by pressing Insert.
Clone	Enabled when the element is selected in the list. A new element will be created. The new element derives all properties from the cloned element. The name will be changed to “<element_name><number>”.

Copyright © 1998-2015 No Magic, Inc.

Redefine	Enabled when the element is selected in the list. A new element redefining an existing one will be created.  See an example, where the *Engine:Diesel engine* is an attribute that redefines *Engine:Engine*. It has a type *Diesel engine*, a specialization of *Engine*.
Delete	Remove the selected attribute from the classifier. **TIP!** You may also delete an attribute from the classifier by pressing Delete.
	Click this button to open the attribute Specification window.

Usage in Diagrams tab



For more information about searching for symbol usage in diagrams from the **Usage In Diagrams** branch, see "To search for diagrams in which symbol is used from the element Specification window" on page 577.

Related concepts

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Operations property group

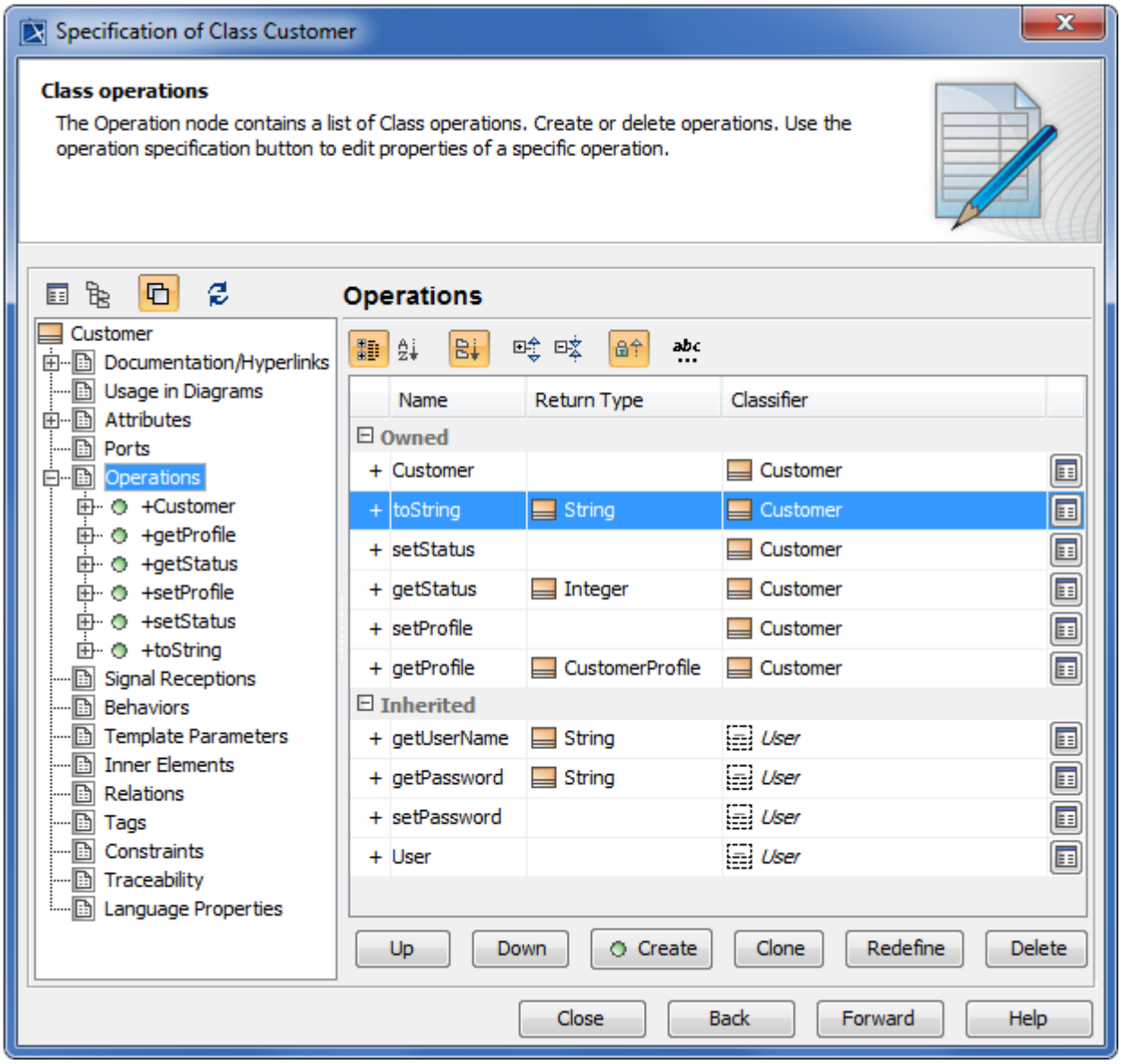


Figure 162 -- Operations property group

The **Operations** property group contains the model element operations list and buttons for managing this list.

Name	Operation name.
Return type	Operation return type.
Classifier	The name of the classifier containing the current operation.
Up	Move item to upper position in the list.
Down	Move item to lower position in the list.
Create	Add a new operation to the model element. The **Operation** Specification window opens. **TIP!** You may also add a new operation from the classifier by pressing Insert.
Clone	Enabled when the element is selected in the list. A new element will be created. The new element derives all properties from cloned element. The name will be changed to “<element_name><number>”.

Copyright © 1998-2015 No Magic, Inc.

Redefine	Enabled when the element is selected in the list. A new element redefining an existing one will be created.
Delete	Remove the selected operation from the model element. TIP! You may also delete an operation from the classifier by pressing Delete.
	Click this button to open the operation Specification window.

Related concepts

Specification Window

Inner Elements property group

In the **Inner Elements** property group, you can:

- Create a new inner element.
- Clone the selected inner element (that is, to create a new element with the copied data of the selected element).
- Delete the selected inner element.

Related concepts

Specification Window

Relations property group

The **Relations** property group contains the list of relationships in which the appropriate model element participates.

Copyright © 1998-2015 No Magic, Inc.

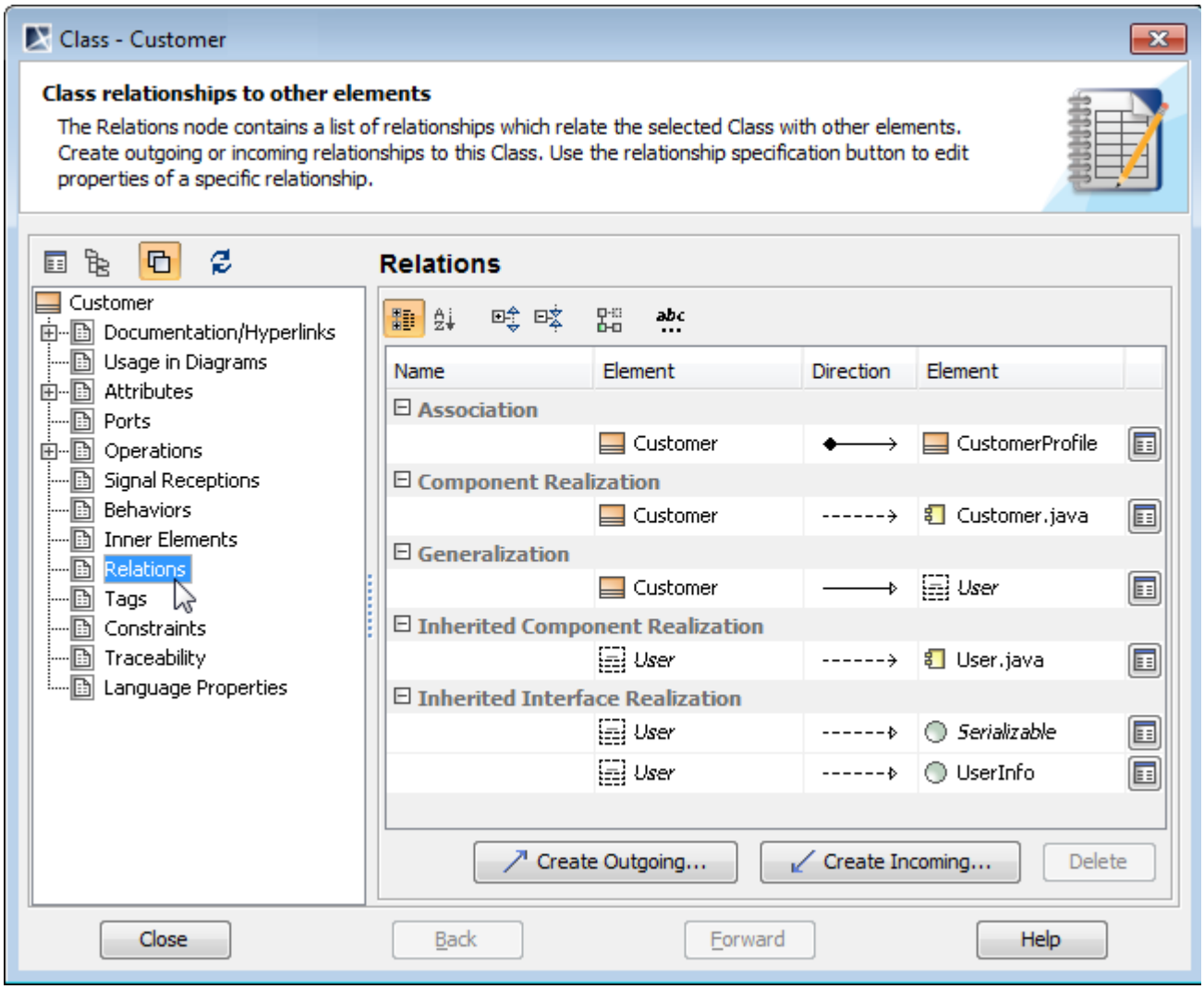
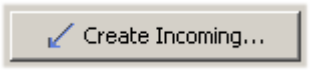
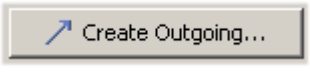


Figure 163 -- Specification window. Relations property group

Element name	Function
Name	Name of the relationship (optional).
Element	One of the relationship endings.
Direction	Shows a relationship's direction, helps to specify source and target.
Element	Another relationship ending.
Create Incoming 	Create a new incoming relationship by choosing the relationship type from the appeared list.
Create Outgoing 	Create a new outgoing relationship by choosing the relationship type from the appeared list.
	After clicking this button, the relationship Specification window opens.
Delete	Removes the selected relationship from the list.

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Specification Window

Connectors property group

The **Connectors** property group contains the list of connectors. Click the property group, to see the **Context** column, which allows you to distinguish connectors easily.

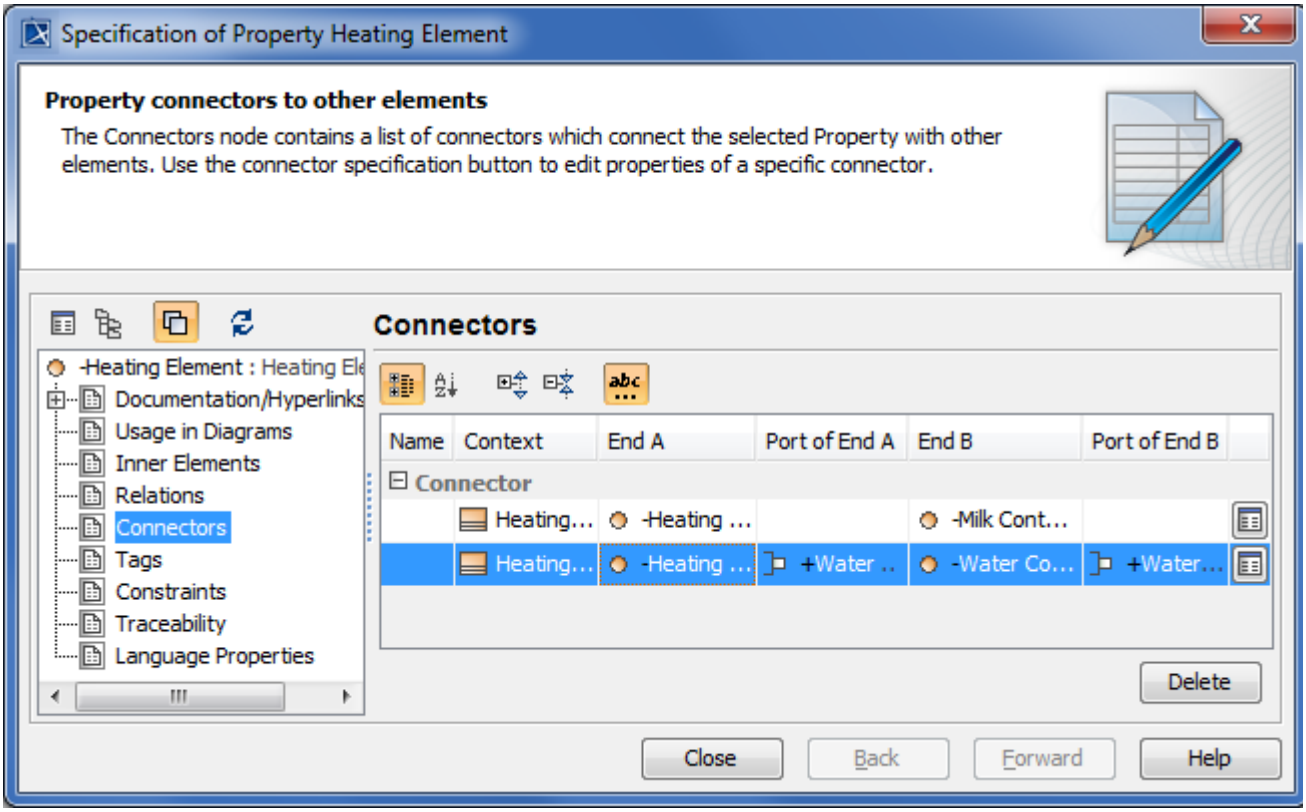


Figure 164 -- Specification window. Connectors property group

Column name	Description
Name	Name of the connector (optional).
Context	Owner of the connector.
End A	Name of the end A.
End B	Name of the end B.
NEW! Port of End A	Port name of the end A.
NEW! Port of End B	Port name of the end B.

Button name	Description
	Opens the Specification window of the selected connector.
Delete	Removes the selected connector from the list.

Related concepts

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Tags property group

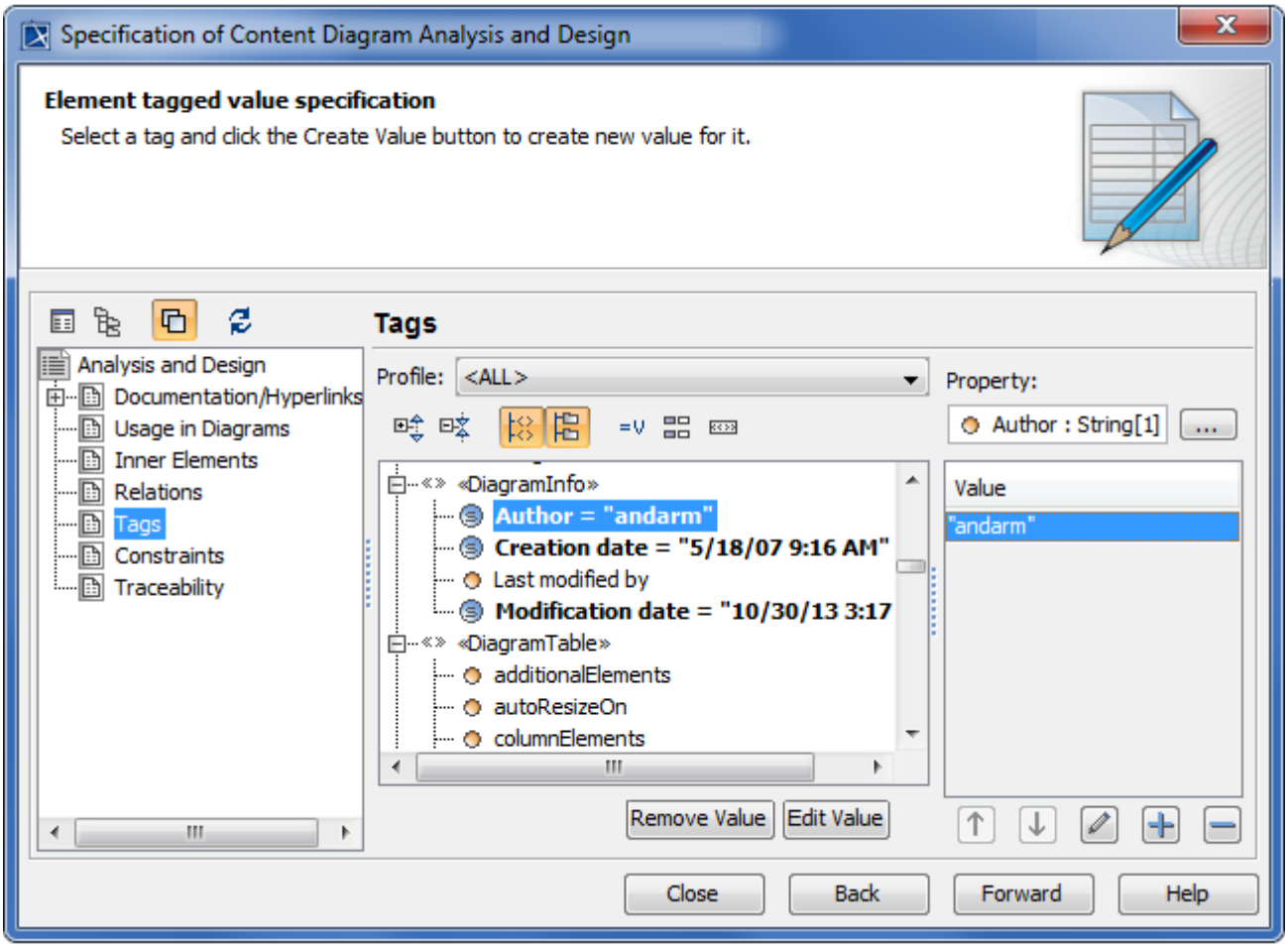


Figure 165 -- Specification window. Tags property group

Element name	Function
Profile	Lists the profiles available for the current project.
Expand	If tag definitions are grouped and those groups are collapsed, expands the groups.
Collapse	If tag definitions are grouped and those groups are expanded, collapses the groups.
Show Tags with Values	Only displays in the list those tag definitions that have values.
Show Tag Group	If tag definitions are grouped into specific “packages”, shows those “packages” on the list by grouping tag definitions.
Group by Stereotype	Sorts tag definitions by the assigned stereotypes.
Show Tag Type	Displays types of tag definitions in the list.

Copyright © 1998-2015 No Magic, Inc.

Element name	Function
Show Only Assigned Stereotypes Tags	In the list of tags, the assigned tag value is highlighted in black.
Create Value	Creates a value for the selected tag definition. The right pane of the dialog is activated. Select or enter the value. All data types and types of metamodel can be types of values. TIP! You may also create value by dragging and dropping an element from the Browser.
Remove Value (available only when the tag definition has a value)	Removes the value(s) from the selected tag definition.
Edit Value	The Slot Specification window opens, allowing you to edit or extend the selected tagged value.
Right pane of the dialog	
Property "..."	Click the "..." button and edit the selected property in the Property Specification window.
HTML	Set the tagged value text as HTML.
Value (if the value is added)	Type or select the value.
Edit	Edit the selected value.
Add	Add a new value.
Remove	Remove the selected value.



For more information about how to create a new tagged value, see "To create a new tag definition" on page 992.

Related concepts

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Constraints property group

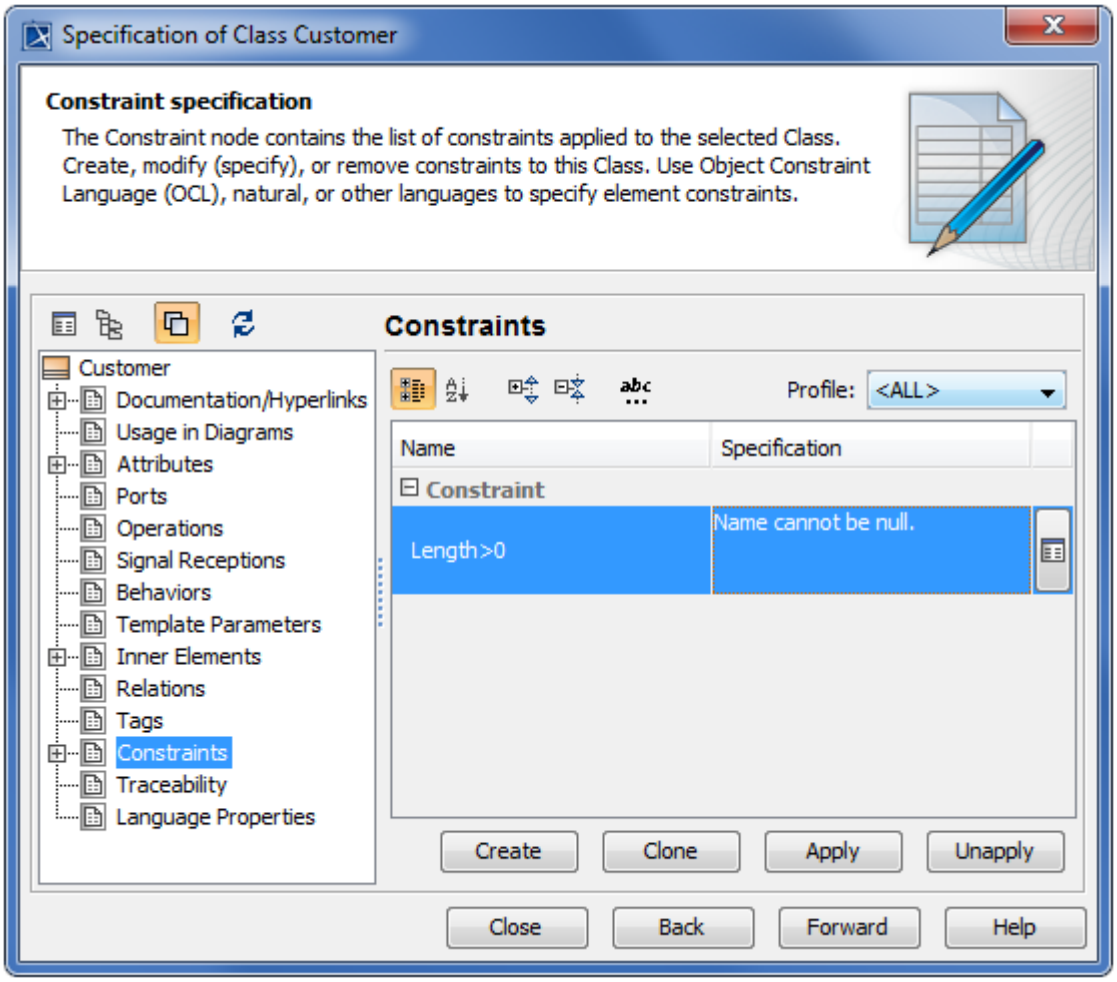


Figure 166 -- Specification window. Constraints property group

Element name	Function
	The list of constraints assigned to the current model element.
Name	Enter the name of the constraint.
Specification	Click the "…" button to open the **Edit Specification** dialog. This allows you to edit expressions and select the Object Constraint Language (OCL) to check the expression syntax.
	Click this button to open the constraint Specification window.
Create	Opens the menu with the available to create constraints. Click to create the duration, interaction, time, simple or custom constraint.
Apply	The **Select Elements** dialog opens. Select an existing constraint from the model or create a new one and apply it to the element.
Unapply	Removes the selected constraint from the list.

Related concepts

Specification Window

Copyright © 1998-2015 No Magic, Inc.

NEW! Instances property group

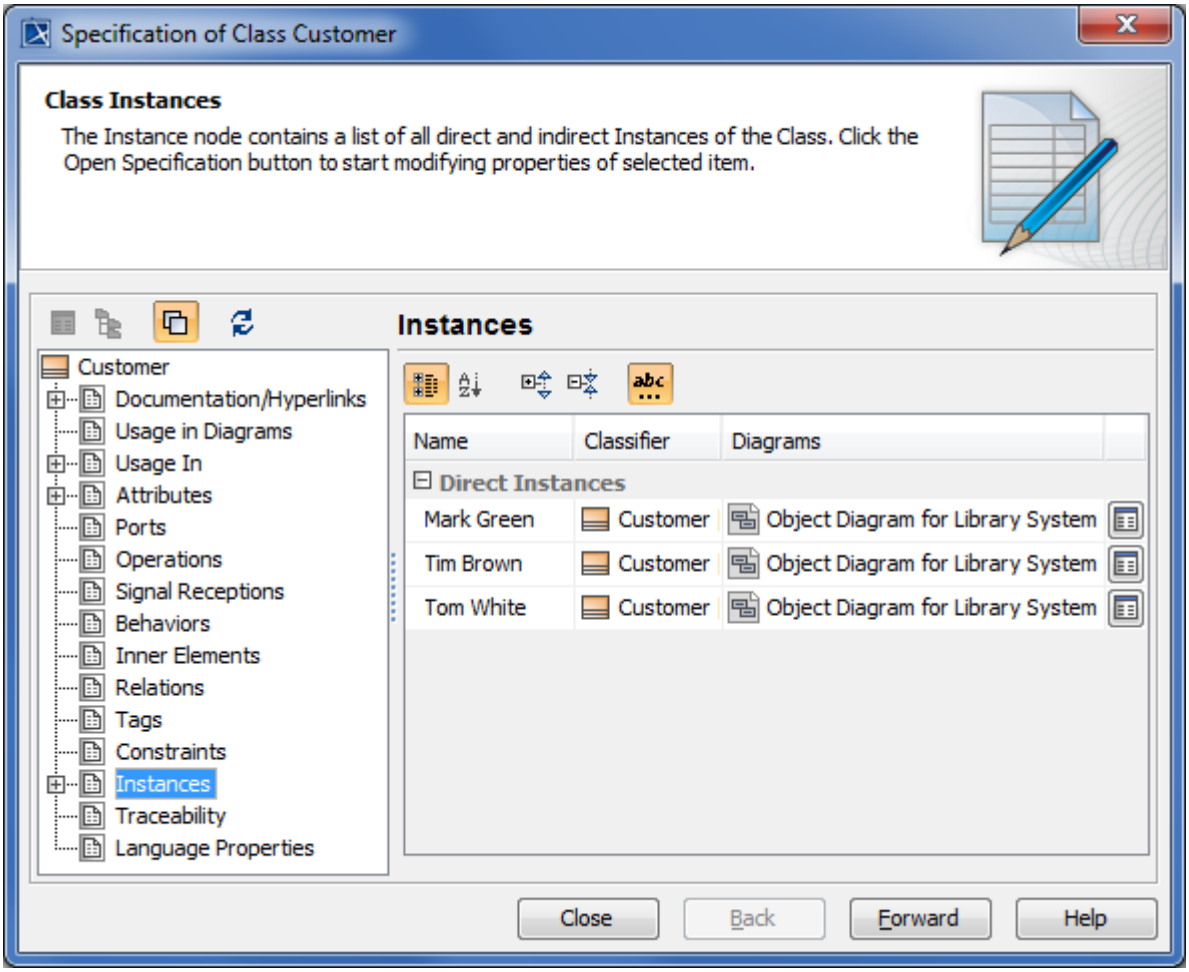


Figure 167 -- Specification window. Instances property group

Column name	Description
Name	Name of the instance.
Classifier	Owner of the instance.
Diagrams	Name of diagram that displays the usage of the selected instance.

Button name	Description
	Opens the Specification window of the selected instance.

Related concepts

Specification Window

Traceability property group

The **Traceability** property group in element's Specification window is one of the places, wherein the element's traceability properties, showing its realizing and/ or more specific elements, are represented.

Copyright © 1998-2015 No Magic, Inc.

For more information about traceability feature, see "Traceability" on page 514.

Related concepts

Specification Window

Buttons Available in Specification Window



Button	Function
Close	Save changes and exit the dialog.
Back	Return to the previous dialog.
Forward	Proceed to the next dialog.
Help	Display MagicDraw Help.

Related concepts

Specification Window

Default Property Values

MagicDraw allows for defining the initial (default) properties for elements.

The Default element properties can be defined for:

- The whole project.
- The specific diagram.

To set the default properties for the whole project

1. From the **Options** main menu, select **Project**. The **Project Options** dialog opens.
2. Expand **Default model properties**. Select the exact element and in the right pane side, change the property value.

After creating a new element it will have new property values. Values for previously created elements will not be changed.

To reset element properties to the default value, click the **Reset to Defaults** button. To reset property values for all elements select the **Default model properties** branch and click the **Reset to Defaults** button.

(Exception: interface attribute default visibility will always be #public, no matter what your settings.)

To set default properties for the specific diagram

1. From the **Diagrams** main menu, select the **Customize** command. The **Customize Diagrams** dialog opens.
2. Select the diagram type you want to customize.
3. Click the **Edit** button. The **Customize Diagram Wizard** opens.
4. In the fourth **Specify toolbar buttons** step, click the **Add** button. The menu appears.
5. In the appeared menu, select the **New Button** command. The **Edit Button** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

6. In the **General** tab, click to expand the **Model Element Type** drop down list and select the element, which default properties you want to customize.
7. In the **Edit Button** dialog, open the **Element Properties** tab.
8. Select the **Specify own values** radio button. Now change the default property values of the chosen element.

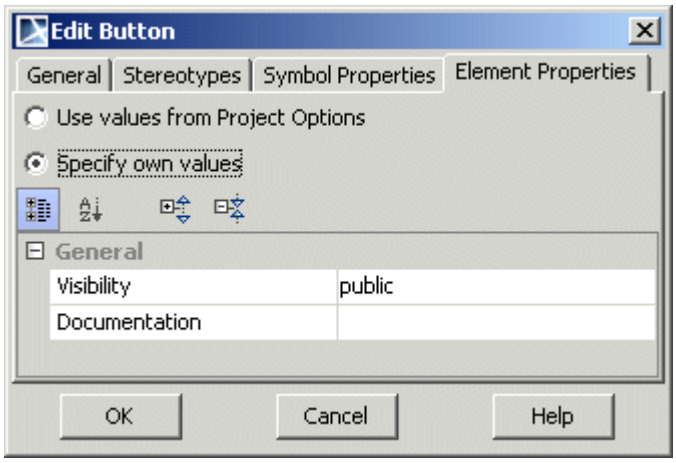


*Figure 168 -- The **Edit Button** dialog. **Element Properties** tab*

Create a new element from the customized diagram toolbar and the element default properties will be the same as you had defined.

TIP!

For more information about **Customize Diagram Wizard**, see MagicDraw UMLProfiling&DSL UserGuide.pdf, which is located in <*MagicDraw installation directory*>/*manual*.

Sharing the default property values

If you want to share the default properties with other users for their new project, change the property values in the **Project Options** dialog and then create a project template, which other users may use:

1. From the **File** main menu select the **Export** > **Template...** command and save the project as a template. A template will be created in <*MagicDraw installation directory*>*templates*.
2. To import the created template to a new project from the **File** main menu, select the **New Project** command. The **New Project** dialog opens. Select the **Project from Template** icon from the **Other** domain and in the **Select template** tree, select your template. The project options are imported to the project together with the template.

Copyright © 1998-2015 No Magic, Inc.

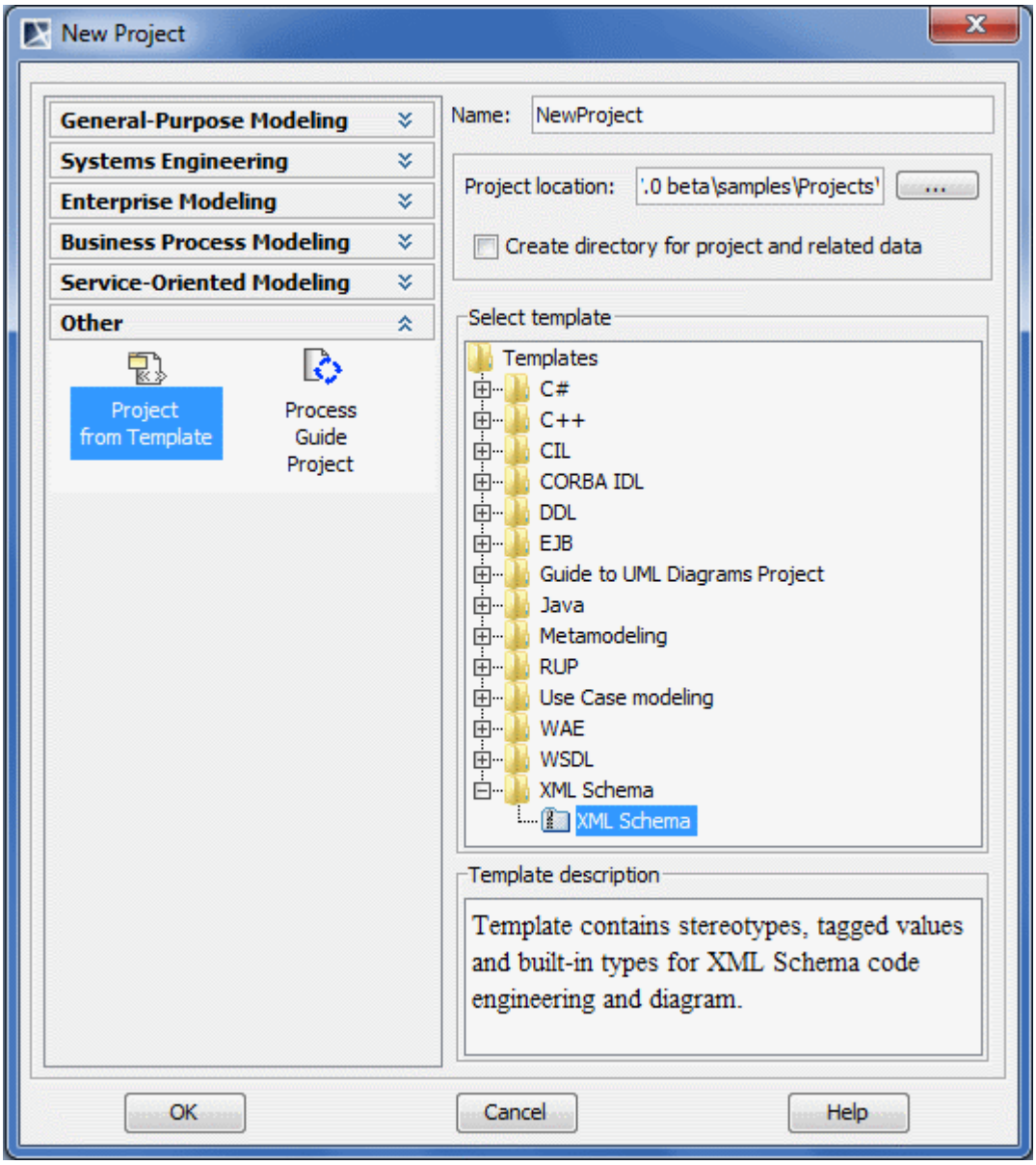


Figure 169 -- Importing template using New Project dialog

Editing Property Values

You can edit property values in:

- Specification window (see "Specification Window" on page 273).
- Generic table (see "Generic Table" on page 785).
- **Element** tab on the **Properties** panel in the Model Browser (see "Properties tab" on page 95).

A property value can be edited in one of the following ways:

- Directly in the value cell.
- Using the **Edit** command on the property's shortcut menu. Different types of properties have different command names on their shortcut menus.

Copyright © 1998-2015 No Magic, Inc.

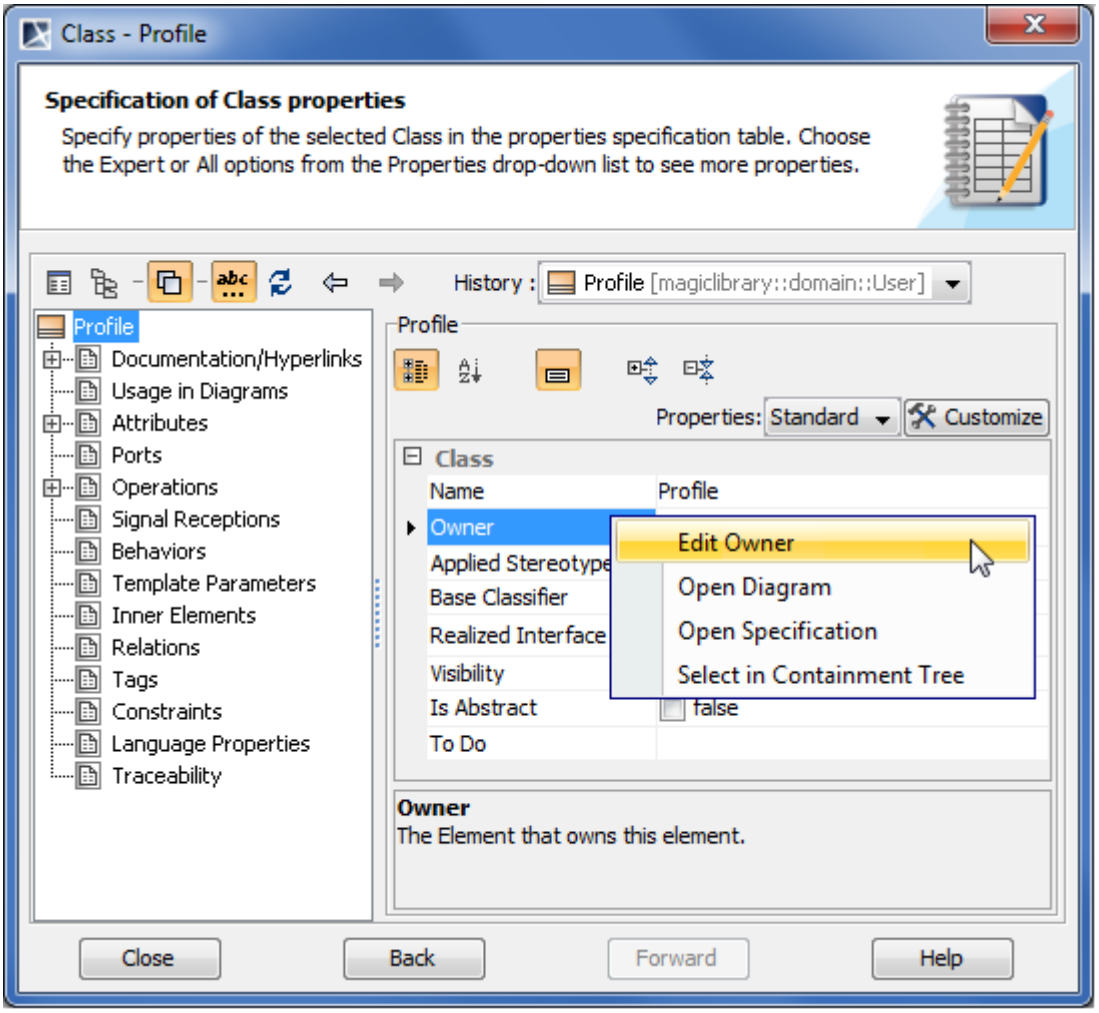


Figure 170 -- Selecting Edit Owner command on property's shortcut menu in Specification window

The way of entering a property value depends on the selected property type.

NOTE

Property values in gray are not editable. They are derived from other elements used in your model.

Learn about editing values of different property types in:

- "Editing textual properties" on page 299.
- "Editing logical values" on page 304.
- "Selecting values" on page 305.
- "Cases of advanced editing" on page 311.

Editing textual properties

Textual type property values are usually simply typed in a property value cell. In some specific cases you can also use advanced editing features, i.e., long textual values editorand multiline textual values editor (both available the HTML mode).

Examples of textual type properties: **Name**, **Body**, and other.

Copyright © 1998-2015 No Magic, Inc.

To edit a textual property value

1. Click an appropriate property value cell.
2. Type a value.

Learn about advanced cases of editing textual values in:

- "Editing long textual values" on page 300.
- "Editing multiline textual values" on page 301.

Editing long textual values

In case of editing a value that consists of several words, the value cell may seem too short. In order to avoid this inconvenience you can edit the property value in a dialog specially designated for typing long values.

To edit a textual value consisting of several words

1. Click an appropriate property value cell.
2. Click the Edit button (see the highlighted button in the following figure).
3. In the opened dialog, type a value and click **OK**.

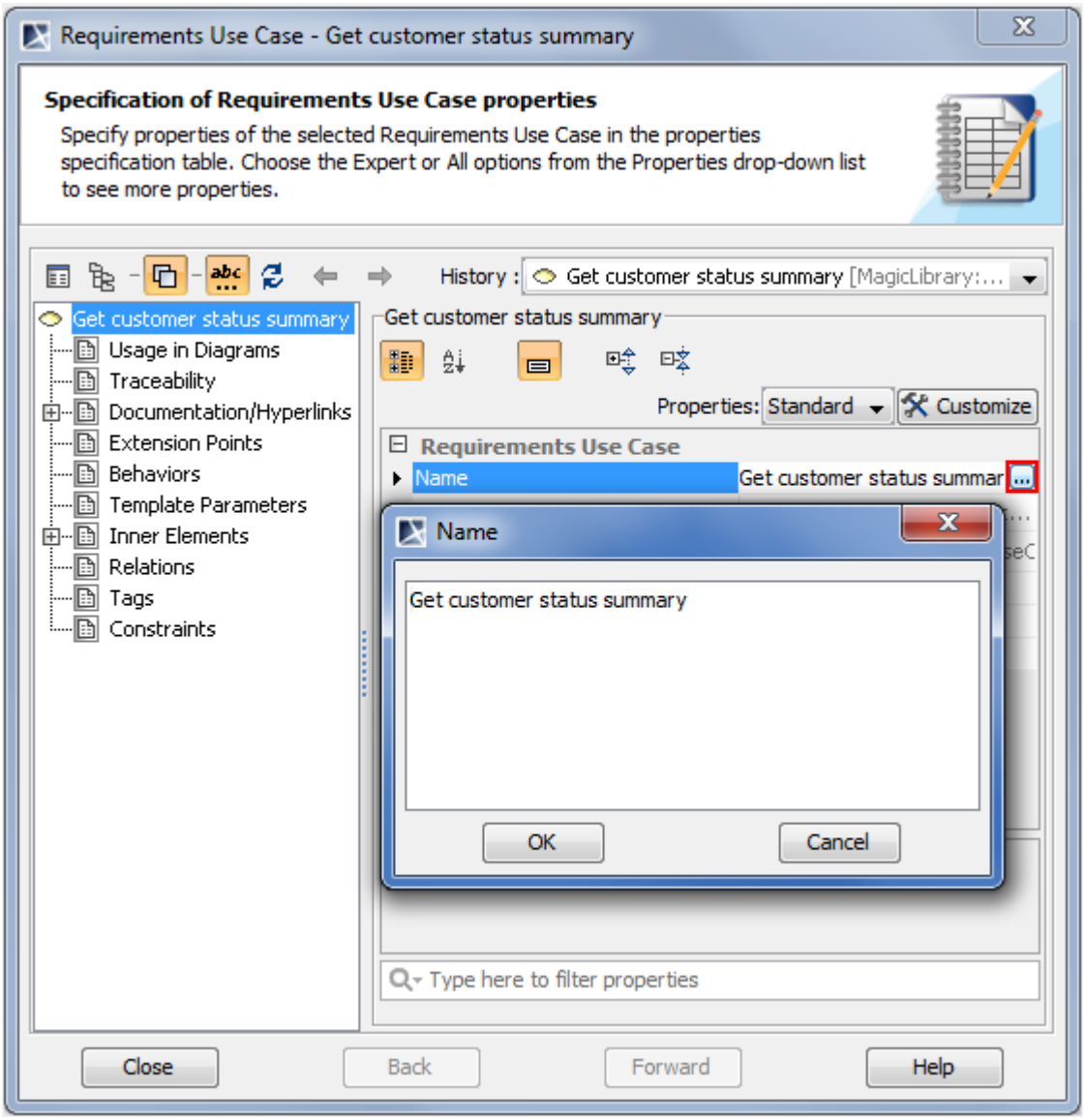


Figure 171 -- Editing value that consists of several words in Specification window

Copyright © 1998-2015 No Magic, Inc.

There are some textual properties, whose values can keep text formatting settings. You can use the HTML editor in such cases. To use the HTML editor, select the **HTML** check box in the dialog opened for editing this special kind of textual property value.

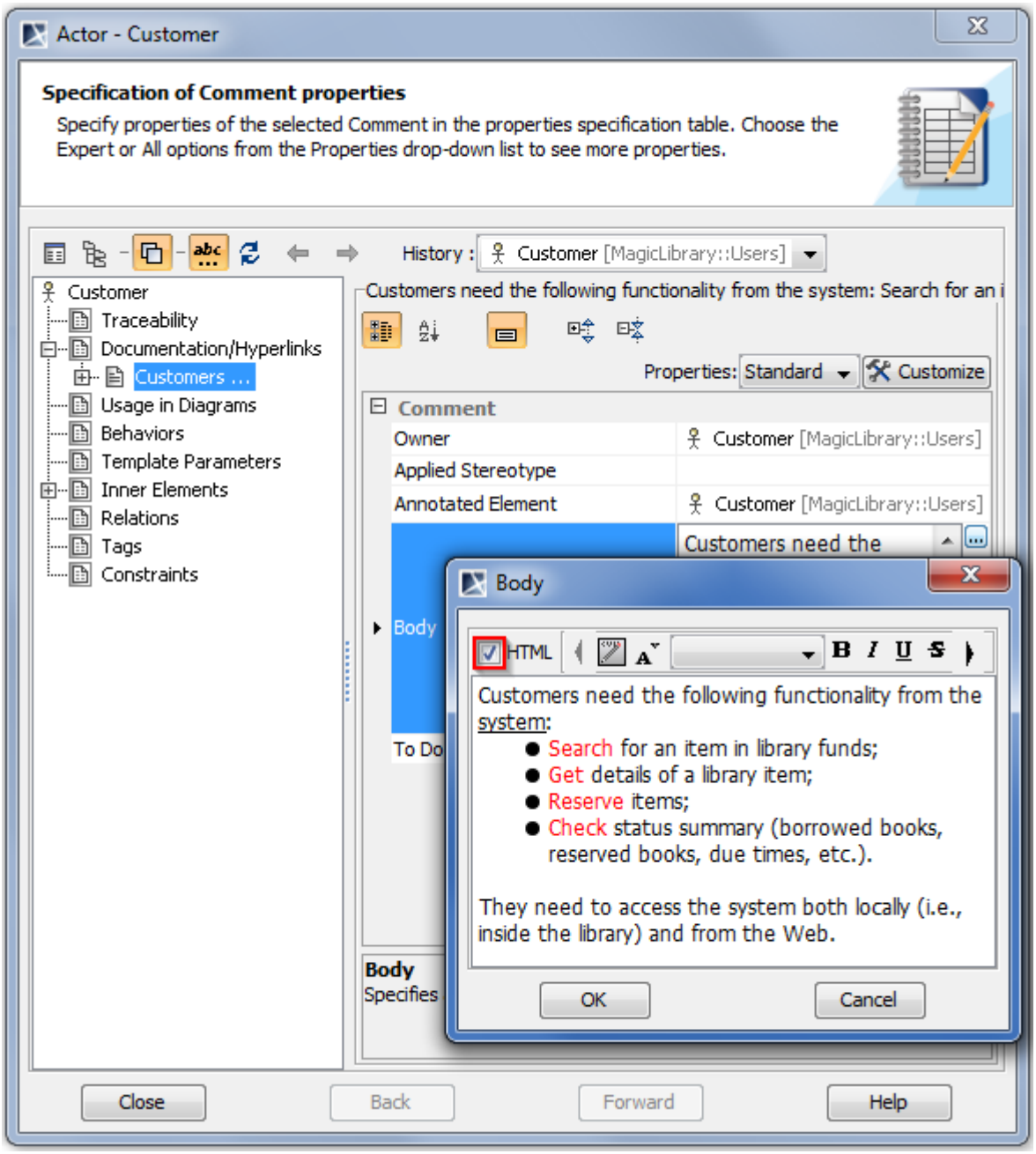


Figure 172 -- Editing textual value in HTML editor

TIP!

For more information about using the HTML editor, see "HTML Editor" on page 422.

Editing multiline textual values

Some properties can have more than one textual value, for example, the **Body** property of an opaque expression, the **Pre Condition** and **Post Condition** properties of a use case, and other. You can use a special editor to edit multiline textual values.

Copyright © 1998-2015 No Magic, Inc.

Using the editor for multiline textual values, you can perform the following actions:

- Create a new value.
- Remove a selected value.
- Edit each value in the HTML editor separately.
- Reorder values.
- Strip long values (available only in the Specification window).

IMPORTANT!

To strip long values, make sure that the Strip Multiline Text mode is turned on in the Specification window. For more information about buttons on the Specification window toolbar, see "Specification Window Toolbar" on page 277.

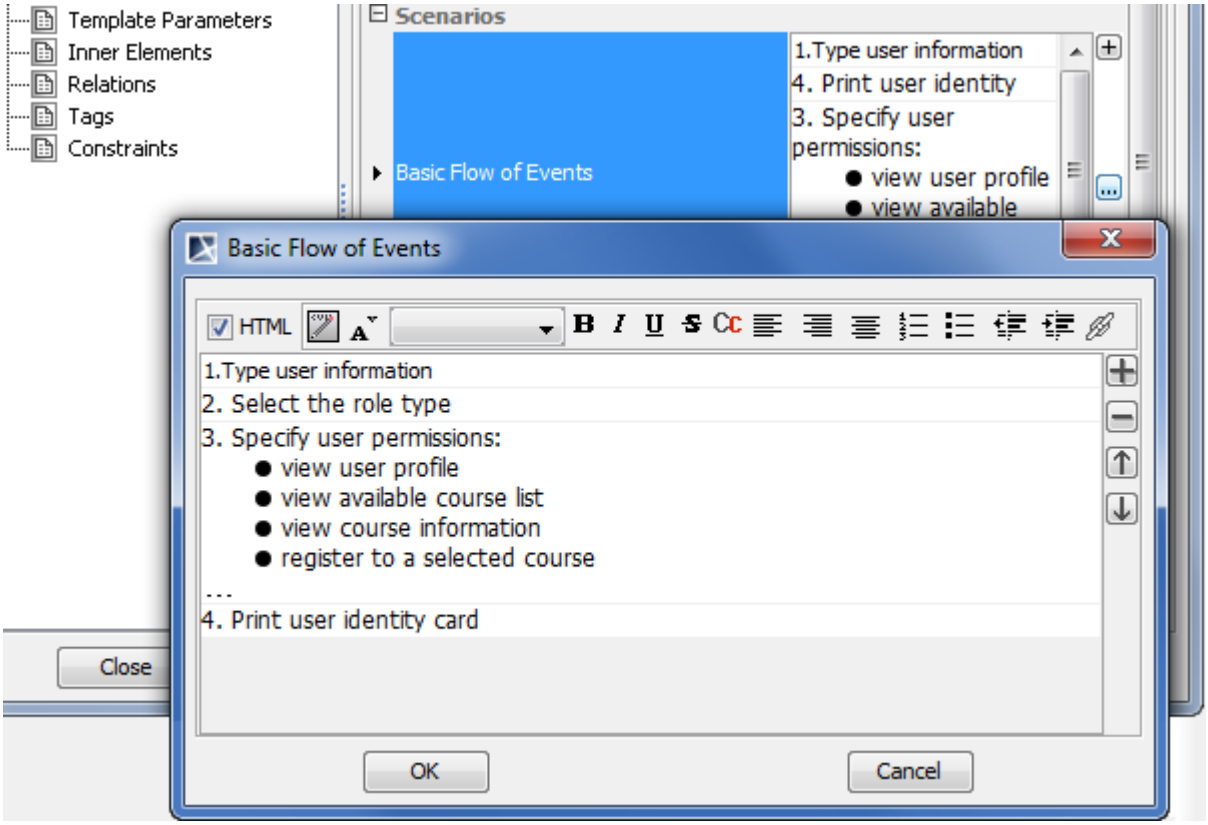


Figure 173 -- Multiline textual values editor

The following table describes functions of buttons used in the editor.

Button	Shortcut Key	Description
HTML		Select the HTML check box to edit a selected row in the HTML mode. The HTML editor toolbar will appear. For more information about the HTML toolbar, refer to "HTML editor toolbar" on page 427.
	ALT+ENTER	Click to add an empty row after the selected one to type a new value.
	ALT+BACKSPACE	Click to remove a selected row.
	ALT+SHIFT+UP ARROW	Click to moves up a selected row.

Copyright © 1998-2015 No Magic, Inc.

Button	Shortcut Key	Description
	ALT+SHIFT+DOWN ARROW	Click to moves down a selected row.



Under the Mac OS platform the MAC shortcut key should be used instead of the ALT key.



To select the previous row, press ALT+UP ARROW.

To select the subsequent row, press ALT+DOWN ARROW.

To open the multiline textual values editor

1. Click an appropriate property value cell.
2. Do one of the following:
   - Double-click one of the values of the selected property.
   - Click the Edit selected value button (see the highlighted **...** button in the following figure).
   - Click the Add new value button (see the **+** button in the following figure). The editor will be opened with an empty line to type a new value.

Copyright © 1998-2015 No Magic, Inc.

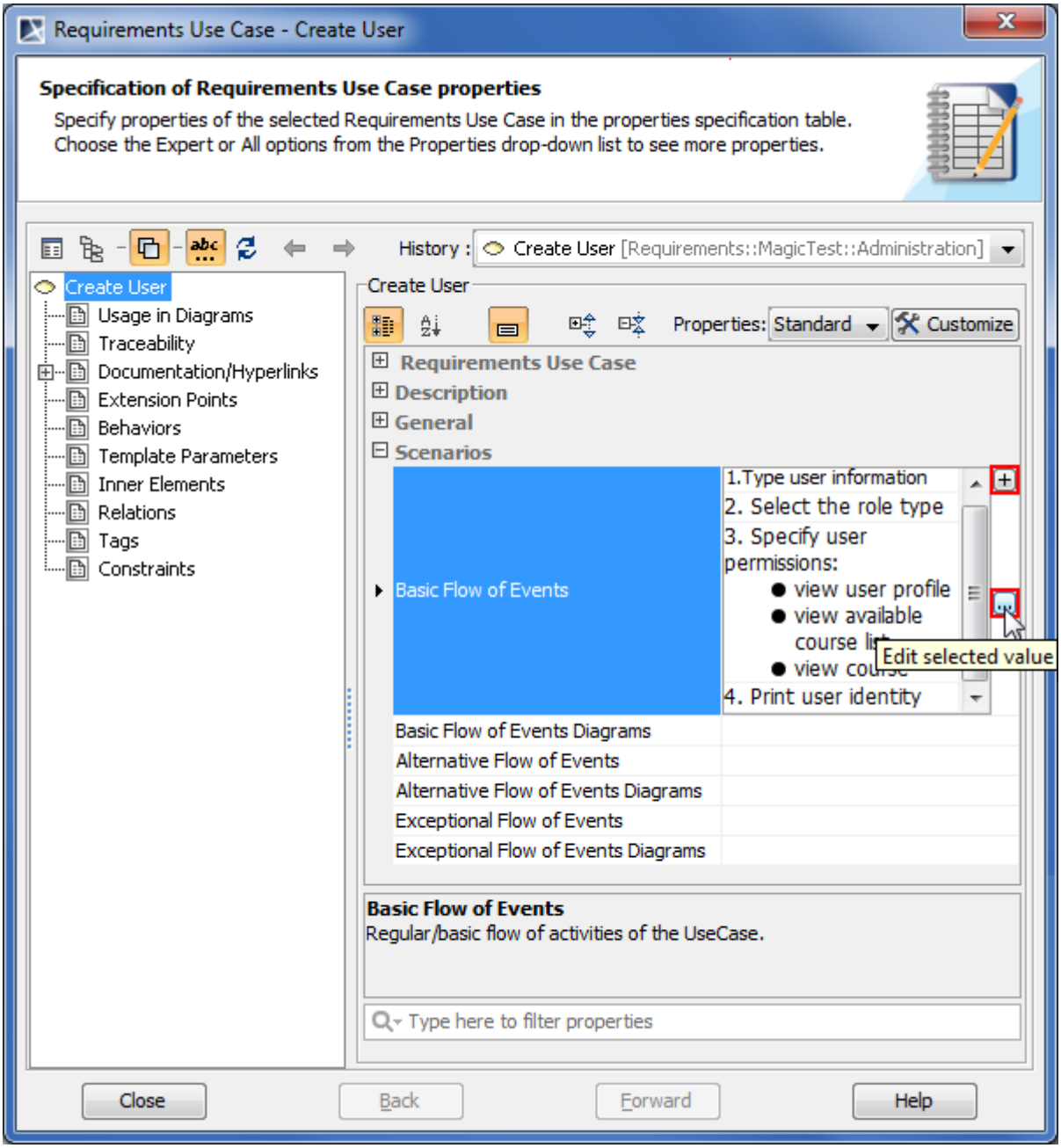


Figure 174 -- Opening multiline textual values editor in Specification window

Editing logical values

Logical (boolean) type property value can be either *true* or *false*.

Examples of logical type properties: **Is Abstract**, **Is Read Only**, and other.

To edit a logical property value

Do one of the following:

- Select the check box to set the value to *true*.
- Clear the check box selection to set value to *false*.

Copyright © 1998-2015 No Magic, Inc.

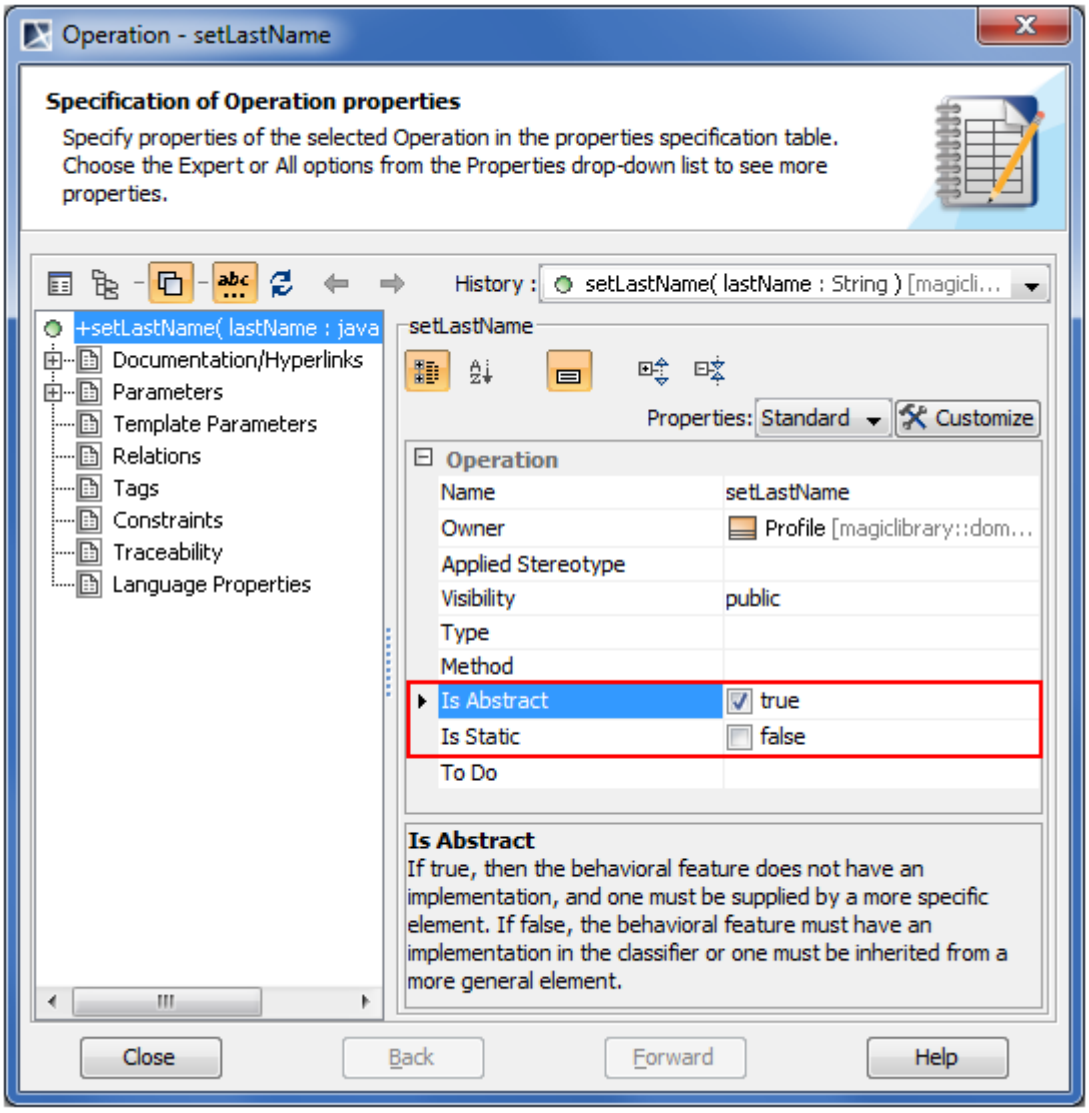


Figure 175 -- Editing logical values in Specification window

Selecting values

Selectable value type property values are selected from lists. There can be two types of lists:

- **Non-editable** - for properties whose value ranges are restricted by UML (for example, Visibility, Message Sort, Event Type, and other).
- **Editable** - for properties whose value ranges are not restricted by UML and can be defined by the user (for example, Type Modifier, Multiplicity, and other).

Learn about selecting property values from lists in:

- "Selecting single property value from non-editable list" on page 305.
- "Selecting single property value from editable list" on page 306.
- "Selecting multiple property values" on page 308.

Selecting single property value from non-editable list

In this case you can choose one of predefined values.

Copyright © 1998-2015 No Magic, Inc.

To select a property value from a non-editable predefined list

1. Click an appropriate property value cell to open the list.
2. In the drop-down list, select the value.

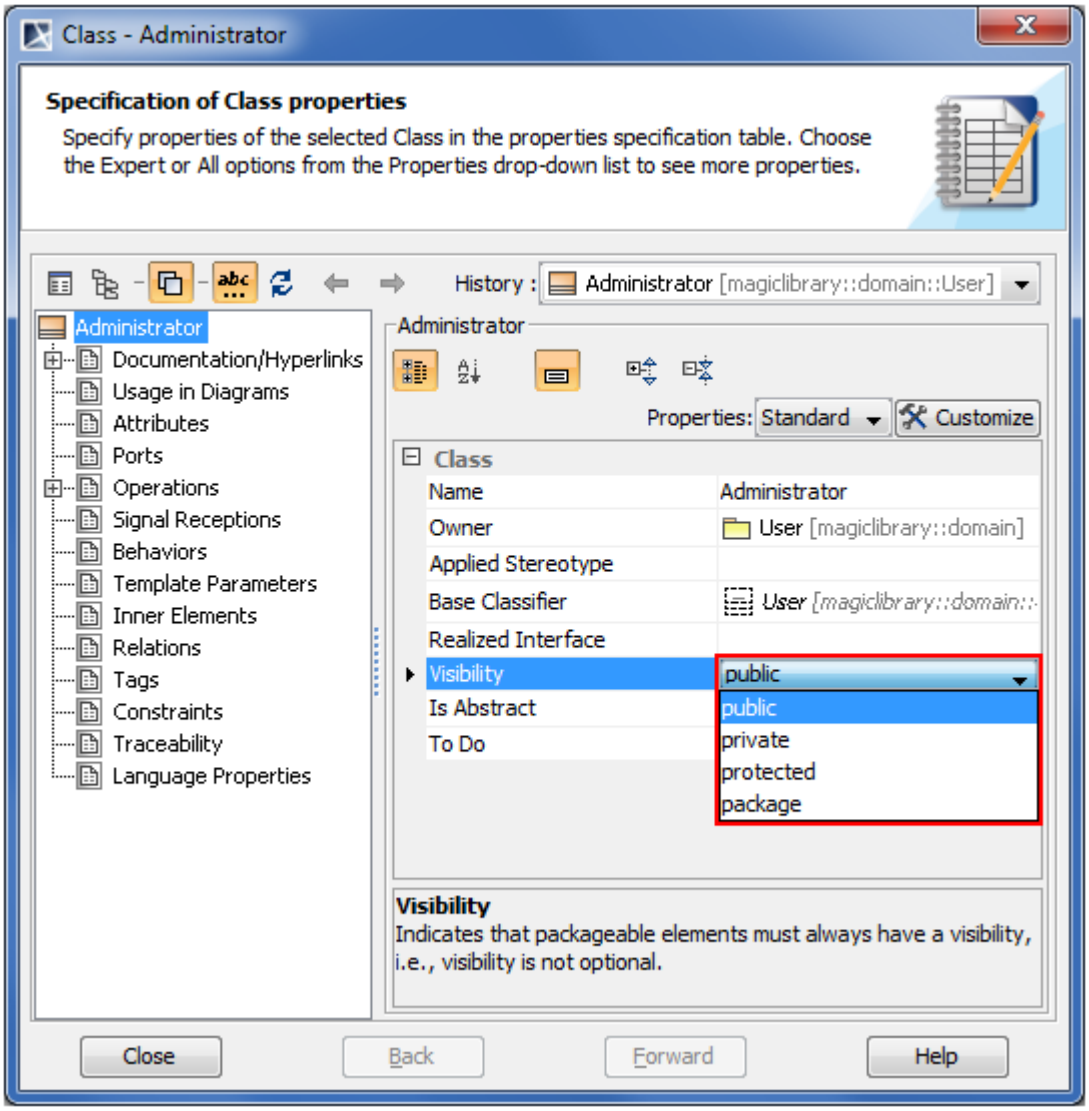


Figure 176 -- Selecting value from non-editable list in Specification window

To remove a property value selected from a non-editable predefined list



You can remove the property value only if the property can have an empty value.

1. Click an appropriate property value cell.
2. Click the drop-down arrow to open the list of available values.
3. Select *<UNSPECIFIED>* from the list.

Selecting single property value from editable list

This is the case of assigning a value to a property in one of the following ways:

- Selecting a value either from a predefined value (autocompletion) list or from the whole model via the element Selection dialog.
- Creating a new element and selecting it as property value.

Copyright © 1998-2015 No Magic, Inc.

To select a property value from a predefined (autocompletion) list

1. Click an appropriate property value cell.
2. Click the drop-down arrow (see the highlighted button in the following figure) to open the list of available values.
3. Select a value from the list.

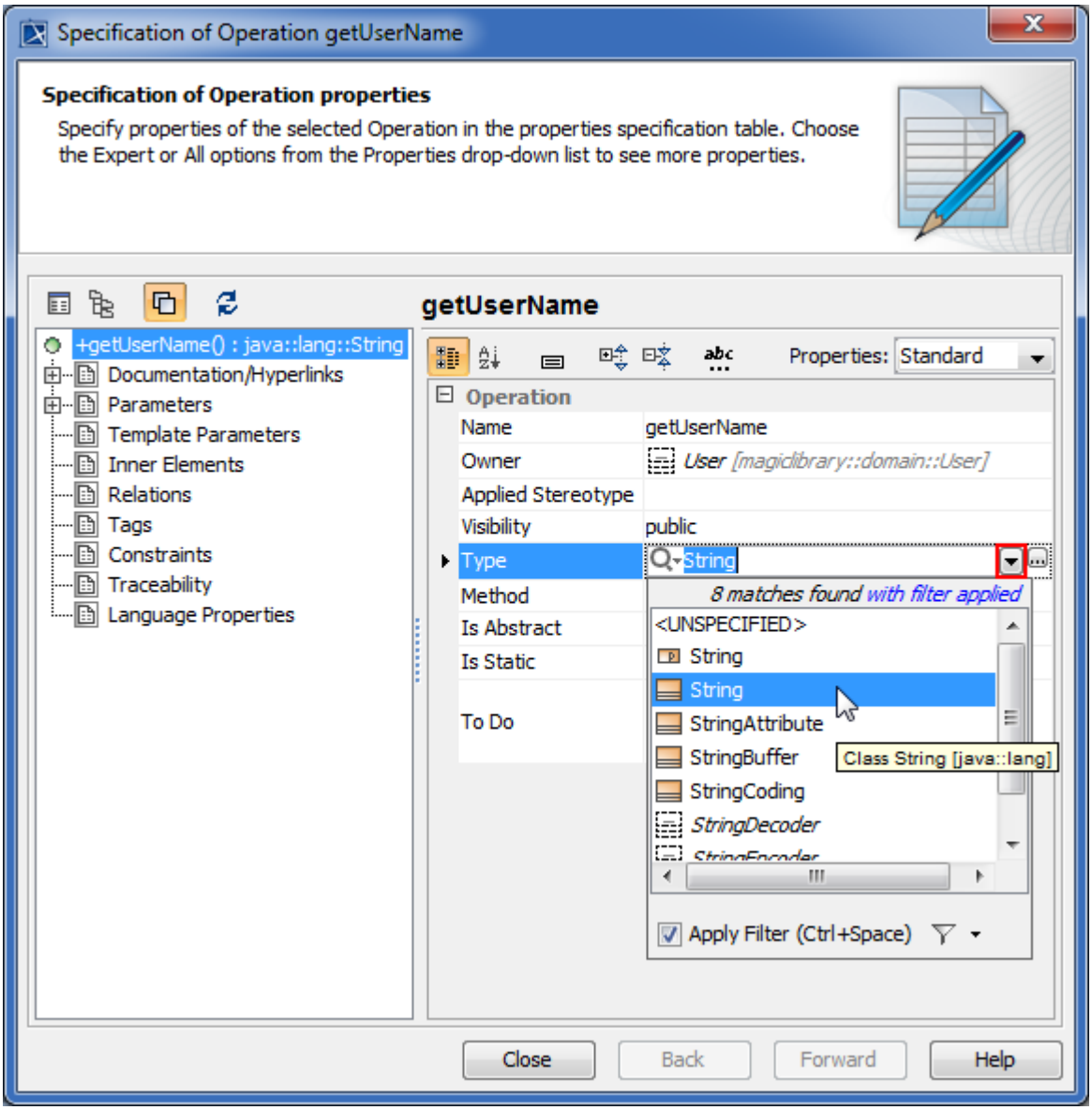


Figure 177 -- Selecting single property value from predefined (autocompletion) list in Specification window

TIP!

For information about the type and mode of searching for an element, see "To create a new element in the element Selection dialog" on page 356.

To assign a property value via the element Selection dialog

1. Click an appropriate property value cell.
2. Click the Edit button (see the highlighted button in the following figure). The element Selection dialog opens.
3. Select an existing element or create a new one.

TIP!

For more information about the element Selection dialog, see "Selecting an Element" on page 352.

Copyright © 1998-2015 No Magic, Inc.

4. Click **OK** when you are done.

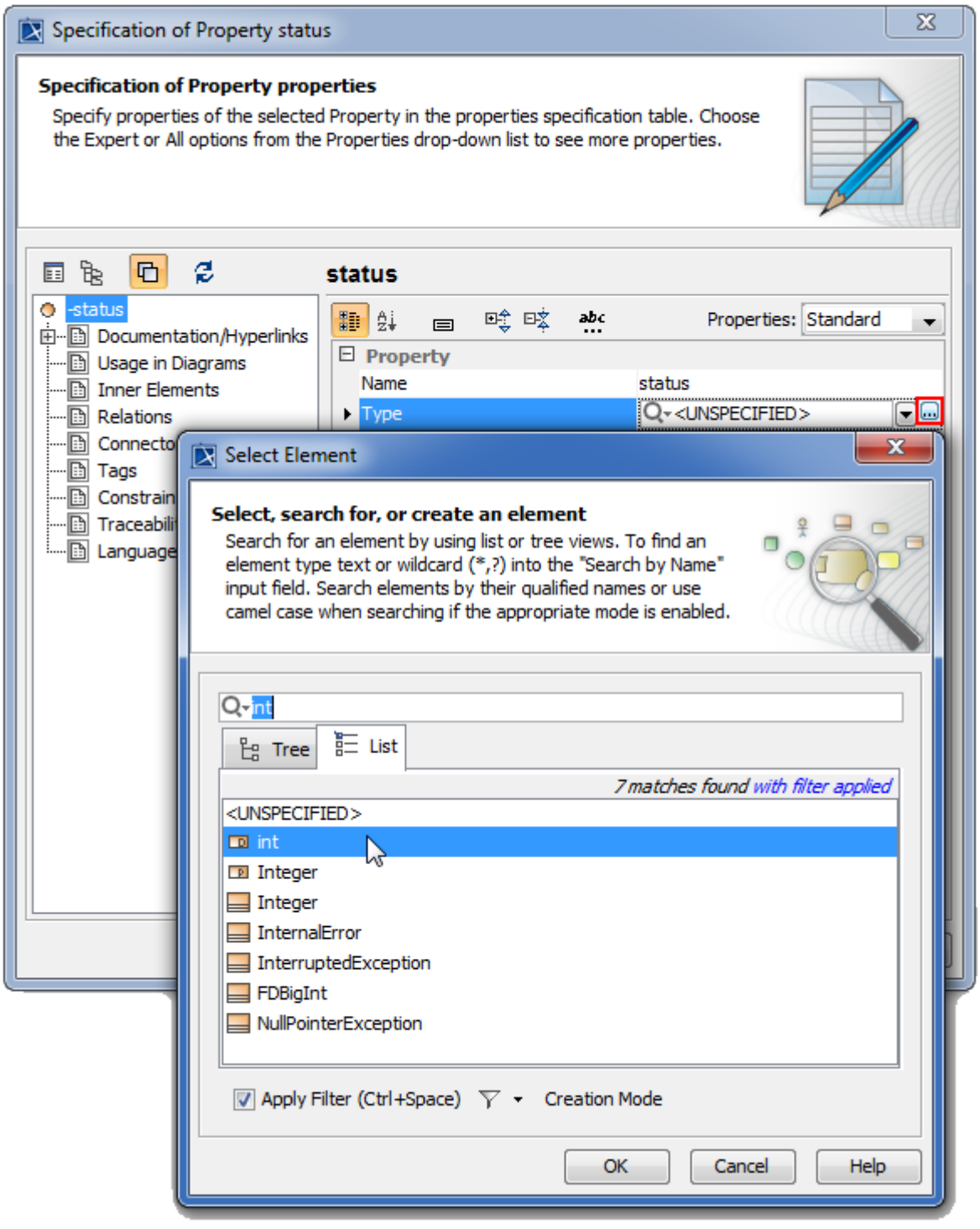


Figure 178 -- Selecting single property value via element Selection dialog

To remove a value selected for a single value type property

1. Click an appropriate property value cell.
2. Click the drop-down arrow to open the list of available values.
3. Select <*UNSPECIFIED*> from the list.

Selecting multiple property values

In this case you can assign more than one value to a property via the element Selection dialog.

Copyright © 1998-2015 No Magic, Inc.

Examples of properties that may have multiple values: **Classifier**, **Method**, and other.

To assign multiple property values

1. Click an appropriate property value cell.
2. Click the Edit button (see the highlighted button in the following figure).The element Selection dialog opens.
3. Select existing elements and / or create new ones.



For more information about the element Selection dialog, see "Selecting an Element" on page 352.



Make sure the Multiple Selection mode is turned on.

Copyright © 1998-2015 No Magic, Inc.

4. Click **OK** when you are done.

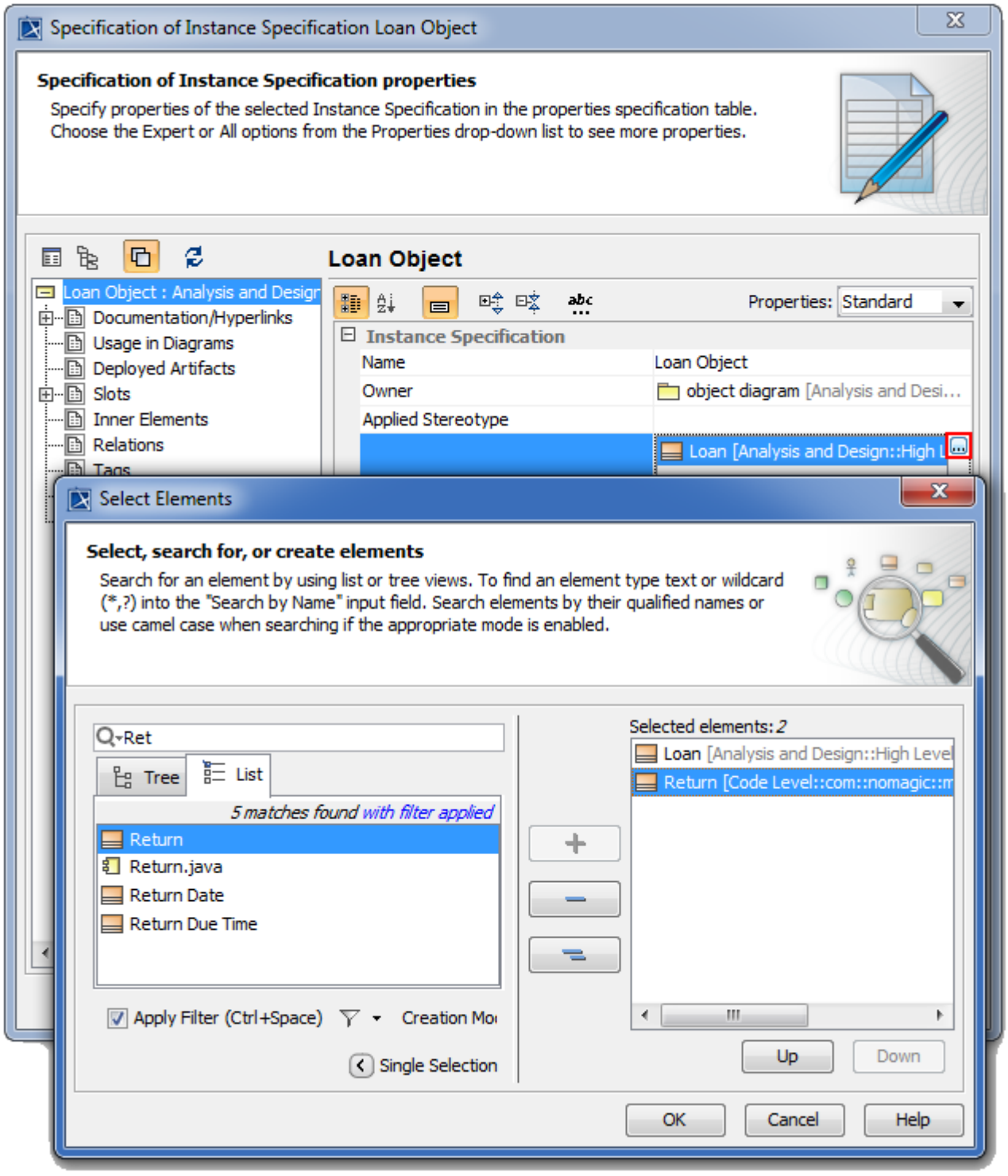


Figure 179 -- Selecting multiple property values in the element Selection dialog

Copyright © 1998-2015 No Magic, Inc.

Cases of advanced editing

You may have noticed that editing some property values is a sophisticated process. These are the cases of editing such property values:

- "Creating inner element as property value" on page 311.
- "Assigning value specification as property value" on page 314.
- "Editing property values in property groups" on page 314.

Creating inner element as property value

There are some properties whose values can be their inner elements. Specifying such property value requires to create a new element. The element, which has the property with the inner element assigned as its value, becomes the owner of this inner element.

Examples of properties whose values are their inner elements: **Owned Comment**, **Owned Attribute**, and other.

To create an inner element as a property value

1. Click an appropriate property value cell.
2. Click the Add button (see the highlighted button in the following figure). The Specification window for creating a new element opens.

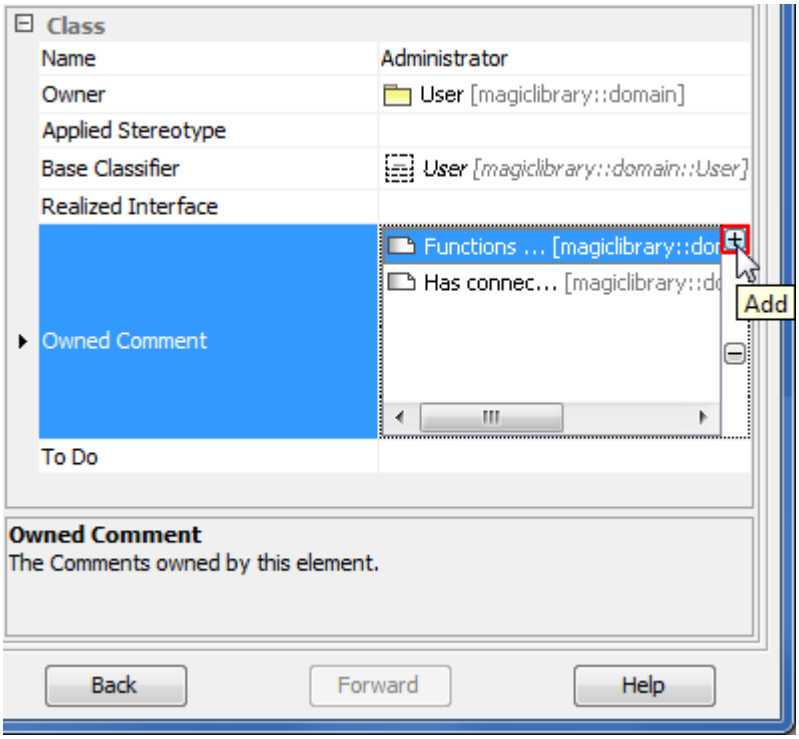


Figure 180 -- Creating inner element (fragment of Specification window)

3. Specify properties of the new inner element's Specification window.
4. When you are done, do one of the following:
   - Close the Specification window.
   - Click **Back** to return to the Specification window of the owning element.

To remove an inner element from a property value list

1. Click an appropriate property value cell.

Copyright © 1998-2015 No Magic, Inc.

2. Select the value you want to remove.
3. Click the Remove button.

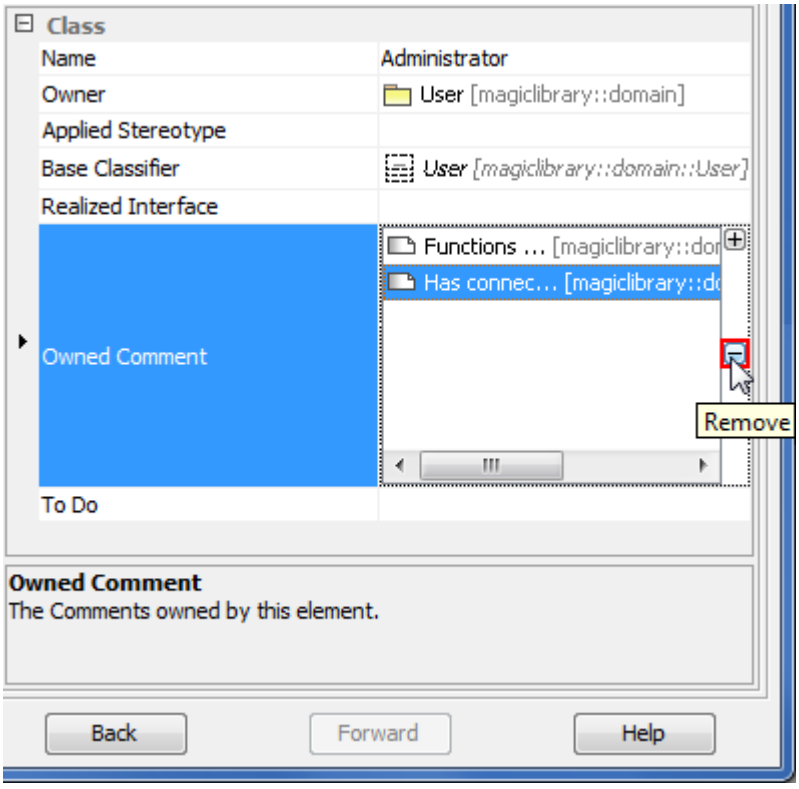


Figure 181 -- Removing inner element (fragment of Specification window)

IMPORTANT!

The element is removed from both the property value list and the model.

Assigning value specification as property value

There are some properties whose values can be value specifications.

TIP!

For more information about value specifications, see "Value Specification" on page 1027.

Examples of properties whose values are value specifications: **Default Value**, **Value**, and other.

To assign a value specification

1. Click an appropriate property value cell.
2. Click the Show Shortcut Menu button (see the highlighted button in the following figure).
3. On the shortcut menu, click **Value Specification** and then select a value specification.

Copyright © 1998-2015 No Magic, Inc.

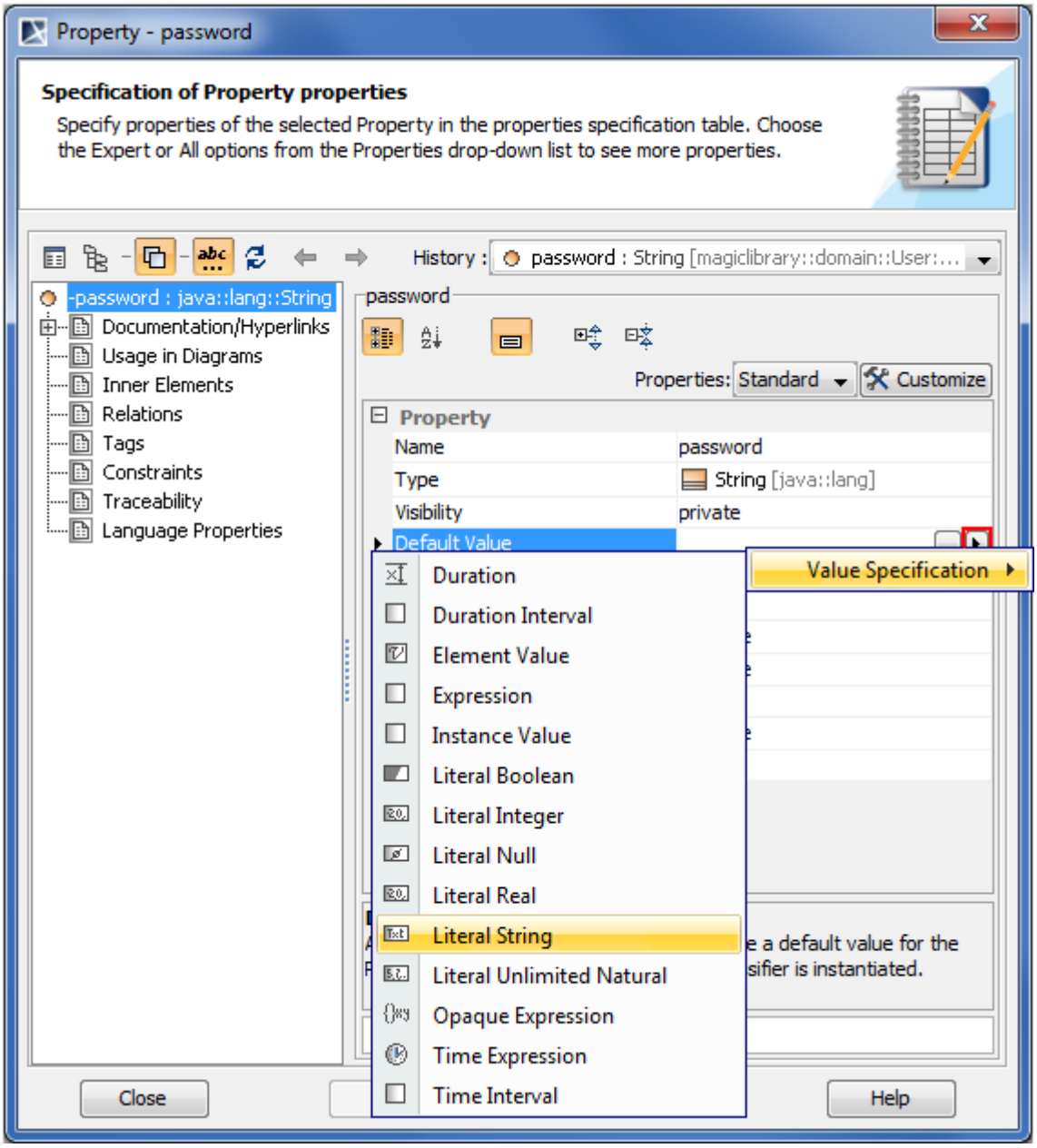


Figure 182 -- Assigning value specification in Specification window

To change an assigned value specification

1. Click an appropriate property value cell.
2. Click the Show Shortcut Menu button (see the highlighted button in the preceding figure).
3. From the shortcut menu, select **Value Specification** > **Delete** <value specification> (see the following figure).
4. Assign a new value specification. See the procedure "To assign a value specification" on page 312.

Copyright © 1998-2015 No Magic, Inc.

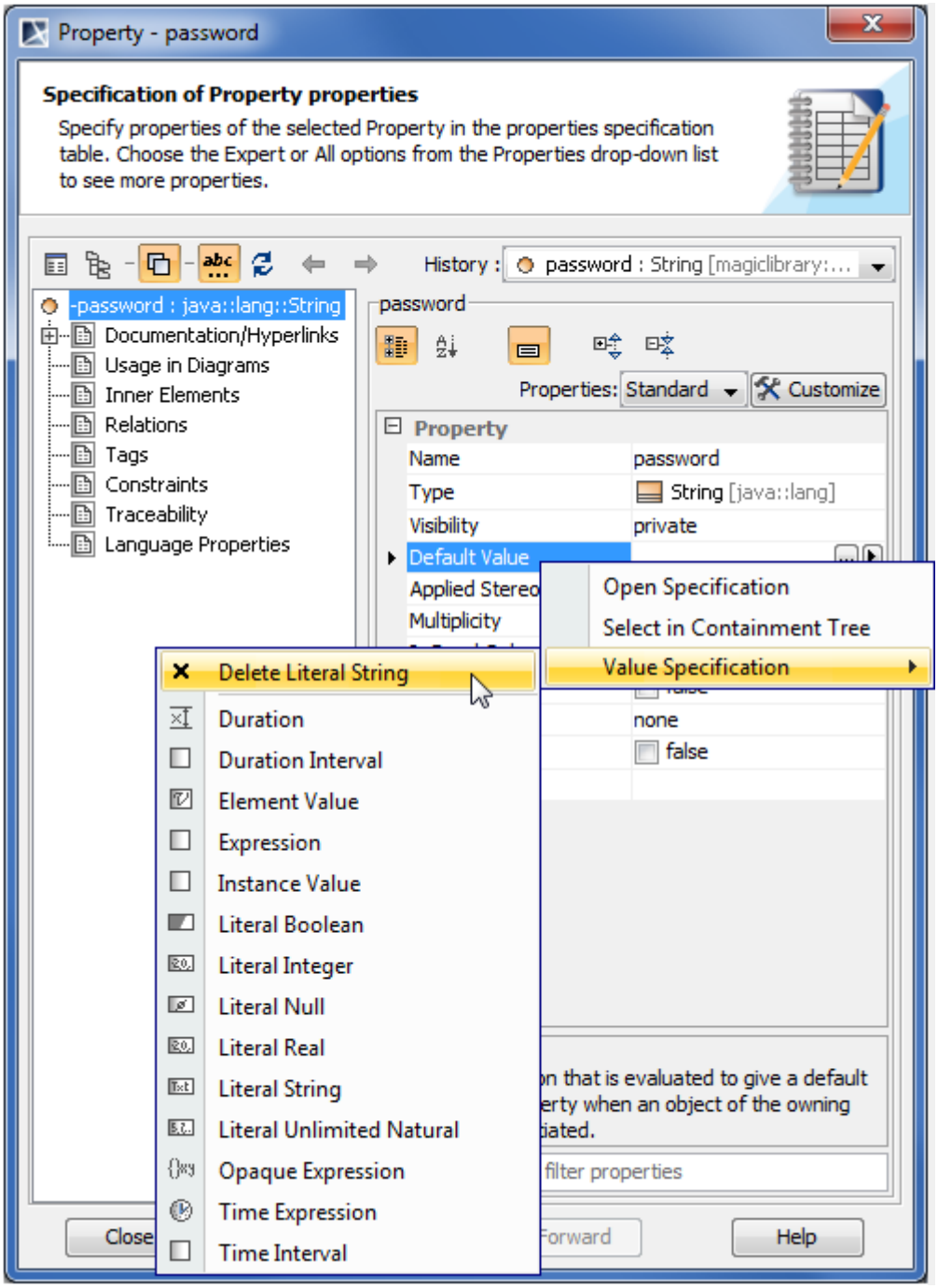


Figure 183 -- Deleting value specification in Specification window

Editing property values in property groups

Some elements can be related to each other as the owner and the owned one, for example, a transition can be an owner of a trigger, and the trigger can be owned by the transition.

MagicDraw allows the handy editing of owned element property values via owner's properties. Owned element properties are available as appropriately named property groups, for example, the **Entry**, **Do Activity**, and **Exit** property groups in the state's Specification window (see the following figure).

IMPORTANT! When editing owned element properties, keep in mind that you create a new element in your model.

Copyright © 1998-2015 No Magic, Inc.

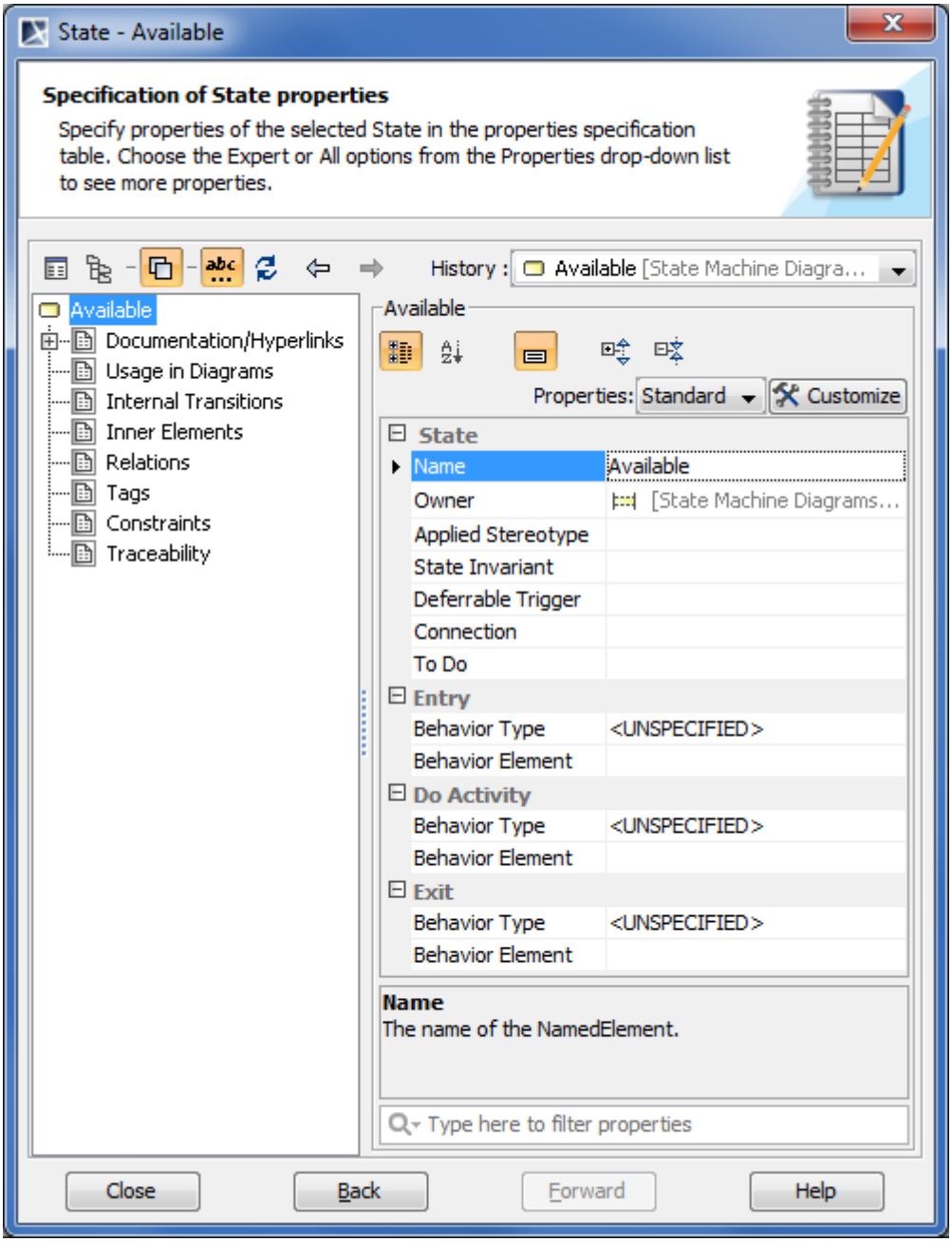


Figure 184 -- Owned element property groups in state's Specification window

Property set in a property group may vary before and after owned element specification. For example, let's say, we have an activity assigned as behavior type in the **Entry** property group. As a result to this certain additional properties for creating a new activity in your model appear in the **Entry** property group: **Name** and **Owned Diagram** (see the following figure).

NOTE

Keep in mind that the owner's specification contains only ownership-relevant properties of an owned element.

To view and edit all other properties of the owned element, open its Specification window.

Copyright © 1998-2015 No Magic, Inc.

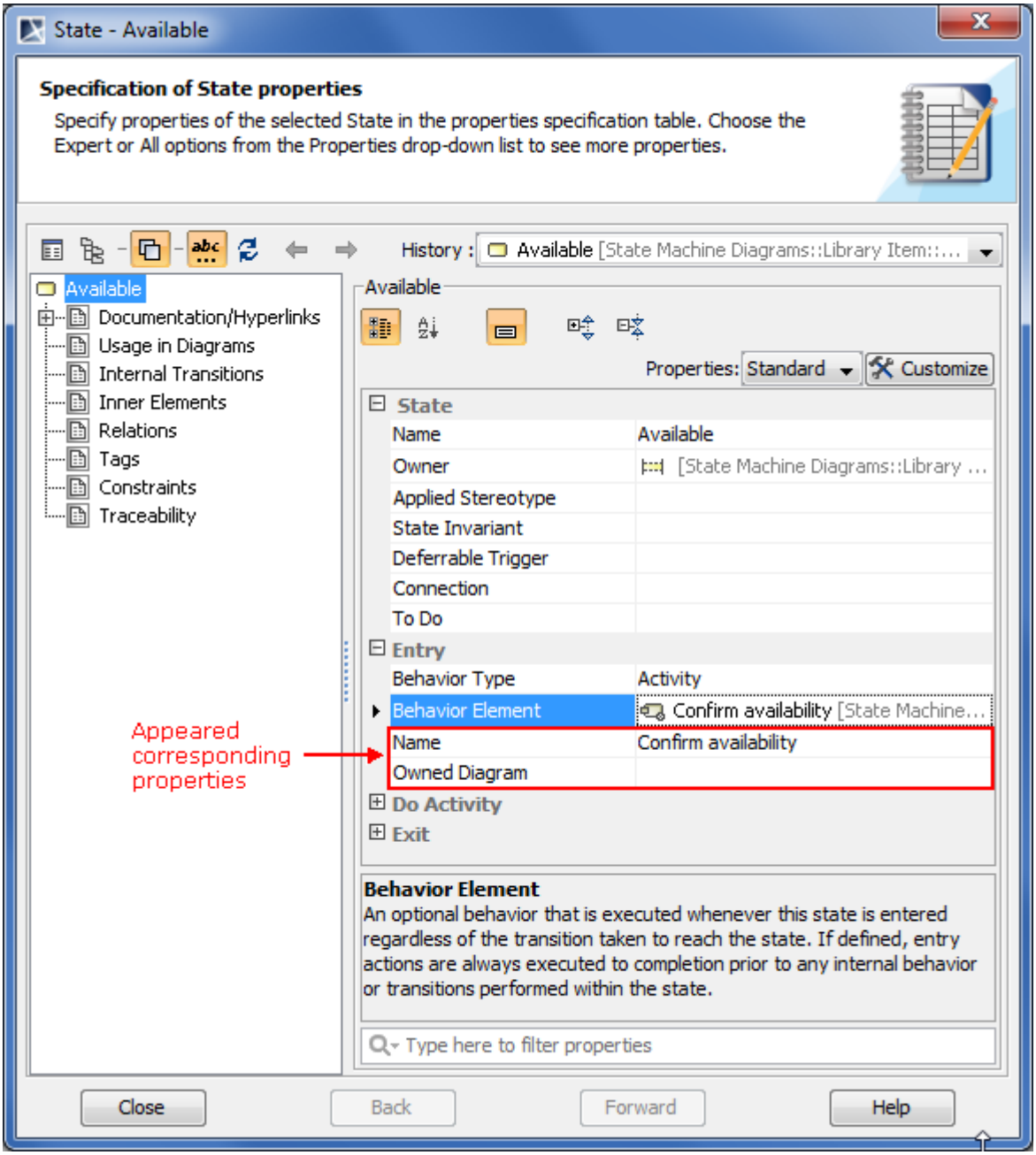


Figure 185 -- Specifying activity as entry in state's Specification window

To create an owned element as a property value

1. In the property group, wherein you want to create the owned element, click any editable property value cell.
2. Specify additional properties corresponding to the created element.

To remove an owned element

1. Click the **<owned element type>** property value cell in the desired to remove owned element property group.
2. In the opened drop-down list, select <*UNSPECIFIED*>.

Copyright © 1998-2015 No Magic, Inc.

Formatting Symbols

Definition	Description
Symbol	The term "symbol" means a visual representation of some model elements in the diagram. Symbols are subdivided into shapes and paths (lines in the model, for displaying various relationships).
Symbol properties	Every symbol may have its own style: color, font, size, and so forth. Symbol properties may be defined for the concrete symbol, for all symbol of one element, or according to the diagram type. For more information about symbol properties definition, see "Formatting Symbols" on page 317, about style engine, see "Style Engine" on page 324.

Every symbol in MagicDraw can have its own style: color, font, size, and so forth. You may define the symbol properties in the following ways:

- In the **Symbol Properties** dialog. See the "Symbol Properties dialog" on page 317..
- In the **Symbol** tab on the **Properties** panel at the bottom of the Model Browser.
- In the **Project Options** dialog. Using this dialog you can change all available symbol properties, create your own style for the project, apply different symbol properties for different diagrams, define stereotype properties that may be bound to the symbol, and more.



For more information about creating, editing, cloning, importing/exporting, or removing symbol styles, see "Style Engine" on page 324.

- On the diagram toolbar. Using this toolbar you can change the color, font, path style of a symbol, and paste the symbol properties.



For more information about diagram toolbar, see "Diagram toolbars" on page 73.

Symbol Properties dialog

In the **Symbol Properties** dialog, you may define various symbol (or group of symbols) properties, like fill color, font type, whether to wrap text and other appearance parameters. The **Symbol Properties** dialog is a non-

Copyright © 1998-2015 No Magic, Inc.

modal window - you can edit the symbol properties and see the changes on the diagram instantly. In addition, while the dialog is active you can still work with diagrams or other dialogs.

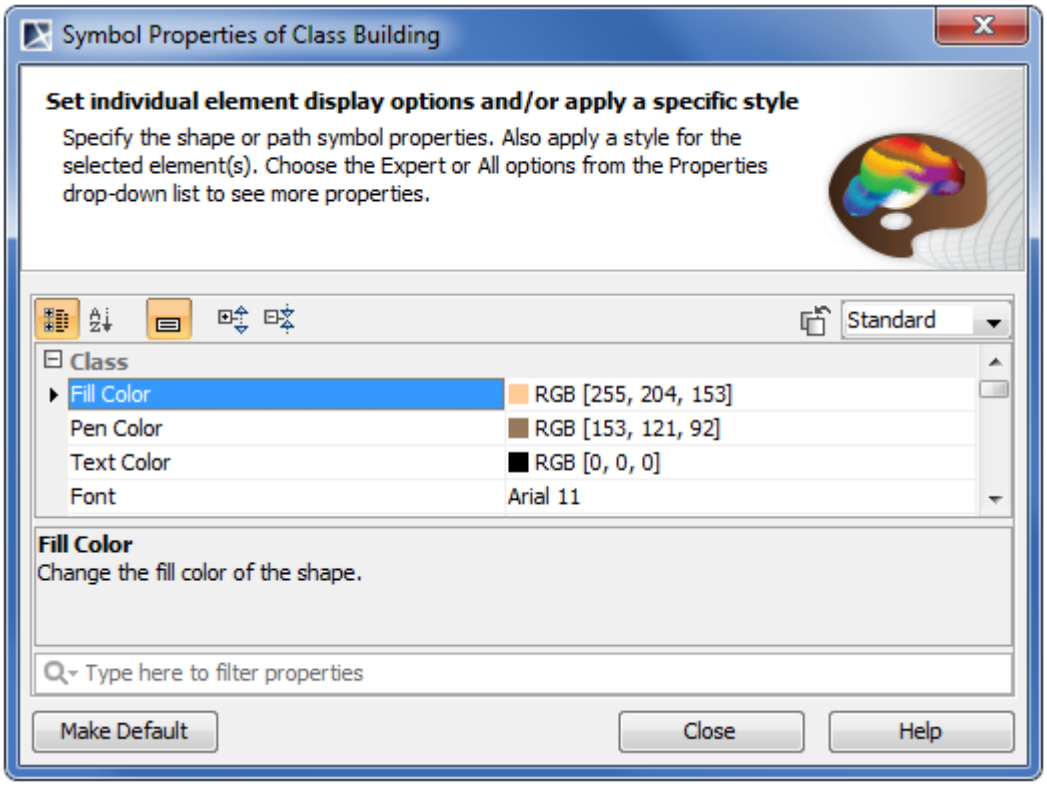


Figure 186 -- Symbol Properties dialog

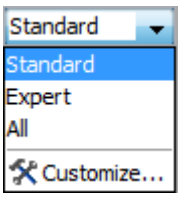
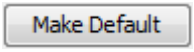
To open the **Symbol Properties** dialog

1. On the diagram, select a symbol or a group of symbols.
2. Do one of the following:
   - On the main menu, click **Edit** > **Symbol** > **Symbol Properties**.
   - Right-click and on the shortcut menu, click **Symbol Properties**.
   - Press Alt+Enter.

In the following table, you will find the described common functions that **Symbol Properties** dialog provides.

Icon	Title	Type	Description
	Apply Style	Button	Click to open the menu with the available to apply styles. For more information, see "Style Engine" on page 324.
	Show All Properties	Button	Click to display all the properties for the selected symbol types. Note that the *Show All Properties* button is displayed only when at least two shapes of different element types are selected, for example, a class and a package shapes.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Type	Description
		Drop-Down list	Click to change the properties display mode. The properties display modes has three available modes: *Standard*, *Expert* and *All*. The *Expert* mode offers more options for specifying properties and the *All* mode lists all the properties. Click the **Customize** command to customize the properties lists for the particular mode. The **Customize Properties** dialog appears. For more information, see "Customizing Lists of Properties and Commands" on page 336.
	**Make Default**	Button	Click to change the default symbol properties to the properties that are specified in this dialog. We highly recommend to use the **Make Default** button only if you fully understand the symbol styles and how your symbol styles can be affected. For more information, see "Collecting symbol styles from modules" on page 331 and "To copy a symbol style from a module to the project by using the Symbol Properties dialog" on page 335.

Displaying icon or image

You may display icon or image on the symbol or instead the symbol.

Copyright © 1998-2015 No Magic, Inc.

Get acquainted with the following definitions in order to understand this functionality better.

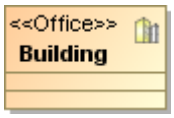
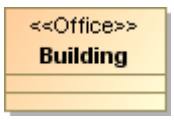
Definition	Description
Stereotype	Every element can be extended by applying a stereotype. For more information about stereotype, see "Stereotype" on page 985. For more information about applying stereotype properties, see "Stereotype properties" on page 330.
Icon	Icon of stereotype. Icon is a small image displayed in the top-right corner of shape. To assign icon to stereotype in the **Stereotype** Specification window, define the **Icon** property. For more information on how to assign Icon for stereotype, see "To create a stereotype with an image" on page 986.
Text	Stereotype name, displayed on the symbol.
Image	Image which can be assigned to element and displayed as icon or instead of element shape. To assign image to element in the element Specification window, assign the **Image** property. For more information on how to assign icon for element, see "Image" on page 282.

To change the icon visibility mode on the element shape

Do one of the following:

- On the diagram, right-click the element shape and then from the shortcut menu, select the **Symbol Properties** command. Then in the **Symbol Properties** dialog, click the cell of the **Show Stereotypes** property, and select the icon visibility mode.
- On the diagram, click the element shape, and then, click the *Compartments* button. From the menu, select **Stereotypes** and then select the icon visibility mode.

Select one of the six property modes for **Show Stereotypes**. The property modes are described in the table below:

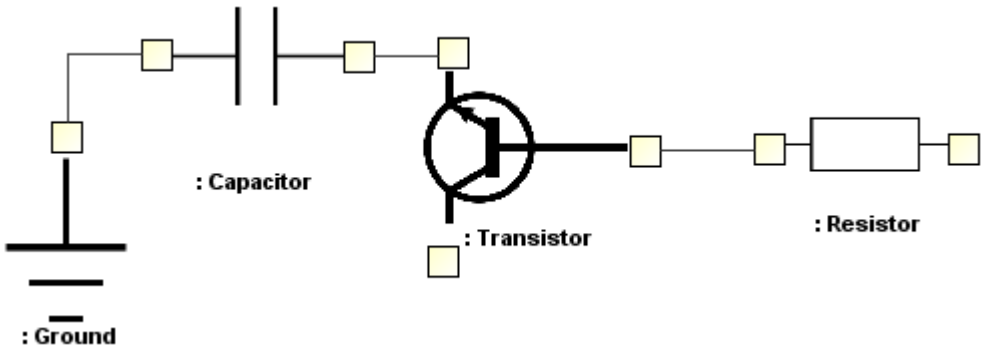
Show Stereotypes Property Mode	Displayed on the diagram pane	Icon of the stereotype and stereotype name are assigned to element	Image to element is assigned
Icon and Text		Icon of stereotype displayed. Name of stereotype displayed.	Image of element is displayed in the corner of shape.
Icon		Icon of stereotype displayed in the corner of symbol. Name of stereotype is not displayed.	Image of element is displayed in the corner of shape.
Text Only		Icon of stereotype is not displayed. Name of stereotype is displayed.	Image of element is not displayed.

Copyright © 1998-2015 No Magic, Inc.

Show Stereotypes Property Mode	Displayed on the diagram pane	Icon of the stereotype and stereotype name are assigned to element	Image to element is assigned
Shape Image and Text*		Icon of stereotype is displayed instead of shape. Name of stereotype is displayed.	Image of element is displayed instead of shape.
Shape Image*		Icon of stereotype is displayed instead of shape. Name of stereotype is not displayed.	Image of element is displayed instead of the shape.
Do Not Display		Icon of stereotype is not displayed. Name of stereotype is not displayed.	Image of element is not displayed.

IMPORTANT!

* - When the **Shape Image and Text** or **Shape Image** property is selected, to display the icon of a stereotype instead of the element symbol, all symbol compartments should be suppressed. If the symbol contains port or pins, these ports or pins are displayed around that symbol (see the following figure).



NOTE

Shape Image and Text and **Shape Image** properties are not added to the Path element properties list.

If element has assigned both - image and stereotype icon - then image of element will be displayed on the shape.

Copyright © 1998-2015 No Magic, Inc.

Displaying image of the type

You can display the image of the type for the following elements:

- Call Behavior Action
- Swimlane
- Object Node
- Activity Parameter Node
- Collaboration Use
- Instance Specification
- Part
- Lifeline

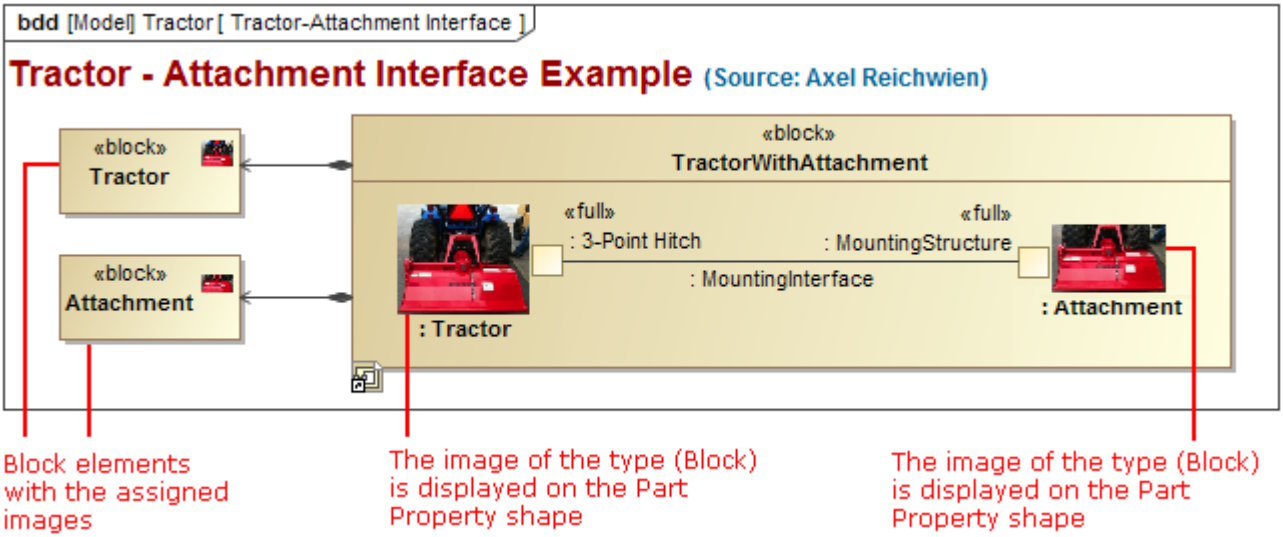


Figure 187 -- Example of the image of the type displayed on the Part

To display the image of the type on the shape

Do one of the following:

1. On the diagram, right-click the element shape and then from the shortcut menu, select **Symbol Properties**.
2. In the **Symbol Properties** dialog, at the right-top corner, click to expand the properties list and select *All*.
3. Click to select the **Show Secondary Image and Stereotypes** property and select *Image and Stereotypes*.

NOTE

- In the projects, imported from 17.0.5 or earlier versions, the default **Show Secondary Image and Stereotypes** property value is **Stereotypes**, therefore on the shape the image of the type is not displayed.
- Note that the **Show Secondary Image and Stereotypes** property value depends on the **Show stereotypes** property value - if the **Show stereotypes** property value is *Do Not Display*, the image of the type is not displayed on the shape too.

NEW! Displaying rake icon on symbol

You can use rake icon for faster navigation in the model. The rake icon on the symbol indicates that the appropriate element is decomposed and has an inner structure. Double-click the element to open the internal diagram. The rake icon isn't shown by default.

Copyright © 1998-2015 No Magic, Inc.

To show rake icon

1. Right-click the symbol.
2. Select **Symbol Properties** from shortcut menu or press Alt+Enter.
3. Set the **Show Rake Icon** property value to *true*.

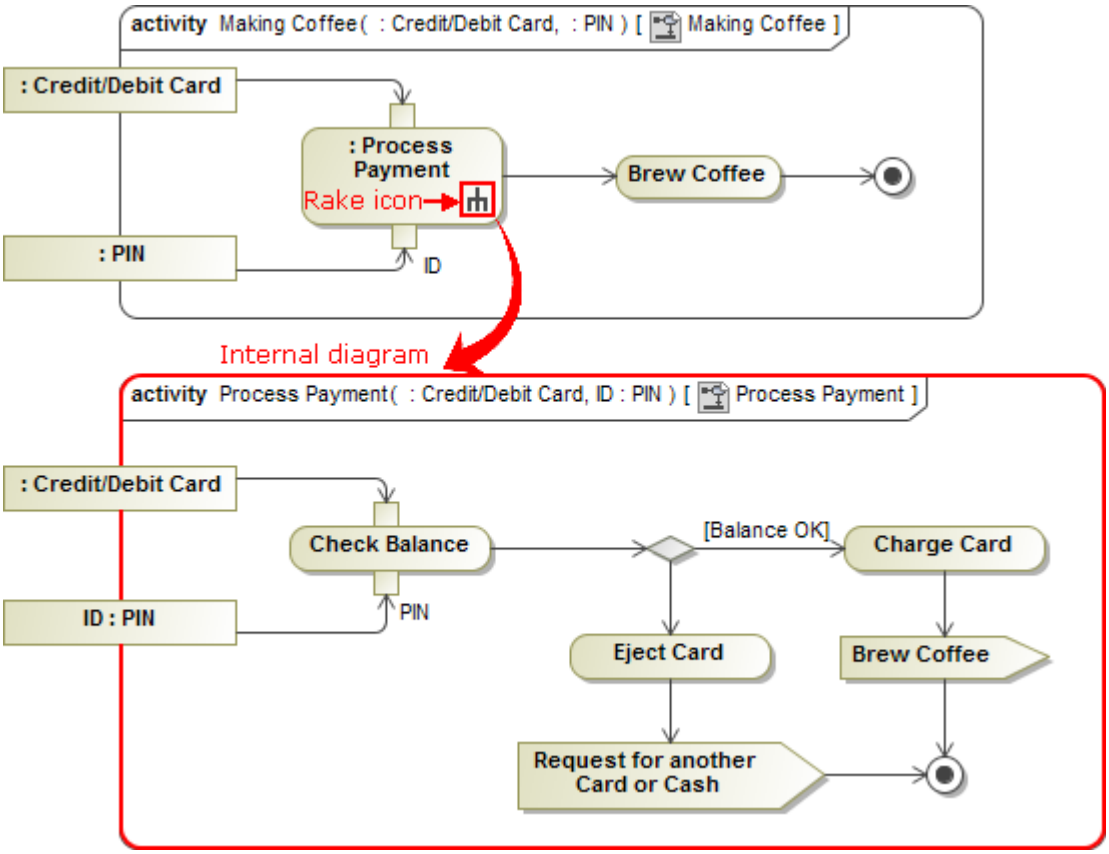


Figure 188 -- Using rake icon

Copy and paste symbol style

A symbol style defined in symbol properties can now be copied and pasted on another symbol. Copy a symbol and then select one or more other symbols of the same type where you need to paste copied properties.



Figure 189 -- Example of copied and pasted style

To copy and paste symbol style

1. On the diagram pane, select symbol(s) which properties you want to copy.
2. To copy symbol properties do one of the following:
   - From the **Edit** menu, select the **Copy** command.
   - Press CTRL+C.
   - Click the Copy button on the diagram toolbar.

Copyright © 1998-2015 No Magic, Inc.

3. On the diagram pane, select the symbol(s) on which you want to paste symbol properties.
4. To paste symbol properties on the selected symbols do one of the following:
   - From the **Edit** menu, select the **Paste Style** command.
   - Press CTRL+SHIFT+V.
   - Click the  Paste Style button on the diagram toolbar.



You can copy and paste symbol properties of a more than one symbol at the same time. Symbol properties will be pasted according the element type. For example, on a class symbol the copied symbol properties will be pasted, on a package symbol the copies symbol properties will be pasted and so on.

Related references

Formatting Symbols

Style Engine

Style Engine

The Style engine is a part of MagicDraw, that defines diagrams, shapes, paths, and stereotype properties. There may be few symbol styles defined, but all symbols are created according to the style that is selected as default. There is a possibility to apply different presentation styles for diagram/shape/path/stereotype depending on the diagram type.

Symbol Styles option group

In the **Project Options** dialog, click the **Symbol Styles** option group, to expand the tree hierarchy of all the styles defined within the project. You may use as many of these styles as you wish.

Copyright © 1998-2015 No Magic, Inc.

Shape and **Path** trees have the inner structure to help you find the model element, the representation of which must be changed. The right side of the dialog contains possible choices and instruments to manage them.

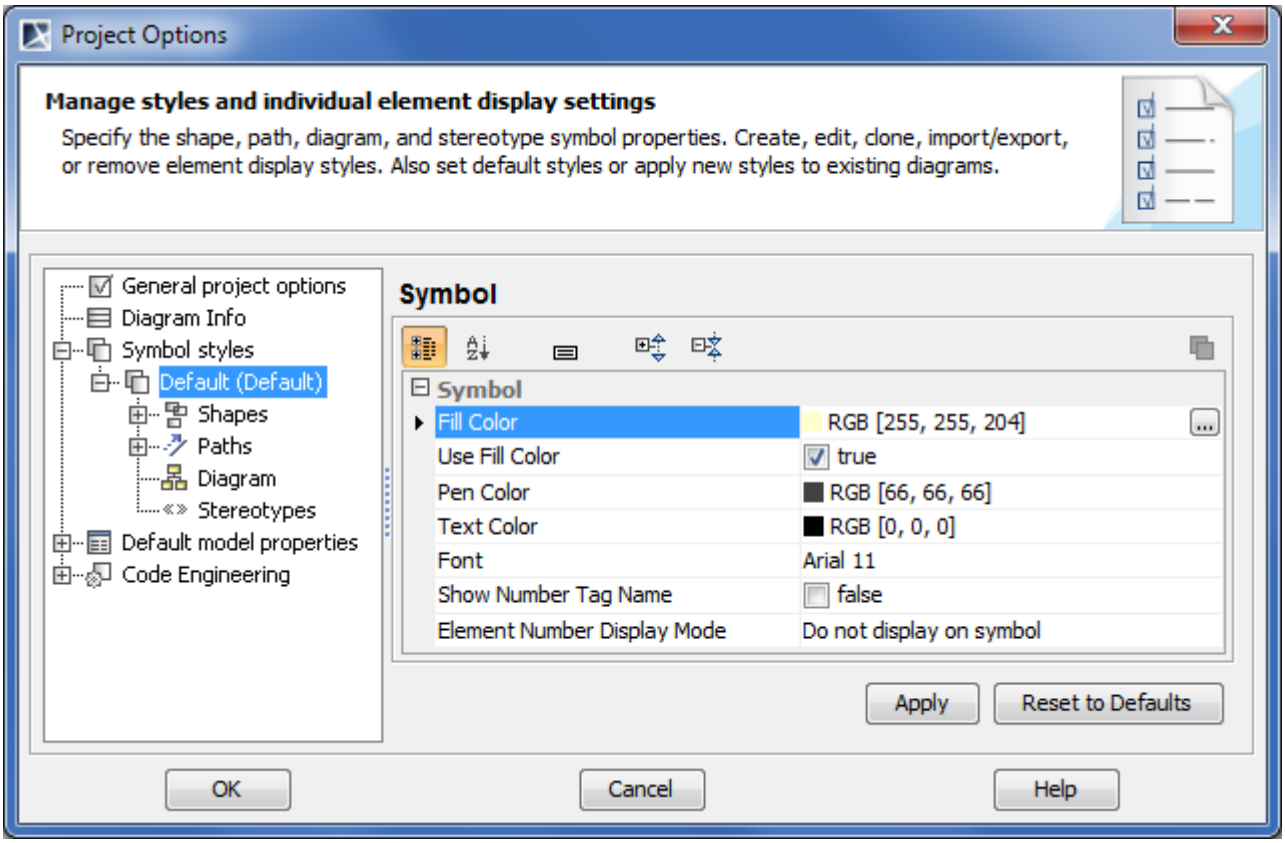


Figure 190 -- Project Options dialog. Default style of symbol properties

To open the **Symbol styles** option group

1. On the main menu, click **Options** > **Project**. The **Project Options** dialog opens.
2. In the tree on the dialog's left side, select **Symbol styles**.

To create a new style by cloning the existing one

1. Select the default style in the **Styles** list box and click the **Clone** button.
2. Type a name for the new style in the **Enter Style Name** dialog.
3. Change options of the new style.

To change the name of the selected style

1. Select a style you want to rename and click the **Rename** button.
2. Type a new name for the style in the **Enter Style Name** dialog.

To remove the selected style

- Click the **Delete** button in the **Project Options** dialog.

To make a selected style your default style

- Click the **Make Default** button in the **Project Options** dialog.

Copyright © 1998-2015 No Magic, Inc.

To apply the selected style or changed option to a current project

- Click the **Apply** button in the **Project Options** dialog, **Symbol Styles** pane.



You can also apply the desired options to the selected diagram model elements. Click the **Apply** button in the specific elements pane.

To import an already created (and exported) project style

- Click the **Import** button. The **Open** dialog opens. Select the style you wish to import (*.stl).

To save the created style (export) for a later usage or for other users

- Click the **Export** button. The **Save** dialog opens. Select the directory where you wish to export a style.

Working with Properties Styles

All symbols in MagicDraw are created according to active properties styles. There may be more than one property style in the same project, and the whole style may be applied for the project.

Every style has its own presentation of Diagram, Shape, Path, and Stereotype that you can modify using the **Project Options** dialog in the **Symbol styles** option group. You can set your own options for every model element to the current style.

Path, Shape, and **Stereotype** branches have the inner structure that helps you find the model element, the representation of which must be changed. The section on the right side of the dialog contains possible choices and instruments to manage them.

The following properties are defined for the formatting symbols:

- **Shapes**. Set general options for the shapes in the right pane of the **Project Options** dialog. You can set options for all shapes that appear on the Diagram pane.
- **Paths**. Set general options for the paths in the right pane of the **Project Options** dialog. You can set options for all paths that appear on the Diagram pane.
- **Diagram**. Set general options about a diagram.
- **Stereotypes**. Set general options for the stereotypes in the right pane of the **Project Options** dialog. You can set options for all stereotypes that may be applied to elements on the Diagram pane.

Copyright © 1998-2015 No Magic, Inc.

Changing properties for multiple element

To change properties for multiple symbols, using Ctrl or Shift key select few elements in the **Project Options** dialog, **Symbol styles** option group.

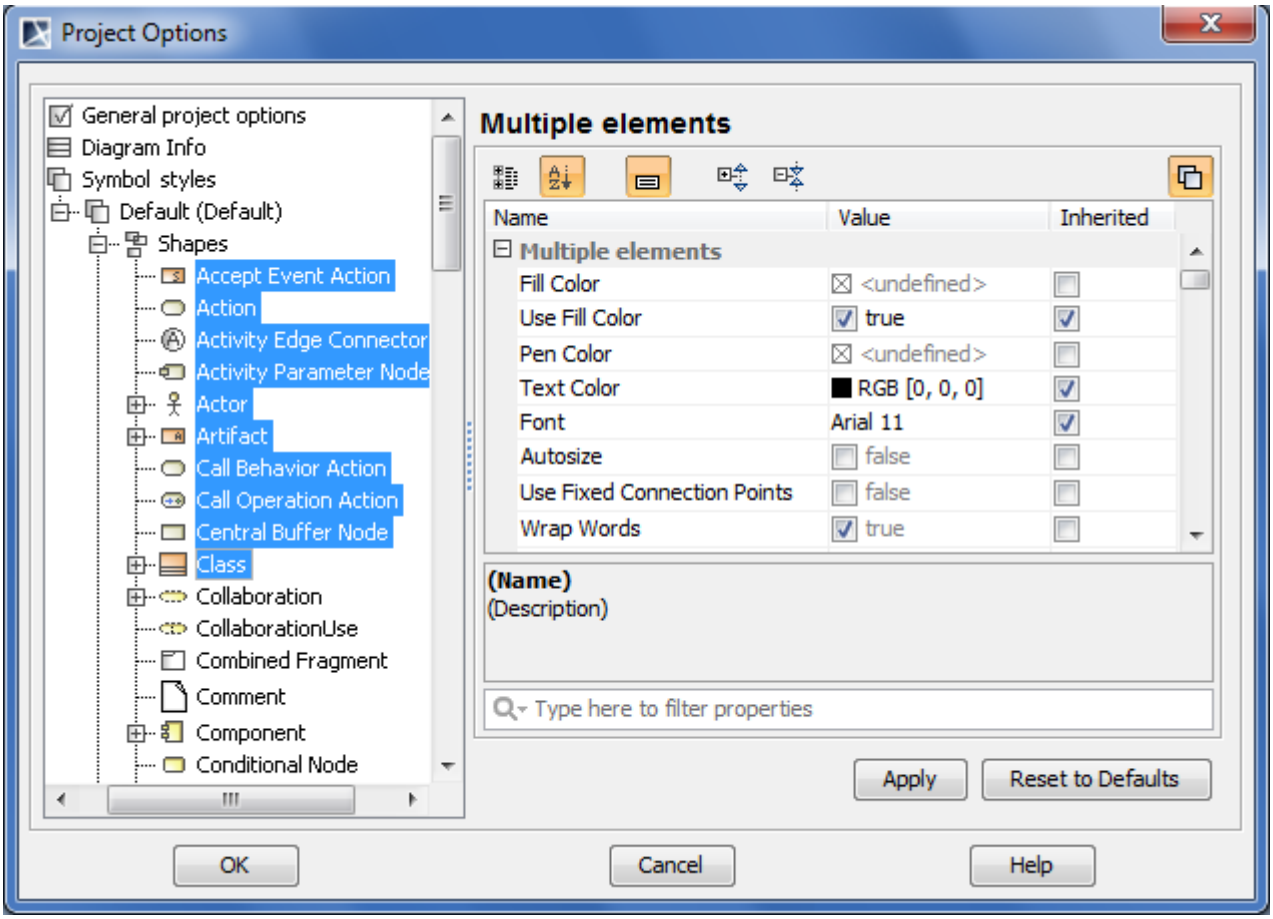


Figure 191 -- The Project Options dialog, multiple element symbols style is selected

Properties extension by diagram

Diagram, shape, path, and stereotype properties can be extended by the particular diagram type. This means that presentation style options will be applied only for the specified element symbol in the specific diagram.

To extend the element properties by diagram

1. In the **Project Options** dialog, the **Symbol Styles** option group, expand an option group, select the specific element (shape, path, diagram, or stereotype) and right click the mouse button. The list of diagrams in which the element symbol may be created, opens.

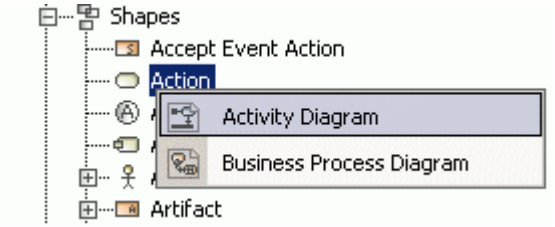


Figure 192 -- Element extension by diagram

Copyright © 1998-2015 No Magic, Inc.

2. Select the diagram type. The Diagram is added as an additional branch to the section.

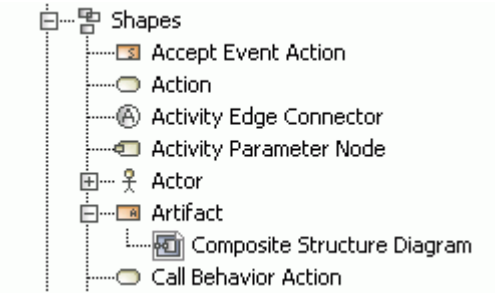


Figure 193 -- Extended diagram addition

3. Set the style properties for the element in the right pane of the **Project Options** dialog. The properties will be applied only in the specified type of diagram.
- The element can be extended by diagram in the **Project Options** dialog, specific elements pane, by clicking the **Extend by Diagram** button. The **Extend by Diagram** dialog opens. Click the **Add Diagram** button and select a diagram from the list.

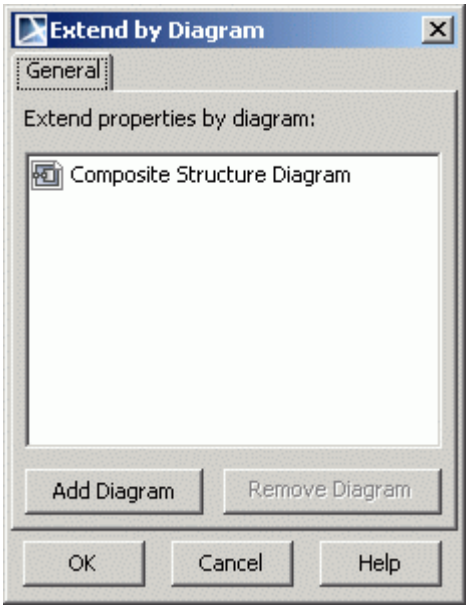


Figure 194 -- Extend by Diagram dialog

To remove the extended diagram from the tree

- Select the extended diagram and right click on the mouse, then select **Remove**.

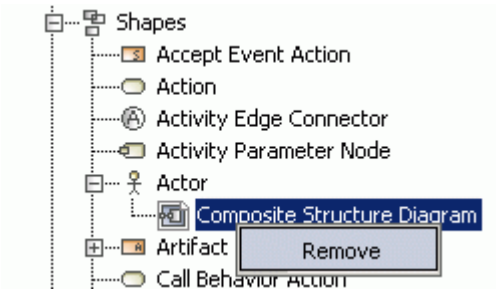


Figure 195 -- Remove extended diagram

- In the **Project Options** dialog, the extended diagram style properties pane, click the **Remove** button.

Copyright © 1998-2015 No Magic, Inc.

Properties Inheritance

All element properties have the “inherited” check box. This check box indicates the property is derived from the base element properties or is it specific.

The **Inherited** column check box value in the elements properties pane specifies if the current property is synchronized with its parent property. When the **Inherited** value is “true”, the element property is changed after changing the parent property.

If the property has no correspondent property in the upper (parent) level, the **Inherited** column check box will be cleared and disabled.

If the property is modified for the specific element and the value differs from the upper level current property value, the **Inherited** column check box is cleared automatically.

General Style Properties

You can define the common properties for the whole style. The style properties are displayed when the properties style is selected in the **Project Options** dialog styles tree.

Shape, Path and Diagram Properties

All shapes, paths and diagrams that can be created in the project, are listed in the **Project Options** dialog. If the **Shape, Path,** or **Diagram** branches are selected in the tree, the general properties can be set in the right pane of this dialog.

When expanding any of these branches, the style for a concrete element (diagram) can be created.

To apply a new style to a previously created element symbol

1. In the **Project Options** dialog, change the element style properties and click the **Apply** button. The **Select Diagrams** dialog opens. The list of diagrams created in the project is displayed.

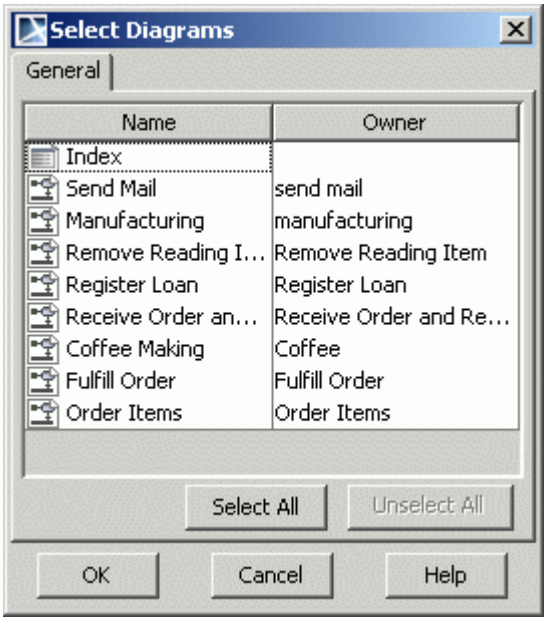


Figure 196 -- Select Diagrams dialog

Copyright © 1998-2015 No Magic, Inc.

2. Select the diagrams to which the element properties will be applied and click **OK**. The **Select Properties to Apply** dialog opens.

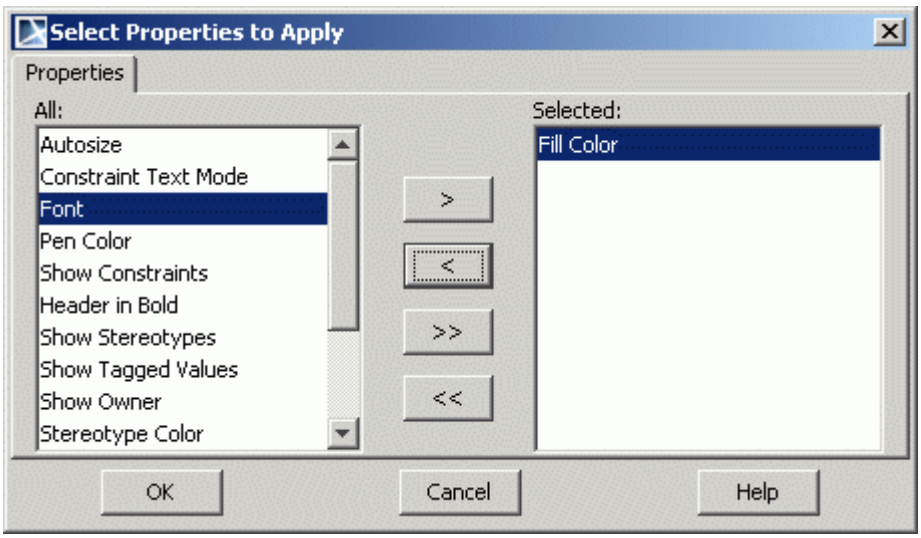


Figure 197 -- Select Properties to Apply dialog

3. Select the properties to be applied to the element symbol by moving them from the **All** list to the **Selected** list. Click **OK.**

NOTE

If a new style was set, it will be applied for all newly created elements after drawing them on the diagram pane. You can always set the default symbols style to the element by clicking the **Apply Default Symbol Style** button in the main toolbar.

Stereotype properties

The Stereotype properties can be applied only if the stereotype properties style is created in the **Project Options** dialog.

The Stereotype properties are derived from their base class. The Stereotype base class is defined in the label of the right pane of the **Project Options** dialog.

The same element can have several stereotypes assigned. In this case, the style of the first stereotype will be applied to the element symbol. If the stereotype is removed from the element, the next (first) stereotype properties are applied. If the last stereotype is removed from the element, the base class (shape or path) properties are applied to the element symbol.

Stereotypes may be extended by diagram.

All stereotypes that have defined symbol properties are included in the **Stereotypes** branch. The default style is created for these stereotypes.

To add a stereotype to the **Stereotypes** branch

1. In the **Project Options** dialog, the **Symbol styles** option group, right-click the **Stereotypes** branch. The **Select Stereotypes** dialog opens.
2. Select the stereotypes you want to add and click the **OK** button.
   The stereotypes are included into the **Stereotypes** branch. Set the stereotype style properties in the right pane of the **Project Options** dialog.

Copyright © 1998-2015 No Magic, Inc.

To remove a stereotype from the branch

1. In the **Project Options** dialog, the **Symbol styles** option group, select the **Stereotypes** branch. The list of stereotypes opens.
2. Clear the check box near the stereotype and click **Apply.** The stereotype is removed from the branch.

To change stereotype properties

1. Expand the **Stereotypes** branch and select a stereotype.
2. Set properties in the right pane of the **Project Options** dialog.

To apply the stereotype properties to a previously created symbol with assigned stereotype

1. When the stereotype style properties in the right pane of the **Project Options** dialog are changed, click the **Apply** button. The **Select Diagrams** dialog opens.
2. Select the diagrams to which the stereotype properties will be applied and click **OK**.
3. In the **Select Properties to Apply** dialog, select which properties will be applied to the stereotype. Click **OK.**



You can apply stereotype properties to a symbol after changing the style properties and in the **Project Options** dialog, click **OK.** The style will be applied when selecting symbol on the diagram pane and clicking the **Apply Default Symbol Style** button on the main toolbar.

To apply the stereotype properties when assigning a stereotype to an element

1. In the created diagram, draw an element.
2. From the element shortcut menu, select **Stereotype**. The list of available stereotypes opens.
3. Select the check box near the stereotype you want to assign to the element. Click **Apply.** The stereotype properties are applied automatically when assigning the stereotype to the element.

Collecting symbol styles from modules

As of MagicDraw version 17.0.3, a module[7] in which the symbol style is defined is displayed next to the symbol style in the **Project Options** and **Symbol Properties** dialogs. The new functionality also allows for copying a symbol style from a module to the current project symbol style.

This new functionality solves the following problem: if you had defined symbol style in the project, after a module usage you couldn't know the source (module) of the symbol style. The symbol style from your project

7. Starting from version 18.1, referred as "used project" in MagicDraw UI. This section mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

was changed and you couldn't change it back. Now this problem is solved - you can see the concrete module from which the symbol style is used and copy the symbol style from modules.

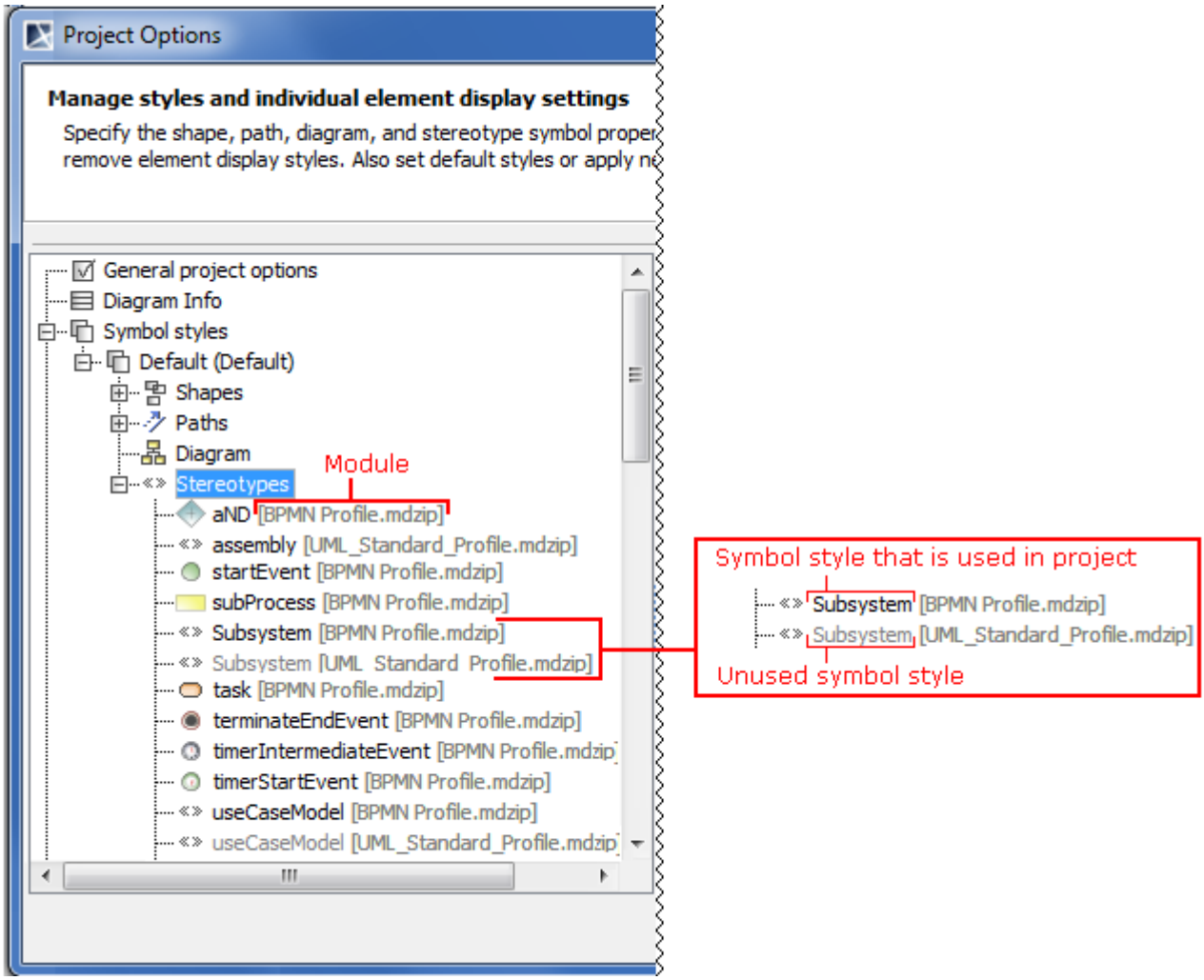


Figure 198 -- Example of symbol styles from modules that are displayed in the Project Options dialog

Copyright © 1998-2015 No Magic, Inc.

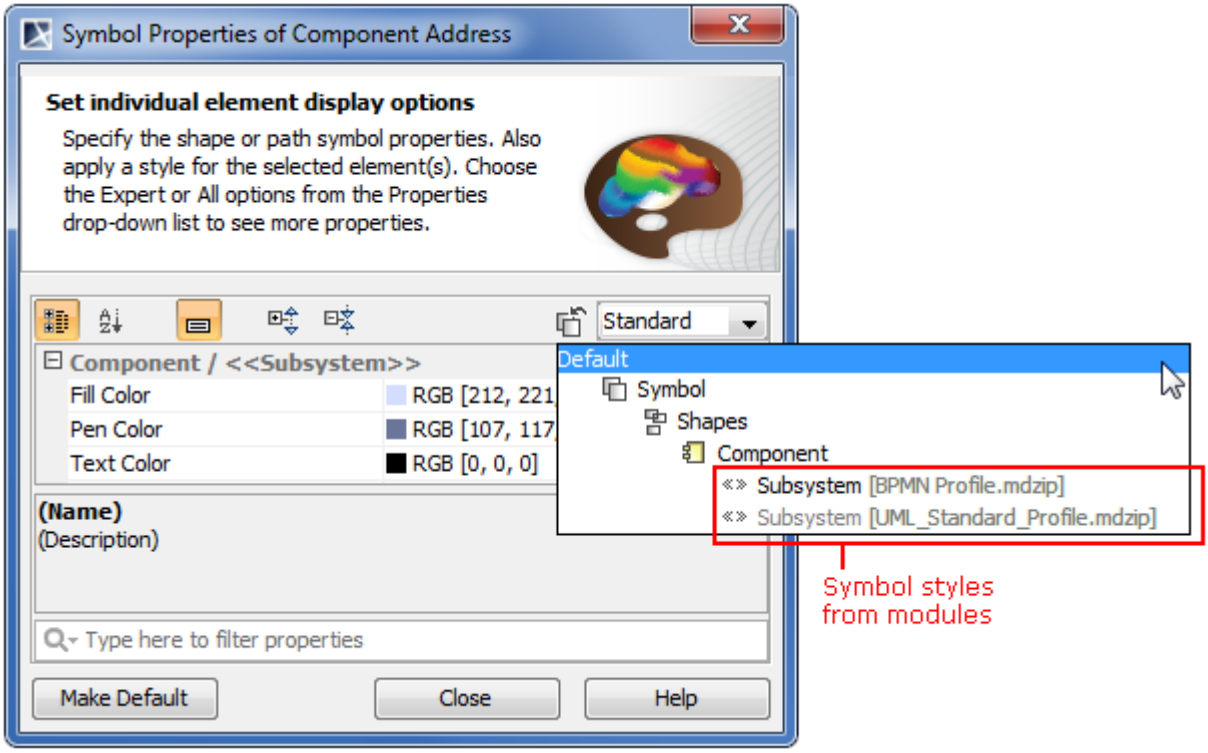


Figure 199 -- Example of symbol styles from modules that are displayed in the Symbol Properties dialog

In the example, you can see that in the project there are two Subsystem symbol styles - one style from the *BPMN Profile.mdzip* module and the other from the *UML_Standard_Profile.mdzip* module. The Subsystem symbol style from the *BPMN Profile.mdzip* module is used in the current project.

The symbol style from module is not editable. This is so, because the module by itself is read-only. By copying the symbol style from the module to the project you can create a new symbol style and then edit it.

IMPORTANT!

Note that there are no rules for the symbol style usage from modules - symbols styles are used according to modules loading order.

TIP!

For more information on how to do this, see the following procedures:

- "To copy a symbol style to the project style by using the Project Options dialog" on page 333.
- "To copy a symbol style from a module to the project by using the Symbol Properties dialog" on page 335.
- For more information about modules, see "Project Partitioning" on page 147.
- For more information about the **Project Options** dialog, the **Symbol styles options** group, see "Symbol Styles option group" on page 324.
- For more information about the **Symbol Properties** dialog, see "Formatting Symbols" on page 317.

To copy a symbol style to the project style by using the Project Options dialog

1. In the **Project Options** dialog, under the **Symbol styles** > **Default** > **Stereotypes** options group, select a symbol style that you want to copy. For example, select the *Subsystem [BPMN_Profile.mdzip] (see* Figure 200 on page 334*).*
2. Click the **Copy to Project Style** button. The new *Subsystem* symbol style is created in the project. This symbol style is now used in the project and this is the editable symbol style (see

Copyright © 1998-2015 No Magic, Inc.

Figure 201 on page 335).

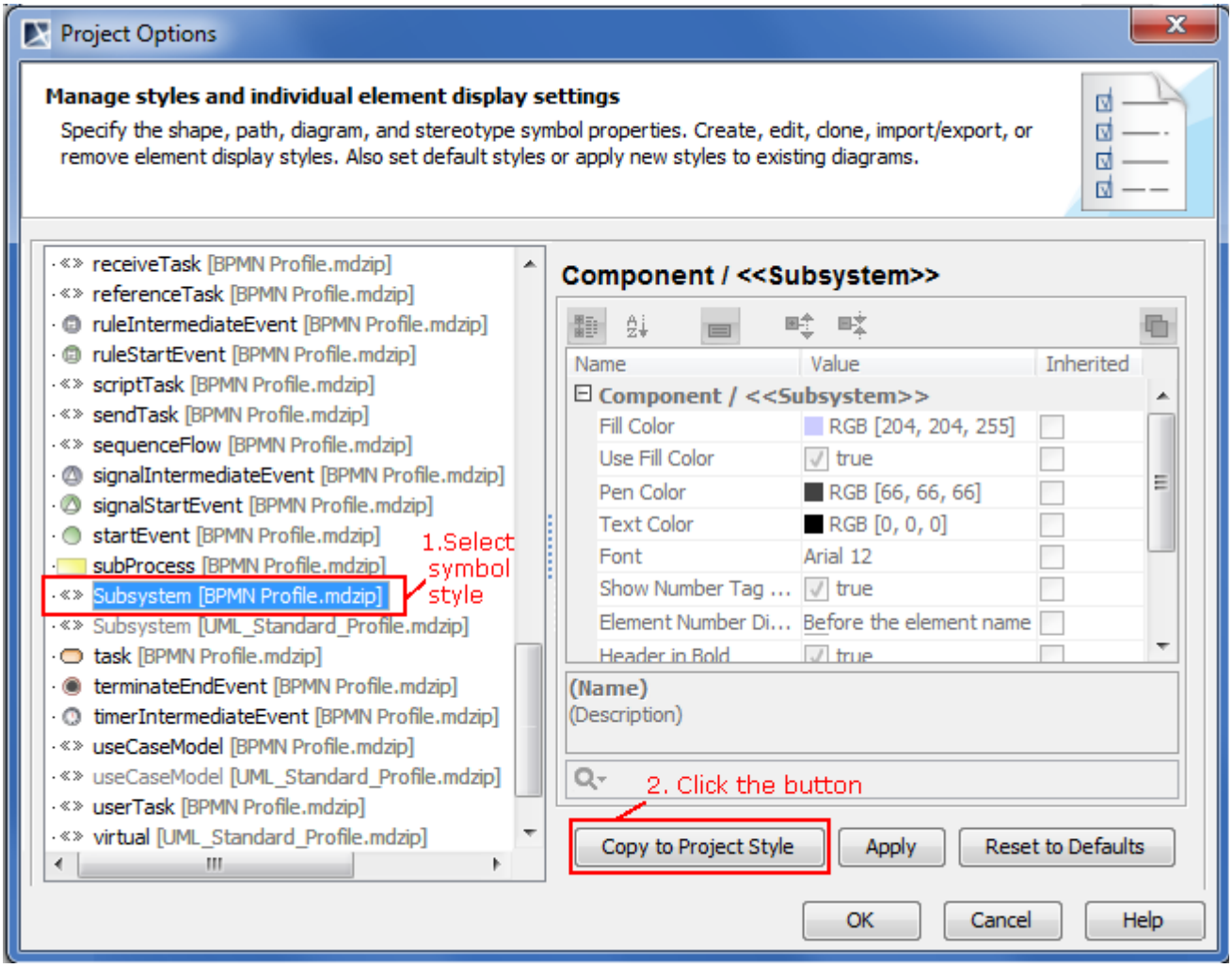


Figure 200 -- Example of copying symbol style from module

Copyright © 1998-2015 No Magic, Inc.

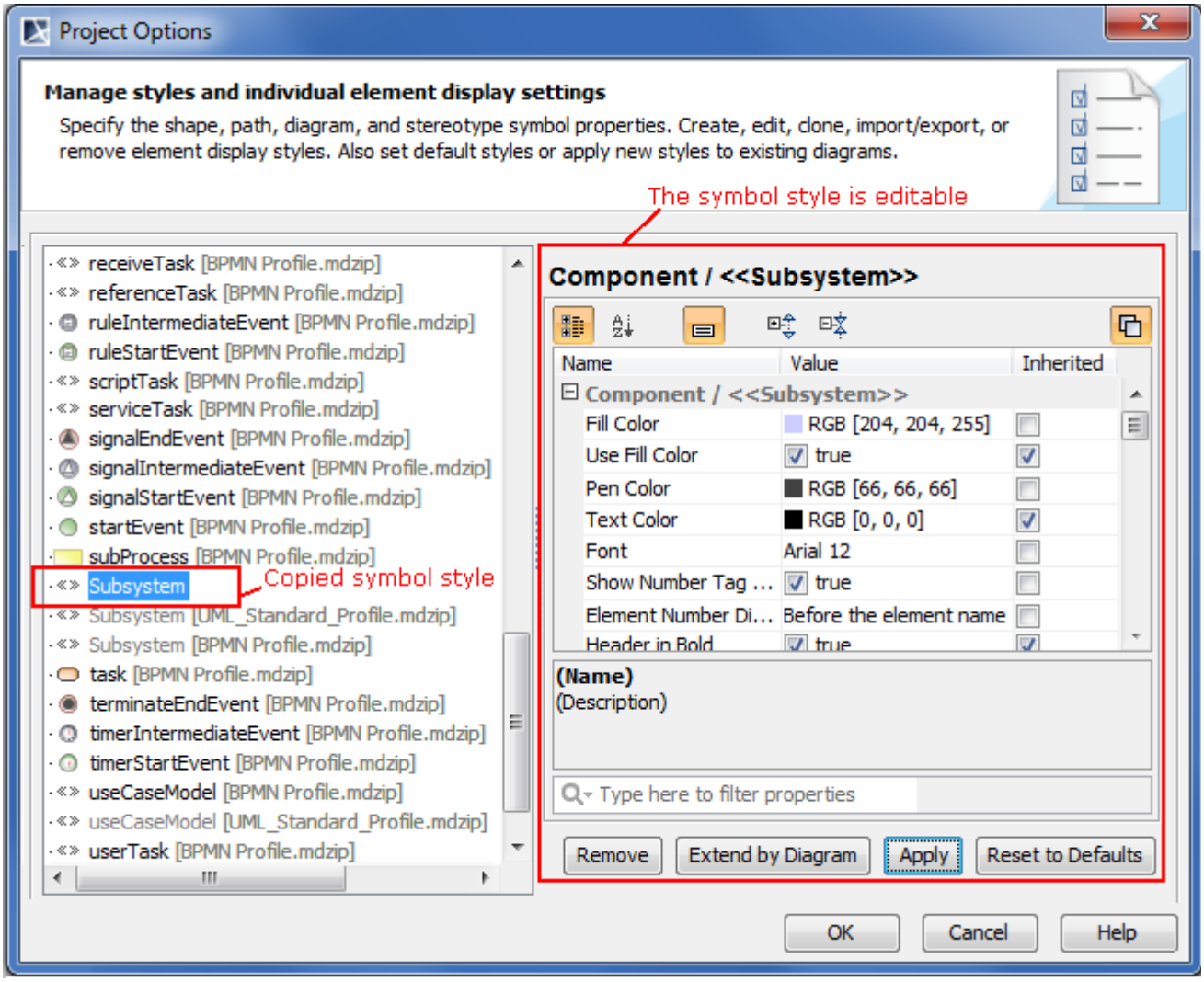


Figure 201 -- Example of copied symbol style

To copy a symbol style from a module to the project by using the Symbol Properties dialog

1. In the **Symbol Properties** dialog, click the *Apply Style* button and from the drop-down list, select a style from a module that you want to copy.
2. Click the **Make Default** button.
3. Edit the symbol style.
4. Click **OK** to close the dialog.
5. Open again the **Symbol Properties** dialog. In the expanded *Apply Style* drop-down list, you can see the newly created symbol style. This is the copy of the selected style from the module.

Copyright © 1998-2015 No Magic, Inc.

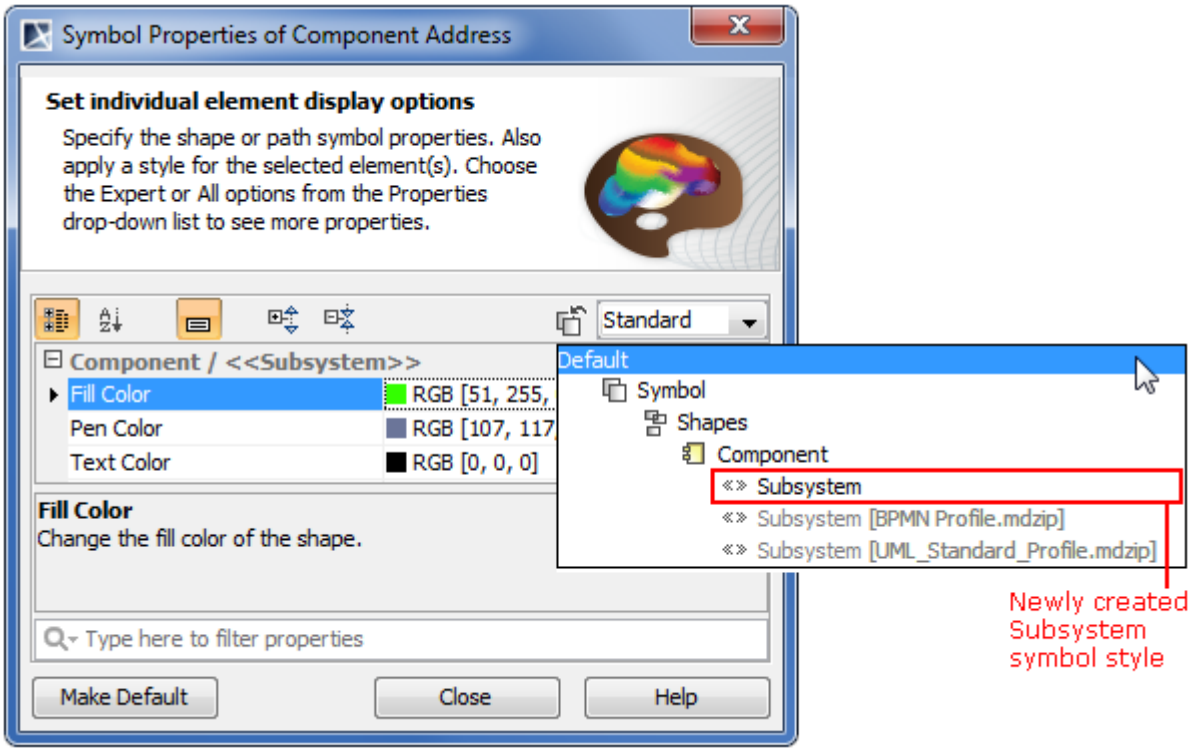


Figure 202 -- Example of copied symbol style that is listed in Symbol Properties dialog

Related concepts

Style Engine

Stereotype properties

Related references

Project Partitioning

Symbol Styles option group

Formatting Symbols

Customizing Lists of Properties and Commands

You can customize the following:

- The list of properties in the element Specification window
- The list of symbol properties in the **Symbol Properties** dialog
- The list of properties in menus that opens after clicking the *Compartments* smart manipulators and also after clicking the *Create Element* smart manipulator
- The list of symbol properties in the symbol shortcut menu

Copyright © 1998-2015 No Magic, Inc.

You can customize lists of properties and commands by using the **Customize Properties** dialog.

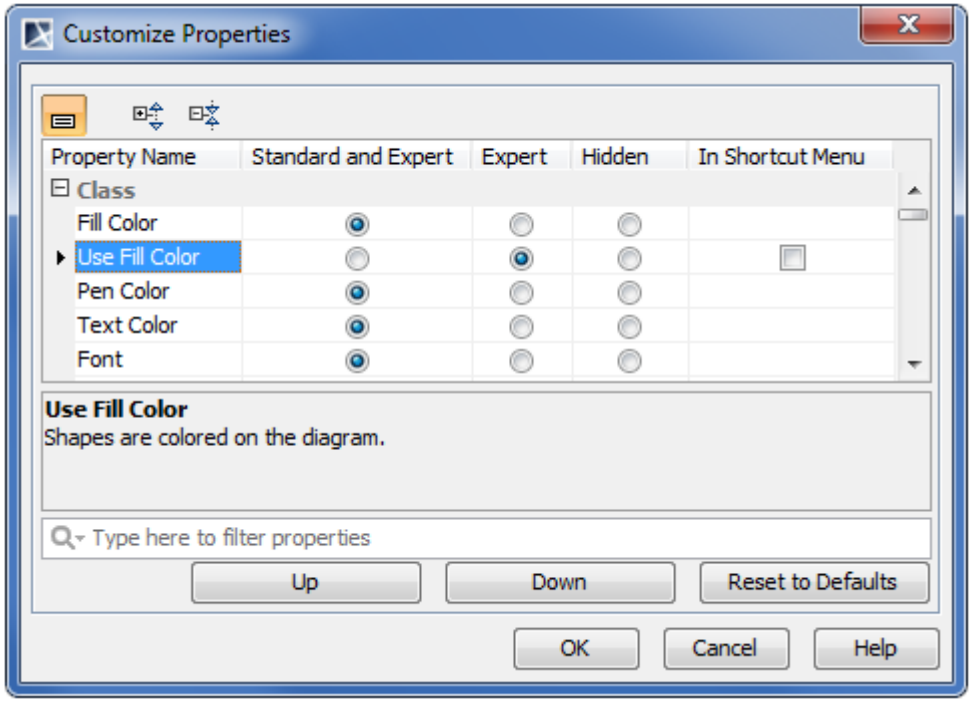


Figure 203 -- Example of Customize Properties dialog for symbol properties

To open the **Customize Properties** dialog

Do one of the following:

- In the element Specification window, at the right-top corner, click to expand the *Properties* list, and click **Customize**.
- In the Model Browser, in the **Properties** tab, under the **Element** tab, at the right-top corner, click to expand the list and click **Customize**.
- On the diagram, right-click the symbol and select **Symbol Properties**. In the **Symbol Properties** dialog, at the right-top corner, click to expand the list, and click **Customize**.
- Select the symbol on the diagram. In the Model Browser, in the **Properties** tab, under the **Symbol** tab, at the right-top corner click to expand the list and click **Customize**.

TIP!

When the **Customize Properties** dialog is opened from the element Specification window or the Element tab from the Model Browser, you can customize the list of properties in the element Specification window.

When the **Customize Properties** dialog is opened from the **Symbol Properties** dialog or the **Symbol** tab in the Model Browser, then you can customize the following:

- The list of properties in the element Specification window.
- The list of properties in the menus that opens after clicking the *Compartments* smart manipulator and after clicking the *Create Element* smart manipulator.
- The list of properties in the symbol shortcut menu.

The following table describes the columns in the **Customize Properties** dialog.

Column name	Type	Description
Property Name	**Text**	The property, which visibility mode you will customize.

Copyright © 1998-2015 No Magic, Inc.

Column name	Type	Description
Standard and Expert	**Option button**	Click to display the symbol or specification property in the *Standard*, *Expert* and in *All* visibility modes.
Expert	**Option button**	Click to display the symbol or specification property only in the *Expert* and *All* modes and do not display in the *Standard* mode.
Hidden	**Option button**	Click to display the symbol or specification property only in the *All* mode.
In Shortcut Menu *Note that this column is only when customizing the list of symbol properties*	**Check box**	Click to add the selected symbol or specification property as a command or menu into the symbol shortcut menu. Note that if there are 8 or more selected check boxes, the properties in the symbol shortcut menu are added under the **Presentation Options** submenu.

See the following examples of customizing list of properties in the element Specification window, in the *Compartments* and *Create Element* menus, and in the symbol shortcut menu.

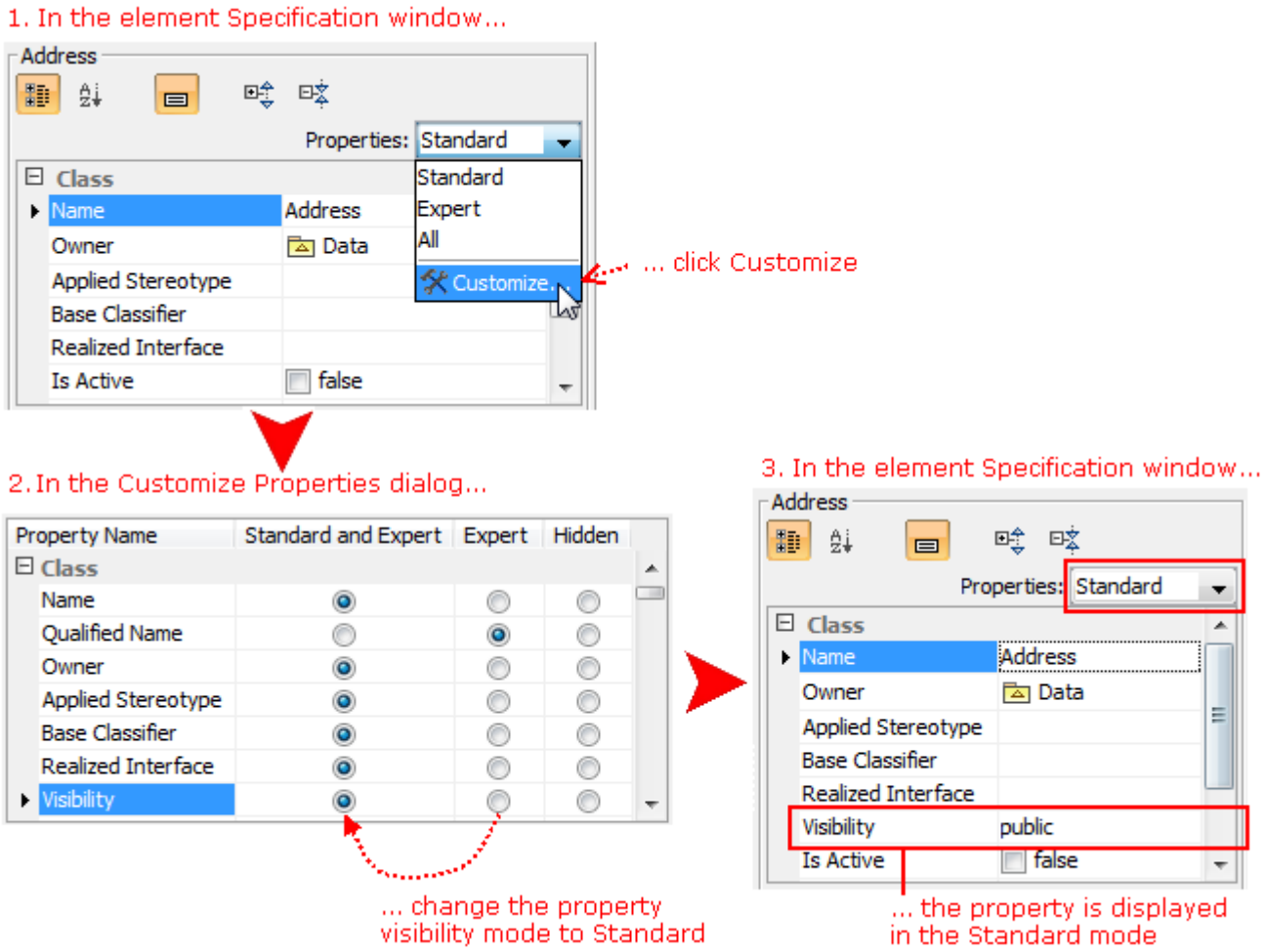


Figure 204 -- Example of customizing list of properties in element Specification window

Copyright © 1998-2015 No Magic, Inc.

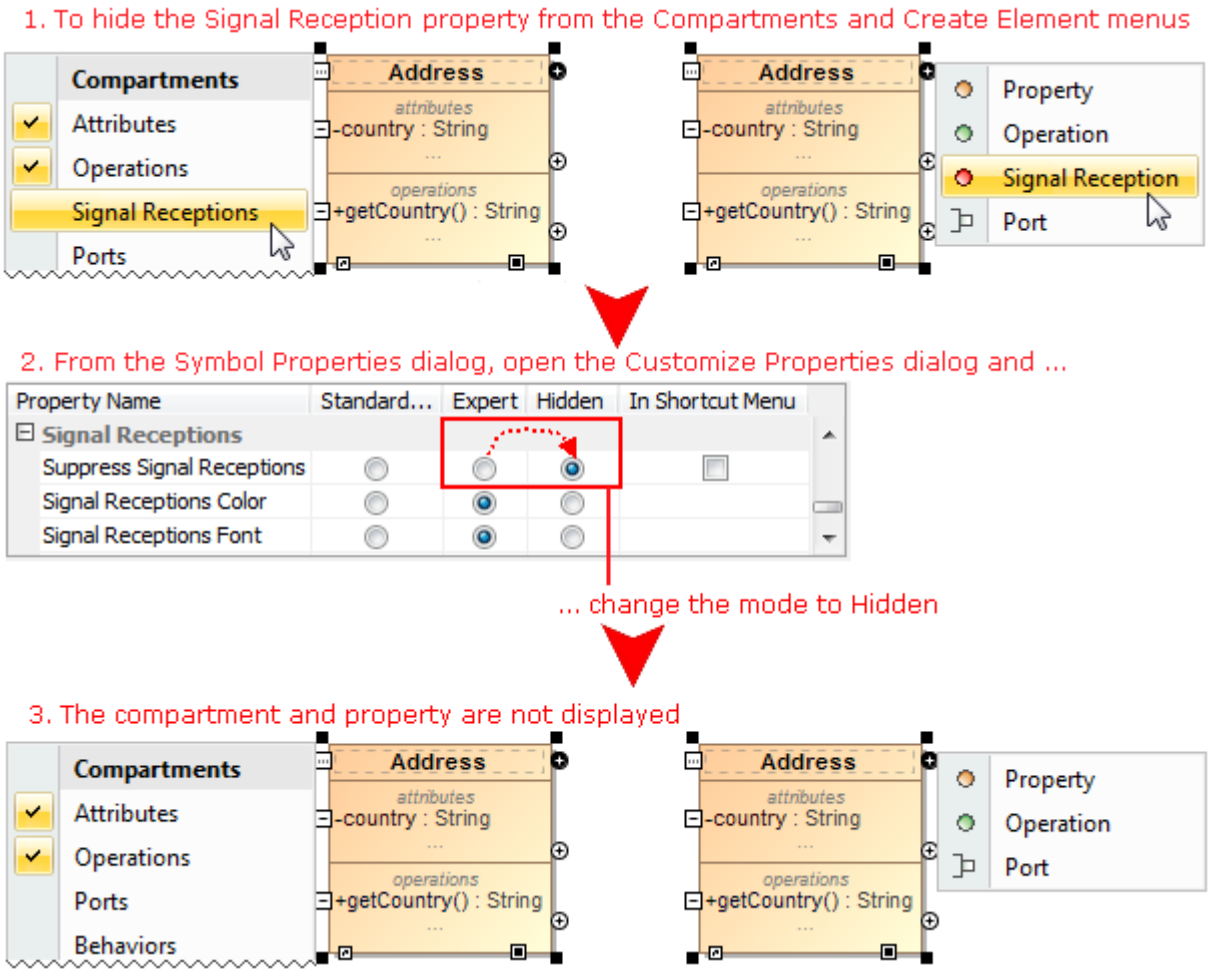


Figure 205 -- Example of customizing list of properties in Compartments and Create Element menus

Copyright © 1998-2015 No Magic, Inc.

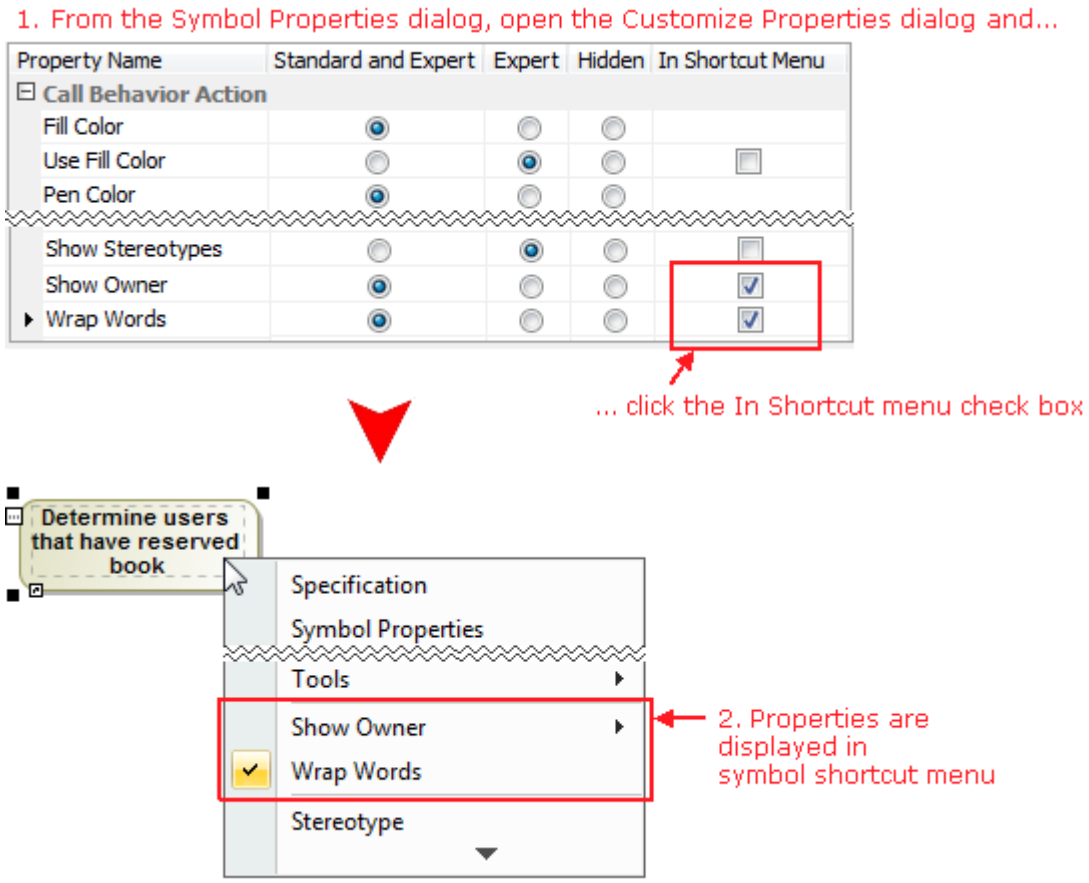


Figure 206 -- Example of customizing list of symbol properties in symbol shortcut menu

Related concepts

Specification Window

Symbol Properties dialog

Properties tab

Smart Manipulation

Related external resources

“Standard Expert Configuration” and “Symbol Standard Expert Configuration” sections in MagicDraw UMLProfiling&DSL UserGuide.pdf

Defining Hyperlinks

You can add a hyperlink from any model element to the following target:

- File
- Address
- Element/Symbol (including diagram)

TIP!

For more information about adding a hyperlink,

see "Adding a hyperlink to the model element" on page 341.

Copyright © 1998-2015 No Magic, Inc.

After a hyperlink is added to an element, you will see the Hyperlinks/Go To icon on the element symbol in the Model Browser. On the diagram pane, on the element symbol, you will see the icon too and this icon will represent the subject to which the hyperlink is added. For example, - this icon means, that the hyperlink is added to the file.

Double click the element with the hyperlink and the hyperlink target will open:

- If the hyperlink was added to the folder/file - the folder/file will open.
- If the hyperlink was added to the web page - the web page will open.
- If the hyperlink was added to the element (for example, to the class) - the element (class) Specification window will open.
- If the hyperlink was added to the diagram - the diagram will open in the same diagram tab.
- If the hyperlink was added to any kind of address - the specific launcher will be started.

You can manage hyperlinks, using the same way how the hyperlink was added.



For more information, see "Adding a hyperlink to the model element" on page 341.

You can edit hyperlinks, using the **Edit Hyperlink** dialog.



For more information, see "Edit Hyperlink dialog" on page 344.

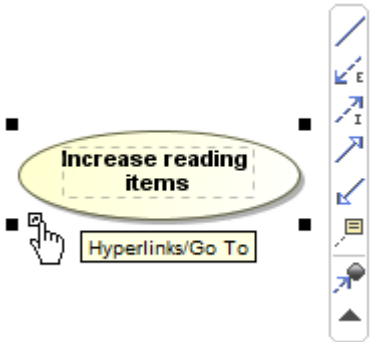
Adding a hyperlink to the model element

There are the following ways to add a hyperlink to the model element:

- from diagram,
- from **Specification** window,
- from the Containment tree.

To add a hyperlink from the diagram

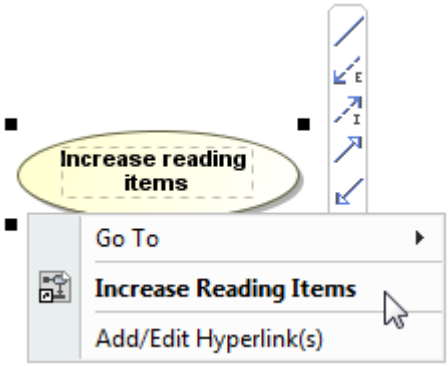
1. Select the element and click the smart manipulator for Hyperlink.



2. The hyperlink menu opens, listing:
   - Previously created hyperlinks with icons corresponding to element type, diagram type, external file, or requirement.
   - Menu item for adding and editing existing hyperlinks - **Add/Edit Hyperlink(s)**

Copyright © 1998-2015 No Magic, Inc.

- If there are no hyperlinks yet defined, only the menu item to add a hyperlink will be in the hyperlink menu.



3. Click **Add/Edit Hyperlink(s)** item. The hyperlinks creation editing dialog opens.

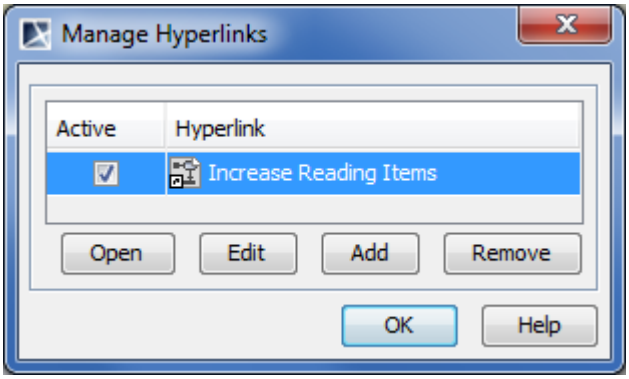


Figure 207 -- Manage Hyperlinks dialog

4. Click **Add** and define the hyperlink to any model element, file, or address in the **Edit Hyperlink** dialog. If you want this hyperlink to be active, select the **Active** check box.
5. Click **OK**.

To add a hyperlink from Specification window

1. Open the model element **Specification** window and select the **Documentation/Hyperlinks** property group.
2. Click the **Add** button. The **Edit Hyperlink** dialog opens.
3. Define the hyperlink to any model element, file, or address. If you want this hyperlink to be inactive, click to clear the **Active** check box.
4. Click **OK**.

Copyright © 1998-2015 No Magic, Inc.

To add a hyperlink from the Model Browser

Now hyperlinks can be created and edited straight from the element shortcut menu:

- Invoke element shortcut menu from the Model Browser.
- Choose **Go To** and **Hyperli**nks. See the following figure.

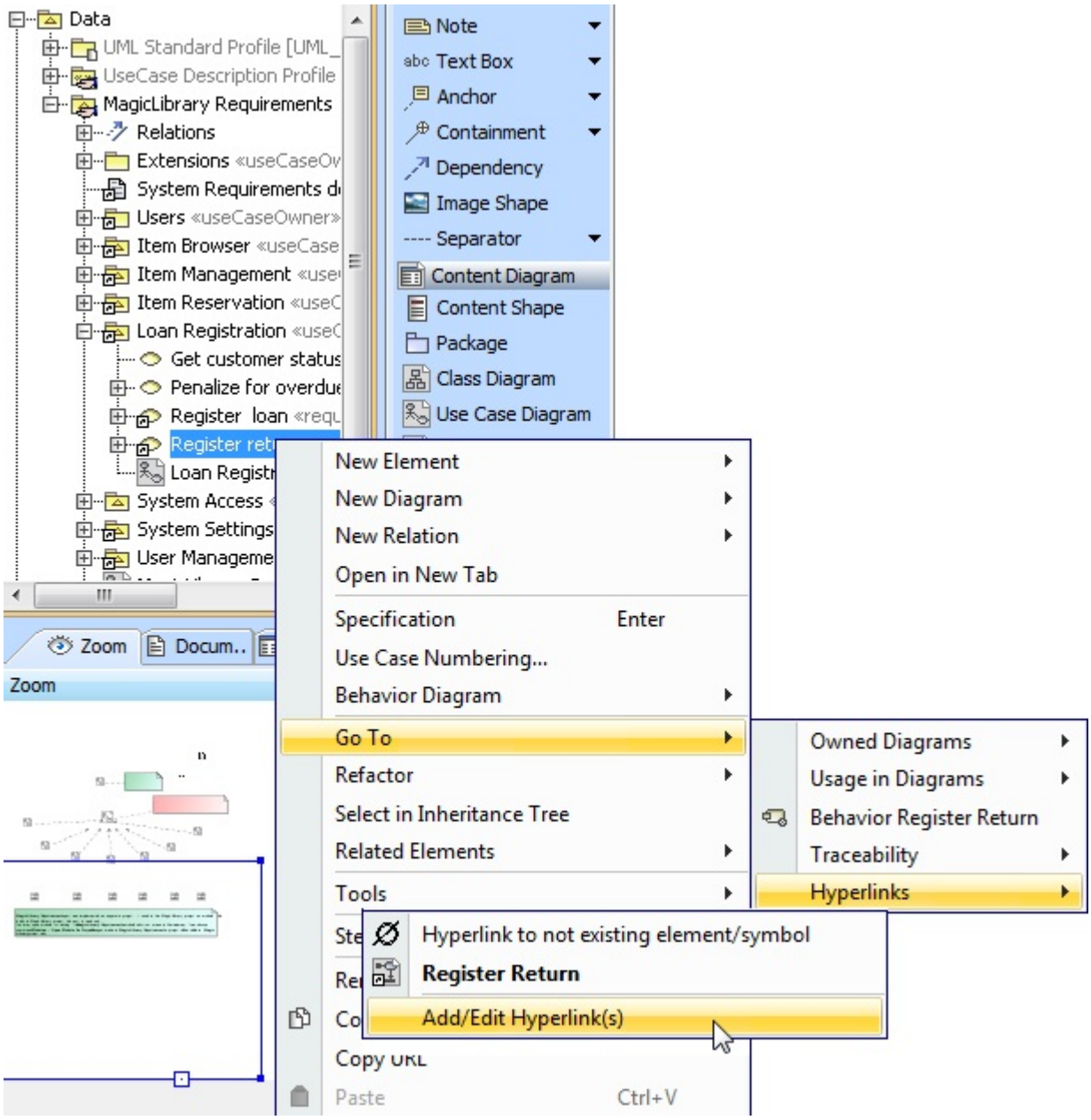


Figure 208 -- Hyperlinks creation from the Browser tree

TIP!

For more detailed description on managing hyperlinks, see "To add a hyperlink from the diagram" on page 341.

To add a hyperlink on a note, text box, or separator text using the Edit Hyperlink dialog

1. Select the text where you want to add a hyperlink and click **Insert Hyperlink** .
2. The **Edit Hyperlink** dialog opens. Select the hyperlink you want to insert, either to an address, another model element, or a file:

Copyright © 1998-2015 No Magic, Inc.

- To link to an existing model element, click the **Element/Symbol** tab. Click the **Select Element/Symbol …** button and select the model element or symbol you want to link to in the element Selection dialog.

TIP! More information about this dialog find in the Section "Selecting an Element" on page 352.

- To link to an existing address, click the **Address** tab, and in the **Type the address** text box, type the protocol or the Web page you want to link to. You can click the **…** button and browse to the desired address.

NOTE Set the path of the HTML viewer in the **Environment Options** dialog (for a description, see “Customizing Environment Options” on page 96.)

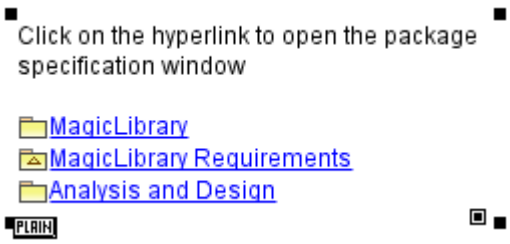
- To link to an existing file, click the **File** tab and enter the path to the file you want to link to. Or, click the **Type the file name …** button and, in the **Open** dialog, select the file you want to link to.

NOTE
- The selected file opens in the HTML browser.
- Set the path of the HTML viewer in the **Environment Options** dialog.
- You can only link to an existing file. New files are not created for you.

Using the HTML editor toolbar, you can change the font, color, size, and the alignment of the selected text.

To add a hyperlink to an element on a note, and text box

1. Select one or more elements in the Model Browser, or Specification window.
2. Drag the selected elements on the note or the text box on the diagram pane. The hyperlinks to specifications of dragged elements are created on the note or the text box. Click the link to open the Specification window of the selected element.



NOTE After the hyperlink to the element on the note or the text box is added, the text format changes to HTML.

Related procedures

Formatting Symbols

HTML Editor

Edit Hyperlink dialog

In the **Edit Hyperlink** dialog, you can add a new or manage an existing hyperlink(s).

Copyright © 1998-2015 No Magic, Inc.

To open the Edit Hyperlink dialog

NOTE The procedures bellow are valid when the element has no hyperlinks created yet.

Do one of the following:

- In the Model Browser or on the diagram pane, select the element and from its shortcut menu, select the **Go To** > **Hyperlinks** > **Add Hyperlink** command.

  TIP! For more information, see "Adding a hyperlink to the model element" on page 341.

- On the diagram pane, select the element symbol, and then click the smart manipulator button. In the opened menu, select the **Add Hyperlink** command.

  TIP! For more information, about the smart manipulators, see "Smart Manipulation" on page 233.

- In the element Specification window, the **Documentation/Hyperlinks** property group, in the **Hyperlinks** group box, click the **Add** button.

  TIP! For more information, see "Documentation/Hyperlinks property group" on page 282.

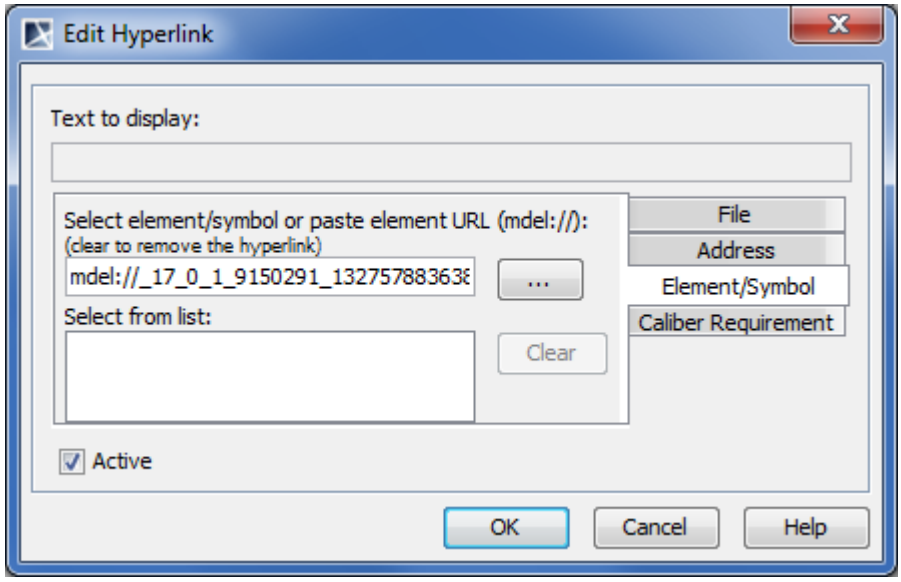


Figure 209 -- The Edit Hyperlink dialog

Copyright © 1998-2015 No Magic, Inc.

Tab name	Box	Function
Element/ Symbol Creates a hyperlink that goes to the selected model element.	**Text to display**	A text that will be displayed as a hyperlink.
	Select Element / Symbol or paste element URL …	The element Selection dialog opens. Select the model element you want to link to. You can also paste URL to element. For more information about element URL, see "Copying/ Opening Element URLs" on page 429.
	Select from list	A list of all items that have previously been selected as links.
	Clear	Remove all items from the **Select from list**.
	Active	If selected, activates the hyperlink on the diagram. Only one hyperlink can be active. Target referenced by the active hyperlink is accessed after double clicking an element with a hyperlink. By default the last added hyperlink is the active one.
Address Creates a link that goes to the specified path.	**Text to display**	The text that will be displayed as a hyperlink.
	Type the address …	Type the web page URL or any other address.
	Select from list	A list of all items that have previously been selected as links.
	Clear	Remove all items from the **Select from list**.
File Creates a hyperlink that goes to a specified file.	**Text to display**	A text that will be displayed as a hyperlink.
	Type the file name …	Type the path to the file you want to be opened or click the **...** button. The **Open** dialog opens. Select the file you want to link to.
	Select from list	A list of items that have previously been selected as links.
	Clear	Remove all items from the **Select from list**.

Owner of the Model Element

Model elements and diagrams belong to a package, model (system boundary), subsystem, or other appropriated model element, which is called *owner*.

The name of the owner is displayed in the model element name compartment in parentheses.

Copyright © 1998-2015 No Magic, Inc.

To add a model element to a package, model (system boundary), or subsystem

Do one of the following:

- Drag a model element to the desired package on the Diagram pane or in the Model Browser.
- Open the **Package**, **Model**, or **Subsystem** Specification window, the **Inner Elements** property group. Click **Create** and select a model element or diagram you want to add. Define a model element or diagram in the open Specification window and click **OK**.
- From the selected owner shortcut menu in the Model Browser, select **Create Element**. From the list, select the desired model element. The element is created.

To display/hide the package/system boundary/subsystem name (the owner of an actor) on a model element

- From the symbol shortcut menu, select **Symbol Properties**. The **Symbol Properties** dialog opens. In the **Show Owner** drop down list, select one of the owner display mode: Do Not Display, Below Element Name, In Same Line With Name, Above Element Name. For more information about the owner display mode, see "Owner display mode" on page 347.
- From the **Options** menu, select **Project**. The **Project Options** dialog opens. Select the desired model element and in the **Show Owner** drop down list, choose one of the owner display mode. If you want to apply changes for previously created model elements, click **Apply**.

Owner display mode

You can change the display position of the qualified name on the element shape.

To change the qualified name position

- On the diagram, right-click the shape and from the shortcut menu, select the **Symbol Properties** command.
- In the **Symbol Properties** dialog, change the **Show Owner** property.

Select one of the four property modes for **Show Owner**. The property modes are described in the following table.

Show Owner Property Mode	Shape	Description	Notation
Do Not Display		Only element name is displayed on the element shape. This is the default value.	-
Below Element Name		Owner is displayed below the element name.	This is MagicDraw style notation. The owner name is constructed from the names of the containing namespaces starting at the root of the hierarchy and ending with the owner of the NamedElement itself. Containing namespaces are separated by dot and owner is displayed in brackets.

Copyright © 1998-2015 No Magic, Inc.

Show Owner Property Mode	Shape	Description	Notation
In Same Line With Name		Element owner is displayed in the same line as the element name.	This is a notation from UML specification. The qualified name is constructed from the names of the containing namespaces, starting at the root of the hierarchy and ending with the name of the NamedElement itself. Containing namespaces are separated by double colons. The double colon is shown to separate containing namespaces and element name.
Above Element Name		Owner is displayed above the element name.	Notation is the same as **In Same Line With Name** option notation.

Qualified name starting from model library

Concepts

Model Library

A package with the «modelLibrary» stereotype applied.

MagicDraw includes an option to show the owner hierarchy starting from the model library as the root.

This option is called **Qualified name display style**. To change its value:

1. From the **Options** menu select the **Project** command. The **Project Option** dialog opens.
2. Select the **General Project Options** branch. In the right side pane, you can modify the option property.

The **Qualified name display style** property is added to the **Project Options** dialog, **General Project Options** branch.

If the **Model Library Relative** property value is selected (default value for a new project), then the full qualified name hierarchy is displayed on the shape, starting from the model library as a root. The model library itself is not displayed. The **Qualified name display style** property allows for having the relative path for library items used in the project.

Changes of Relationships' Ownership when Client or Supplier is Moved to Another Owner

Some issues related to relationships have been addressed to improve usability.

Copyright © 1998-2015 No Magic, Inc.

Now relationships will not get lost in Containment tree while changing the element ownership. Relationships will also be moved together with the client or supplier (or both) so that all the related elements can be grouped together in one place. This will also prevent unexpected dependencies on model partitioning.

For example, if you move two Classes which are connected to the Association relationship to another Package in the Containment tree, a question dialog will open, asking if you want to move the relationship as well

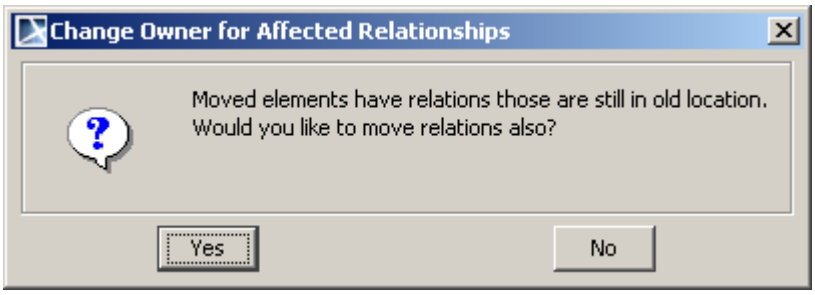


Figure 210 -- Change Owner for Affected Relationships message

NEW! Creation of Elements from other Resources

It is possible to copy a listed text from other resources (Word, Excel, HTML, etc.) and create as many elements as you need by pasting it. Do this either in the Containment tree or on the diagram pane. Each item of this list becomes a new element in the model. The basic numbering and bullets are not reproduced in the model.

IMPORTANT! Avoid names duplication, because elements with the same name cannot be created for the same owner.

To create the elements from the clipboard in the Containment tree

1. Copy a list.
2. Select the possible owner in the Containment tree.
3. From the shortcut menu select **Paste** or press Ctrl+V.
4. From the **Select Type** dialog choose the element type.

The named elements are created in the model.

Copyright © 1998-2015 No Magic, Inc.

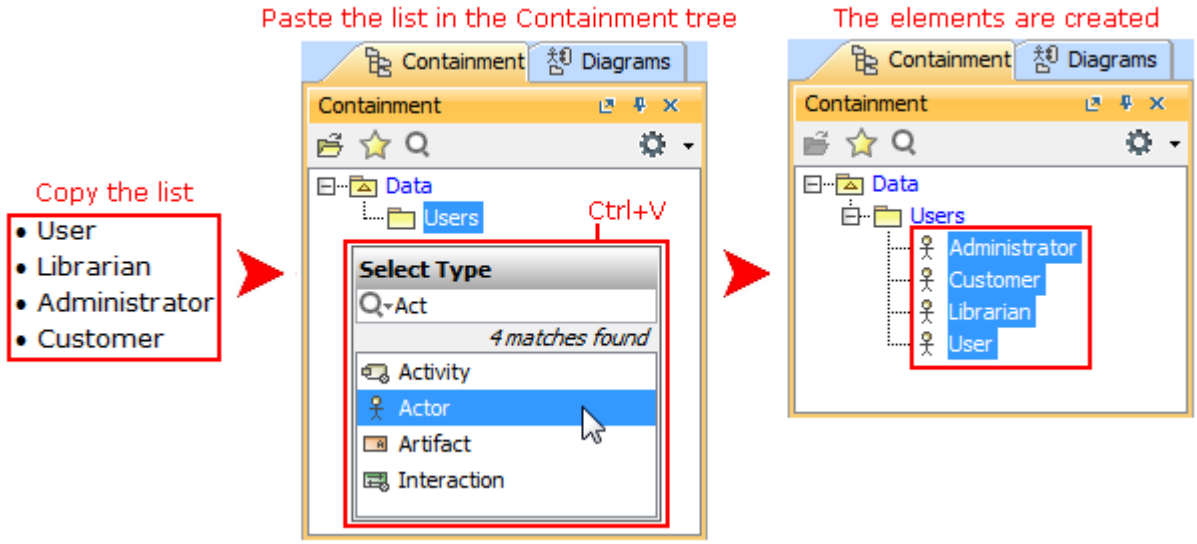


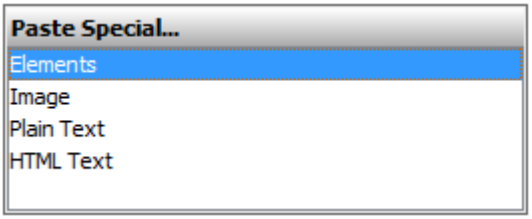
Figure 211 -- Creation of elements from clipboard in Containment tree

There are two ways to create the elements from the clipboard on the diagram pane:

- Using **Select Type** dialog.
- Using diagram pallet.

To create the elements from the clipboard on the diagram pane using **Select Type** dialog

1. Copy a list and click a free space on the diagram pane.
2. Press Ctrl+V and from the **Paste Special** dialog select **Elements**.



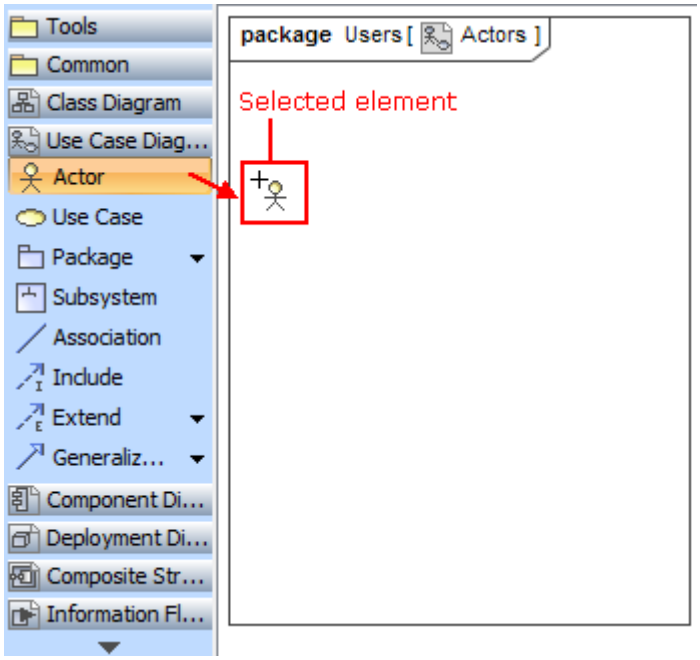
3. From the **Select Type** dialog choose the element type.
The named elements are displayed on the diagram pane and are created in the model.

To create the elements from the clipboard on the diagram pane using diagram pallet

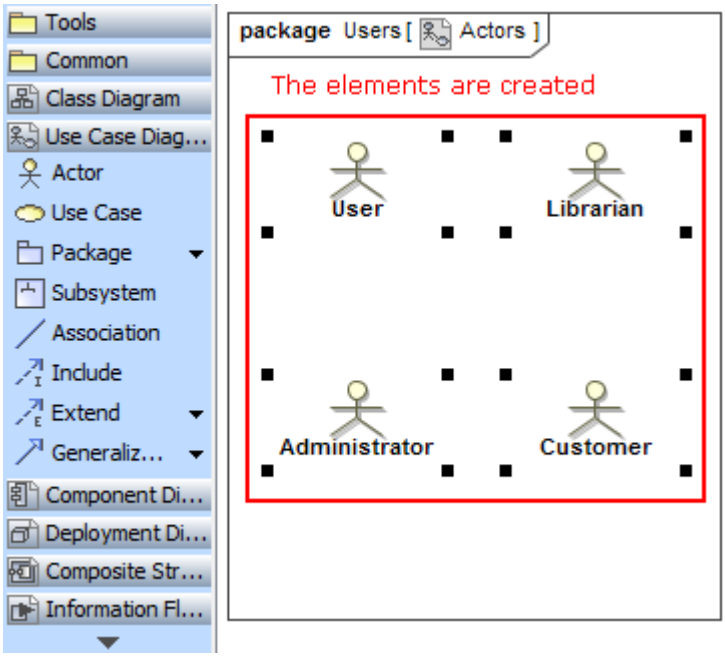
1. Copy a list.

Copyright © 1998-2015 No Magic, Inc.

2. From the diagram pallet select the element type.



3. Press Ctrl+V.
The named elements are displayed in the diagram pane and are created in the model.



Copyright © 1998-2015 No Magic, Inc.

Selecting an Element

The element Selection dialog is used for selecting elements from available contents and adding them to a certain destination scope. The title of the dialog varies depending on the way it is invoked. Depending on the content, the element Selection dialog is used for selecting an element, diagram, or owner.

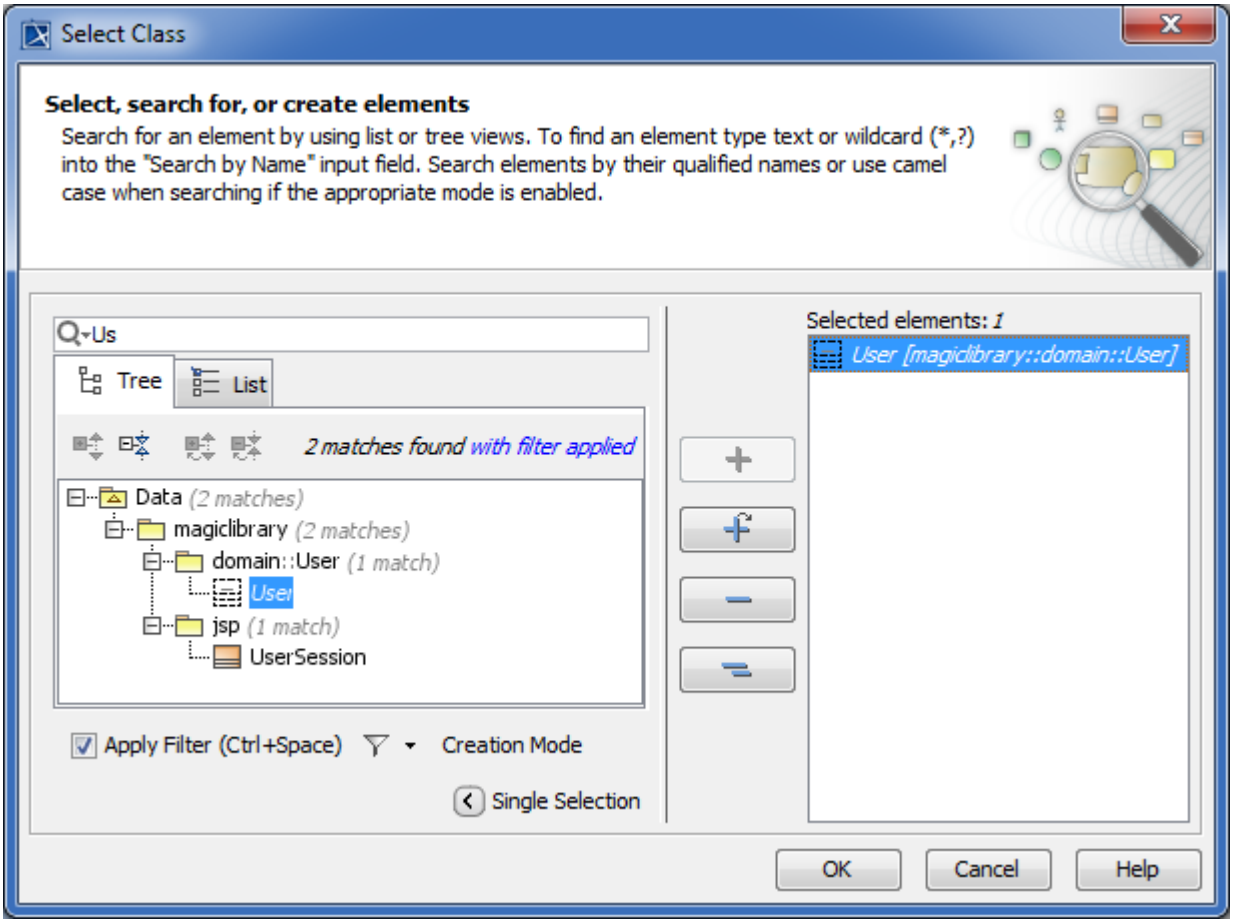


Figure 212 -- Element Selection dialog

The element Selection dialog opens then assigning a property value - the concrete element or elements. For example, the element Selection dialog opens then assigning the a base classifier in the **Class** Specification window.

In the element Selection dialog, you can perform the following actions:

- Select one or more elements to assign them as property values. For more information about selecting elements, see "Element Selection Views" on page 353 and "Elements Multiple Selection" on page 354.
- Search for an element. For more information about searching for elements, see "Searching for Elements in Element Selection Dialog" on page 355.
- Create new elements.
- Clone a selected element.

Related procedures

Searching for Elements in Element Selection Dialog

Creating New Elements

Cloning Selected Element

Copyright © 1998-2015 No Magic, Inc.

Related references

Element Selection Views

Elements Multiple Selection

Element Selection Views

There are two different views for elements selection:

- The **Tree** view displays the hierarchical structure of the elements and enables to create a new one.
- The **List** view displays the elements listed in the alphabetical order.

The **Library** view is displayed only if the UPDM plugin is used.

Element selection view	Description
 **Tree** view	The **Tree** view displays all selectable items and their owners. The following commands helps to manipulate in the Tree view: •  **Expand All / Collapse All** - all nodes are expanded/ collapsed recursively. •  **Expand selected recursively / Collapse selected recursively** - a selected node is expanded/collapsed recursively. • The **Load** button - click the **Load** button to load the selected used project which is not loaded. You can find those commands in the shortcut menu. Right-click an empty space near the model tree in the Selection dialog to open the shortcut menu.
 **List** view	The **List** view displays all the items that can be selected in a particular case. Elements are sorted in the alphabetical ascending order. The first 30 items are displayed in the list. Other items can be seen after clicking the **click here to show the remaining matches** at the end of the list.
Library view	Available only if the UPDM plugin is used. For more information about the Library view, see “Library Support” in UPDM Plugin UserGuide.pdf.

Related concepts

Selecting an Element

Related procedures

Searching for Elements in Element Selection Dialog

Creating New Elements

Cloning Selected Element

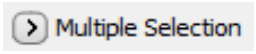
Related references

Elements Multiple Selection

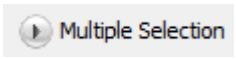
Copyright © 1998-2015 No Magic, Inc.

Elements Multiple Selection

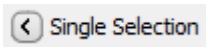


The **Multiple Selection**  button is available only when it is allowed to select more than one element.

To select more than one element

1. Click the **Multiple Selection**  button. The **Selected elements** area appears in the right side of the element Selection dialog.
2. In the model tree or list, select one or more elements at a time by holding the CTRL or SHIFT keys.
3. Move selected elements to the **Selected elements** area.

To return to the single selection

- Click the **Single Selection**  button. The first element that is in the **Selected element** list is selected in the **Tree** view.



When more than one element is selected in the tree or in the list in the multiple selection mode, the selection will be reduced to one element (the first one based on alphabetical order), when switching to the single selection mode.

The following buttons are available in the Multiple selection mode:

Command Name	Button	Description
Add		Adds the selected element(s) to the **Selected elements** list. **TIP!** You can also perform this action by double-clicking an element.
Add Recursively		Adds all the elements owned by selected element and by the elements owned by the selected element and so forth, to the **Selected elements** list.
Remove		Removes selected elements from the **Selected elements** list. **TIP!** You can also perform this action by double-clicking an element.
Remove All		Removes all the elements from the **Selected elements** list.

Related concepts

Selecting an Element

Related procedures

Searching for Elements in Element Selection Dialog

Creating New Elements

Cloning Selected Element

Related references

Element Selection Views

Copyright © 1998-2015 No Magic, Inc.

Searching for Elements in Element Selection Dialog

When searching for an element, type the search phrase in the **Search By Name** box. The following table describes the search types that can be used.

Search type	Description
Simple search in the List and Tree views	In the **Search By Name** box, type a phrase to search for in element names. Elements matching the search phrase will be displayed in the **List** or **Tree** view. When switching between element selection views, the selected elements are remembered. That is, if the element is selected in the **Tree** view, it will be selected in the **List** view as well.
Search using wild cards	In the **Search by name** box, type a phrase using the following wildcard characters: • "*" substitutes any range of characters. So if your search phrase is "*dd", all the elements containing "dd" in their names can be found. For example, "Address" and "ThreadDeath". • "?" substitutes exactly one character. So if your search phrase is "?dd", only the elements with "dd" in their names starting from the second character can be found. For example, "Address". "ThreadDeath" cannot be found in this case, because it contains more than one character before the search phrase "dd". Be sure the **Match Text Anywhere** command is unchecked, when using wildcard characters!

To optimize the search, you can customize search options. For this, click **NEW!** on the **Search By Name** box. The following table describes the commands for the search customization.

Command	Description
Match Text Anywhere	• To search for matching phrase in any part of element name, click the command to add the check mark. • To explicitly search from the beginning of element name, click the command to remove the check mark.
Use Camel Case	• To search for matching phrase typed in camel case, click the command to add the check mark. • To search for matching phrase typed in regular case, click the command to remove the check mark. You can use camel case as follows: • Type only the first letters (either capitalized or not) of each word, for example, "and" or "AND" to find "Analysis and Design". • Type skipping spacers, for example, "cv3" to find "CV-3". • Perform partial search, for example, "ibd" to find SysML Internal Block diagram. Be sure the **Match Text Anywhere** command is checked!
Search in Qualified Name	• To search for matching phrase in qualified names of elements, click the command to add the check mark. • To search for matching phrase in element names only, click the command to remove the check mark.

Copyright © 1998-2015 No Magic, Inc.

To narrow the search scope, you can customize scope filter options. For this, click **NEW!**  below the **Tree** or **List** view. Be sure the **NEW!** **Apply Filter** check box is selected! The following table describes the commands for the search scope filter customization.

Command	Description
Hide Uncommon Elements	To reduce the search results list by hiding uncommon elements, click the command to add the check mark. Elements, such as Interface, Association Class, Class, and Component, remain in the list.
Exclude Used Projects	To reduce the search results list by excluding elements from the standard/system profiles and used projects, click the command to add the check mark.
Search in Favorites	Select one or more favorite packages as the search scope. Be sure you have at least one package marked as favorite! For more information, see "Favorites" on page 416.
Remember Selected Options	To make the element Selection dialog remember the specified search scope filter options for the next time it opens, click the command to add the check mark. The same scope filter settings appear in autocompletion lists and the **Quick Find** dialog as well.

Creating New Elements

You can create a new element directly in the element Selection dialog. To create a new element, turn on the element creation mode. The **Tree** view is opened automatically, when the element creation mode is turned on.

To turn on the element creation mode in the element Selection dialog

Do one of the following:

- Click the **Creation Mode** button.
- Click the **Tree** tab and right-click the empty space near the model tree to open the shortcut menu. In the shortcut menu, click **Creation Mode**.

To create a new element in the element Selection dialog

1. Click the **Tree** tab and turn the element creation mode on.
2. In the model tree, select the owner for a new element and do one of the following:
   - Click the **Create** button.
   - Right-click the selected element or empty space near the model tree to open the shortcut menu. In the shortcut menu, **Create**.
3. The list of elements available to create appears. Select the element you want to create.



The list of elements available to create elements is generated automatically according to the particular situation.

4. The Specification window of a new element opens. Specify desired element properties and click **Close**. The new element is included in the model tree.

Related concepts

Selecting an Element

Related procedures

Searching for Elements in Element Selection Dialog

Copyright © 1998-2015 No Magic, Inc.

Cloning Selected Element

Related references

Element Selection Views

Elements Multiple Selection

Cloning Selected Element

In the element Selection dialog, you can clone an existing element. It means you can create a new element on the basis of the selected element having the same data as the selected element.

To clone a selected element in the element Selection dialog

1. Click the **Tree** tab and turn on the element creation mode. See the procedure "To turn on the element creation mode in the element Selection dialog" on page 356.
2. In the model tree, select the element which you want to clone and do one of the following:
   - Click the **Clone** button.
   - Right-click the selected element or empty space near the model tree to open the shortcut menu. In the shortcut menu, click **Clone**.
3. The Specification window of a new element opens with all properties derived from the cloned element. By default, the number is added to the element name.
4. Change or define desired properties and click **Close** when you are done. The new element is included to the model tree.

Related concepts

Selecting an Element

Related procedures

Searching for Elements in Element Selection Dialog

Creating New Elements

Related references

Element Selection Views

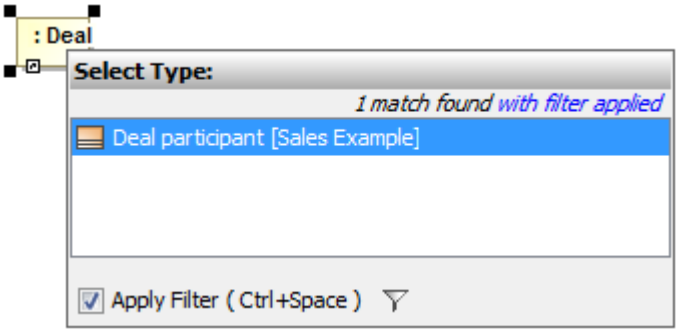
Elements Multiple Selection

Selecting Port and Part Type

The **Type Selection Mode** function turns on the automatic port and part type selection. When the function is turned on:

Copyright © 1998-2015 No Magic, Inc.

- On a part creation, the **Select Type** dialog appears. The same is valid when creating ports.



- If a part does not have a specified type, then on a nested port creation, a new type for the part will be created. A type for the new port will be created too.
- If a part is without name, type the name on the part shape on the diagram and the type will be created automatically. The typed name will be the name of the part type (note, that part will be without name).

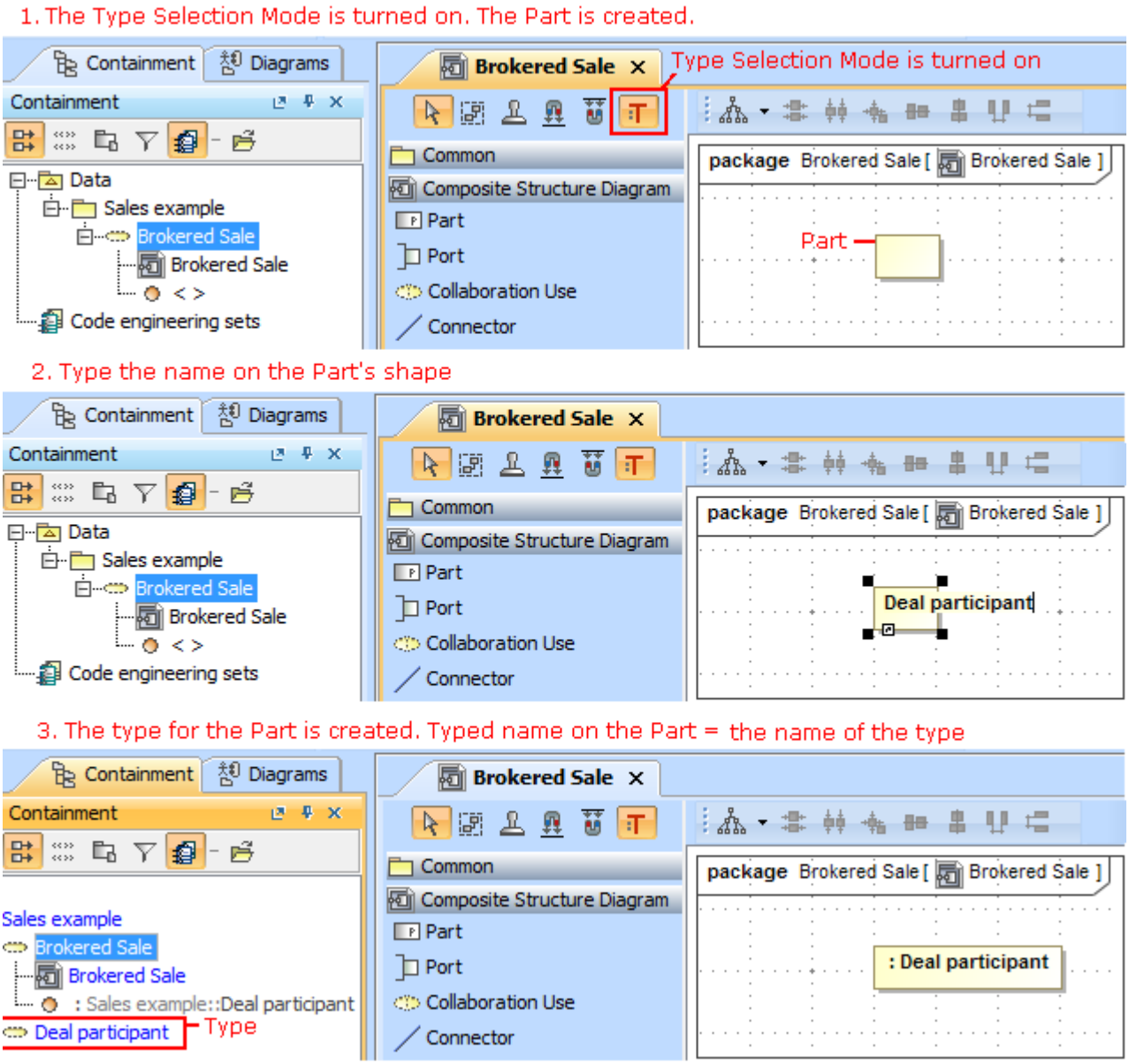


Figure 213 -- Example of typing name on part when Type Selection Mode is turned on

Copyright © 1998-2015 No Magic, Inc.

To turn on the type selection mode

- On the Composite Structure, **NEW!** Class, **NEW!** Component, or **NEW!** Package diagram pallet, click the **Type Selection Mode** button .

Refactoring

Refactoring allows you to modify model elements including all related data. Within refactoring, you can convert element types, replace elements, decompose diagrams, and perform other actions.

All the refactoring commands have been moved under the **Refactor** command in the shortcut menu of the particular element. Now the refactoring covers the following commands:

- **Convert To**. For more information, see "Converting Elements" on page 360.
- **Replace With**. For more information, see "Replacing Elements" on page 363.
- **Reverse Direction**. For more information, see "Reversing a Relationship Direction" on page 365.
- **Extract**. For more information, see "Extracting" on page 368.
- **Split Control Flow** for the selected Control Flow edge.
- **Split Object Flow** for the selected Object Flow edge.
- **Join Connector** for the selected split Control Flow or Object Flow edge.
- **Swap**. For more information, see "Swapping" on page 366
- **Redefinition**. For more information, see "NEW! Redefinition" on page 368 .

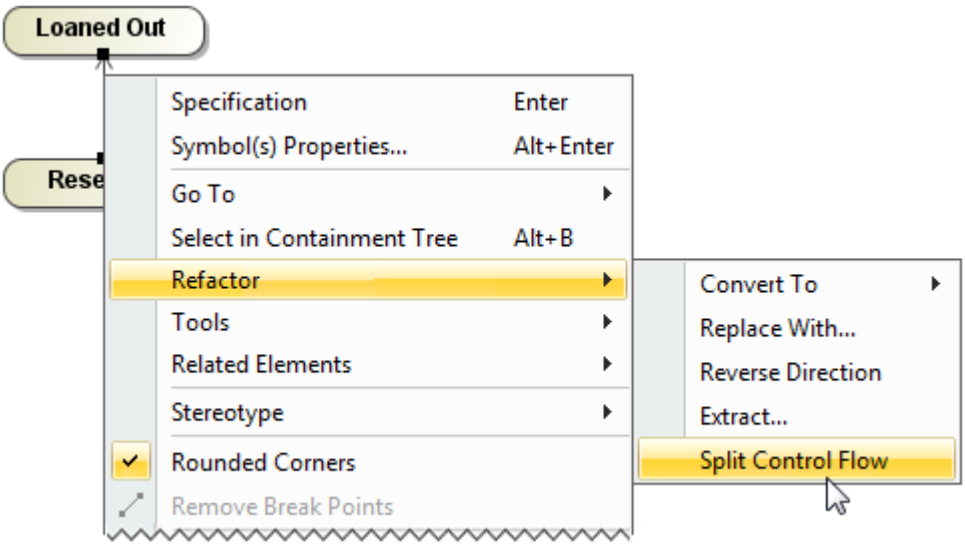


Figure 214 -- Example of Refactor command

To perform refactoring

1. Select an element in the Model Browser or on the diagram pane.
2. Open the selected element's shortcut menu.
3. Select **Refactor** and then one of commands available for the selected element.

NOTE Some of the **Refactor** subcommands are available only in element shortcut menu invoked from the diagram pane (not available in the Model Browser).

Copyright © 1998-2015 No Magic, Inc.

Related procedures

Converting Elements

Replacing Elements

Reversing a Relationship Direction

Component Diagram

Converting Elements

Original element

The original element is the element that will be converted. In other words, it is the conversion source.

Conversion target

The conversion target is the element type, to which the original element is going to be converted.

Converted element

The converted element is the element that type has been changed during the conversion.

Related elements

The related elements are elements having relations to the original element. Related elements can be the following:

- If the original element is a relationship, elements connected by this relationship are considered as related element to relationship.
- If the original element is not a relationship, relationships going from or to the original element are the related elements (such as an association, dependency, generalization, and others).
- Other related elements associated to the original element. They are properties of the original element, such as inner elements, types, and others.

Compatible properties

The compatible properties are original element specification that are compatible with the conversion target. The term “properties” includes related elements, inner elements, and all other elements listed in the element Specification window.

Compatible means that the converted element can have or own the particular property or properties of the original element.

Incompatible properties

Incompatible properties are original element specification that are incompatible with the conversion target. Incompatible means that the converted element can NOT have or can NOT own the particular property(-ies) of the original element.



For more information, see "Incompatible properties and related elements" on page 362.

Element conversion allows converting one element type to another. To be more specific, the element conversion functionality allows the UML model element conversion from one metaclass to another. For example, a class element can be converted to a use case, component, interface, or other. All compatible

Copyright © 1998-2015 No Magic, Inc.

properties and all compatible related elements are moved from the source element to the converted element during the conversion.

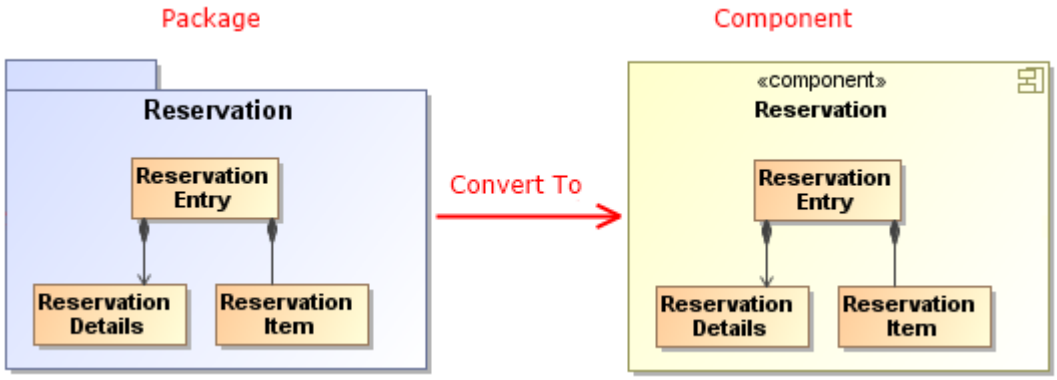


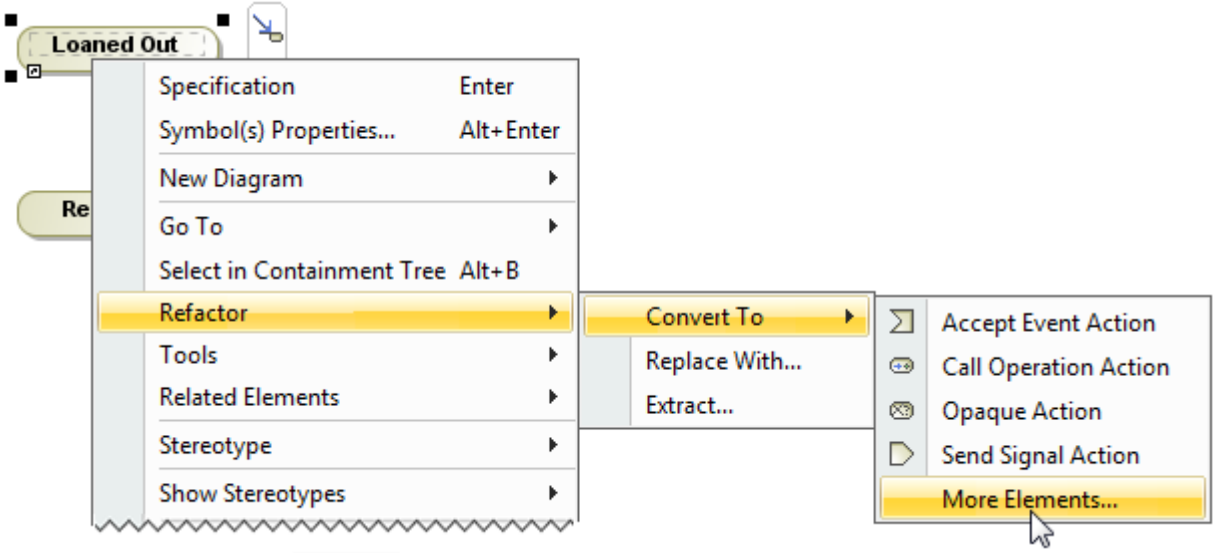
Figure 215 -- Example of Package conversion to Component

TIP!

- Relationships conversion is available.
- Multiple elements can be selected on the diagram or in the Model Browser to convert them to other types.

To convert elements

1. Select a symbol on the diagram pane or an element in the Model Browser.
2. Open the selected element(s) shortcut menu and click **Refactor** > **Convert To**. The list with available conversion target types opens.



3. Do one of the following:
   - Select the element type from the commonly used element type list. The element is converted.
   - Click **More Elements** to see the whole list of available conversion targets. In the opened **More Elements...** dialog, click the element type you need and click **OK**. The element is converted and selected element type is included as commonly used element type.

Copyright © 1998-2015 No Magic, Inc.

TIP!

As of the MagicDraw version 17.0.2, the following commands have been moved under the **Refactor** command in the shortcut menu of the particular element:

- **Convert To** > **Output Pin** for the selected Object Node or Input Pin.
- **Convert To** > **Input Pin** for the selected Object Node or Output Pin. For more information, see
- **Convert To** > **Object Node** for the selected Input Pin or Output Pin. For more information, see "To convert a pin to an object node" on page 964.
- **Convert To** > **Attribute(s)** for the selected Association that has defined roles, that is, roles have names. For more information, see "To show association ends as attributes on linked class shapes" on page 843.
- **Convert To** > **Association(s)** for the selected Attribute. **NOTE**: The Association(s) command is available only if the attribute(s) type is specified.

NOTE

- If the **Refactor** command is unavailable, make sure you have permissions to edit the selected element or related element(s).
- Use **Undo** to restore the original element and its data.

Related concepts

Refactoring

Incompatible properties and related elements

After the conversion, a converted element owns the same compatible specification properties and are connected with the same related elements as the original element was connected. The incompatible properties of the original element are lost, that is, are deleted from the project.

Incompatible means that the converted element can NOT own particular properties or can NOT be associated with particular related elements of the original element.

For example, let's convert the component to the package. See an example in the following figure.

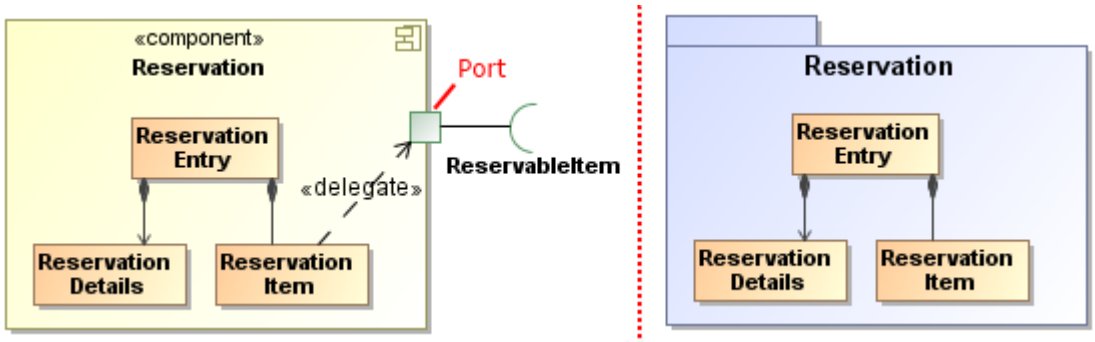


Figure 216 -- Example of conversion when original and converted elements has incompatible property (Port)

As you see, the component has the assigned port. According to UML guidelines, a package cannot own ports. So, in this case, ports are incompatible elements with the packages. After the conversion, the package, that is, a converted element, has no the assigned port. The port is deleted from the project.

To prevent from losing model data, you always get a message with a warning that incompatible properties will be lost after the element conversion.

Related concepts

Refactoring

Related procedures

Converting Elements

Copyright © 1998-2015 No Magic, Inc.

Changes after conversion

Converting a property to a port

If the property visibility is *private*, *protected* or *package*, then after the conversion visibility of the created port will be set to public.

Related concepts

Refactoring

Related procedures

Converting Elements

Replacing Elements



This functionality is available in Standard, Professional, Architect, and Enterprise editions.

Original element

The original element is the element that is going to be replaced. In other words, the replacement source.

Replacement target

The replacement target is the element with which the original element is going to be replaced.

Replaced element

The replaced element is the element to which the original element has been changed during the replacement.

Element properties

The element properties are so called the element specification properties (specified in the element Specification window) and the element symbol properties (defined in the element **Symbol Properties** dialog).

Inner elements

Inner elements are the elements stored inside the element (that is, owned by that element), for example, the class as inner package element.

You can replace one model element with another of the same type (metaclass) element. The model element replacement is useful when during the modeling process you notice that one model element needs to be replaced with another. All relations and references of the original element are transferred to the newly replaced element.

Copyright © 1998-2015 No Magic, Inc.

After the replacement, an original element will be replaced with a replacement target:

- The replacement target will be displayed in all diagrams instead of the original element.
- All references to the original element will be replaced by references to the replacement target.
- The replacement target after the replacement will have all the relationships of the replacement target and the original element.



Please note that on a diagram relationships to and from the replacement target element does not display automatically after the replacement.

To display all the relationships of the replacement target on the diagram pane, you can use the Display path function. For more information, see "To display paths among shapes that already exist in the model" on page 509.

- The original element will be deleted from the project.
- The original element properties and inner elements will be deleted from the project.

See the replacement example in the following figure.

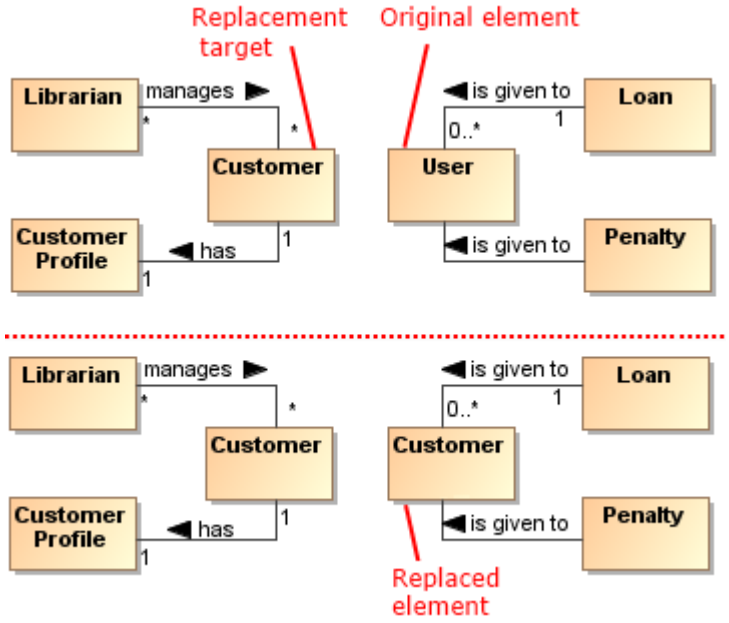


Figure 217 -- Model before element replacement (at the top) and model after element replacement (at the bottom)

The preceding figure illustrates the replacement of the *User* class instead of the *Customer* class.After the replacement the *User* class has been removed and the *Customer* class has been created instead of it.

To replace one element with another

1. Select a symbol on the diagram pane or an element in the Model Browser and open its shortcut menu.
2. Click **Refactor** > **Replace With**. The warning message appears:
   - Click **Yes** to continue the element replacement. The properties and inner elements of the original element will be lost.
   - Click **No** to cancel the element replacement. The selected element will not be replaced.

If you decided to continue the replacement, the element Selection dialog opens.

3. In the element Selection dialog, select the element with which you want to replace the original element. For more information about the element Selection dialog, see "Selecting an Element"

Copyright © 1998-2015 No Magic, Inc.

on page 352.

4. Click **OK**. The original element is replaced with the target element.

NOTE

Use **Undo** to restore the original element and its data.

Related concepts

Refactoring

Reversing a Relationship Direction

NOTE

This functionality is available in Standard, Professional, Architect, and Enterprise editions.

The direction of the particular relationship can be reversed. You can reverse the direction of the transition, control flow, object flow, and other relationships.

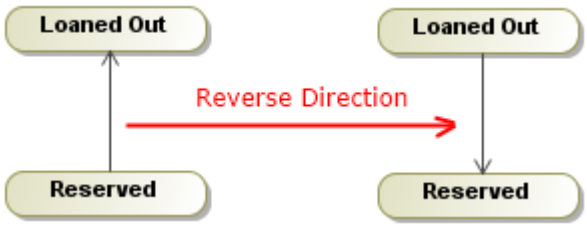


Figure 218 -- Example of direction reverse

The relation can be reversed if:

- The selected relationship and elements connected by this relationship are editable.
- The source and target elements are compatible, that is, they can be counterchanged.

To reverse a relationship direction

1. Select the relationship in the Model Browser or the relationship's path on the diagram pane.
2. From the element's shortcut menu, select **Refactor,** and then select **Reverse Direction**. The direction of the relationship changes.

NOTE

The direction of the information flow will not be changed on the direction reverse action. Additionally, in some cases when the information flow is not valid anymore, the information flow will be removed from the project after the direction reverse action.

For more information about the information flow, see "Information Flow" on page 900.

Related concepts

Refactoring

Copyright © 1998-2015 No Magic, Inc.

Swapping

Swapping allows counter-change symbols with each other on the diagram.

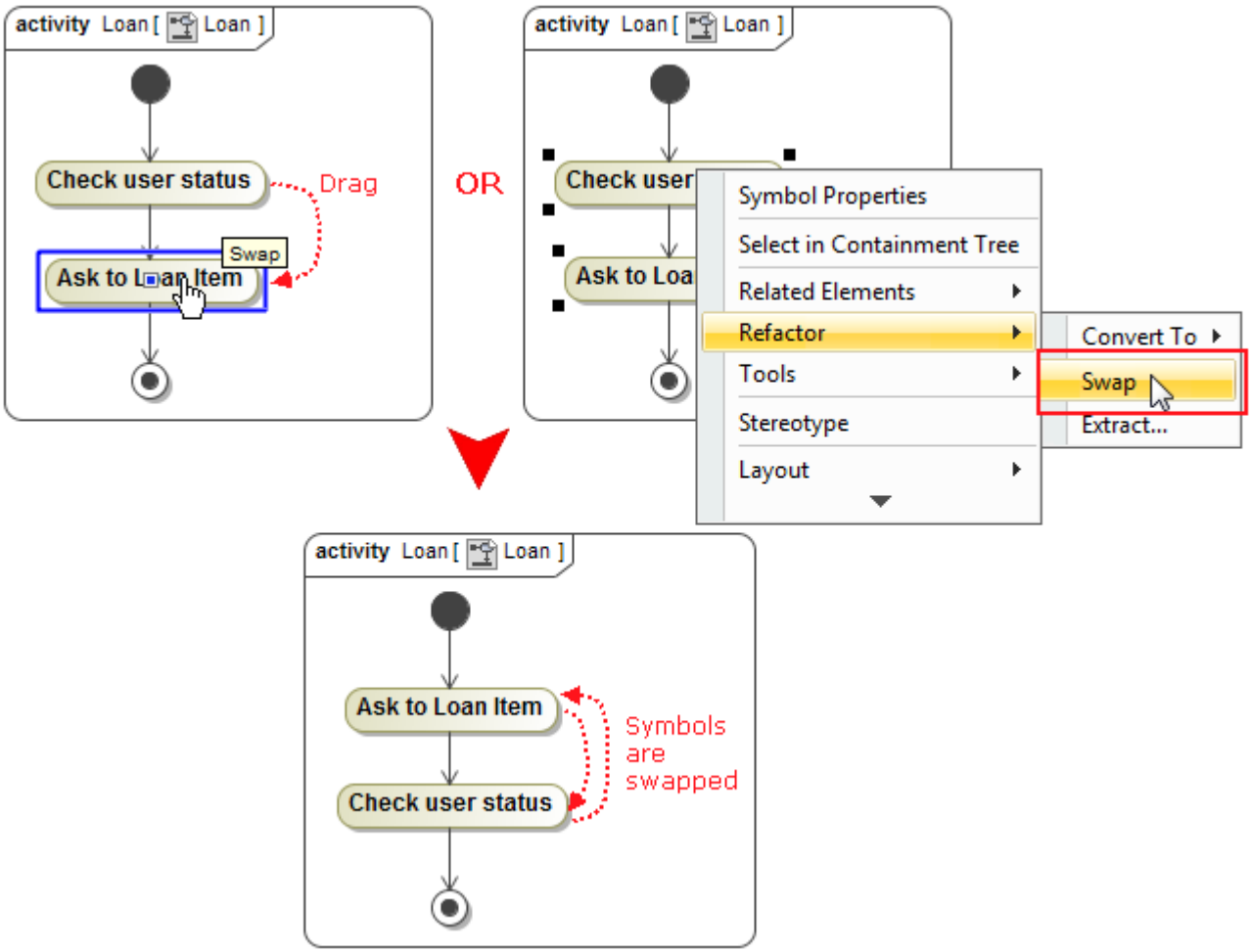


Figure 219 -- Example of swapping actions in Activity diagram

You can swap the symbols in the following diagrams:

- In the Activity diagram, you can swap *actions*, *merge* or *decision nodes*, *join* or *fork nodes* with each other, if the symbols are connected through **control flow**.
- In the State diagram, you can swap *states* or *pseudo states* (*initial state*, *deep history*, *join*, *fork*, *junction*, *choice*, *entry point*, etc.) with each other, if symbols are connected through **transition**.

To swap elements

Do one of the following:

- Drag-and-drop one shape onto another.

NOTE

Swap does not work when dragged shape can be added into the target shape (or in other words, can be added as a child).

- On the diagram, select the two shapes that you want to swap, right-click, and from the shortcut menu, select **Refactor** > **Swap**.

NOTE

- Swapping does not work when symbols have different parents.
- If you cannot swap within a teamwork project, make sure you have the right to edit the model of this project and then try to lock for edit this diagram.

Copyright © 1998-2015 No Magic, Inc.

In the following example, you can see the cases when you can swap the elements and the cases when you cannot swap the elements.

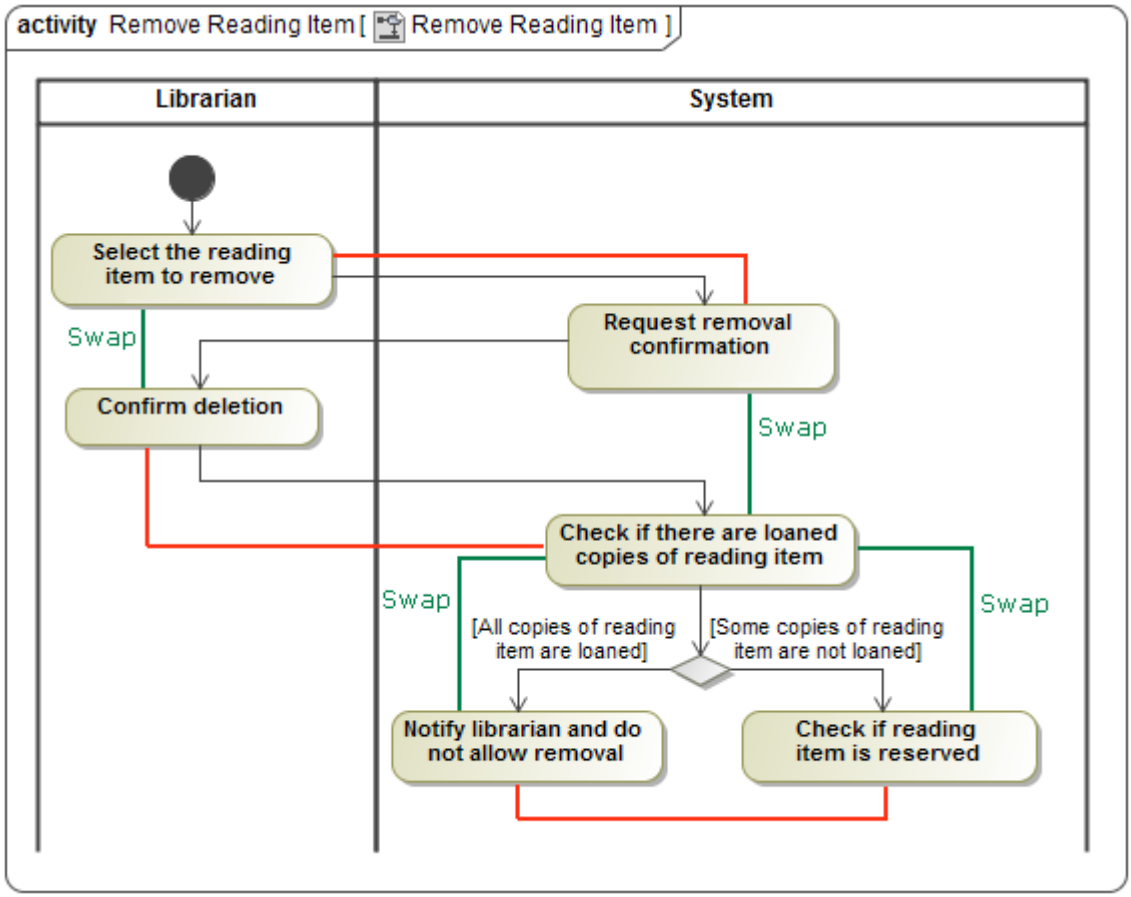


*Figure 220 -- Example of swapping. The green lines are drawn between the elements that you **can** swap, and the red lines are drawn between the elements that you **cannot** swap.*

Rules	Example
You **can** swap the elements that are in the same branch	You **can** swap the following elements: • "Select the reading item to remove" with "Confirm deletion" • "Request removal confirmation" with "Check if there are loaned copies of reading item" • "Check if there are loaned copies of reading item" with "Notify librarian and do not allow removal" • "Check if there are loaned copies of reading item" with "Check if reading item is reserved"
You **cannot** swap the elements that have different parents	You **cannot** swap the following elements: • "Select the reading item to remove" with Request removal confirmation, because one element is in the "Librarian" partition and the other is in the "System" partition. • "Confirm deletion" with "Check if there are loaned copies of reading item" because one element is in the "Librarian" partition and the other is in the "System" partition.
You **cannot** swap the elements that are in different branches	You **cannot** swap the following elements: • "Notify librarian and do not allow removal" with "Check if reading item is reserved".

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Refactoring

Related diagrams

Activity Diagram

State Machine Diagram

NEW! Redefinition

Any inherited part can now be quickly redefined directly in the composite structure diagrams. A new part redefining the original one will be created and graphically replaced in the diagram with all ports and connectors remaining in their places.

To redefine an inherited part

1. Select a part and from the shortcut menu, choose **Refactor** > **Redefine To**.
2. Select one of the following:
   - **New Type**, if you want to create a new type
   - One of the existing types

Extracting



Extracting feature is available in Architect and Enterprise editions.

The extraction functionality allows moving a selected part of a diagram to a newly created diagram. A referencing element will be created in place of the extracted element(s) in the source diagram.

In order to make the diagrams more readable and usable, you can extract a specific part of a diagram to another diagram. By using the extraction, you can reuse that part of the diagram to simplify a complicated system.

Definitions

Source diagram

The source diagram is the diagram from which the elements are to be extracted.

Target diagram

The target diagram is the newly created diagram resulting from the extraction. The selected elements are moved from the source diagram to the target diagram.

Extracting

Extracting means moving selected elements from one location to another. In other words, the selected elements are deleted from the source diagram and moved to the newly recreated target diagram.

Referencing element

A referencing element is created as the result of the extraction. The referencing element is created in the source diagram and represents the elements moved to the target diagram during the extraction.

The referencing element contains a reference to the target diagram. Therefore, double-clicking the referencing element on the diagram pane or in the Model Browser will open the target diagram.

Copyright © 1998-2015 No Magic, Inc.

Intersected relationship

An intersected relationship is the relationship selected in the source diagram. The intersection appears when the selection intersects the relationship - one end element of the relationship is selected, but not both.

You can use the extraction functionality in the following diagrams:

- Extracting in Sequence Diagram
- Extracting in Activity Diagram
- Extracting in State Machine diagram
- Extracting in Composite Structure diagram

To extract a selected part from a diagram, on the diagram select a part that you want to extract, and then on the shortcut menu, click **Refactor** > **Extract**. An extract wizard specific to the particular type of diagram then opens.

Related diagrams

Sequence Diagram
State Machine Diagram
Activity Diagram
Composite Structure Diagram

Extracting in Sequence Diagram

The extraction functionality allows extracting a selected part of a sequence diagram to a newly created diagram. After extraction, an interaction use is created in the source diagram in place of the extracted part(s). That is, the interaction use is a referencing element that references the newly created sequence diagram.

Copyright © 1998-2015 No Magic, Inc.

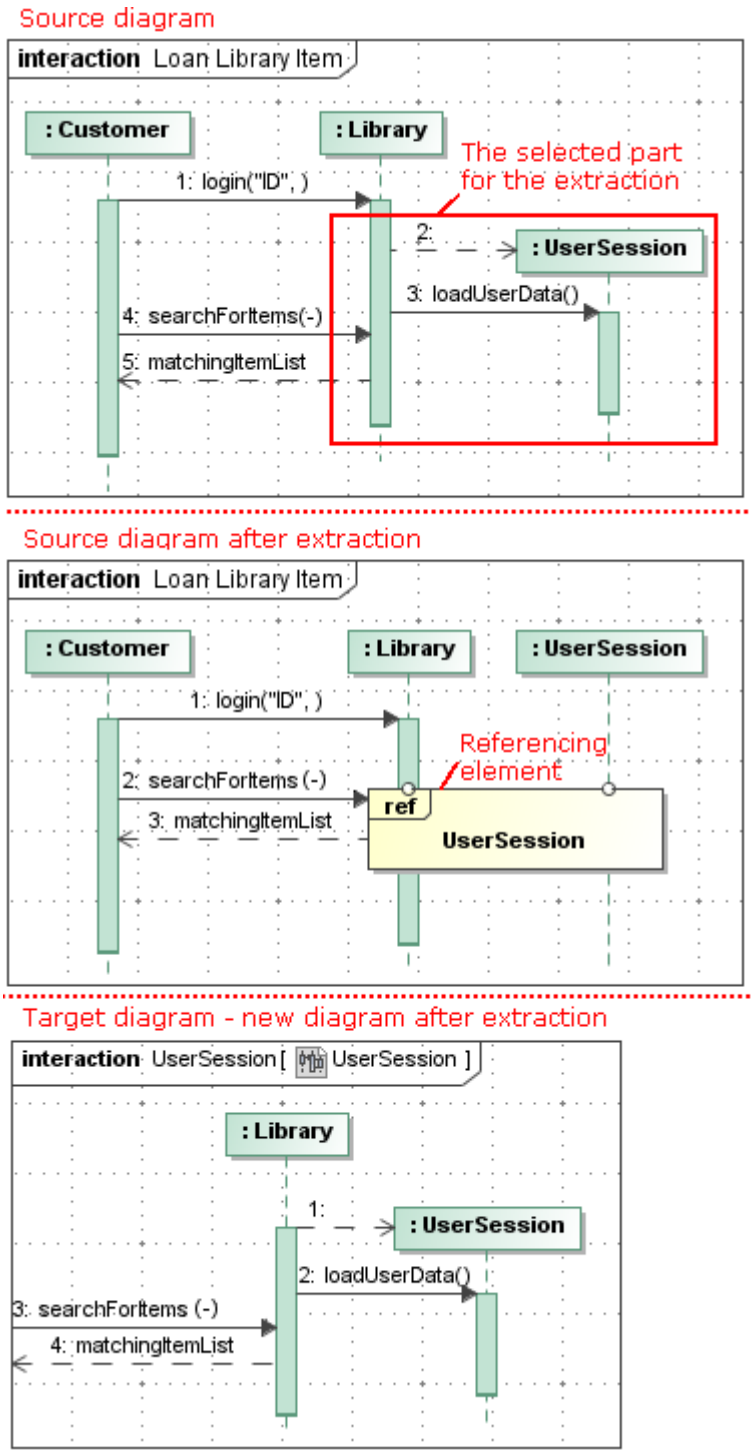


Figure 221 -- Example of extraction in Sequence diagram

As you can see in the example, during the extraction, the selected part of the sequence diagram is moved to the newly created *UserSession* sequence diagram. The interaction use element that references the *UserSession* interaction is created in the source sequence diagram.

Related concepts

Extracting

Related references

Rules of extracting in Sequence diagram

Copyright © 1998-2015 No Magic, Inc.

Extract Sequence Wizard

Sequence Diagram

Rules of extracting in Sequence diagram

Outgoing intersected messages are not selected for the extraction. The exception to this rule is the reply message. You can see an example in the following figure.

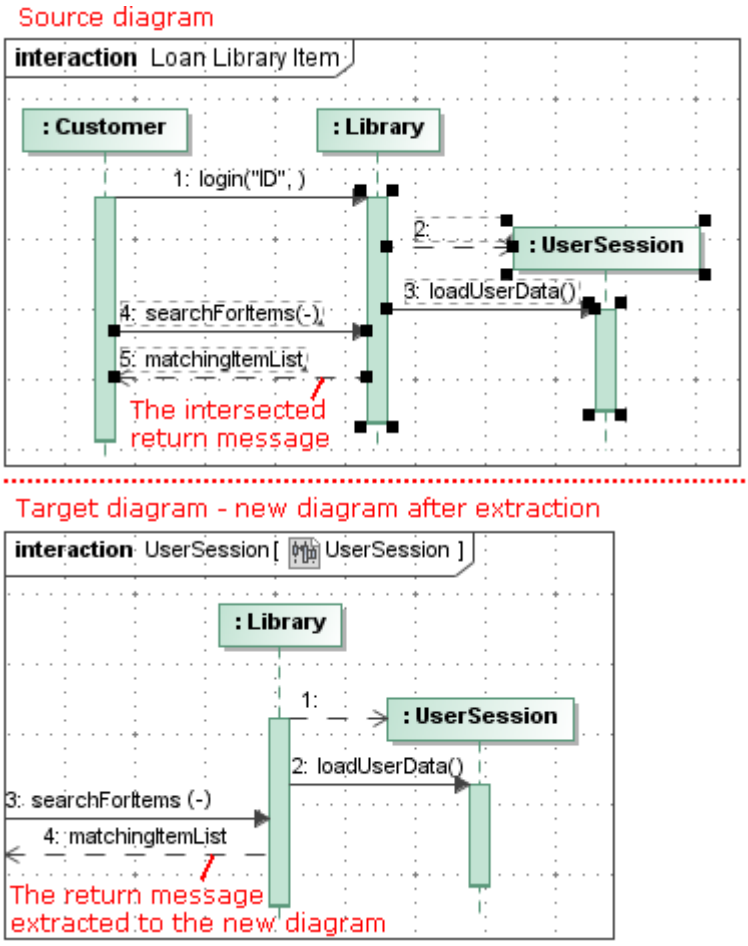


Figure 222 -- Example of the return message extraction

Related concepts

Extracting

Related references

Extracting in Sequence Diagram

Extract Sequence Wizard

Sequence Diagram

Extract Sequence Wizard

The **Extract Sequence Wizard** is used to extract a selected part of a sequence diagram to a separate diagram. Using this wizard, you can:

- Create and specify the new element wherein the extracted part will be moved.
- Select the intersected incoming messages for which the formal gates will be created, and that will be created in the new target diagram.

Copyright © 1998-2015 No Magic, Inc.

To start the Extract Sequence Wizard

1. On an sequence diagram, select a part that you want to extract.
2. From the selected part's shortcut menu, select **Refactor** > **Extract**. The **Extract Sequence Wizard** opens.

The **Extract Sequence Wizard** consists of two steps:

1. Specification of a new element
2. Selection of messages

STEP #1. Specification of a new element

In this step, you need to specify a diagram into which the extracted part will be moved and an owner wherein this new diagram will be created. You can also define the type of the newly created element that will own the diagram. The interaction element and the sequence diagram type are selected by default. The name of the sequence diagram will be the same as the specified name of the interaction element.

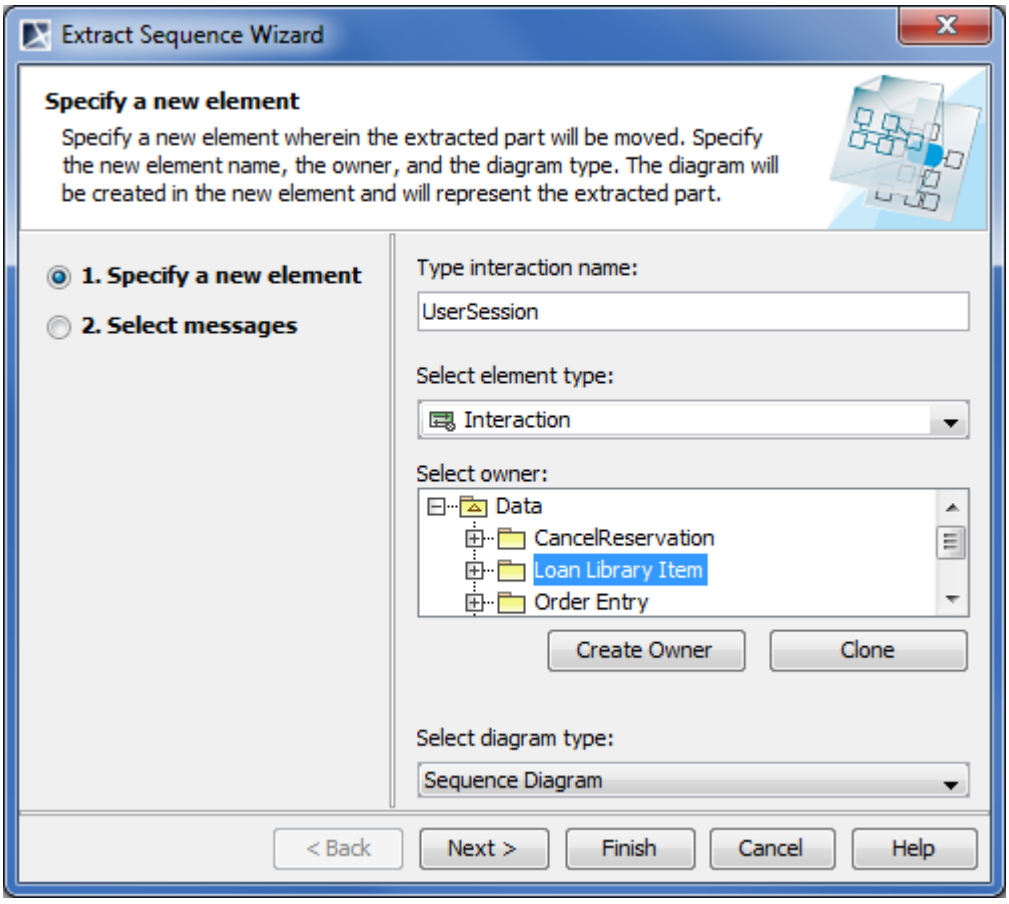


Figure 223 -- Extract Sequence Wizard. Specify a new element

Wizard element name	Element type	Description
Type interaction name	Text box	Type the name of the new interaction, wherein the extracted part will be stored and the new sequence diagram will be created. The newly created sequence diagram name will be the same as the interaction name.

Copyright © 1998-2015 No Magic, Inc.

Wizard element name	Element type	Description
Select element type	Drop-down list	Open the list and select the element type in which the extracted part and new diagram will be created. The default element type is an interaction. By using the customization plugin, you can add custom elements to the element type list.
Select owner	Box	In the model tree, select a package, profile, or model as an owner of the newly created element. You may also: • Create a new owner. Click the **Create Owner** button, select an element type, and specify its properties. • Clone an existing package, profile, or model. To clone, in the model tree, select an owner, click the **Clone** button, and specify properties of the cloned element.
Select diagram type	Drop-down list	Open the list and select the diagram type for the extracting part. The default diagram type is a sequence diagram. You can also select to create a new time diagram. By using the customized plugins, you can add custom diagrams to the diagram type list.

After you have specified the new element name, the element type, an owner, and the type of the new diagram, do one of the following:

- Click **Next**, if you want to select intersected messages that you want to create in the new target diagram. The next wizard step opens.
- Click **Finish**. The selected part of a sequence diagram is extracted to a new diagram. The second wizard step is skipped, and the messages that by default are selected in this step, will be checked.

Step #2. Selection of messages

The incoming intersected messages selected in the source diagram are listed in this step. Select the messages that will be extracted to the new target diagram. The formal gates will be created for the selected messages.

You can see an example of intersected connectors in Figure 221 on page 370.



- The **Select messages** list is empty unless there are intersected incoming messages that are selected in the source sequence diagram.
- If there are intersected outgoing replay messages when they are listed in the **Extract Sequence Wizard**, the second step. For more information, see "Rules of extracting in Sequence diagram" on page 371.
- For more information about intersected messages, see concepts in "Extracting" on page 368.

Copyright © 1998-2015 No Magic, Inc.

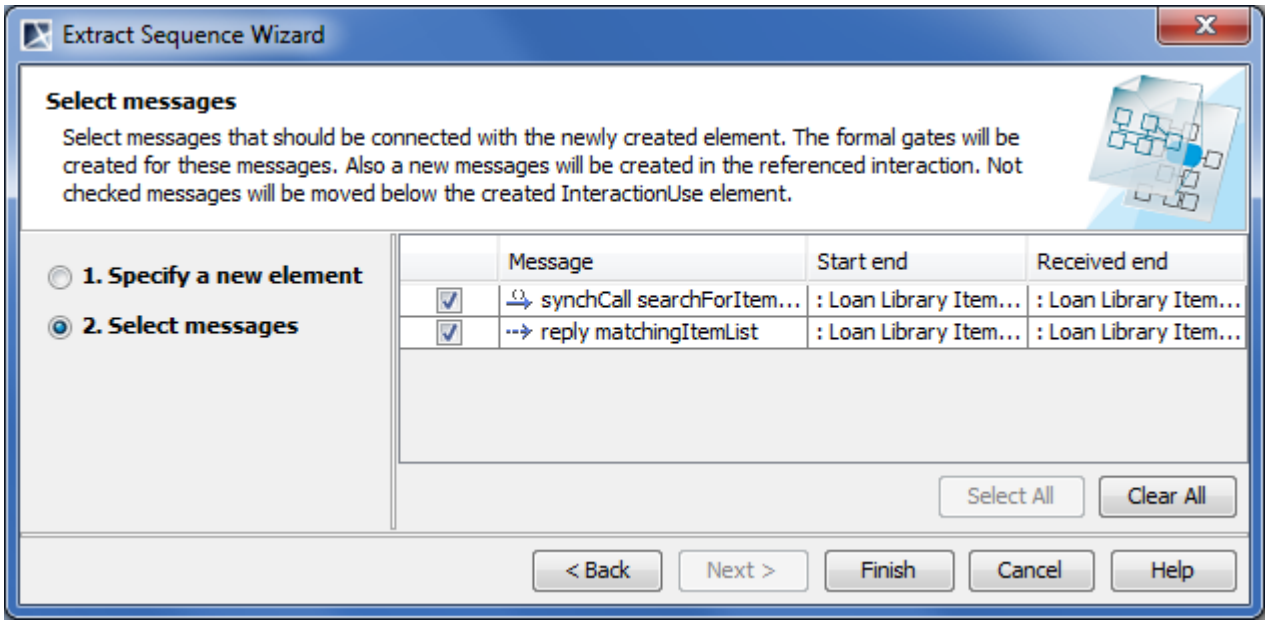


Figure 224 -- Extract Sequence Wizard. Select messages

Wizard element name	Element type	Description
☑	Check box	The check box determinates whether or not an intersected message will be extracted. Unchecked messages will be moved bellow the interaction use (the referencing element) in the source diagram.
Message	Table	The name of the message is listed in the **Message** column. The column is not editable.
Start end	Table	In the **Start end** column, there is the element from which the message is created.
Received end	Table	In the **Receive end** column, there is listed the element to which the message is connected.

After you have selected the messages, do one of the following:

- Click **Back** to return to the first step.
- Click **Finish**. The selected part of the sequence diagram is extracted to a new diagram.

Related concepts

Extracting

Related references

Extracting in Sequence Diagram

Rules of extracting in Sequence diagram

Sequence Diagram

Copyright © 1998-2015 No Magic, Inc.

Extracting in Activity Diagram

The extraction functionality allows extracting a selected part of an activity diagram to a newly created diagram. After extraction, in the source diagram, a call behavior action is created instead of the extracted part. That is, the call behavior action is a referencing element that references the newly created activity diagram.

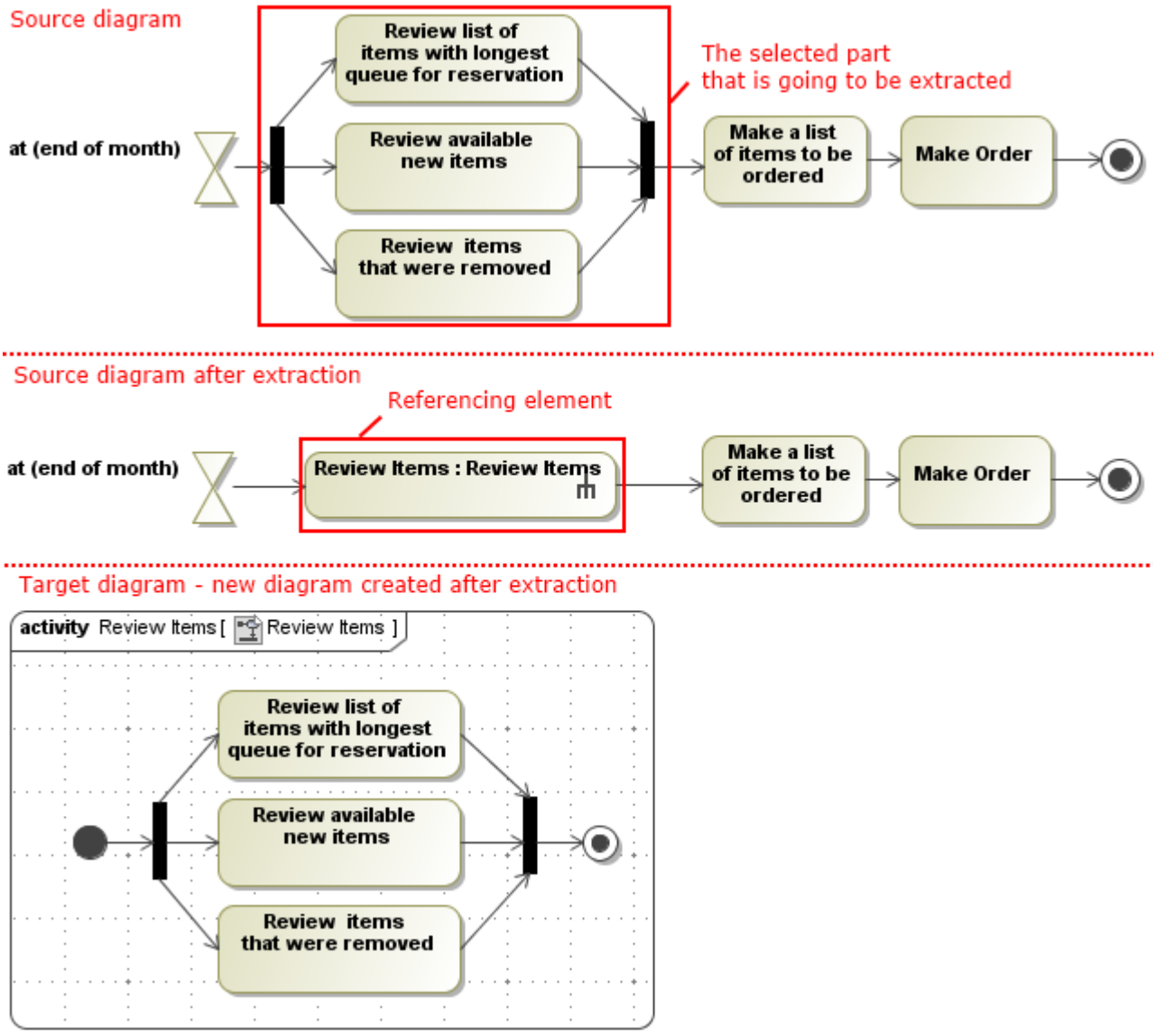


Figure 225 -- Example of extraction

As you can see in the example, during extraction, the selected part of the activity diagram is moved to the newly created *Review Items* activity diagram. In the source activity diagram, a *Review Items* call behavior action is created in place of the selected elements.

Related concepts

Extracting

Related references

Extraction rules in Activity diagram

Extract Activity Wizard

Activity Diagram

Copyright © 1998-2015 No Magic, Inc.

Extraction rules in Activity diagram

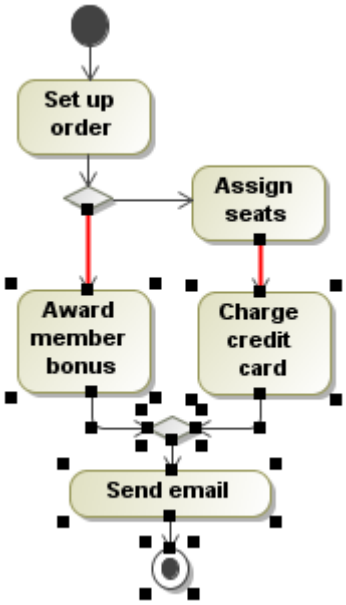
Extracting rules are the rules that should be followed in order to proceed with the correct extraction. For example, there can be invalid selections in a source diagram. Or there are unsupported elements that cannot be extracted - that is, cannot be moved from the source diagram to the new target diagram. Without knowing the rules you will not be able to proceed with extraction (you will have to change your selected elements and/or reselect elements according to the rules) or you may lose some data. Here are the extraction rules:

- Rule 1. Only one incoming and/or outgoing control flow is allowed for a valid diagram extraction.
- Rule 2. The selected elements should exist in the same owner, action, or activity partition.
- Rule 3. Any intersected exception handlers will not be extracted to a new diagram.

Rule 1. Only one incoming and/or outgoing control flow is allowed for a valid diagram extraction

If the selected part of an activity diagram has two or more intersected incoming or outgoing control flows, the extraction is not valid.

Let's analyze an example of the extraction when two incoming control flows are selected in the activity diagram as depicted in the following figure.



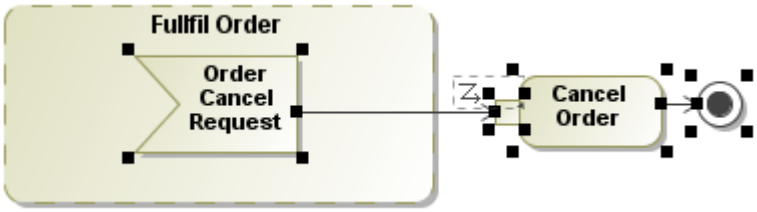
After you start the extraction, you are warned that only one incoming control flow and only one outgoing control flow is allowed. Click the **OK** button to close the message and then change the selected elements in the activity diagram.

Copyright © 1998-2015 No Magic, Inc.

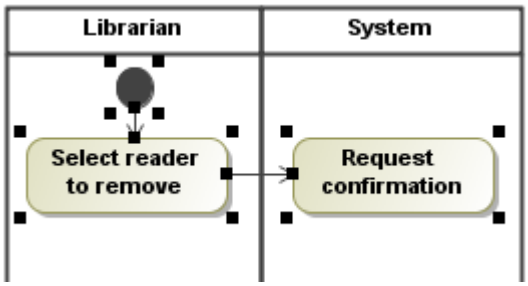
Rule 2. The selected elements should exist in the same owner, action, or activity partition

The extraction will be invalid if:

- The selected elements do not belong to the same owner. For example, if one of the selected actions is located in a structured activity node and the other does not belong to the same structured activity node. See the example in the following figure:



- Selected elements are in different activity partitions. See an example in the following figure.

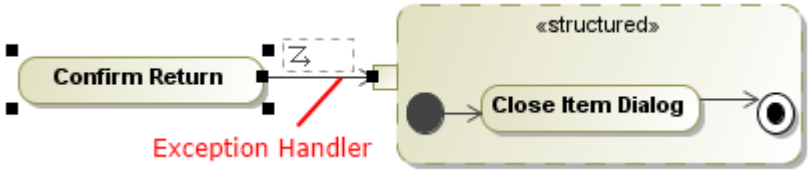


If an attempt is made to extract using an invalid selection, a warning will be displayed. A valid selection of elements will need to be made in order to proceed with the extraction.

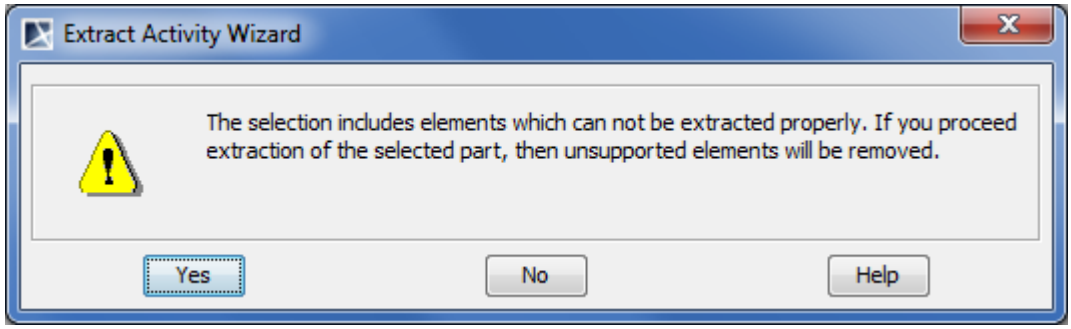
Rule 3. The intersected exception handler is not extracted to a new diagram

Let's analyze an example of the extraction when the intersected exception handler is selected in the source activity diagram.

1. In the source activity diagram, select an exception handler along with an action (see the following figure).



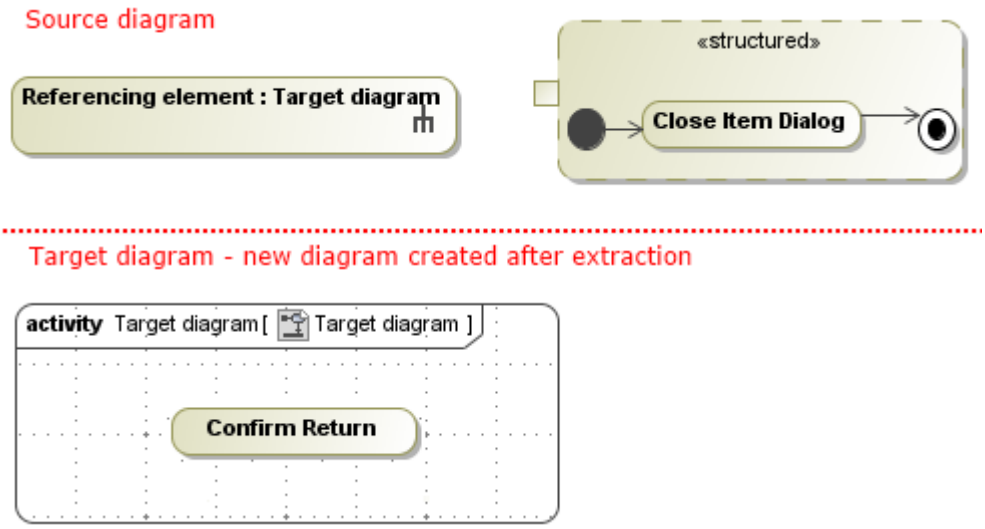
2. Start the extraction. The following message is displayed.



3. Click **Yes** to continue with the extraction. See the samples of the extraction result in the following figures. After the extraction the referencing element is created in the source activity diagram in place of the selected exception handler and the *Confirm Return* action. The path of

Copyright © 1998-2015 No Magic, Inc.

the exception handler is not represented in the new target diagram because in this case the exception handler does not have the two symbols between which it could be drawn.



NOTE

- Click **No** to close the message. The extraction is canceled and you can select elements for extraction again.
- Use **Undo** to restore the original element and its data.

Related concepts

Extracting

Related references

Extract Activity Wizard

Activity Diagram

Extract Activity Wizard

The **Extract Activity Wizard** is used to extract a selected part of an activity diagram to a separate diagram. Using this wizard, you can:

- Create and specify the target diagram that will be created as the result of the extraction. The selected part will be extracted to the newly created diagram.
- Select the parameters that will be created in a new diagram. The parameters are listed for each intersected object flow from the source diagram.
- Specify a referencing element that is created in the source activity diagram and represents the elements moved to the target diagram during the extraction.

To start the Extract Activity Wizard

1. On an activity diagram, select a part that you want to extract.
2. From the selected part's shortcut menu, select **Refactor** > **Extract**. The **Extract Activity Wizard** opens.

This section describes components of the **Extract Activity Wizard**, such as text boxes and buttons. The **Extract Activity Wizard** consists of three steps:

1. Specification of a new element
2. Creation of parameters
3. Creation of a reference element

Copyright © 1998-2015 No Magic, Inc.

STEP #1. Specification of a new diagram

In this step, you need to specify a diagram into which the extracted part will be moved and a package wherein this new diagram will be created. You can also define the type of a newly created diagram here. The activity diagram type is selected by default.

The new diagram will be created along with a new activity - the diagram will be created as an inner activity element. I

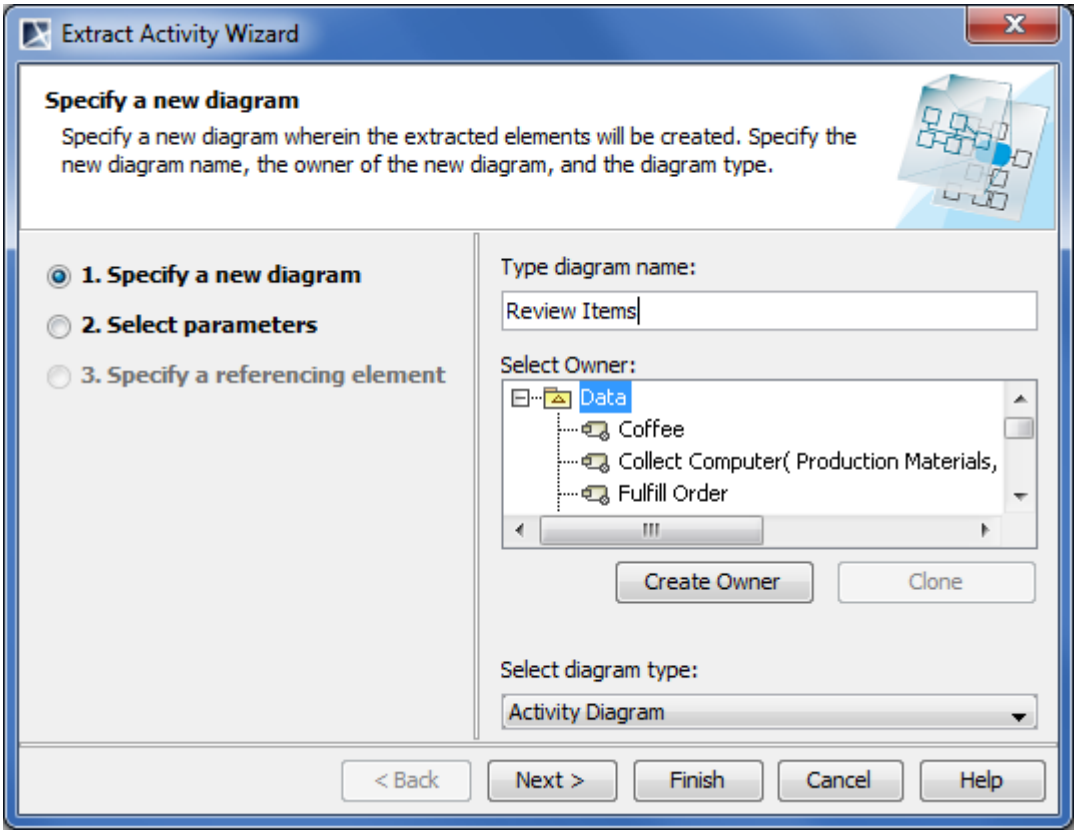


Figure 226 -- Extract Wizard. Specify a new diagram

Wizard element name	Element type	Description
Type diagram name	Text box	Type the name of the new diagram wherein the extracted part will be stored.
Select owner	Box	In the model tree, select a package, profile, or model as an owner of the newly created activity. You may also: • Create a new owner. Click the **Create Owner** button, select an element type, and specify its properties. • Clone an existing package, profile, or model. In the model tree, select an owner, click the **Clone** button, and specify properties of the cloned element.
Select diagram type	Drop-down list	Open the list and select the diagram type for the extracting part. The default diagram type is activity diagram. You can also select to create an interaction overview diagram. By using the customized plugins, you can add custom diagrams to the diagram type list.

Copyright © 1998-2015 No Magic, Inc.

After you have specified the new diagram name, an owner of the diagram, and the type of the new diagram, do one of the following:

- Click **Next**, if you want to specify parameters and to define a referencing element. The next wizard step opens.
- Click **Finish**. The selected part of an activity diagram is extracted to a new diagram that is specified in this step. Instead of the extracted part, the Call Behavior Action is created in the activity diagram referencing to the newly created diagram. The second and the third wizard steps are skipped - that is the default options are taken.

Step #2. Selection of parameters

The parameter(s) that will be created for the intersected object flow is listed in this step. Along with the parameter(s) in the newly created target diagram, the activity parameter node will be created. The activity parameter node has the pin created as an inner element, as well as the assigned parameter (see the following figures).

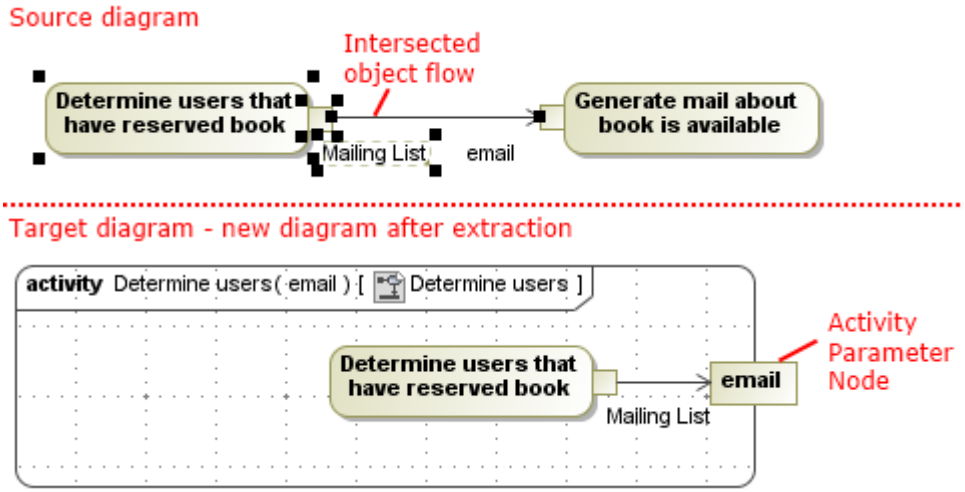


Figure 227 -- Example of intersected object flow extraction

NOTE

- The **Select parameters** list is empty unless there are intersected object flows selected in the source activity diagram.
- For more information about intersected object flows, see concepts in "Extracting" on page 368.

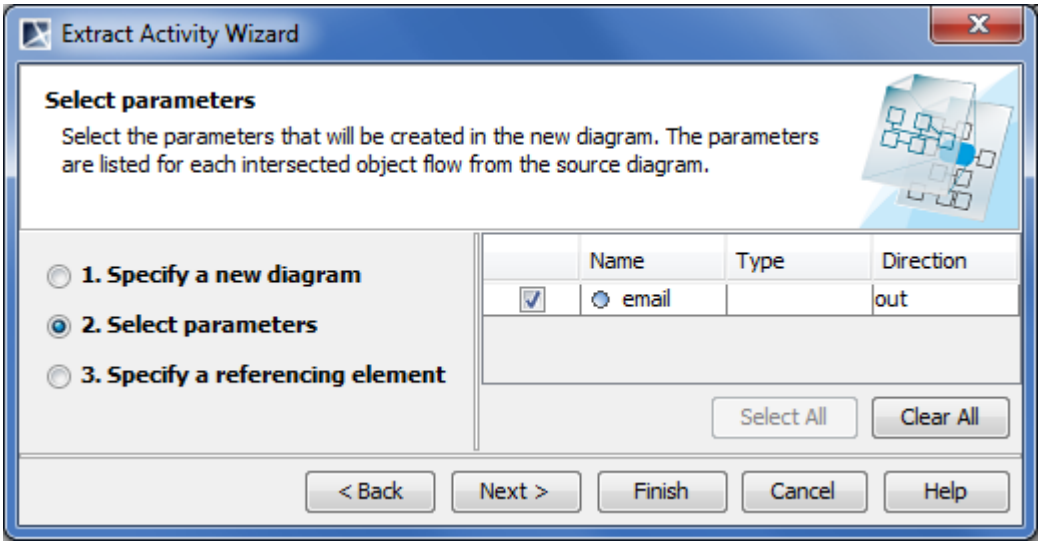


Figure 228 -- Extract Wizard. Select parameters

Copyright © 1998-2015 No Magic, Inc.

Wizard element name	Element type	Description
	Check box	The check box determinates whether or not pin will be created for each parameter.
Name	Table	The name of the parameter is listed in the **Name** column. This column is not editable.
Type	Table	The type of the parameter is listed in the **Type** column. This column is editable - click the **Type** property cell, to open the drop-down list with the ability to search for a type. For more information on using this type of drop down list, see "Selecting single property value from editable list" on page 306.
Direction	Table	The direction of the pin is specified in the **Direction** column. This will be created after the extraction. The *in* or *out* direction is defined according to the intersected object flow direction.

After you have selected the parameters, do one of the following:

- Click **Back** to return to the first step.
- Click **Next,** if you want to define a referencing element. The next wizard step opens.
- Click **Finish**. The selected part of the activity diagram is extracted to a new diagram. The third wizard step is skipped - that is the default options are taken.

Step #3. Specification of a referencing element

A referencing element is created in the source sequence diagram instead of the extracted part. Double-click the referencing element to navigate to the extracted part.

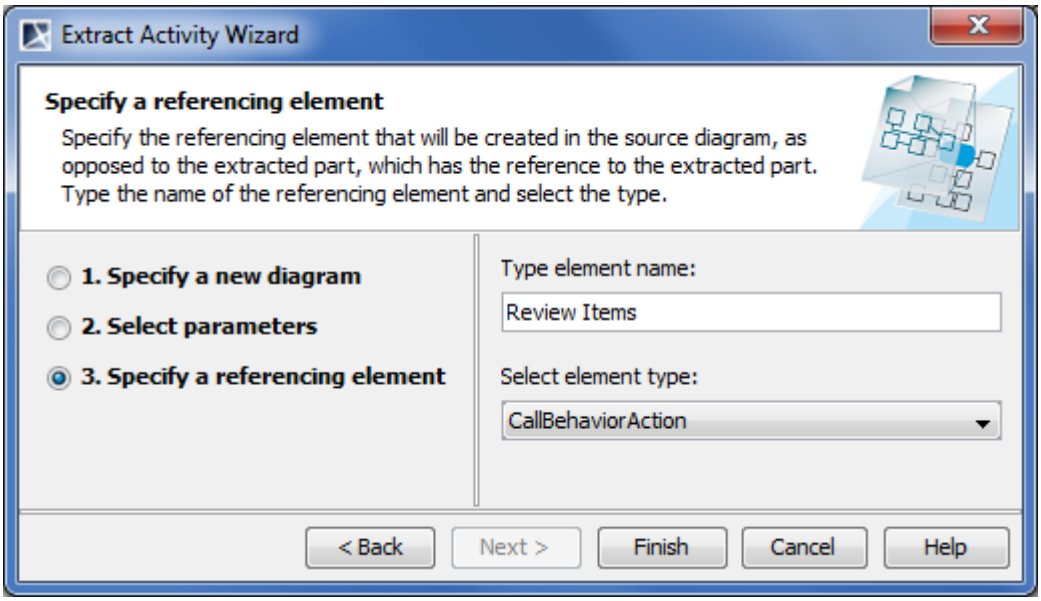


Figure 229 -- Extract Wizard. Specify a referencing element

Wizard element name	Element type	Description
Type element name	Text box	Type the name of the referencing element that will be created in the source activity diagram in place of the selected elements.

Copyright © 1998-2015 No Magic, Inc.

Wizard element name	Element type	Description
Select element type	Drop-down list	Open the list and select the referencing element type. The default referencing element type is a call behavior action. Custom elements can be added to the referencing element type list using customized plugins.

After you have selected the parameters, do one of the following:

- Click **Back** to return to the first step.
- Click **Finish**. The selected part of an activity diagram is extracted to a new diagram.

Related concepts

Extracting

Related references

Extracting in Activity Diagram

Activity Diagram

Extracting in State Machine diagram

The extraction functionality allows extracting a selected part of a state machine diagram to a newly created diagram. After extraction, a submachine state is created in the source diagram in place of the extracted part(s). The submachine state is a referencing element that references the newly created state machine diagram.

Copyright © 1998-2015 No Magic, Inc.

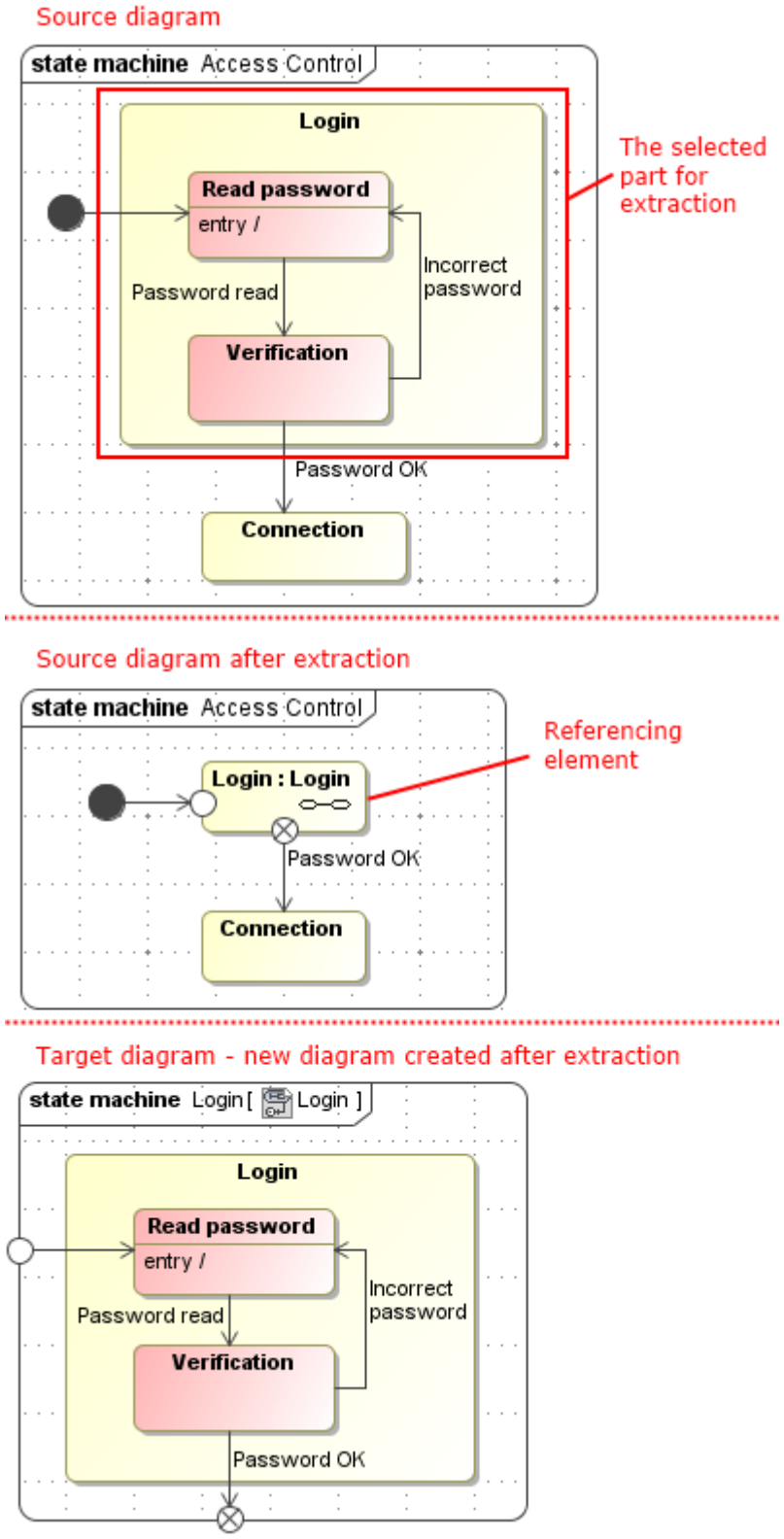


Figure 230 -- Example of state machine diagram extraction

As you can see in the example, during the extraction, the selected part of the state machine diagram is moved to the newly created *Login* state machine diagram. In the source state machine diagram, a *Login* submachine state is created in the place of the selected elements.

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Extracting

Related references

Extract State Machine Wizard

State Machine Diagram

Extract State Machine Wizard

The **Extract State Machine Wizard** is used to extract a selected part of a state machine diagram to a separate diagram. Using this wizard, you can:

- Create and specify the target diagram that will be created as the result of the extraction. The selected part will be extracted to the newly created diagram.
- Select the entry and exit points that will be created in a new diagram. The entry and exit points are listed for each intersected transition from the source diagram.
- Specify a referencing element that is created in the source state machine diagram and represents the elements moved to the target diagram during the extraction.

To start the Extract State Machine Wizard

1. On a state machine diagram, select a part that you want to extract.
2. From the selected part's shortcut menu, select **Refactor** > **Extract**. The **Extract State Machine Wizard** opens.

The **Extract State Machine Wizard** consists of three steps:

1. Specification of a new element
2. Creation of entry/exit points
3. Specification of a submachine

STEP #1. Specification of a new element

In this step, you can specify a diagram into which the extracted part will be moved. The new diagram will be created along with a new state machine - the diagram will be created as an inner state machine element. You can also define the type of the element, the name, and the owner. Note, that the diagram name will be the same as the element name. The state machine type is selected by default

Copyright © 1998-2015 No Magic, Inc.

.

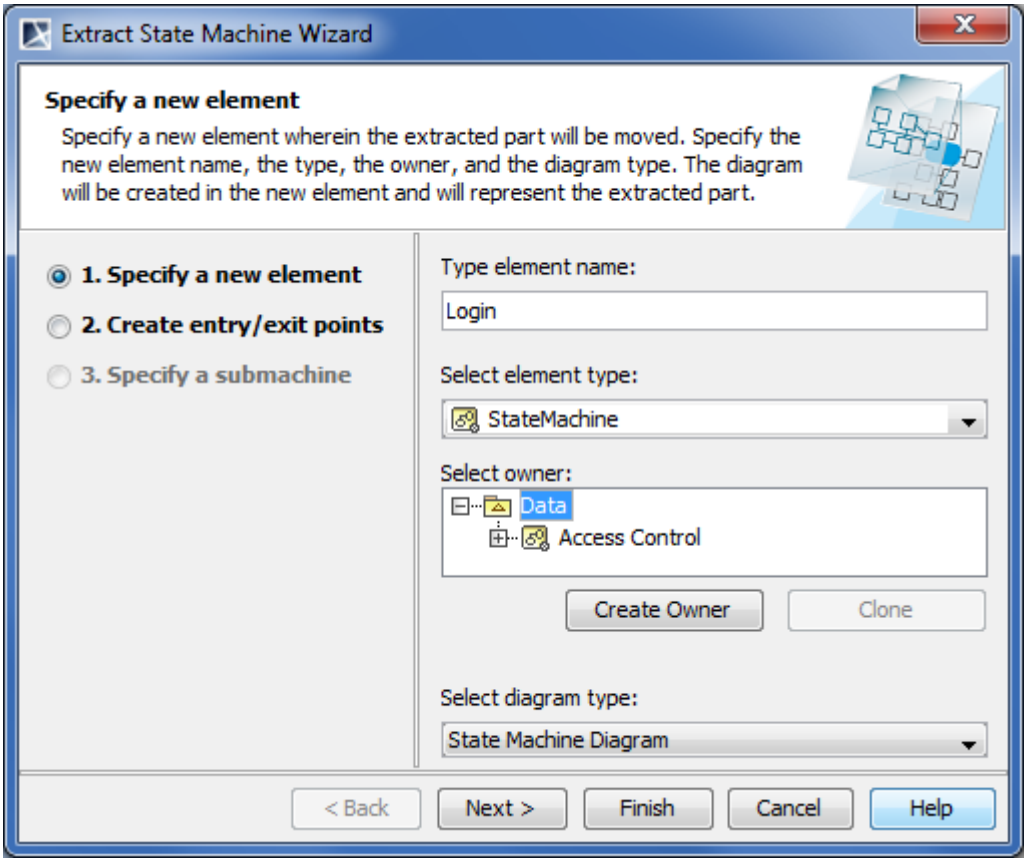


Figure 231 -- Extract State Machine Wizard. Specify a new element

Wizard element name	Element type	Description
Type element name	Text box	Type the name of the new element wherein the extracted part will be stored. The new element will contain the new diagram.
Select element type	Drop-down list	Open the list and select the element type in which the extracted part and new diagram will be created. The default element type is a state machine. By using customized plugins, you can add custom elements to the element type list.
Select owner	Box	In the model tree, select a package, profile, or model as an owner of the newly created state machine. You can also: • Create a new owner. Click the **Create Owner** button, select an element type, and specify its properties. • Clone an existing package, profile, or model. To clone, in the model tree, select an owner, click the **Clone** button, and specify properties of the cloned element.
Select diagram type	Drop-down list	Open the list and select the diagram type for the extracting part. The default diagram type is a state machine diagram. By using customized plugin, you can add custom diagrams to the diagram type list.

Copyright © 1998-2015 No Magic, Inc.

After you have specified the new element name, an owner of the element, and the type of the new diagram, do one of the following:

- Click **Next**, if you want to specify entry/exit points, and to define a referencing element. The next wizard step opens.
- Click **Finish**. The selected part of a state machine diagram is extracted to a new diagram that is specified in this step. Instead of the extracted part, the submachine state is created referencing to the newly created diagram. The second and the third wizard steps are skipped - that is the default options are taken.

Step #2. Creation of entry/exit points

The entry and exit points that will be created for the intersected transitions are listed in this step.

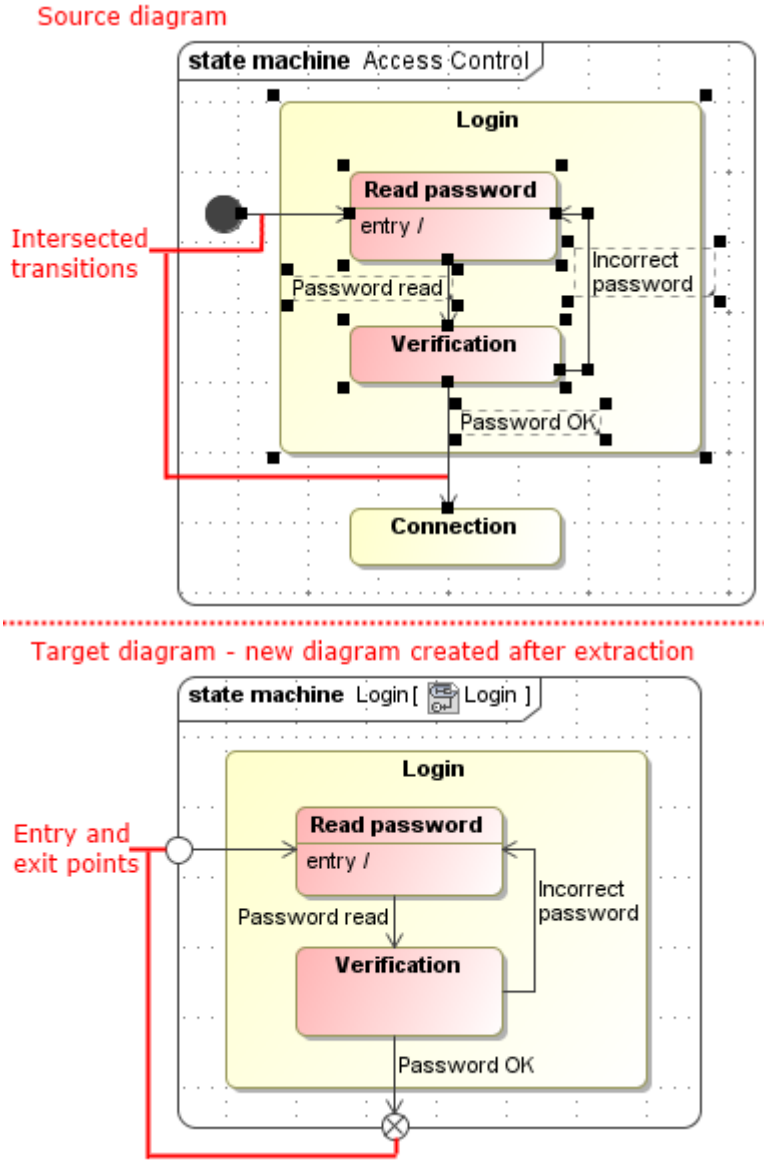


Figure 232 -- Example of intersected transition extraction

Copyright © 1998-2015 No Magic, Inc.

As you can see in the example, in the *Access Control* source diagram, the intersected transitions are selected for the extraction. After the extraction, in the target diagram, the entry and exit points are created for each intersected transition.

NOTE

In the **Extract State Machine Wizard**, the **Create entry/exit point** step is empty unless there are intersected transitions selected in the source state machine diagram.

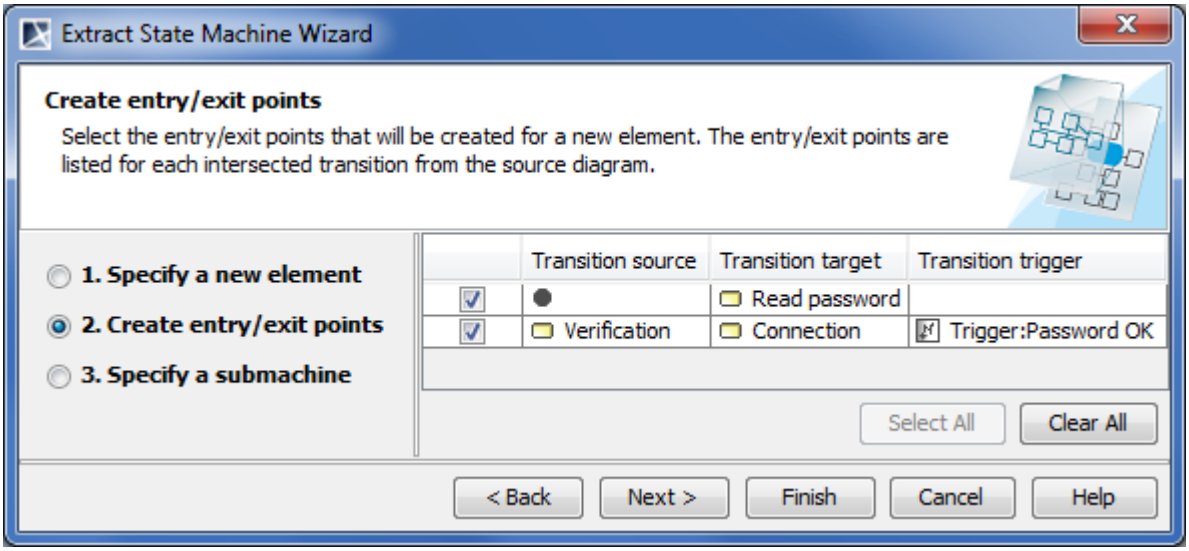


Figure 233 -- Extract State Machine Wizard. Create entry/exit points

Wizard element name	Element type	Description
☑	Check box	The check box determines whether or not an entry/exit point will be created for each intersected transition.
Transition source	Table	The source element of the transition.
Transition target	Table	The target element of the transition.
Transition trigger	Table	Trigger created on the transition.

After you have selected the parameters, do one of the following:

- Click **Back** to return to the first step.
- Click **Next** to define a referencing element. The next wizard step opens.
- Click **Finish**. The selected part of the state machine diagram is extracted to a new diagram. The third wizard step is skipped - that is the default submachine name is taken.

Copyright © 1998-2015 No Magic, Inc.

Step #3. Specification of a submachine

A referencing element is created in the source state machine diagram instead of the extracted part. Double-click the referencing element to navigate to the extracted part.

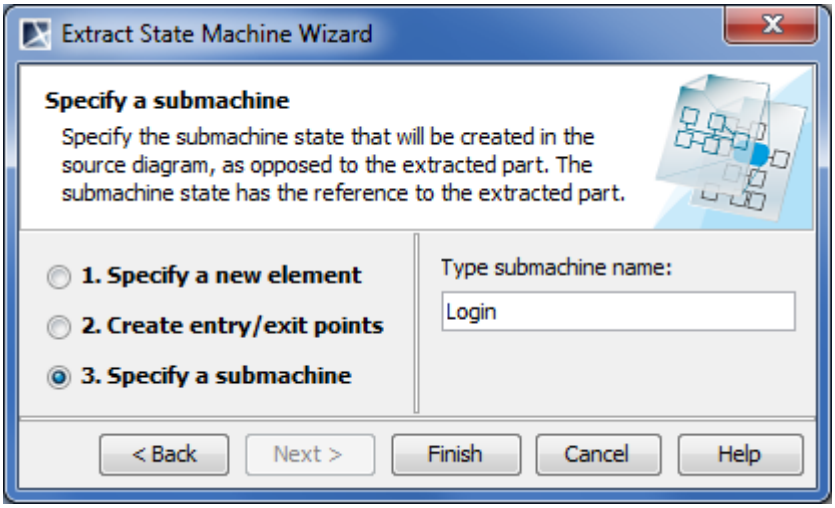


Figure 234 -- Extract State Machine Wizard. Specify a submachine

Wizard element name	Element type	Description
Type submachine name	Text box	Type the name of the referencing element that is created in the source state machine diagram in place of the selected part after the extraction.

After you have typed the submachine name, do one of the following:

- Click **Back** to return to the second step.
- Click **Finish**. The selected part of the state machine diagram is extracted to the new state machine diagram.

Related concepts

Extracting

Related references

Extracting in State Machine diagram

State Machine Diagram

Copyright © 1998-2015 No Magic, Inc.

Extracting in Composite Structure diagram

NOTE

With the SysML plugin, the extracting in composite structure diagram feature is available in Standard, Professional, Architect, and Enterprise editions.

The extraction functionality allows extracting a selected part of a composite structure diagram to a newly created diagram. After extraction, a property is created in the source diagram in place of the extracted part(s). That is, the property is a referencing element that references the newly created activity diagram.

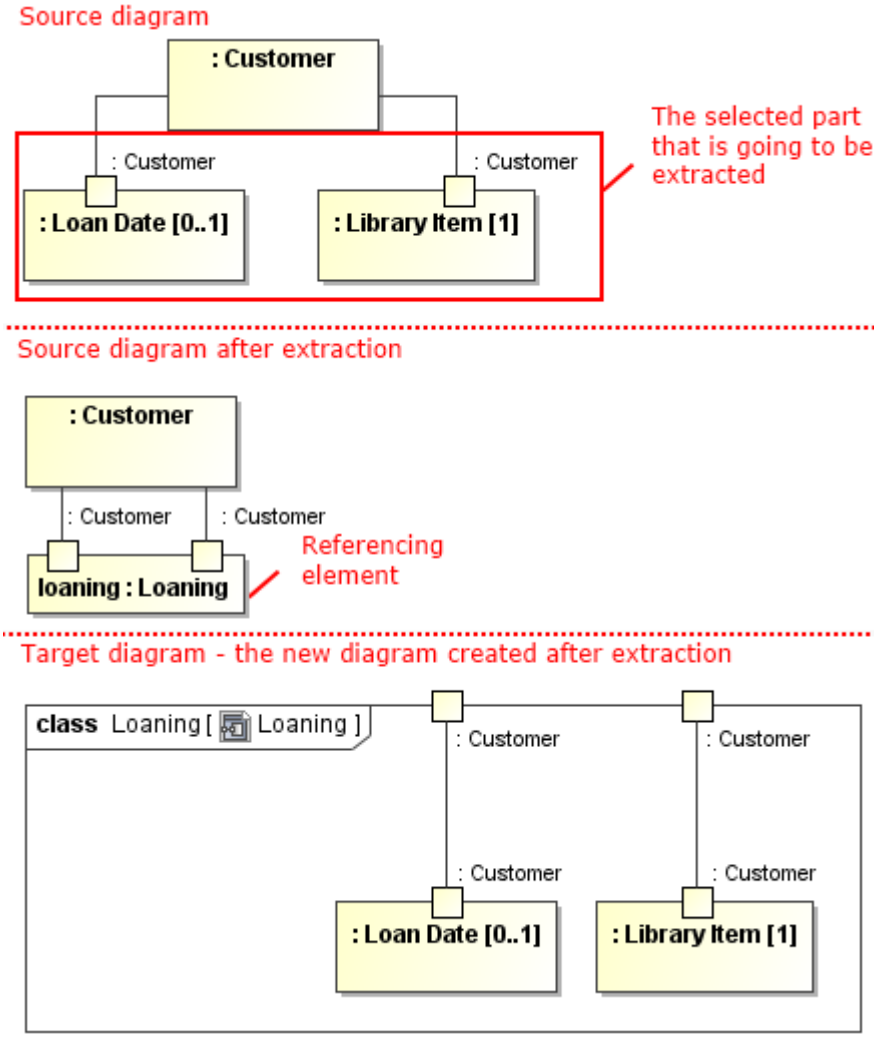


Figure 235 -- Example of extraction

As you can see in the example, during extraction, the selected part of the composite structure diagram is moved to the newly created *Loaning* composite structure diagram. In the source composite structure diagram, a *loaning* property is created in a place of the selected elements.

Related concepts

Extracting

Related references

Extract Structure Wizard

Composite Structure Diagram

Copyright © 1998-2015 No Magic, Inc.

Extract Structure Wizard

The **Extract Structure Wizard** is used to extract a selected part of a composite structure diagram to a separate diagram. Using this wizard, you can:

- Create and specify the target diagram that will be created as the result of the extraction. The selected part will be extracted to the newly created diagram.
- Select the ports that will be created in a new diagram. The ports are listed for each intersected connector from the source diagram.
- Specify a referencing element (that is, a property) that is created in the source composite structure diagram and represents the elements moved to the target diagram during the extraction.

To start the Extract Structure Wizard

1. On an composite structure diagram, select a part that you want to extract.
2. From the selected part's shortcut menu, select **Refactor** > **Extract**. The **Extract Structure Wizard** opens.

This section describes components of the **Extract Structure Wizard**, such as text boxes and buttons. The **Extract Structure Wizard** consists of three steps:

1. Specification of a new element.
2. Creation of ports.
3. Creation of a property.

STEP #1. Specification of a new element

In this step, you need to specify a diagram into which the extracted part will be moved and an owner wherein this new diagram will be created. You can also define the type of the newly created classifier element that will own the diagram. The class element and the composite structure diagram type are selected by default. The name of the composite structure diagram will be the same as the specified name of the classifier element

Copyright © 1998-2015 No Magic, Inc.

.

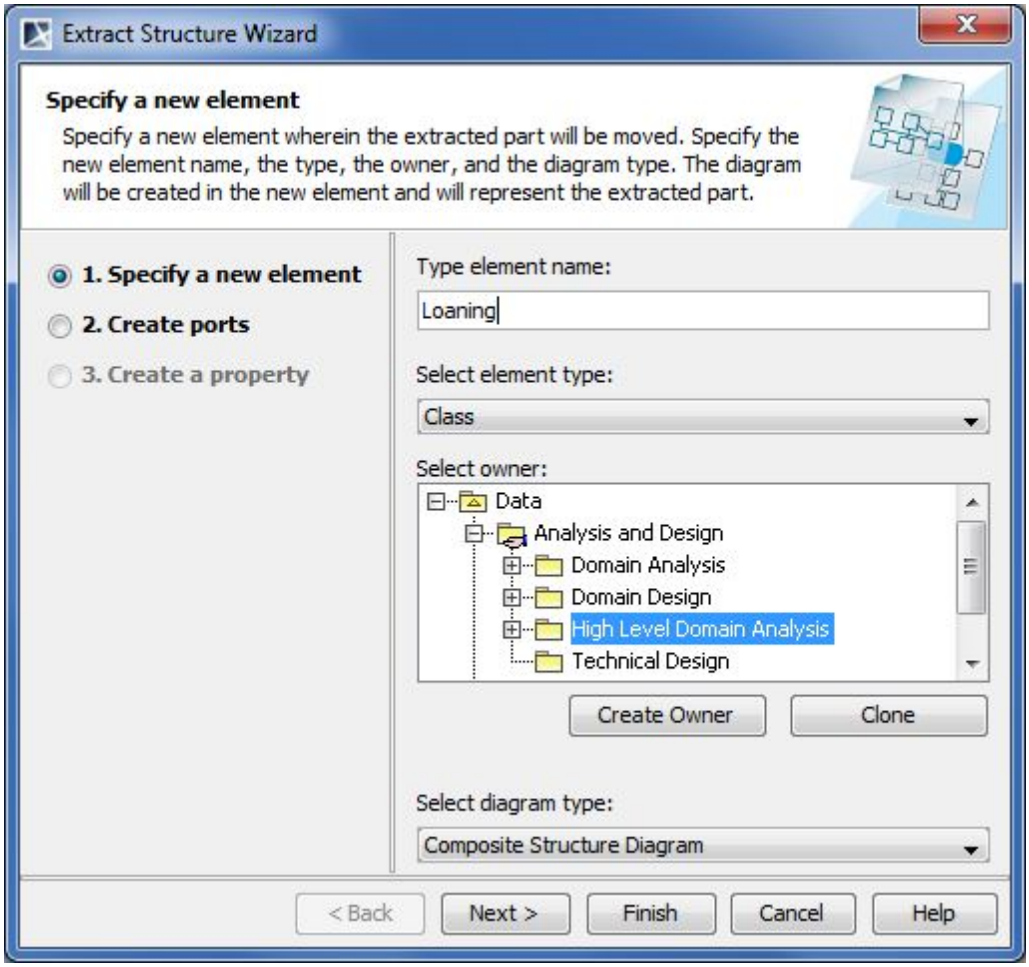


Figure 236 -- Extract Structure Wizard. Specify a new element and a diagram

Wizard element name	Element type	Description
Type element name	Text box	Type the name of the new element, that is, classifier, wherein the extracted part will be stored.
Select element type	Drop-down list	Open the list and select the element type. The default element type is a class. By using the customized plugins, you can add custom elements to the element type list.
Select owner	Box	In the model tree, select a package, profile, or model as an owner of the newly created element. You may also: • Create a new owner. Click the **Create Owner** button, select an element type, and specify its properties. • Clone an existing package, profile, or model. In the model tree, select an owner, click the **Clone** button, and specify properties of the cloned element.
Select diagram type	Drop-down list	Open the list and select the diagram type for the extracting part. The default diagram type is a composite structure diagram. By using the customized plugins, you can add custom diagrams to the diagram type list.

Copyright © 1998-2015 No Magic, Inc.

After you have specified the new element name, the element type, an owner of the diagram, and the type of the new diagram, do one of the following:

- Click **Next**, if you want to specify ports and to define a referencing element. The next wizard step opens.
- Click **Finish**. The selected part of a composite structure diagram is extracted to a new diagram that is specified in this step. Instead of the extracted part, the selected element type is created in the composite structure diagram referencing to the newly created diagram. The second and the third wizard steps are skipped, that is, the default options are taken.

Step #2. Creation of ports

In this step, there are listed the port(s) that will be created for the:

- intersected connector(s). For more information about intersected connectors, see concepts in "Extracting" on page 368.
- incoming connector(s)
- outgoing connector(s)

You can see an example of intersected connectors in Figure 235 on page 389.

NOTE

The **Create ports** list is empty unless there are intersected, incoming and/or outgoing connector(s) selected in the source composite structure diagram.

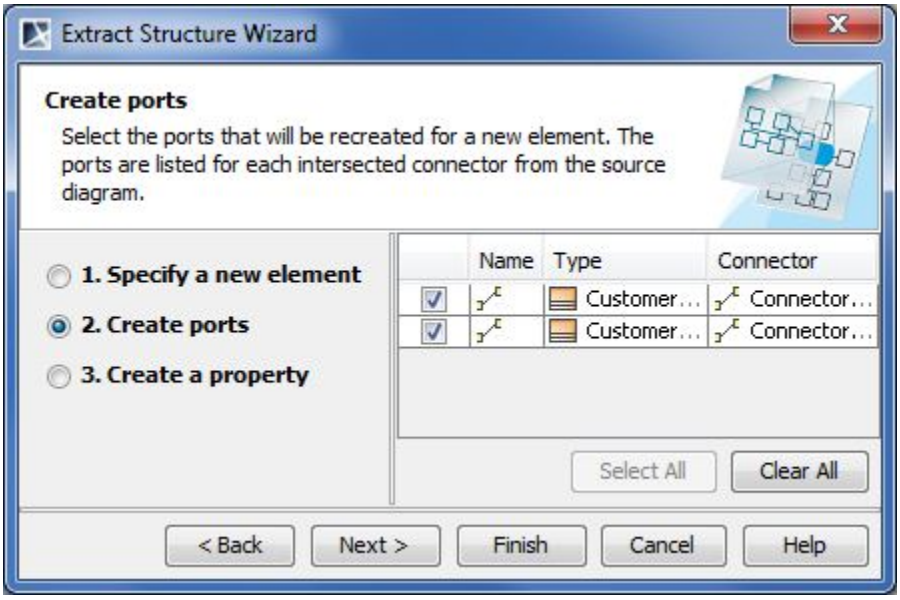


Figure 237 -- Extract Wizard. Create ports

Wizard element name	Element type	Description
☑	Check box	The check box determines whether or not a port will be created for each intersected connector.
Name	Table	The name of the connector is listed in the **Name** column. This column is not editable.

Copyright © 1998-2015 No Magic, Inc.

Wizard element name	Element type	Description
Type	Table	The type of the connector is listed in the **Type** column. This column is editable - click the **Type** property cell to open the drop-down list with the ability to search for a type. For more information on using this type of drop down list, see "Selecting single property value from editable list" on page 306.
Connector	Table	In the **Connector** column, there is listed the connector, which will be created after the extraction in the target diagram. The connector will connect the extracted element with the target diagram border. See an example in the Figure 235 on page 389.

After you have selected the ports, do one of the following:

- Click **Back** to return to the first step.
- Click **Next,** if you want to define a property (that is, referencing element). The next wizard step opens.
- Click **Finish**. The selected part of the composite structure diagram is extracted to a new diagram. The third wizard step is skipped, that is, the default options are taken.

Step #3. Creation of a property

A property is created as the result of the extraction. The property is created under the classifier that was specified in the first wizard step. The symbol of the property is created in the source composite structure diagram and represents the elements moved to the target diagram during the extraction.

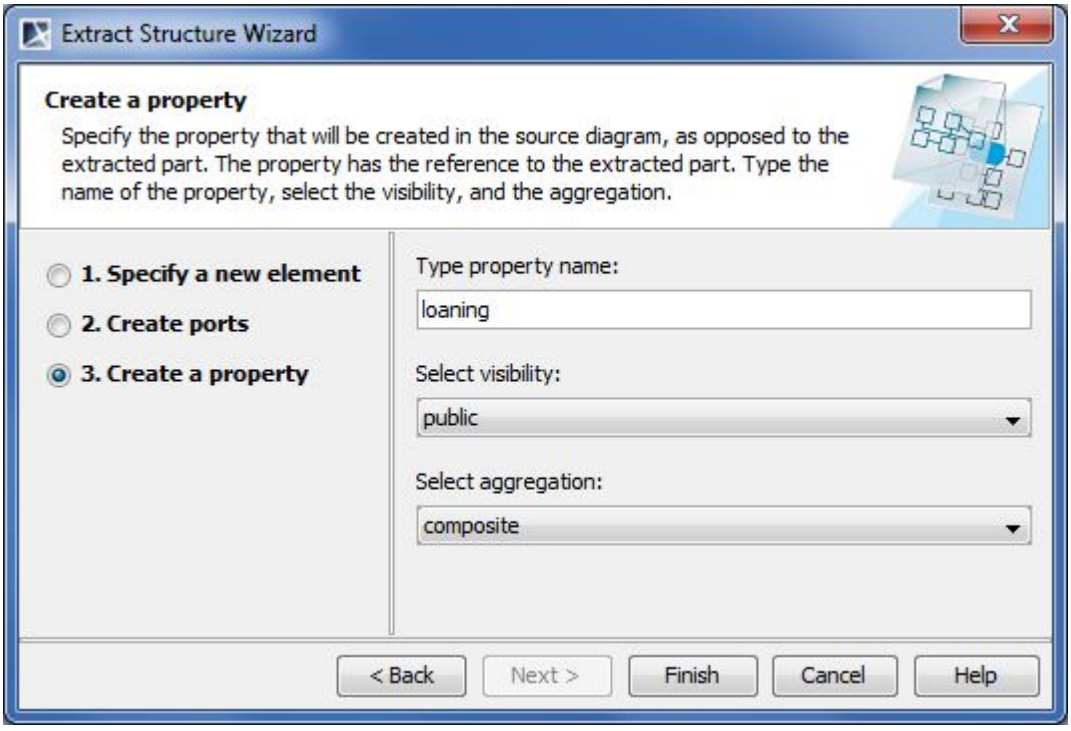


Figure 238 -- Extract Wizard. Create a property

Wizard element name	Element type	Description
Type property name	Text box	Type the name of the property that will be created in the source composite structure diagram in place of the selected elements.

Copyright © 1998-2015 No Magic, Inc.

Wizard element name	Element type	Description
Select visibility	Drop-down list	Open the list and select the visibility for the property. The default visibility is public.
Select aggregation	Drop-down list	Open the list and select the aggregation for the property. The default aggregation is composite.

After you have selected the parameters, do one of the following:

- Click **Back** to return to the previous step.
- Click **Finish**. The selected part of a composite structure diagram is extracted to a new diagram.

Related concepts

Extracting

Related references

Extracting in Composite Structure diagram

Composite Structure Diagram

Generic Numbering Mechanism

MagicDraw provides you with a generic numbering mechanism that can be applied to any elements created in MagicDraw. The generic numbering mechanism allows you to:

- Automatically assign a unique number to the elements in the model when they are created, moved, or your model is refactored.
- Easily identify and find an element in the model when communicating with colleagues or stakeholders.
- Show an element's place in a hierarchy.

With this feature you can easily:

- Customize currently applied numbering formats.
- Create and apply your own numbering formats.
- Change the numbering format while creating numbered elements.

The Generic Numbering Mechanism feature description is organized in two sections. Section "Working with Generic Numbering Mechanism" on page 395 describes how to number elements, apply numbering formats, and modify numbers. Section "Element Numbering dialog" on page 399 explains GUI elements, used while numbering elements.

Before stating to describe the generic numbering mechanism, get acquainted with used concepts that are presented in the following section.

Concepts

Numbering customization

A customization class which specifies element types that will be numbered, numbering format that will be used to number elements, and the property to which the created number will be assigned. The numbering format should be defined prior to numbering elements. Usually the system administrator creates the numbering customization.

Copyright © 1998-2015 No Magic, Inc.

Numbering property

Indicates an element property wherein the element number will be stored and defines a numbering scheme that will be used for the element numbering.

Numbering scheme

A set of rules describing what numbering formats will be applied on the concrete elements. Numbering scheme is defined while creating the numbering customization.

Prefix

An affix which is placed before the element number.

Separator

A symbol which is used to separate of the different level numbers. Separator is defined in the numbering scheme.



For information on how to create the numbering formats (schemes), see “Creating numbering customizations” in MagicDraw UMLProfiling&DSL UserGuide.pdf.

Working with Generic Numbering Mechanism

With the generic numbering mechanism you can number the elements. There are two element numbering modes: automatic and manual. If you choose automatic numbering mode, a number for an element will be created automatically according to predefined numbering format. Using the manual numbering mode you can create a custom number format for the element.



Make sure that numbering customizations are already created.

For more information about creating numbering customizations, see “Creating numbering customizations” in MagicDraw UMLProfiling&DSL UserGuide.pdf.

Automatic numbering

With an automatic numbering feature, you can number the elements of the selected type upon the element creation or modification. You can specify the number display location after the elements are numbered.

To enable automatic element numbering

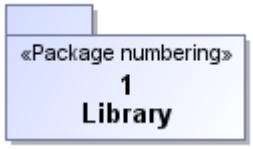
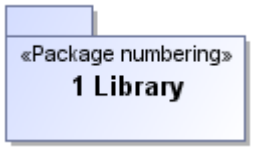
1. From the main menu, select **Options** > **Project**. The **Project Options** dialog opens.
2. Select the **General project options** tab.
3. In the **Numbering** properties group, set the **Use Element Auto-numbering** property value to *true*.
4. Click **OK**.

To specify the element number display location

1. Do one of the following:
   - On the selected element’s shortcut menu, click **Symbol Properties**. The **Symbol Properties** dialog opens.
   - From the main menu, select **Options** > **Project**. Then, in the opened **Project Options** dialog, select the **Symbol styles** > **Default** tab.

Copyright © 1998-2015 No Magic, Inc.

2. Select one of the following **Element Numbering Display Mode** property value:

Property value	Visualization on the element shape
Above the element name	
Before the element name	
Do not display on symbol	

3. Click **OK** after you have selected.

NOTE

You can edit automatically applied element numbers. For information about editing element numbers, see "Manual numbering" on page 396.

Related references

Element Numbering dialog

Related procedures

Manual numbering

Manual numbering

Using the manual numbering, you can:

- Number one or more elements.
- Increase, decrease, and change a number for an element.
- Add a prefix for the element number
- Change a numbering format.

To number the element

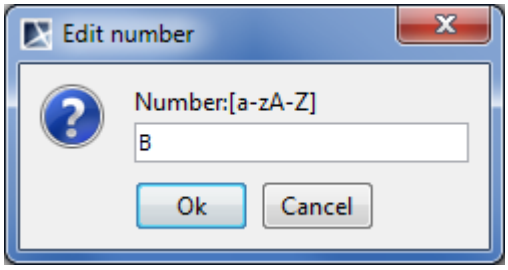
1. On the selected element's shortcut menu, click **Element Numbering**. The **Element Numbering** dialog opens.
2. In the element list on the right side of the dialog, select an element you want to number.
3. Click **Create**.

To change the element's numbering symbol

1. Open the selected element's shortcut menu and click **Element Numbering**. The **Element Numbering** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

2. Click the **Edit** button. The **Edit number** dialog opens.



3. Enter a new numbering symbol.

NOTE

You can enter only the number, if the selected element's numbering value is numerical.

You can enter only a new literal symbol, if the selected element's numbering value is a literal symbol.

4. Click **OK**.

To remove the element's number

1. Open the selected element's shortcut menu and click **Element Numbering**. The **Element Numbering** dialog opens.
2. Select the element from which you want to remove the number and click the **Remove** button.
3. You can select either to remove the number only from the selected element or to remove numbers from all elements owned by the selected elements. Do one of the following:
   - Click the **Remove** button. Only the selected element's number is removed.
   - Click the **Remove All** button. Numbers are removed from the selected element and all elements owned by the selected element.
4. Click **OK** when you are done.

To increase the element number

1. Open the selected element's shortcut menu and click **Element Numbering**. The **Element Numbering** dialog opens.
2. Select the element which number you want to increase.
3. Click the **Increase** button. The element number is increased by one.
4. Click **OK** when you are done.

To decrease the element number

1. Open the selected element's shortcut menu and select **Element Numbering**. The **Element Numbering** dialog opens.
2. Select the element which number you want to decrease.
3. Click the **Decrease** button. The element number is decreased by one.

NOTE

The **Decrease** button is available, when the selected element's number is not the first in the list.

4. Click **OK** when you are done.

To renumber the elements

1. On the selected element's shortcut menu, click **Element Numbering**. The **Element Numbering** dialog opens.
2. You can select either to renumber only elements listed in the element list or to renumber elements listed in the element list including their owning elements. Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

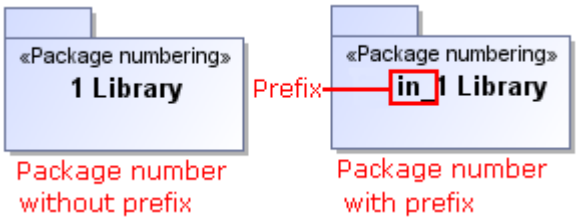
- Click the **Renumber** button. All the elements in the elements list are renumbered successively.
- Click the **Renumber All** button. All the elements and elements owned by the selected element are renumbered successively.

3. Click **OK** when you are done.

To add a prefix to the element number

1. On the element owner's shortcut menu, select **Element Numbering**. The **Element Numbering** dialog opens.
2. In the **Prefix** column, click the appropriate cell (according to the used numbering property and numbering scheme).
3. Type the prefix. It can be any symbol.
4. Click **OK**.

The following example illustrates how numbers with prefixes and without prefixes are depicted on the element shape.



To change a separator in the element number

1. Open the element owner's shortcut menu and click **Element Numbering**. The **Element Numbering** dialog opens.
2. In the **Separator** column, click the appropriate cell (according to the used numbering property and numbering scheme).
3. Type the separator symbol.



A new separator will override the earlier predefined separator.

4. Click **OK**.

The following example illustrates the number separator changing from "." to "/".



To change a numbering scheme

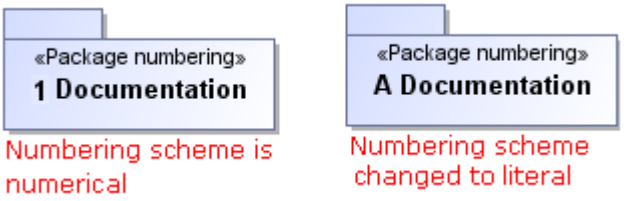
1. On the selected element's shortcut menu, click **Element Numbering**. The **Element Numbering** dialog opens.
2. In the appropriate **Numbered property** row, click the **Numbering Scheme** cell to open the list of the defined numbering schemes.
3. Select the desired numbering scheme in that list.

Copyright © 1998-2015 No Magic, Inc.

4. Click **OK**.

NOTE When you change the numbering scheme for the selected element, the new numbering scheme is applied to all elements owned by the selected element.

The following example illustrates how numbers are depicted on the element shape after changing the numbering scheme from numerical to literal.



Related references

Element Numbering dialog

Related concepts

Numbering customization
Numbering property
Numbering scheme
Prefix
Separator

Element Numbering dialog

The **Element Numbering** dialog is designed to number elements, select different numbering schemes, or modify numbering properties that are defined by the numbering customization. Using this dialog, you can also change, remove, and update the element numbering, as well as you can change the number separator, prefix, and the numbering scheme.

NOTE Numbering changes you make for the selected element will by applied to all elements owned by the selected element.

To open the Element Numbering dialog

1. Select the model element you want to number.
2. Open the selected element's shortcut menu and click **Element Numbering**.

NOTE The command **Element Numbering** is available, if numbering customization for the selected element type is created.

For more information about numbering customization, see "Creating numbering customizations" in MagicDraw UMLProfiling&DSL UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

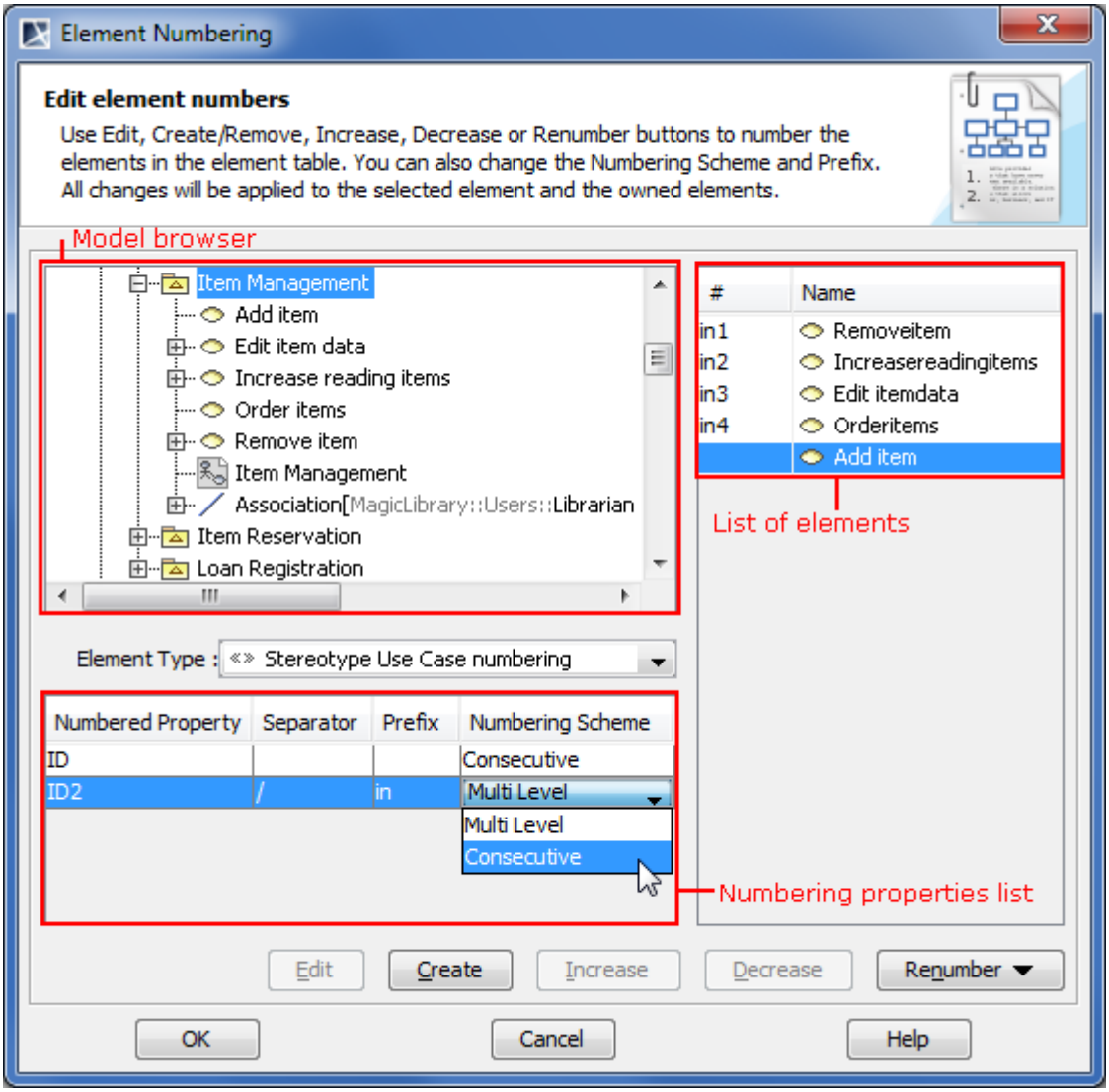


Figure 239 -- Element Numbering dialog

Your model is represented in the Model browser on the left side of the dialog. If the element of the type that is selected in the **Element Type** list is selected in the Model browser, owned elements are listed on the right side of the dialog.

In the **Element Type** list you can select which type of elements you want to number. The selected element type numbered property, separator, prefix, and numbering scheme are displayed in the numbering properties list.

The usability of numbering properties list is described in the following table:

Dialog element name	Dialog element type	Description
Numbered Property	Text box	A name of a property to which the created element number is assigned. Not editable. This property is defined in numbering customization. This column is available, when more that one numbered property is defined for the selected element type.

Copyright © 1998-2015 No Magic, Inc.

Dialog element name	Dialog element type	Description
Separator	Text box	Enter your custom separator which can be any symbol. The separator is changed for all numbers of the selected element type. All previously defined separators will be overridden.
Prefix	Text box	Enter your custom prefix which can be any symbol. Just the one prefix can be entered for the selected numbering scheme. The defined prefix is displayed before the element number. All previously defined prefixes will be overridden.
Numbering Scheme	Drop-down list	Select one of the available numbering schemes for the selected element type. Numbering schemes are defined in numbering customization.



For more information on numbering properties, see “Creating your first numbering customization” in MagicDraw UMLProfiling&DSL UserGuide.pdf.



You can have several numbering schemes defined for one type of elements. All defined numbering schemes for the selected element type are displayed in the numbered properties list.

The Element Numbering dialog buttons are described in the following table:

Button	Description
Edit	Click to change the numbering symbol for the selected element. The button is available, when the selected element has a number and the element is editable.
Create / Remove	The **Create** button is available, if a selected element in the element list is not numbered. Click the **Create** button to number the selected element. A unique successive number will be created for the selected element. The **Remove** button is available, if a selected element in the element list is numbered and editable. This button has two options: Click **Remove** to delete the selected element’s number. Click **Remove All**. Numbers are removed from the selected element and all elements owned by the selected element
Increase	Click to increase an element number. The button is available, if the selected element is numbered and editable. The element’s place in the list changes after increasing the number.
Decrease	Click to decrease an element number. The button is available, if the element is editable, is numbered, and that number is not the first one. The element’s place in the list changes after decreasing the number.
Renumber	This button has two options: Click **Renumber** to update the element numbers. The numbering is updated for all elements in the list. Click **Renumber All** to update all element numbers recursively. The numbering is updated for all elements in the element list, including their owning elements.

Copyright © 1998-2015 No Magic, Inc.

Related procedures

Automatic numbering
Manual numbering

Related concepts

Numbering customization
Numbering property
Numbering scheme
Prefix
Separator

Smart Packages



This feature is available in Standard, Professional, Architect, and Enterprise editions.

Smart package is a special collection of model elements.

An element can be included in the smart package either:

- *Manually* - the user selects to include the particular element.



You can create a smart package named *My Bookmarks* and add frequently used elements to it.

- *Automatically* - the element meets the set of criteria defined by the user.



You can create a smart package named *Requirements v5* with the criteria “all elements of type *Requirement* under the package *Business requirements*, having tag *version=5*”.

The membership in the smart package is not the UML ownership: one element can belong to several smart packages.

Smart packages aggregate relevant elements so that you can

- Easily browse, navigate, list, and discover these elements in the Containment tree.
- Narrow the scope in both the **Find** dialog and the element Selection dialog.
- Define dynamic row and column scopes in dependency matrices.

Concepts

For better understanding the further material, learn the following concepts.

Static contents

A collection of manually included elements.

Dynamic contents

A collection of elements that is automatically calculated according to the set of criteria specified by the user.

Copyright © 1998-2015 No Magic, Inc.

The smart packages feature is in details described in

- "Creating Smart Packages" on page 403
- "Managing Contents of Smart Package" on page 404
- "Smart Package Properties" on page 408
- "Using Smart Packages in Your Model" on page 408

Creating Smart Packages

A smart package can be created in one of the following ways:

- **From the shortcut menu of an element**. The newly created smart package is empty.
- **From the search results** (by saving as a smart package either the search options or the search results). The newly created smart package appropriately has either dynamic or static contents.

To create a smart package from the shortcut menu of an element

1. In the Containment tree, right-click the element. The shortcut menu opens.

   IMPORTANT! The element should be a namespace, for example, a package, a class, or a requirement.

2. Select **Create Element** > **Smart Package**.
3. Type a name for the new smart package.
4. Press Enter to finish.

The new smart package is created, and you can proceed to managing its contents and reviewing its specific properties.

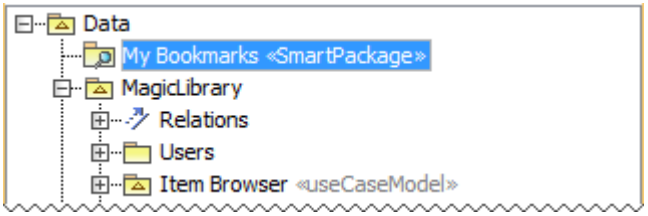


Figure 240 -- New smart package with empty contents

To create a smart package from the search results

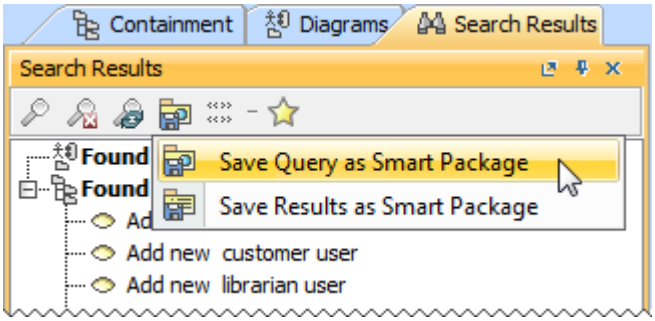
1. Open the **Find** dialog, define search options, and perform the search. The search results are displayed in the **Search Results** tab on the Model Browser.

   TIP! For more information about defining search options and analyzing the search results, appropriately refer to "Find dialog" on page 132 and "Search Results tab" on page 87.

2. On the toolbar of the **Search Results** tab, click and then select one of the following commands:
   - **Save Query as Smart Package** to create a new smart package with dynamic contents. The search options will be saved as the **Query** property value.
   - **Save Results as Smart Package** to create a new smart package with static contents. The search results will be saved as values of the **Additional Elements**

Copyright © 1998-2015 No Magic, Inc.

property.



The element Selection dialog opens.

3. In the dialog, select an owner of the new smart package and click **OK**.
4. In the Containment tree, type a name for the new smart package.
5. Press Enter to finish.

The new smart package is created, and you can proceed to managing its contents and reviewing its specific properties.

Managing Contents of Smart Package

Once the smart package is created, you can manage its contents as is described in the following sections:

- "Including elements into a smart package" on page 404
- "Removing elements from a smart package" on page 406
- "Freezing the contents of a smart package" on page 406
- "Making a snapshot of the smart package" on page 407

Including elements into a smart package

An element can be included in the smart package either:

- *Manually* - the user selects to include the particular element. Manually included elements constitute the static contents.
- *Automatically* - the element meets the set of criteria defined by the user. Automatically included elements constitute the dynamic contents.

IMPORTANT!

The membership in the smart package is not the UML ownership association. The owner of the element will not change after the element is included into the smart package.

Copyright © 1998-2015 No Magic, Inc.

To include an element into a smart package manually

Do either:

- In the Containment tree, select an element and drag it to the smart package.
  See the element included into the contents of the smart package. The element becomes a value of the **Additional Elements** property as well.



You can include more than one element at once.

- To include *adjacent* elements, select a single element, and then hold down the Shift key while you click other elements that you want to select.
- To include *nonadjacent* elements, select a single element, and then hold down the Ctrl key while you click other elements that you want to select.

- Open the Specification window of the smart package and click the cell of the **Additional Elements** property value. Then click ... and in the open dialog, select the element or more. Click **OK** when you are done.
  See the element included into the contents of the smart package.

To include elements into a smart package automatically

1. Open the Specification window of the smart package.
2. Under the **Contents** category, click the cell of the **Query** property value.
3. Click ... . The **Query** dialog opens.
4. In the dialog, define a set of criteria (query) for gathering the elements into the contents of the smart package.



For the instructions about defining criteria in the **Query** dialog, refer to "Specifying Criteria for Querying Model" on page 551.

5. Close the Specification window.
   See the contents of the smart package updated with elements that meet the defined set of criteria.



If the contents of the smart package is not updated, select **View** > **Refresh** from the main menu.



- If the smart package contains manually included elements (values of the **Additional Elements** property), they are joined up with the automatically included elements, skipping duplicates.
- If the smart package has excluded elements (values of the **Excluded Elements** property), they do not appear in the dynamic contents of the smart package.

Copyright © 1998-2015 No Magic, Inc.

Removing elements from a smart package

To remove an element from a smart package

Do either:

- In the Containment tree, under the smart package, right-click the element and from the shortcut menu, select **Delete from Contents**.



You can exclude more than one element at once.

- To exclude *adjacent* elements, select a single element, and then hold down the Shift key while you click other elements that you want to select.
- To exclude *nonadjacent* elements, select a single element, and then hold down the Ctrl key while you click other elements that you want to select.

- Open the Specification window of the smart package and click the cell of the **Excluded Elements** property value. Then click ... and in the open dialog, select the element or more. Click **OK** when you are done.
- Open the Specification window of the smart package and click the cell of the **Additional Elements** property value. Then click ... and in the open dialog, unselect the element or more. Click **OK** when you are done.



This is valid only for manually added elements.

See the element removed from the contents of the smart package, though it is not removed from the model. You can still see the element displayed under the owner of the element in the Containment tree.

Freezing the contents of a smart package

If you no longer need the contents of a smart package to be dynamic, you can simply freeze it. The dynamic contents will be converted to the static list of elements and joined up with the list of manually added elements (values of the **Additional Elements** property), skipping both duplicates and elements defined as excluded (values of the **Excluded Elements** property).

To freeze the contents of a smart package

- In the Containment tree, right-click the smart package with dynamic contents and from the shortcut menu select **Freeze Contents**.

  See that there are no changes visible in the Containment tree. Open the Specification window of the smart package to see that the values of both the **Query** property and the **Excluded Elements** property have passed into empty, and the value of the **Additional Elements**

Copyright © 1998-2015 No Magic, Inc.

property stores the new list of elements.

IMPORTANT! The contents of the smart package will no longer be automatically updated after relevant changes in the model.

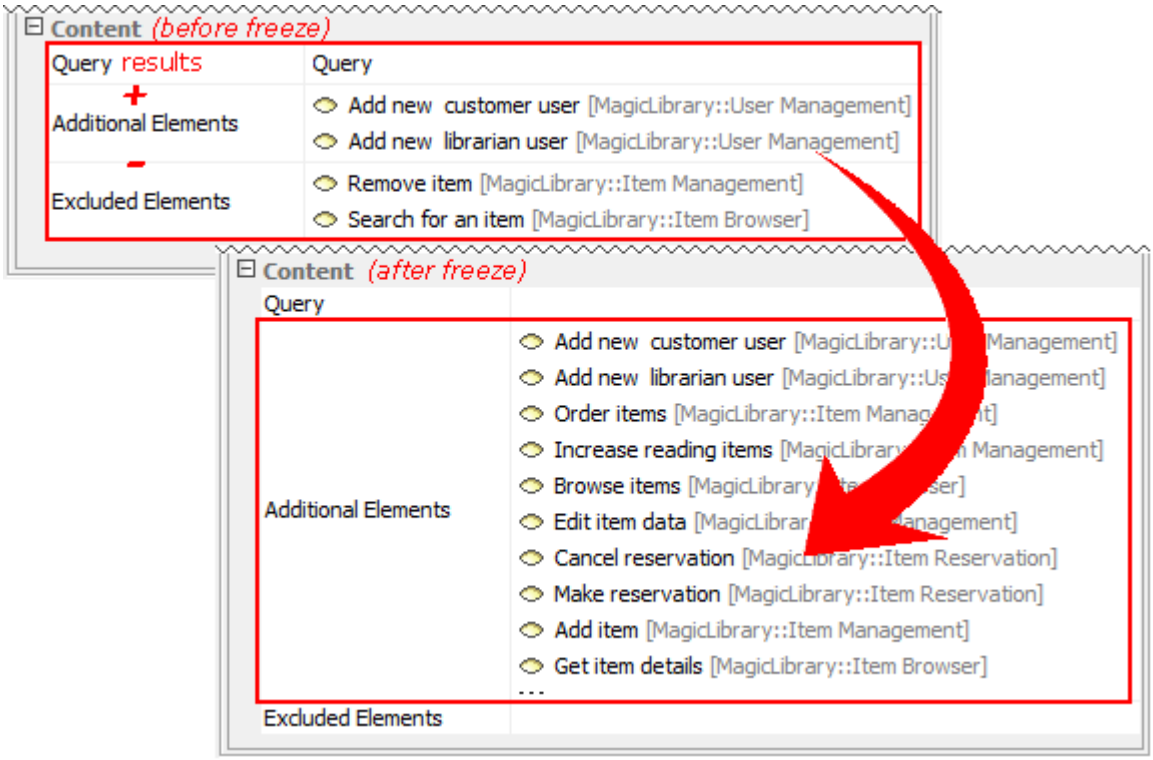


Figure 241 -- Counting Additional Elements property value after freeze

Making a snapshot of the smart package

You can create a copy of the smart package, with the static contents, that is, a snapshot.

To make a snapshot of a smart package

1. In the Containment tree, right-click the smart package with dynamic contents and from the shortcut menu select **Snapshot Contents**. The element Selection dialog opens.
2. In the dialog, select an owner of the new smart package and click **OK**.

   See the new smart package created under the selected owner in the Containment tree with the name *Snapshot of <smart package name>*, for example, *Snapshot of My Bookmarks*. The contents of the snapshot equals the contents of the smart package, but the list of elements is static.

Copyright © 1998-2015 No Magic, Inc.

Smart Package Properties

Every smart package has several properties that allow for specifying the contents of the smart package. These properties are described in the following table.

Property	Description
Query	Stores a set of criteria for automatic inclusion of elements into the contents of the smart package. For more information, see "To include elements into a smart package automatically" on page 405.
Additional Elements	Stores the list of the elements manually included into the contents of the smart package. For more information, see "To include an element into a smart package manually" on page 405.
Excluded Elements	Stores the list of the elements excluded from the contents of the smart package. For more information, see "To remove an element from a smart package" on page 406.

Using Smart Packages in Your Model

For better understanding the usability of the smart packages, analyze the following case studies.

Case study #1: Gathering use cases



The case study uses the sample project *use case diagram.mdzip*, which can be found in *<MagicDraw installation folder>\samples\diagrams*.

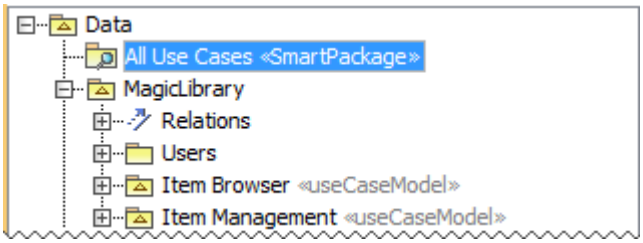
Let's say we have a lot of use cases owned by different system boundaries in the project.

And we need to have all these use cases, except several particular ones, in a single package. Also, we need that all newly created use cases were automatically included into this package.

This is the case, when the smart packages feature is very useful. We will create a smart package with dynamic contents to gather all the use cases in the model and then demonstrate how it manages further changes in the model. Finally we will create a snapshot of the smart package to have a a static list of use cases as a milestone of the model development.

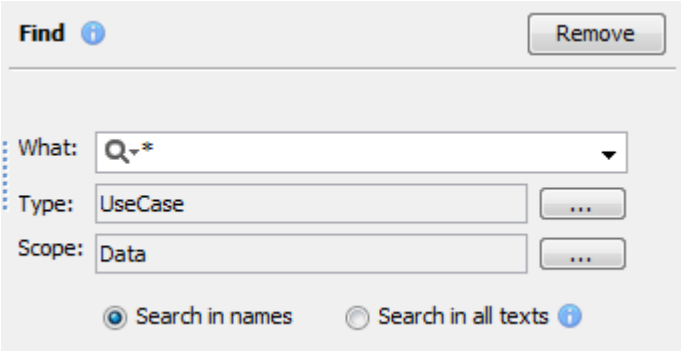
Let's do the following:

1. Create a smart package named *All Use Cases*.

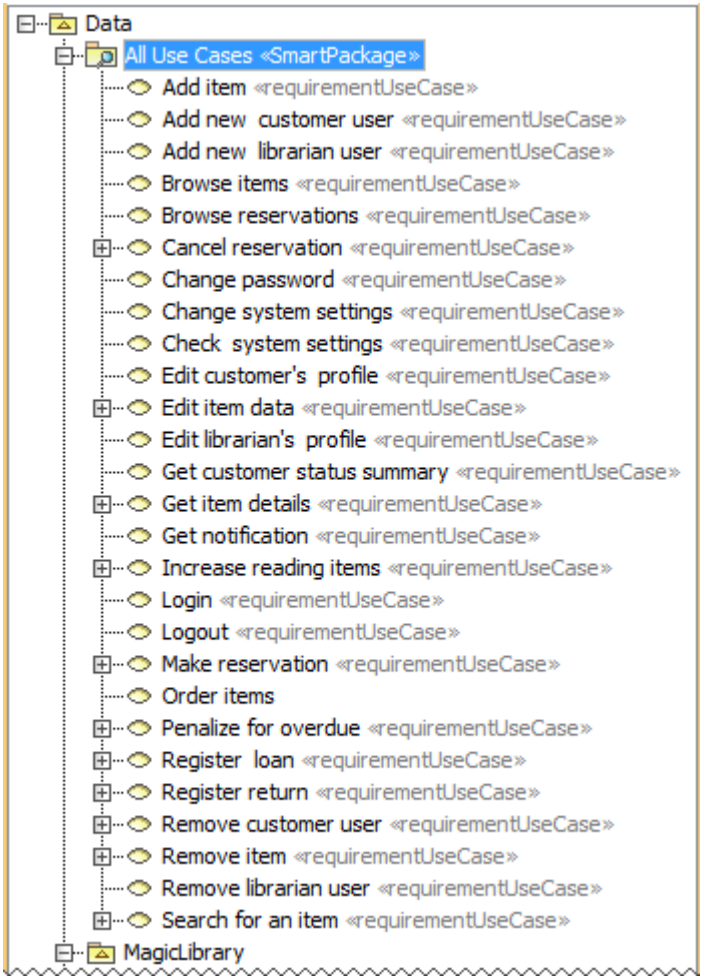


Copyright © 1998-2015 No Magic, Inc.

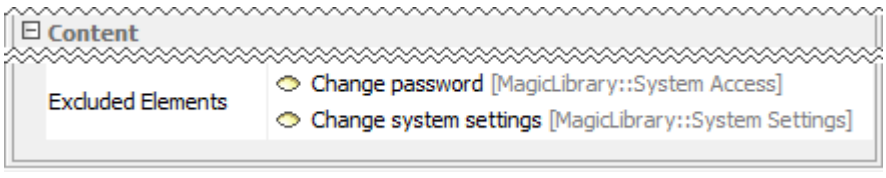
2. Define criteria for gathering the contents of the smart package. Specify search options to find all use case type elements from the root package *Data*.



3. Expand the contents of the smart package *All Use Cases.*



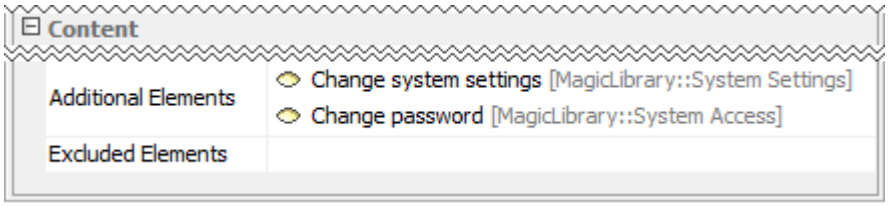
4. Exclude use cases *Change password* and *Change system settings* from the contents. See the contents of the smart package shortened.
5. Open the Specification window of the smart package and see the excluded use cases in the cell of the **Excluded Elements** property value.



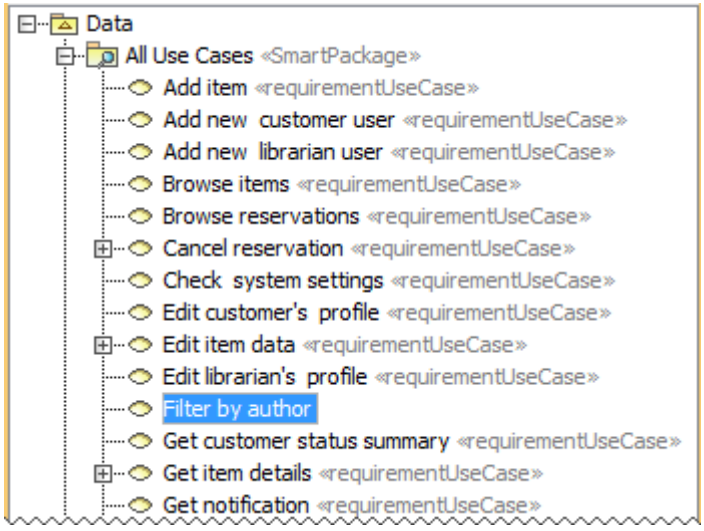
6. Drag use cases *Change password* and *Change system settings* to the smart package. See them in the contents of the smart package again.

Copyright © 1998-2015 No Magic, Inc.

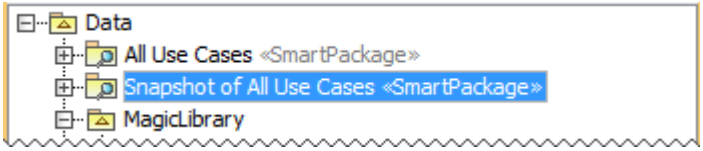
7. Open the Specification window of the smart package again and see these manually included use cases in the cell of the **Additional Elements** property value. Elements that have been excluded from the contents, appear in it after being manually added.



8. Create a new use case named *Filter by author* under the system boundary *Item Browser*. See the new use case appeared in the smart package *All Use Cases*.



9. Create a snapshot of the smart package *All Use Cases* to have a static list of use cases as a milestone of the model development.



10. Expand the contents of the snapshot and see that it equals the contents of the smart package *All Use Cases*.

Case study #2: Performing the requirements coverage analysis



The case study uses the sample project *hybrid sport utility vehicle.mdzip*, which can be found in *<MagicDraw installation folder>\samples\SysML*, if the SysML plugin is installed.

Now let's study another case where the smart package feature is very helpful. Let's say we need to have all unsatisfied requirements in a separate package. Also, we need that requirements automatically disappear from this package after becoming satisfied.

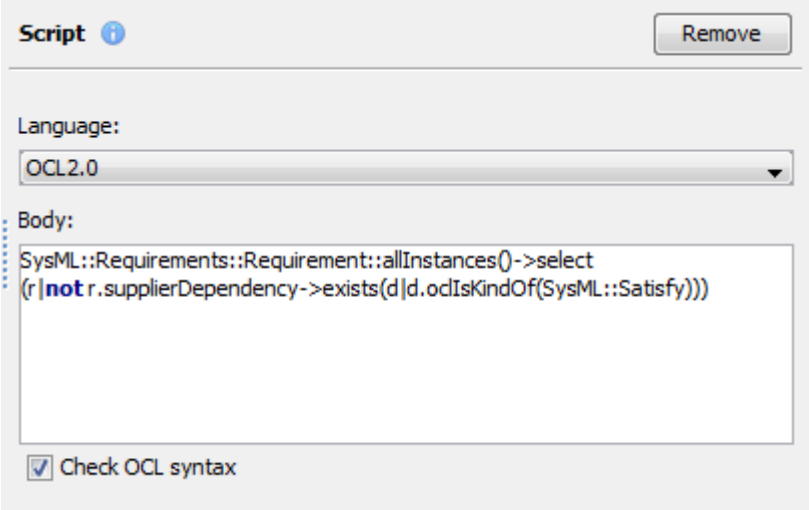
We will create a smart package with dynamic contents to gather all the unsatisfied requirements in the model and a dependency matrix for performing the requirements coverage analysis. Then we will demonstrate how both the smart package and the dependency matrix reflect the transition of a requirement to satisfied.

Let's do the following:

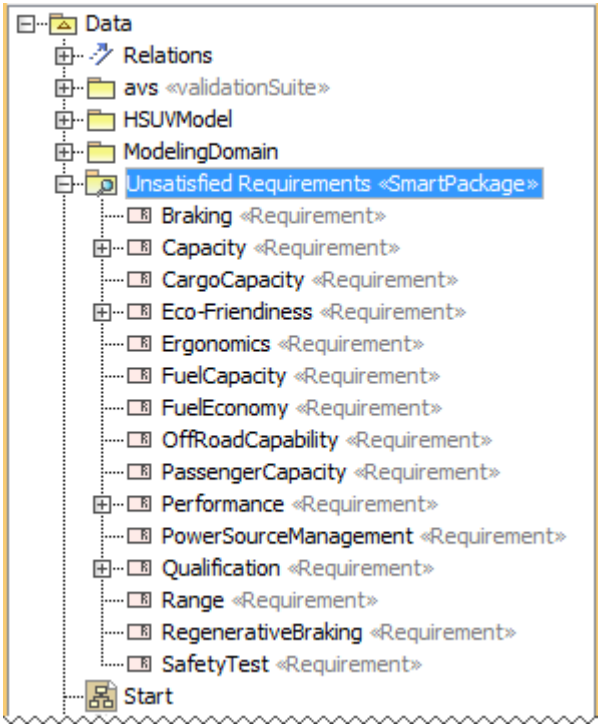
1. Create a smart package named *Unsatisfied Requirements*.
2. For gathering the contents of the smart package, add a new script operation (in the **Expert** mode of the **Query** dialog) and define the following OCL 2.0 expression as the criteria:

Copyright © 1998-2015 No Magic, Inc.

```
SysML::Requirements::Requirement::allInstances()->select(r|not
r.supplierDependency->exists(d|d.oclIsKindOf(SysML::Satisfy)))
```



3. Expand the contents of the smart package *Unsatisfied Requirements*.

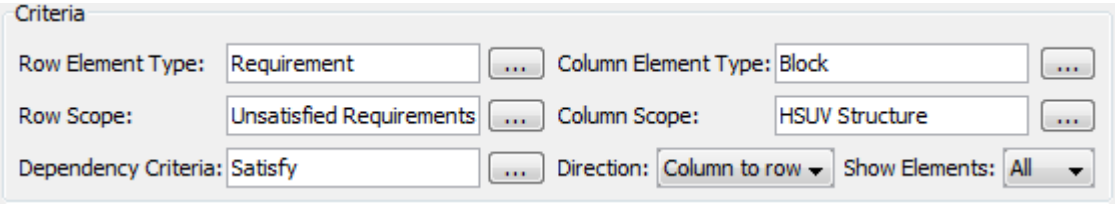


4. Create a dependency matrix and define the following criteria:

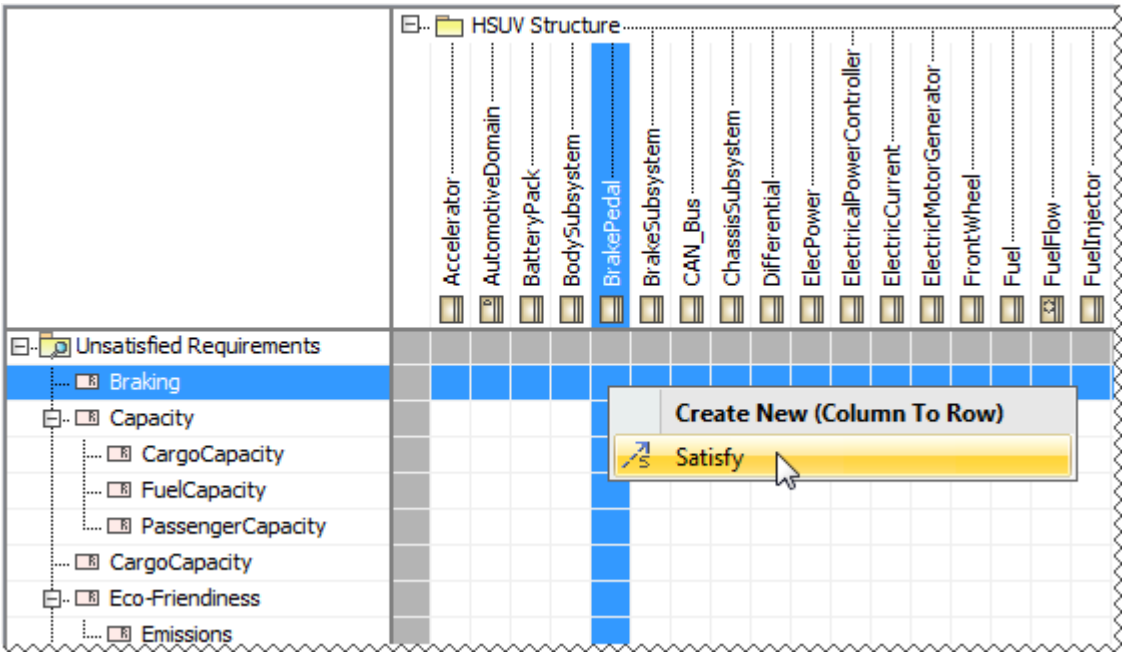
   - Specify Requirement as the row element type
   - Specify Block as the column element type
   - Specify the smart package *Unsatisfied Requirements* as the row scope
   - Specify the package *HSUV Structure* as the column scope
   - Specify the Satisfy relationship as dependency criteria

Copyright © 1998-2015 No Magic, Inc.

- In the **Direction** drop-down list, select **Column to row**.



5. On the dependency matrix, create a Satisfy relationship between the block *BrakePedal* and the requirement *Braking*.



The requirement becomes satisfied and thus disappears from the contents of the smart package *Unsatisfied Requirements* and from the dependency matrix as well.

Case Study #3: Configuration management of the complex system - creating dynamic configuration catalogs

The efficient configuration management process is a challenge in the evolution of any industrial scale product family. Smart packages are a real-life out of the box solution supporting the configuration management approach.

Now let's study the case that illustrates the efficient management of the complex system configurations with the help of smart packages. Let's say we have a library (static package) of system components, which we need to

Copyright © 1998-2015 No Magic, Inc.

see in several different views of the model, that is, catalogs, according to their characteristics. Using the catalogs will not extend the scope of the model, since they do not require the duplication of the elements.

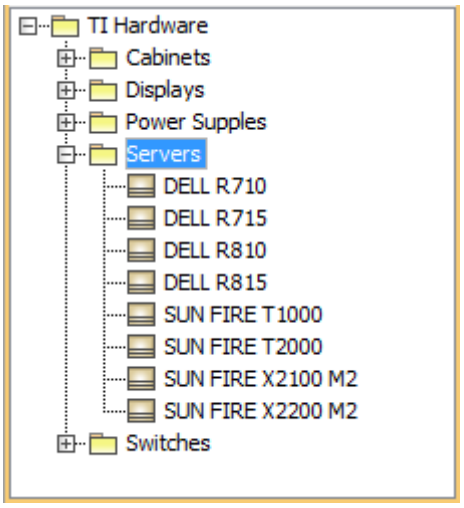
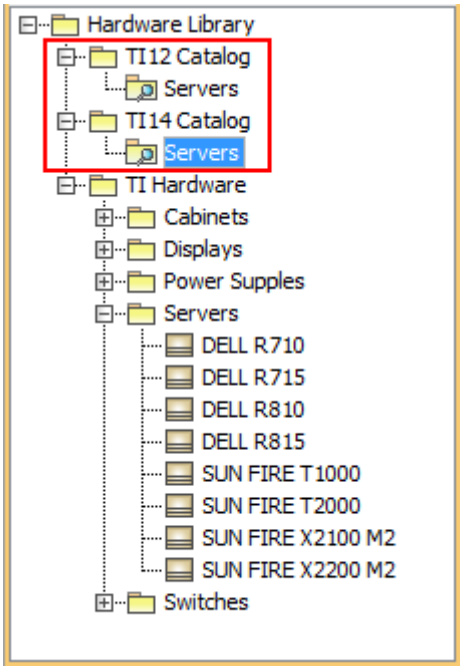


Figure 242 -- Library "TI Hardware" of system components

We will create two dynamic system configuration catalogs, that is, smart packages with dynamic contents to gather the servers from the library *TI Hardware* according to the configuration version defined in a tag value of their specification.

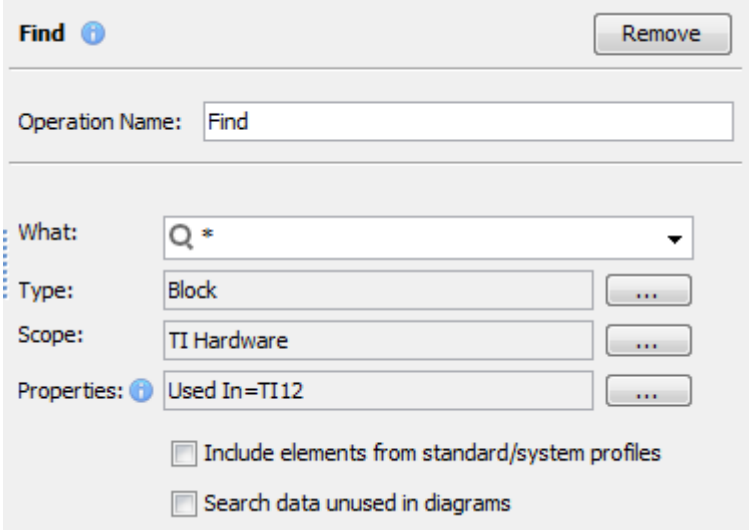
Let's do the following:

1. Create two packages: *TI12 Catalog* and *TI14 Catalog*.
2. In each package, create a smart package named *Servers*.

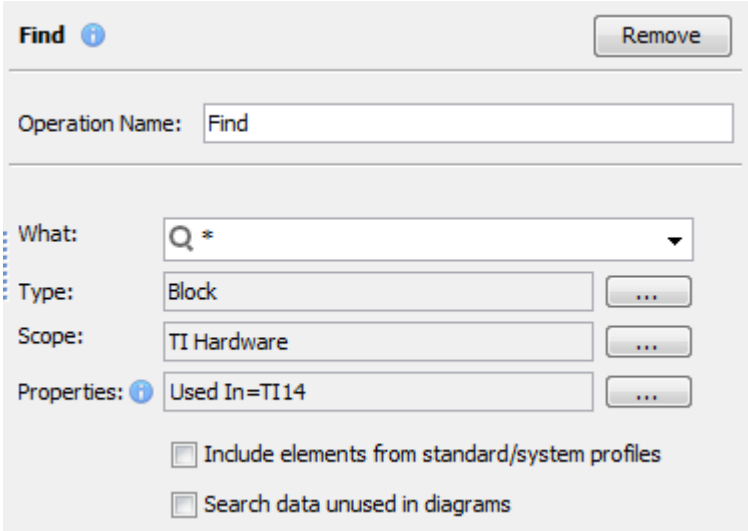


Copyright © 1998-2015 No Magic, Inc.

3. Define criteria for gathering the contents of the smart package *Servers* in *TI12 Catalog*. Specify search options to find in the package *TI Hardware*, all block type elements with tag value *Used In=TI12*.

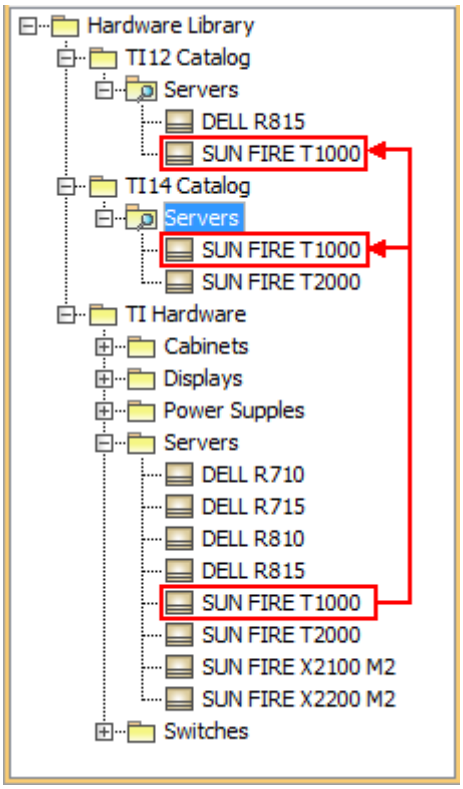


4. Define criteria for gathering the contents of the smart package *Servers* in *TI14 Catalog*. Specify search options to find in the package *TI Hardware*, all block type elements with tag value *Used In=TI14.*



Copyright © 1998-2015 No Magic, Inc.

5. Expand the contents of both smart packages.



NOTE

The block *SUN FIRE T1000* appears in both *TI12 Catalog* and *TI14 Catalog*, since it has both tag values *Used In=TI12* and *Used In=TI14*.

6. Also, you can add a block to a smart package manually. Just drag the block to the smart package.

Copyright © 1998-2015 No Magic, Inc.

Favorites

You can mark as *favorite* any element, including diagrams, types, packages, and so forth that you regularly use in your model and need to access quickly.

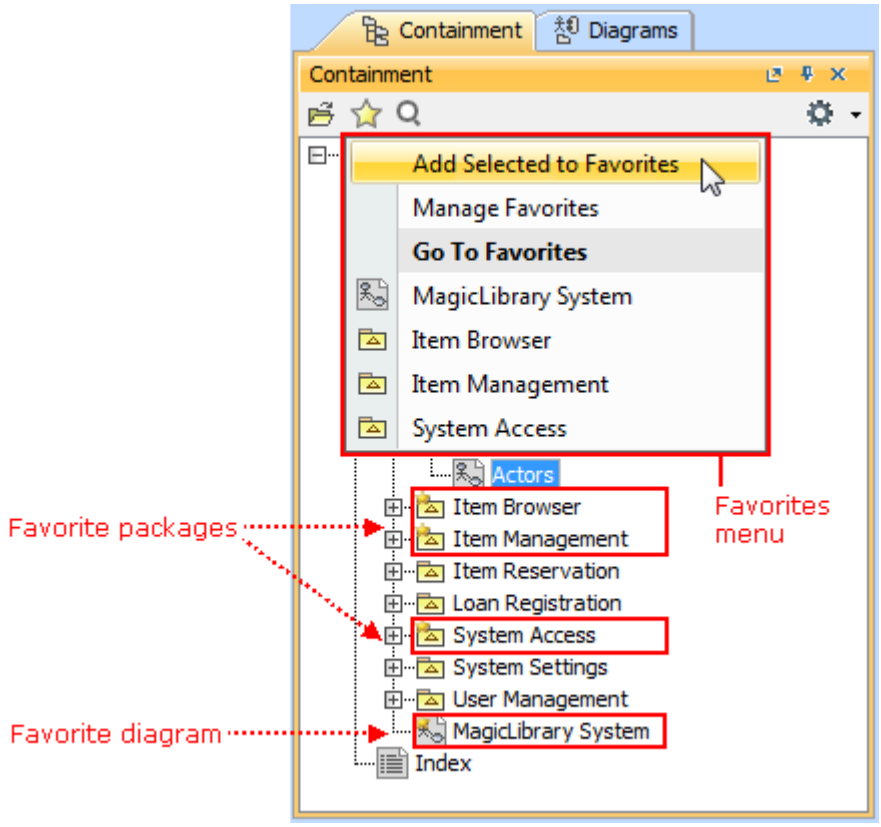


Figure 243 -- Favorites in Containment tree

Using favorites saves your time as you are able to

- Navigate quickly to any desired place in your model.
- Easily narrow the search scope to get rid of unneeded elements and thus find desired elements faster.
- Easily narrow the scope for model analysis.

In a server project, each user has his own list of favorites.

Learn more about using favorites in

- "Adding, Removing, and Changing Order of Favorites" on page 416
- "Manage Favorites dialog" on page page 419

Adding, Removing, and Changing Order of Favorites

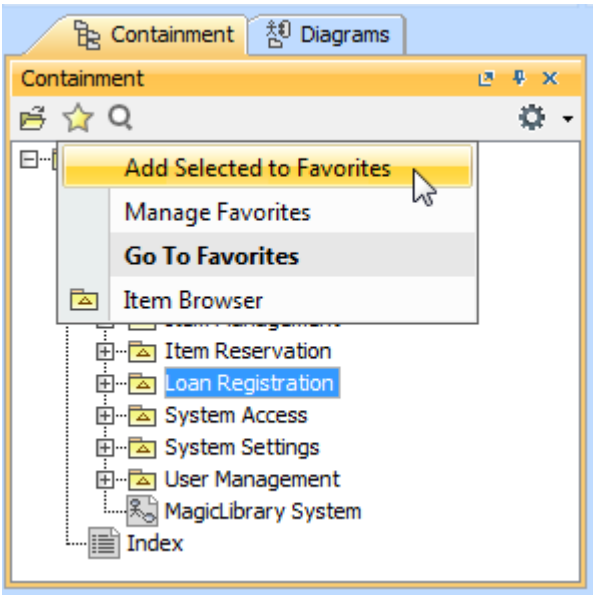
There are a few ways of adding an element to favorites:

- By using the **Add Selected to Favorites** command from the Favorites menu. This way is especially handy for the quick addition to favorites.
- By using the **Manage Favorites** dialog. Choose this way, when you need to add and then perform another manipulations with favorites.

Copyright © 1998-2015 No Magic, Inc.

To quickly add an element to favorites

1. Select the element in the Model Browser and click ☆ on the toolbar of the open tab.



2. Click **Add Selected to Favorites** on the menu.

To add an element to favorites in the **Manage Favorites** dialog

1. Open the **Manage Favorites** dialog.

TIP! To learn the ways of opening the dialog, see the procedure "To open the Manage Favorites dialog" on page 420.

2. In the **Manage Favorites** dialog, select the element and click the + button, which is between two lists of the dialog. The element appears in the list of favorites.

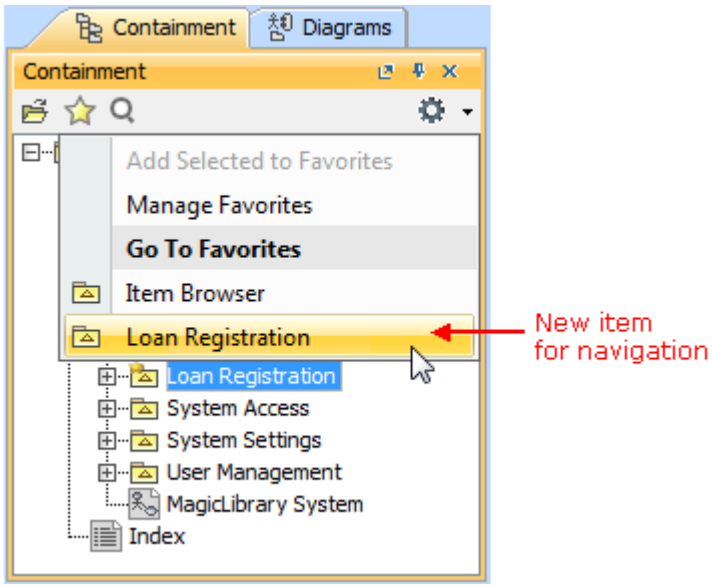
TIP! For more information about the dialog, refer to "Manage Favorites dialog" on page 419.

3. Click **OK** to confirm the addition and close the dialog.

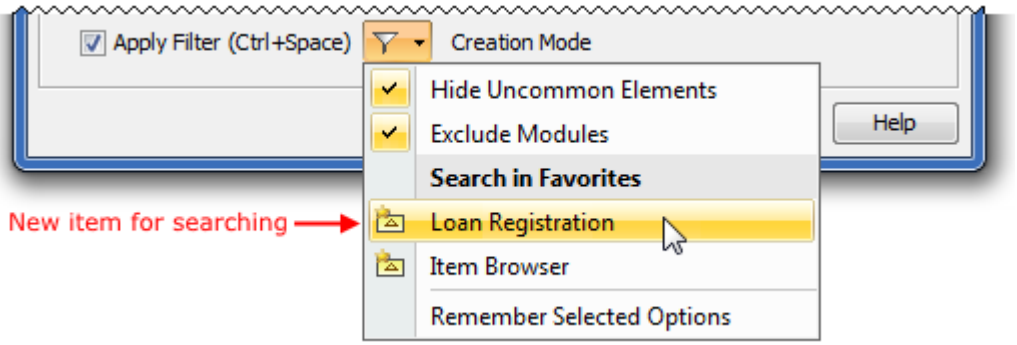
Copyright © 1998-2015 No Magic, Inc.

Once the element is added to favorites, the following changes occur in your project:

- The element becomes marked as favorite in all the trees and lists of the model.
- You can easily navigate to the element in the Model Browser. The element name is added to the **Go To Favorites** command group on the Favorites menu that opens in the Model Browser.



- You can easily specify the element (in case it is a diagram or a package) as the search scope. The element name is added to the **Search in Favorites** command group on the Favorites menu that opens either in the element Selection dialog or in the autocompletion list.



To remove an element from favorites

1. Open the **Manage Favorites** dialog.

   TIP! To learn the ways of opening the dialog, the procedure "To open the Manage Favorites dialog" on page 420.

2. In the list of favorites (on the right side of the dialog), click the element you need to remove and then click the  button.

   TIP! For more information about the dialog, refer to "Manage Favorites dialog" on page 419.

3. Click **OK** to confirm the removal and close the dialog.

Once the element is removed from favorites, it is no longer appropriately marked. Commands for navigation to the element and for searching in it are no longer on the Favorites menu.

Copyright © 1998-2015 No Magic, Inc.

The following procedure describes how to change the order of favorite elements on the Favorites menu. To change the order of favorites on the Favorites menu, you need to make appropriate order changes in the list of favorites on the **Manage Favorites** dialog.

To change the order of favorites

1. Open the **Manage Favorites** dialog.

   TIP! To learn the ways of opening the dialog, see the procedure "To open the Manage Favorites dialog" on page 420.

2. In the list of favorites (on the right side of the dialog), click an element, whose order on the Favorites menu you need to change and then do either:
   - If you need to move the element up, click the **Up** button. Click again, if you need to move the element up by one more position and so on.
   - If you need to move the element down, click the **Down** button. Click again, if you need to move the element down by one more position and so on.
3. Click **OK** to confirm the reorder and close the dialog.

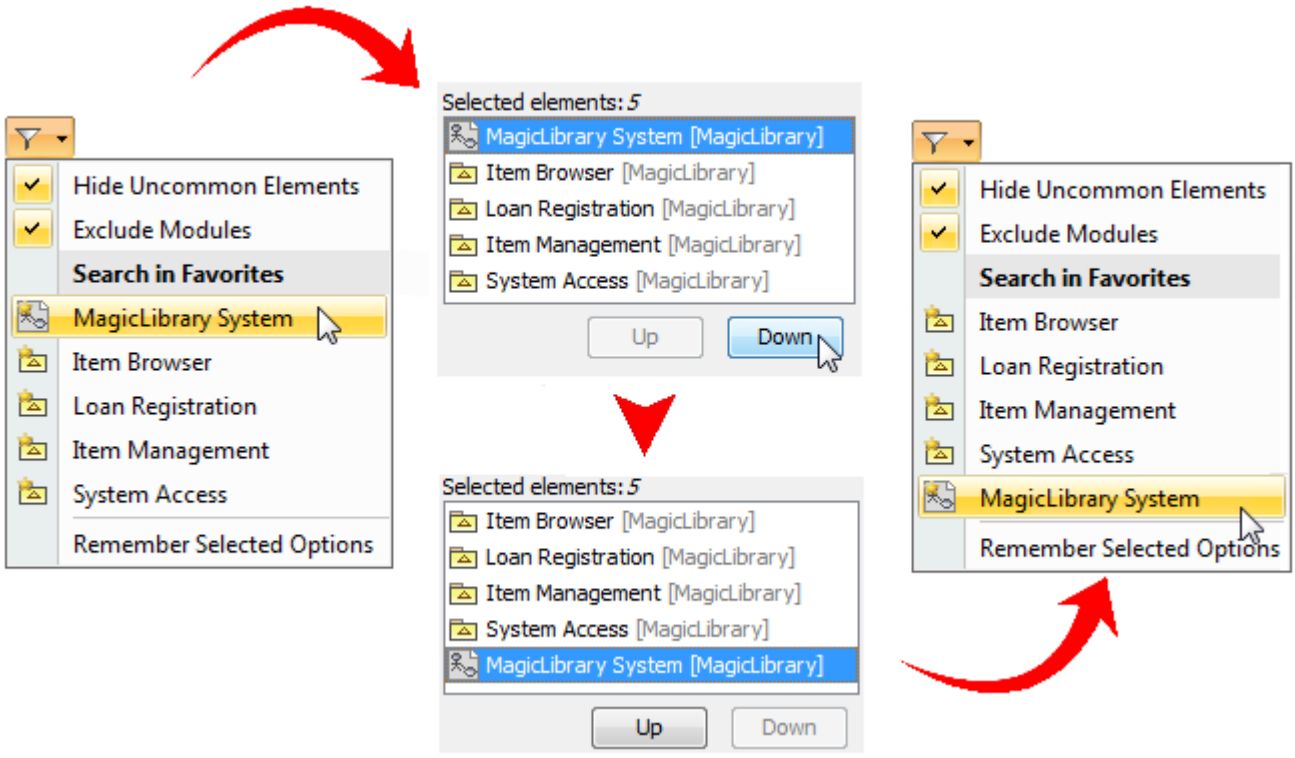


Figure 244 -- Changing order of favorites: moving diagram "MagicLibrary System" down

Manage Favorites dialog

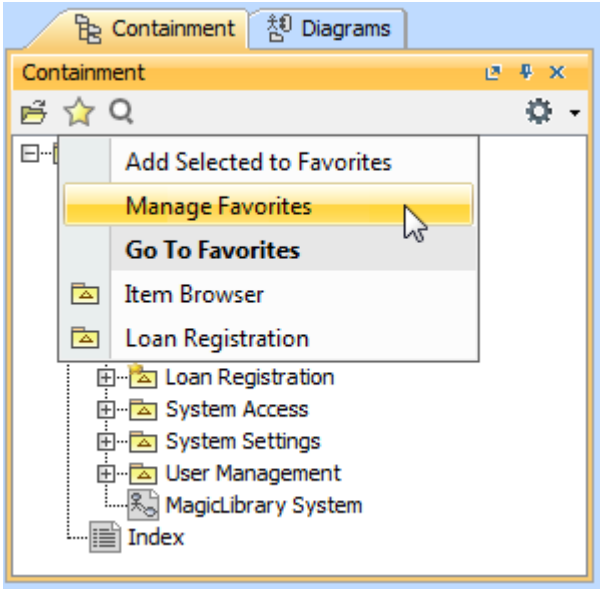
The **Manage Favorites** dialog allows for

- Adding to favorites
- Removing from favorites
- Changing the order of favorites

Copyright © 1998-2015 No Magic, Inc.

To open the **Manage Favorites** dialog

- In the Model Browser, click the button, which is on the toolbar of the open tab, and then select **Manage Favorites**.



The **Manage Favorites** dialog is a modification of the element Selection dialog. The dialog displays

- All elements (in tree or list view on the left side of the dialog)
- Favorite elements (in the list on the right side of the dialog)
- Buttons for adding and removing from favorites
- Search modes and scope filters area
- Buttons for reordering favorites

Copyright © 1998-2015 No Magic, Inc.

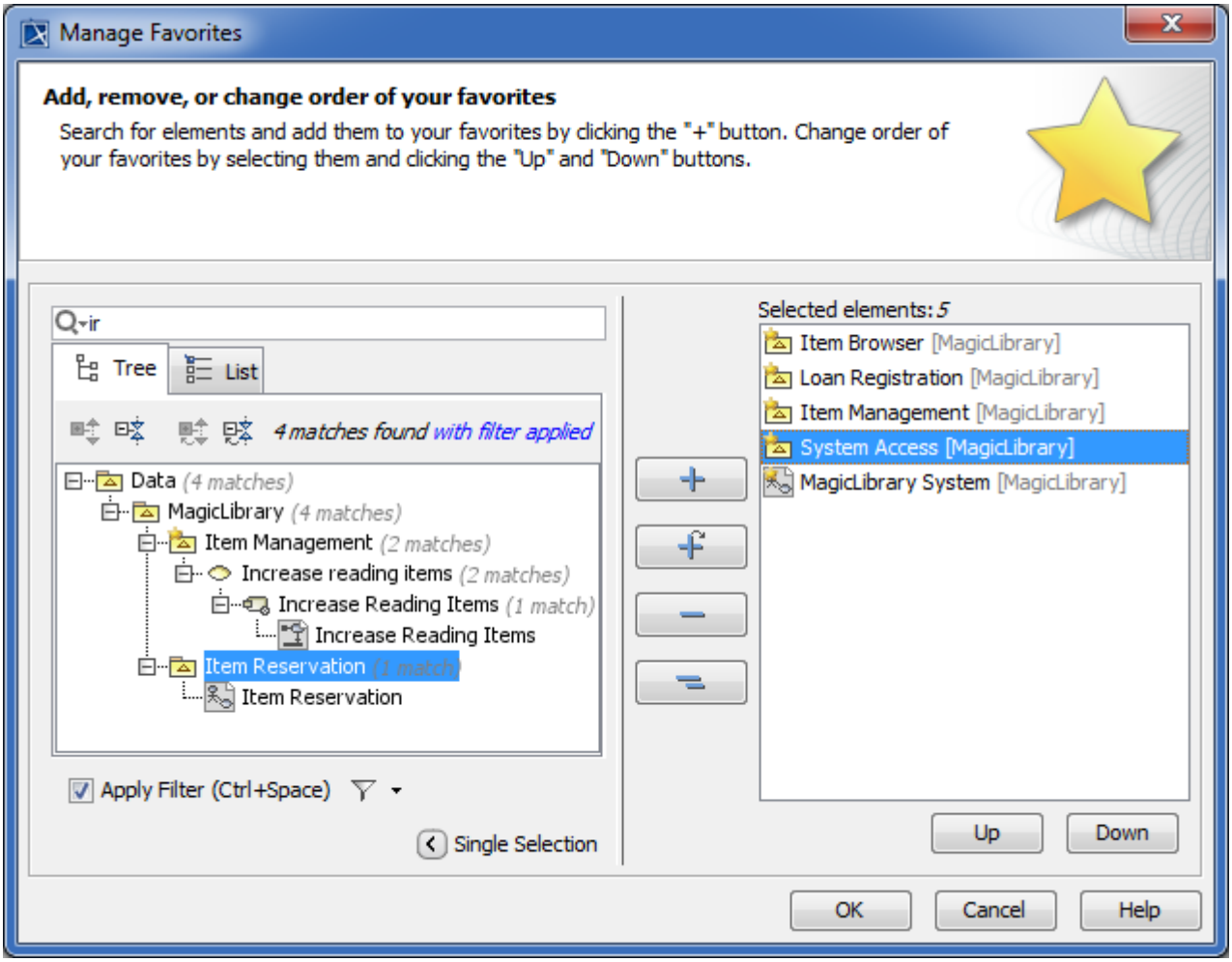


Figure 245 -- Manage Favorites dialog

The order of favorites in the list of the **Manage Favorites** dialog meets the order of favorites on the Favorites menu. Thus if you need to change the order of favorites on this menu, make appropriate order changes in the list of favorites on the **Manage Favorites** dialog. Buttons below the list will help you to change the order of favorites. See their descriptions in the following table.

Button	Description
Up	Click to move a selected element up. The element will accordingly change its position on the Favorites menu. For more information, see "To change the order of favorites" on page 419.
Down	Click to move a selected element down. The element will accordingly change its position on the Favorites menu. For more information, see "To change the order of favorites" on page 419.

TIP!

For the descriptions of other dialog elements, see

- "Elements Multiple Selection" on page page 354
- "Searching for Elements in Element Selection Dialog" on page page 355

Copyright © 1998-2015 No Magic, Inc.

HTML Editor

Using the HTML editor you can easily edit an HTML text. It also allows you to preserve the text format when copying formatted text.

To set the note / text box / separator text as HTML

- Select a shape and click Switch To HTML Text that appears on the lower-left corner of the shape.



- From the note / text box shortcut menu, select **HTML Text**.
- Draw the note or text box, using the **Note(HTML text)** or **Text Box(HTML text)** buttons on the diagram pallet.

Click the text area on the selected shape or start typing letters to open the HTML editor toolbar. For more information about the toolbar buttons, see "HTML editor toolbar" on page 427.



Figure 246 -- HTML editor toolbar

You can write HTML text in various dialogs. To start doing this, you have to turn on the HTML mode first.

To turn on the HTML mode in a dialog

- Click to select the **HTML** check box.
  These are the samples of the **HTML** check box and HTML editor toolbar in various dialogs:
  - Element Specification window, the **Documentation/Hyperlinks** tab (Figure 247 on page 423).
  - In the Model Browser, the **Documentation** panel (Figure 248 on page 424).
  - Element Specification window, the **Tags** property group, when editing a tagged value (Figure 249 on page 425).

Copyright © 1998-2015 No Magic, Inc.

- When specifying the **To Do** property in the element Specification window (Figure 250 on page 426).

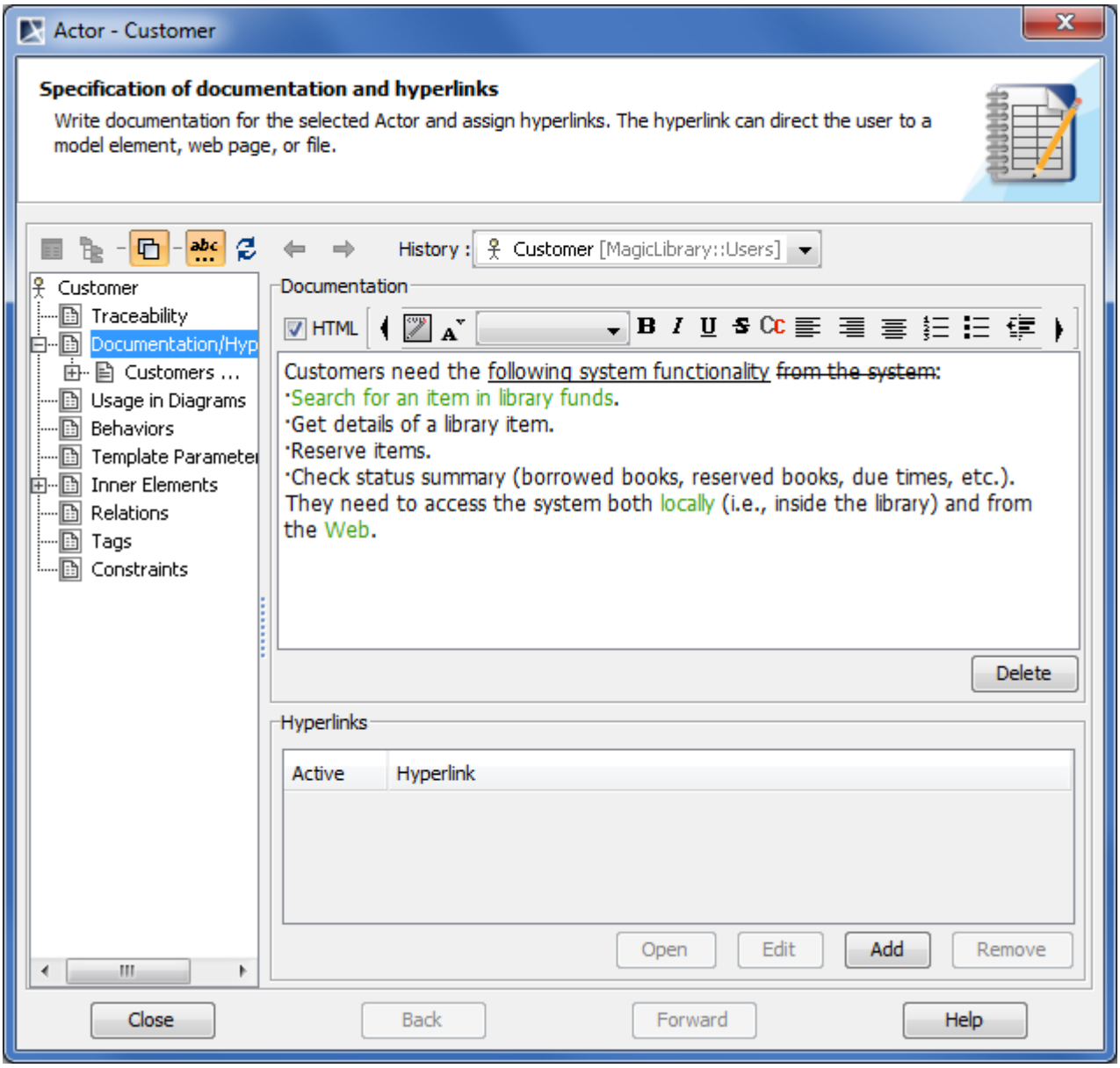


Figure 247 -- HTML editor toolbar in Documentation/Hyperlinks tab of element Specification window

Copyright © 1998-2015 No Magic, Inc.

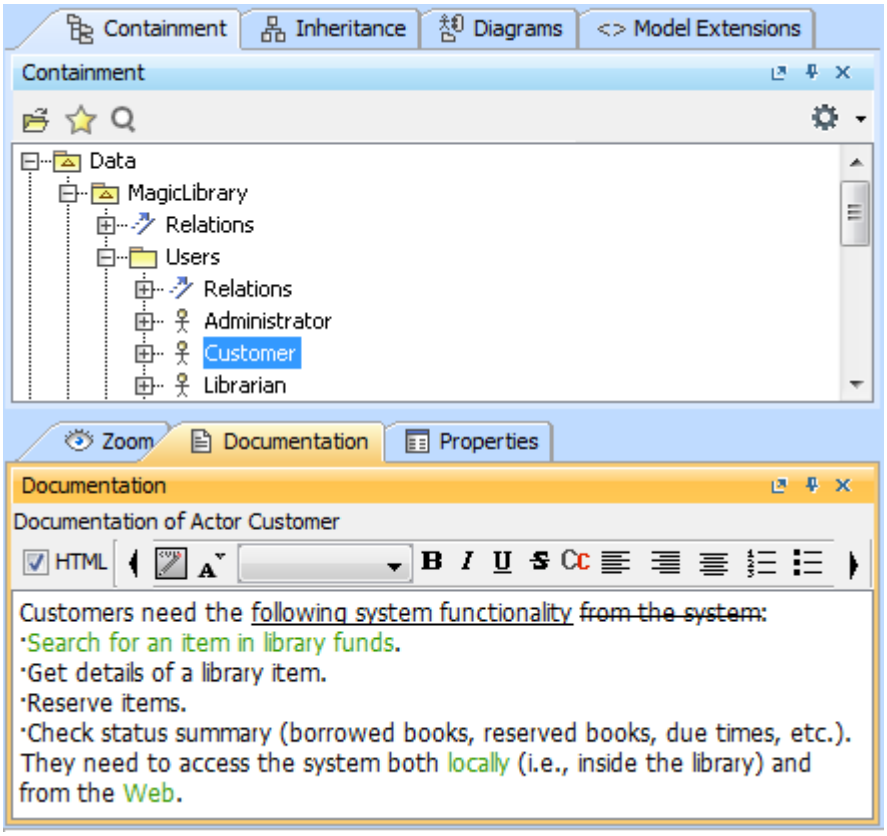


Figure 248 -- HTML editor toolbar in Documentation panel on Model Browser

Copyright © 1998-2015 No Magic, Inc.

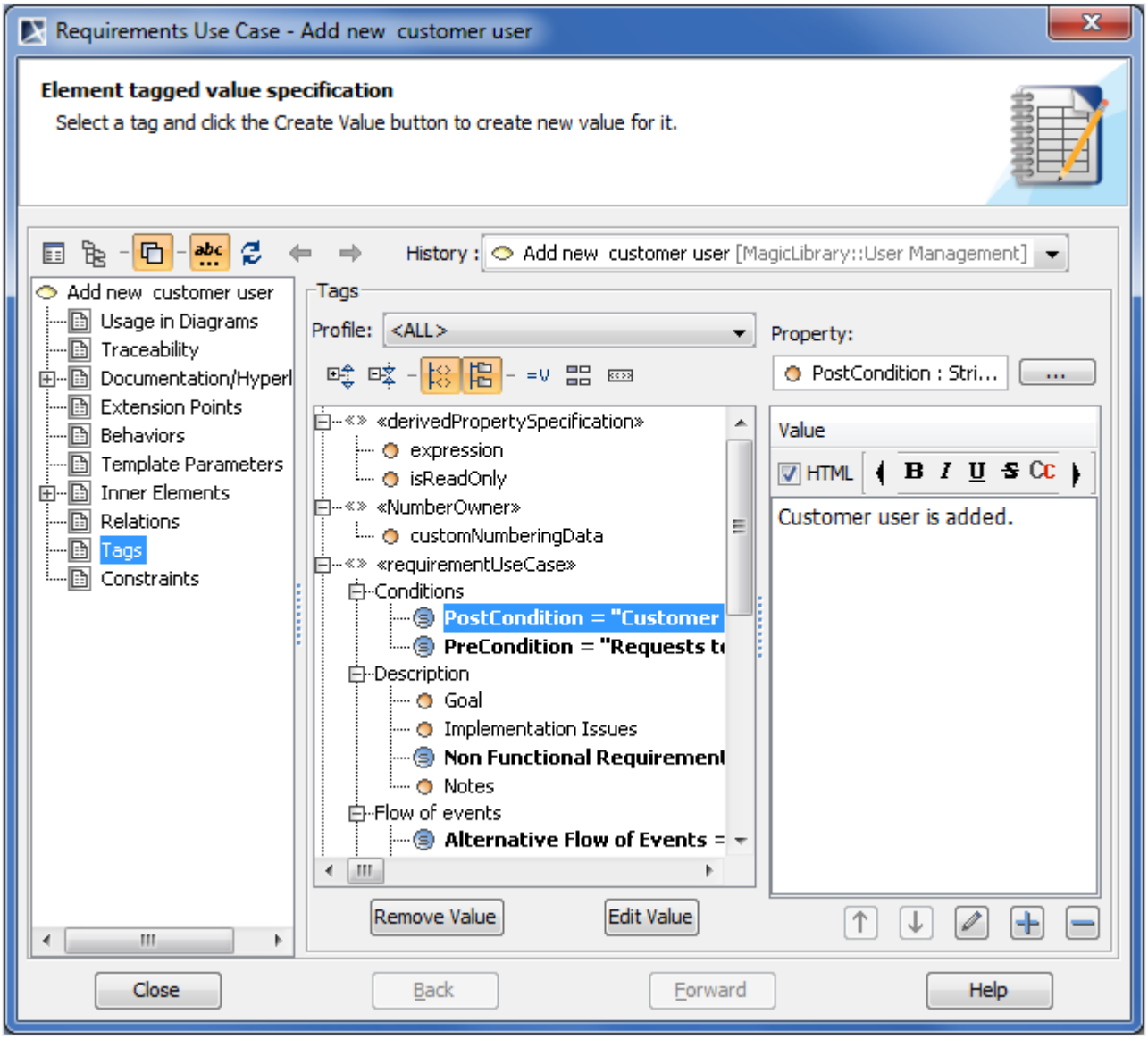


Figure 249 -- HTML editor toolbar in Tags tab of element Specification window when editing tag value

Copyright © 1998-2015 No Magic, Inc.

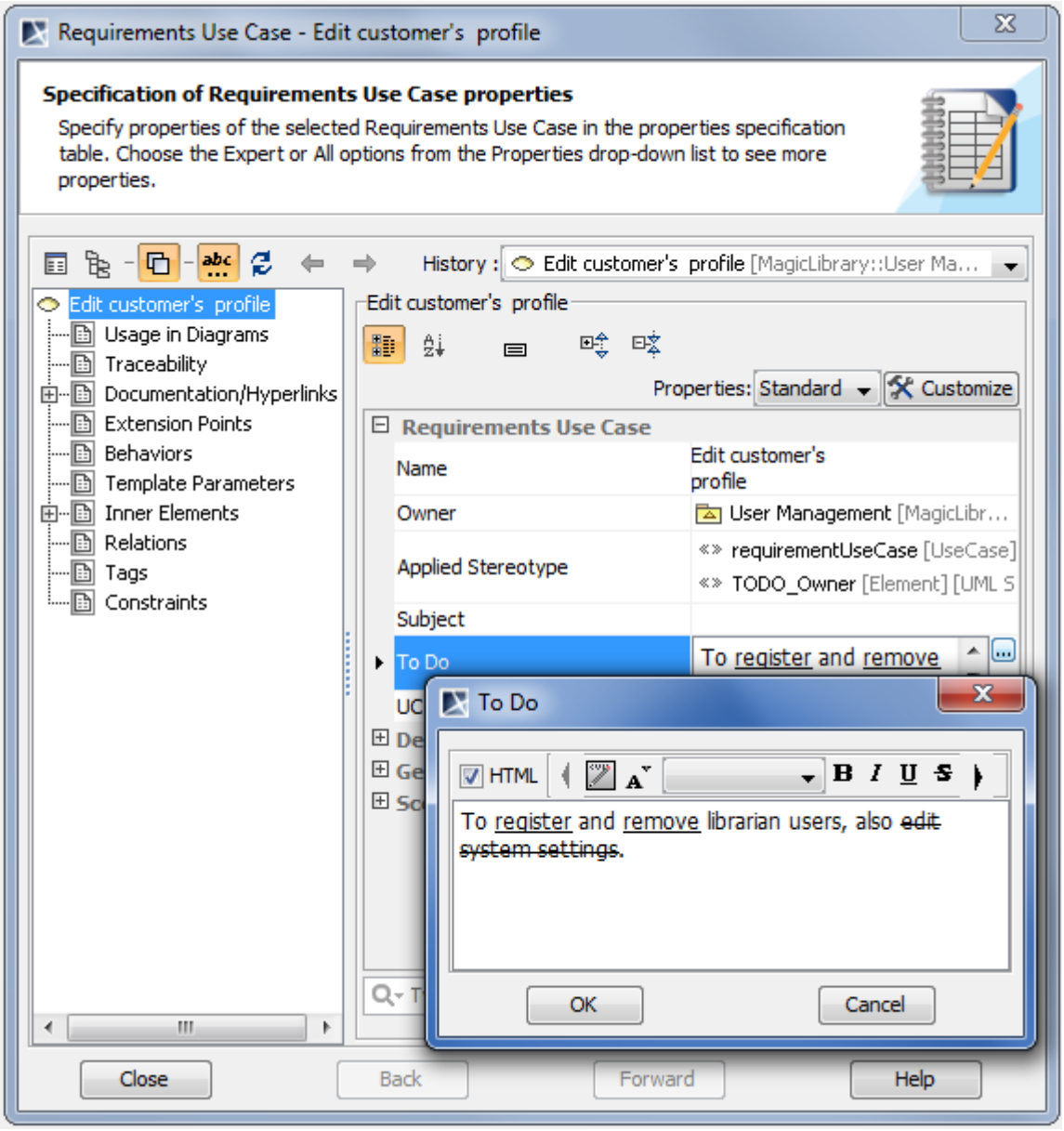


Figure 250 -- HTML editor toolbar in To Do dialog

You can also use the HTML editor toolbar, when editing a tagged value directly on the element's shape.

To use the HTML editor toolbar for editing a tagged value

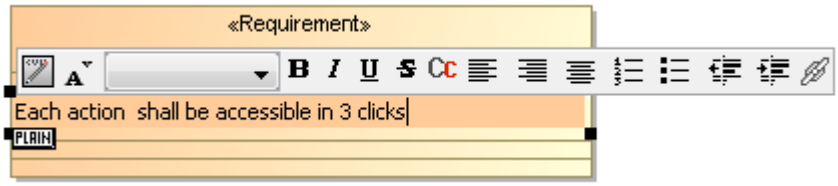
1. Click the tagged value on the element's shape as it is shown in the following picture.



2. Click the tagged value once again. The HTML mode will be turned on and the HTML editor toolbar will open for editing this tagged value.

Copyright © 1998-2015 No Magic, Inc.

HTML editor toolbar



Button	Description
Advanced HTML Editor	Edit text with advanced HTML editor. The **Advanced HTML Editor** dialog opens.
Font properties	Select font style of the text. The **Font Properties** dialog opens.
Font Size	Select font size of selected text.
Bold	Set text as bold.
Italic	Set text as italic.
Underline	Set text as underlined.
Strikethrough	Set text as strikethroughed.
Foreground	Select font color of selected text.
Align left	Align selected text to the left side border.
Center	Center selected text.
Align Right	Align selected text to the right side border.
Numbering	Change text style to numbered list.
Bullets	Change text style to bullet list.
Decrease Indent	Decrease indent by moving text closer to the left border.
Increase Indent	Increase indent by moving text closer to the right border.
Insert Hyperlink	Add the hyperlink to a file, an address, an element, a symbol, or a requirement. The **Edit Hyperlink** dialog opens. For more information, see Section "Adding a hyperlink to the model element" on page 341.

Copyright © 1998-2015 No Magic, Inc.

Advanced HTML Editor dialog

To open the **Advanced HTML Editor** dialog

- Click the Advanced HTML Editor button on the HTML editor toolbar.

TIP! For the information how to turn on the HTML editor, see the procedure "To turn on the HTML mode in a dialog" on page 422.

In the **Advanced HTML Editor** dialog you can change the text style, insert symbols, images, and tables, as well as perform other actions using appropriate buttons.

Click the **HTML source** tab to view HTML source.

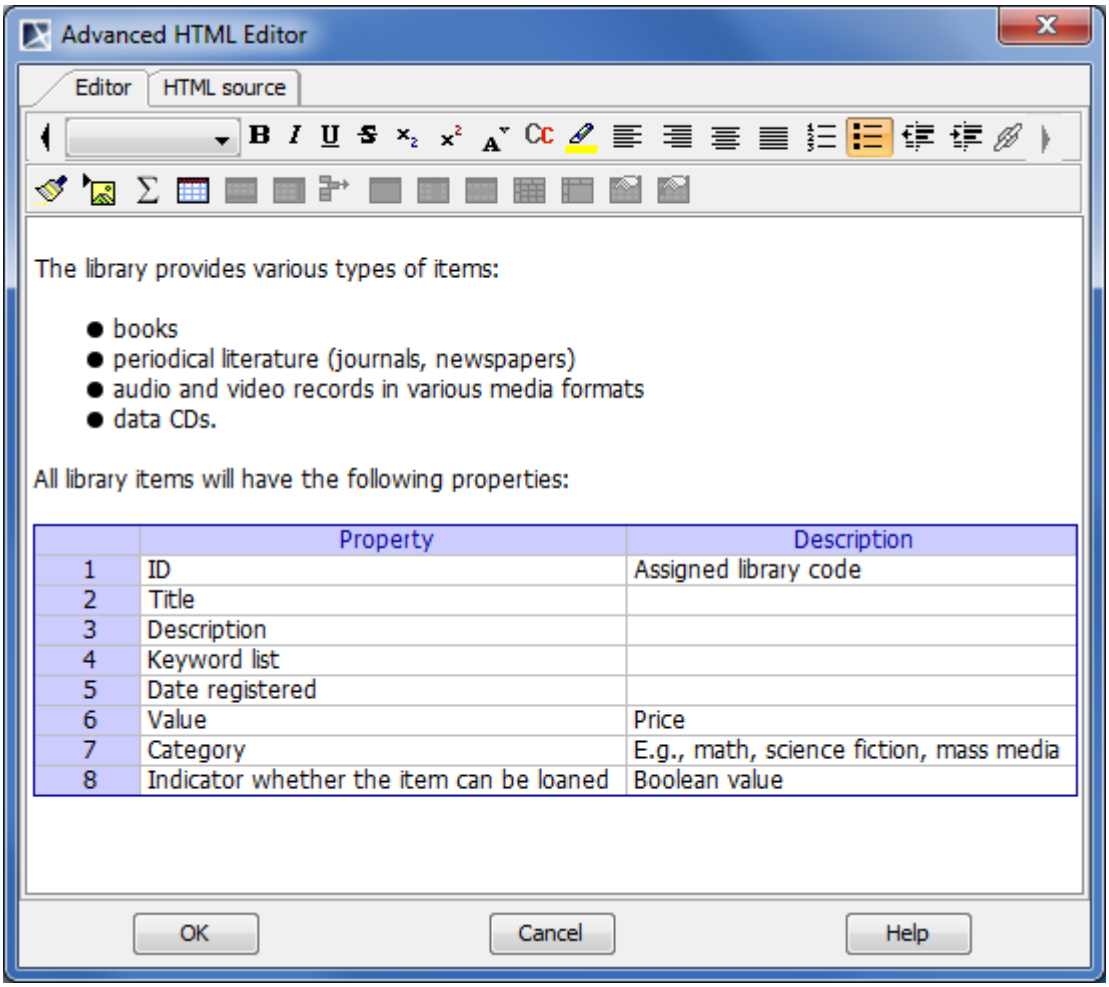
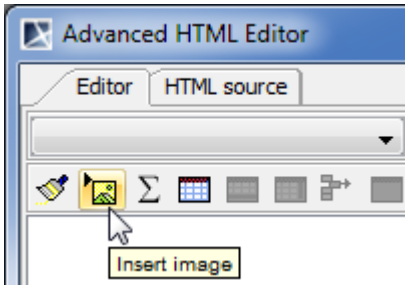


Figure 251 -- Advanced HTML Editor dialog

Copyright © 1998-2015 No Magic, Inc.

To insert an image into HTML text

1. Click the Insert image button as it is shown in the following picture.



2. In the **Picture Properties** dialog, specify the image location. Do one of the following:
   - Type the path to the image location.
   - Click **Browse** and browse to the image you want to insert.

IMPORTANT!

If you are working with a server project, make sure that the path to the image location is accessible from any computer that has Internet connection. Otherwise the image will not be displayed, when the project is opened on another computer.

   - **NEW!** If you want to insert an image, which is an attached file, simply paste its URL to the **Picture Source** box by pressing Ctrl+V (Cmd+V on OS X).

TIP!

For more information, refer to "Inserting into HTML text" on page 128.

3. Click **OK** when you are done.

The image is inserted into HTML text.

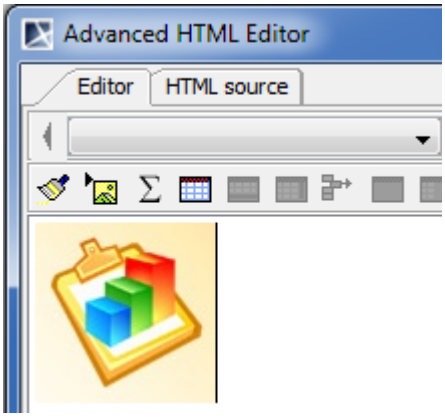


Figure 252 -- Image in HTML text

Copying/Opening Element URLs

You can now copy a project element URL to a clipboard and share it with others as a quick reference to model elements.

Copyright © 1998-2015 No Magic, Inc.

To copy a project element URL, do any of the following

- Select **Copy URL** from the element shortcut menu in the Containment tree to copy the URL to a model element.
- Select the element symbol in a diagram and click **Edit > Copy URL** on the main menu to copy the URL to element symbol.

You can open any elements through their URLs by clicking the **Open Element from URL** command and the element will be highlighted in the Containment tree or in the diagram. Custom URL "mdel://" is registered into windows registry. Activating the URL in other applications will allow you to start MagicDraw, open the project (if possible), and select any elements. You can paste URLs from the clipboard to any MagicDraw diagram. Hyperlinks also can hold URLs of any model elements.

Parameters Synchronization

The parameters synchronization keeps your model valid and synchronized with your changes. How the synchronization works? After you have created a new element or edited a property the associated elements and properties are updated according to your changes.

The following table lists the pairs of elements that are synchronized.

Synchronized Elements		The backward synchronization is available
Behavior (Parameter)	Operation (Parameter)	Yes
Operation (Parameter)	Call Operation Action (Pin)	Yes
Behavior (Parameter)	Call Behavior Action (Pin)	Yes
NEW! Signal (Attribute)	Send Signal Action (Pin)	Yes
NEW! Signal (Attribute)	Accept Event Action (Pin)	Yes
Activity (Parameter)	Activity (Activity Parameter Node)	Yes
NEW! Behavior (Parameter)	Signal Reception (Parameter	Yes
Operation (Parameter)	Message (Argument)	No
Interaction (Parameter)	Interaction Use (Argument)	No
NEW! Signal (Attribute)	Message (Argument)	No

The following table lists the synchronized element properties and conditions on which these properties are synchronized.

Synchronized Elements		Properties			
		Name	Type	Direction	Multiplicity
Behavior (Parameter)	Operation (Parameter)	always	always	always	always
Operation (Parameter)	Behavior (Parameter)	always	always	always	always
Operation (Parameter)	Call Operation Action (Pin)	when source and target have the same values	when source and target have the same values	always	when source and target have the same values
Behavior (Parameter)	Call Behavior Action (Pin)				

Copyright © 1998-2015 No Magic, Inc.

Synchronized Elements		Properties			
		Name	Type	Direction	Multiplicity
Call Operation Action (Pin)	Operation (Parameter)	when target is undefined or unnamed	when target is undefined or unnamed	always	when target is undefined or unnamed
Call Behavior Action (Pin)	Behavior (Parameter)				
Signal (Attribute)	Send Signal Action (Pin)	when source and target have the same values	when source and target have the same values	-	when source and target have the same values
Signal (Attribute)	Accept Event Action (Pin)				
Send Signal Action (Pin)	Signal (Attribute)	when target is undefined or unnamed	when target is undefined or unnamed	-	when target is undefined or unnamed
Accept Event Action (Pin)	Signal (Attribute)				
Activity (Parameter)	Activity (Activity Parameter Node)	when target is undefined or unnamed	always	always	-
Activity (Activity Parameter Node)	Activity (Parameter)	when target is undefined or unnamed	always	not synchronized	-
Behavior (Parameter)	Signal Reception (Parameter)	always	always	always	always
Signal Reception (Parameter)	Behavior (Parameter)	always	always	always	always

To turn on/off the parameters synchronization

1. On the **Options** menu, click **Project**.
2. In the **Project Options** dialog, in the **General project options** group, click to select or clear the **Synchronize Parameters and Arguments** option.

Resolving Not Synchronized Elements

The active validation functionality marks the not synchronized elements in the Model Browser and highlights the shapes on the diagram pane. For more information about the active validation and solving the incorrect model, see "Active Validation" on page 632.

In addition, you can solve the broken parameters synchronization by using the **Synchronization** dialog.

Synchronization Dialog Environment

The **Synchronization** dialog represents the elements those parameters are not synchronized and provides the possibility to restore the synchronization.

Copyright © 1998-2015 No Magic, Inc.

In this section, you will find the brief information on how to open the **Synchronization** dialog and on each button that can be useful while working in the dialog.

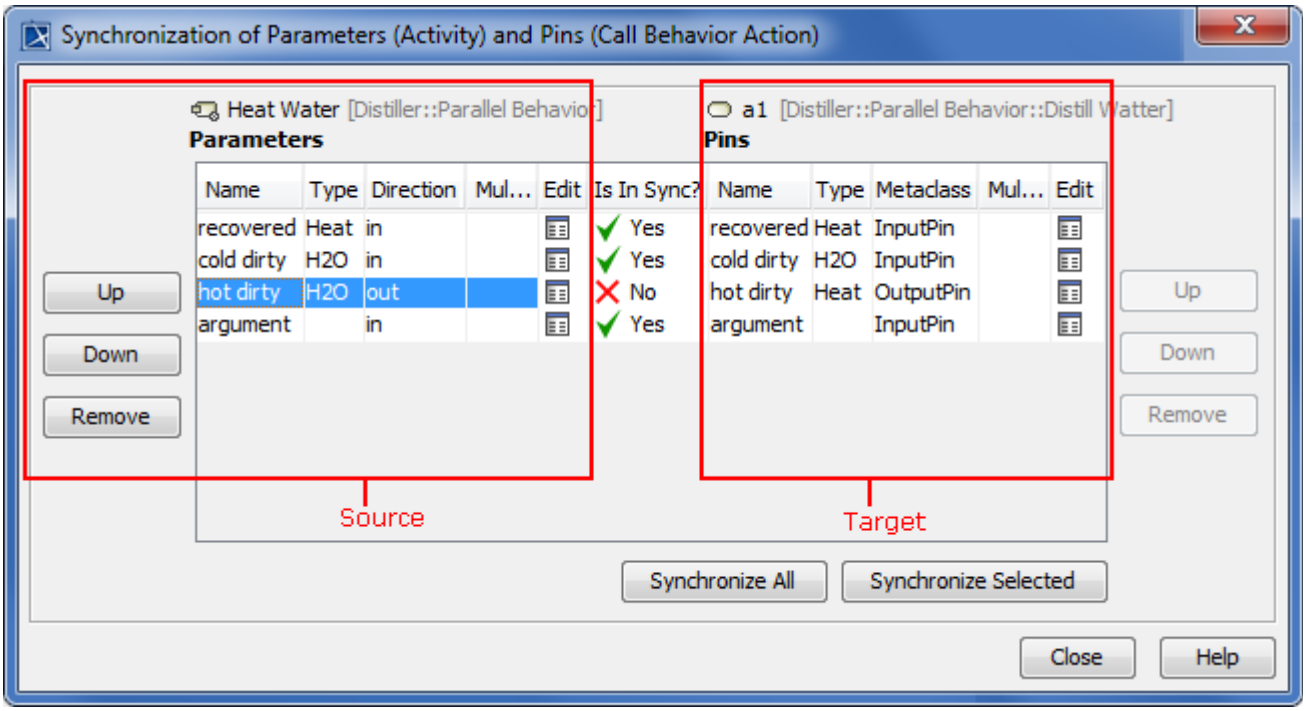


Figure 253 -- Synchronization dialog

To open the **Synchronization** dialog

Do one of the following:

- In the Model Browser, right-click the invalid element, on the shortcut menu, point to **Validation**, then to the validation group, and click the **Synchronize Manually** command.
- On the diagram pane, select the invalid shape, on the smart manipulator toolbar, click the ⚠ button, and then click the **Synchronize Manually** command.
- In the **Validation Results** window, click the **Solve** button and then click the **Synchronize Manually** command. For more information on how to open the **Validation Results** window, see "Validation Results Panel" on page 616.

The source and target elements area

The following table describes the buttons in the source and target areas of the **Synchronization** dialog.

Button	Description
Source elements area:	
Up	Click to shift the selected source element up a row. In this way you can change the order of the source elements.
Down	Click to shift the selected source element down a row. In this way you can change the order of the source elements.
Remove	Click to remove the selected source element from the project.
Edit	Click to open the Specification window of the selected source element.

Copyright © 1998-2015 No Magic, Inc.

Button	Description
Target elements area:	
Up	Click to shift the selected target element up a row. In this way you can change the order of the target elements.
Down	Click to shift the selected target element down a row. In this way you can change the order of the target elements.
Remove	Click to remove the selected target element from the project.
Edit	Click to open the Specification window of the selected target element.

Buttons in the Synchronization dialog

The following table describes the **Synchronization** dialog buttons.

Button	Description
Synchronize All	Click the button to synchronize all the not synchronized elements. Then from the menu select the straightforward or **NEW!** backward synchronization. For example, when synchronizing Parameters (Activity) and Pins (Call Behavior Action) you can select to synchronize one of the following: • Pins (Call Behavior Action) by Parameters (Activity) • Parameters (Activity) by Pins (Call Behavior Action).
Synchronize Selected	Click to synchronize the selected parameter. Note that the button is disabled if no parameters are selected in the list. After you click the **Synchronize Selected** button, the following menu appears: **Create ->** - Click to create the target element with the same properties as the source element. **<-Create** - Click to create the source element with be create with the same properties as the target element. **Update ->** - Click to update the not synchronized properties of the target element according to the properties of the source element. **<-Update** - Click to update the not synchronized properties of the source element according to the properties of the target element. Note that the **Update** command exists only if there are properties to update.

Copyright © 1998-2015 No Magic, Inc.

7 TOOLS

MagicDraw provides the following tools and wizards to help you quickly and easily perform design tasks.

- "Model Merge" - allows for porting changes between different project versions.
- "Pattern Wizard" - creates various GOF, Java, Junit, CORBA IDL, XML Schema, and other design patterns.
- "Creating Setters/Getters" - creates getting and setting operations for attributes defined in the class.
- "Implementing or Overriding Operations" - creates defined operations down the Inheritance tree.
- "Model Transformation Wizard" - enables running one of the predefined transformations to convert the given model part into another model according to the rules of that transformation. Transformations are usually used for converting models between different modeling domains, for example, UML and SQL.
- "Automatic Instantiation Wizard" - allows for creating instances of various entities with just a few clicks. It is especially useful when working with complex models or assembling large systems from parts.
- "Resource Manager" - allows for managing resources (Profiles, Plugins, Templates, Language resources, Case studies/examples, Custom diagrams, and others).
- "Spelling Checker" - checks spelling as you type. Select what you want to be spell checked: either the whole project or some specific parts.

Model Merge



Model Merge



Model Merge functionality is available in Standard Edition and above for an additional fee.

Model Merge allows for copying changes between different project versions. This functionality is usually needed when there are several branches that reflect different releases or versions of the product, for example, when certain fixes have to be copied from a release branch to the mainstream development.

You can merge different versions of both local and server projects.

Projects that use other projects, can be compared and merged in one single operation. You do not need to merge each used project and finally the main project one-by-one.

Concepts

For better understanding the further material, first of all learn the following concepts.

Source

A project version from which changes are copied.

Target

A project version into which changes are copied.

Copyright © 1998-2015 No Magic, Inc.

Ancestor

A common parent project version of both source and target.

Contributor

General description for a project version that participates in the merge as a source or a target.

Getting Started with Model Merge

Before starting to use the model merge feature, we recommend you to gain some knowledge about

- Merge types (see "Choosing merge type" on page 435)
- Change concept in the model merge (see "Understanding change concept" on page 436)

Choosing merge type

There are two types of model merge: 3-way merge and 2-way merge. Choose the way of merge according to a particular case. In order to find out which way best suites your case, learn about each one in the following sections.

3-way merge

Conceptually the 3-way merge is a reconciliation of two difference sets, if we assume that a *difference set* contains changes between a contributor and the ancestor. For example, a difference set v1-v2 represents changes between project v2 and ancestor v1 as well as v1-v3 does for project v3 and the same ancestor.

Use the 3-way merge, if you need to compare and merge changes from two projects into one and also consider the ancestor of both projects. For example, you may need to merge two branches of the same server project that have been simultaneously developed by two different users in a collaborative environment, as is shown in the following figure. As you can see, branch b.1 has been created from project version i. Then both versions have been developed in parallel: i+1, ..., n-1, n and b.1, ..., b.n. The merge operation copies changes that have been made in the branch version to the trunk version.

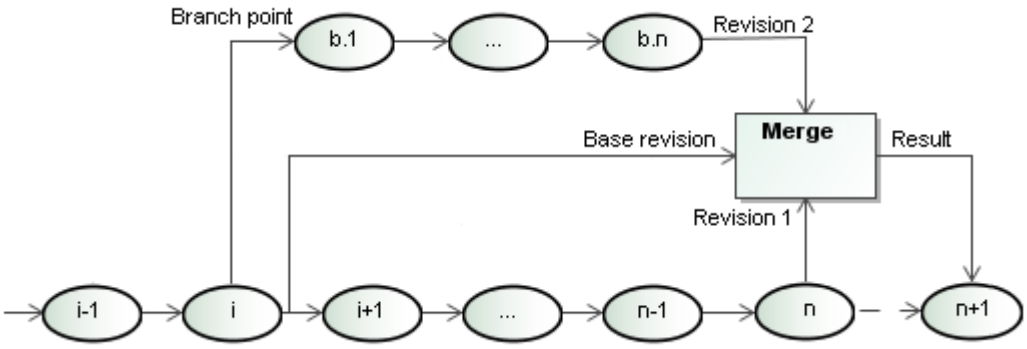


Figure 254 -- Concept of 3-way merge

Copyright © 1998-2015 No Magic, Inc.

2-way merge

The 2-way mere is a specific case of the 3-way merge. Use this type of merge for joining together two separately developed local projects that have no ancestor. The target project will be taken as an ancestor.

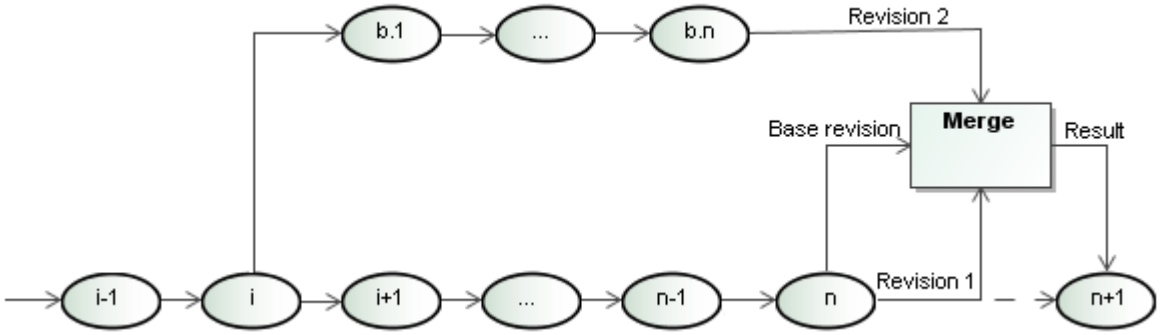


Figure 255 -- Concept of 2-way merge

Understanding change concept

Change is a difference, found between the ancestor and a participant. Changes can be as follows:

- addition
- deletion
- modification
- movement
- order

Every change can be accepted or rejected and have dependent changes. Changes can conflict with each other or be equivalent as well.

Merging begins with building a composite change tree, which consists of model, diagramming, and non-model changes.

Change types

Read the following definitions to get familiar with different change types.

Addition change

If an element has been added to a contributor, an addition change occurs.

Deletion change

If an element has been removed from a contributor, a deletion change occurs.

Modification change

If an element property in a contributor has been modified, a modification change occurs.

EXAMPLE

If the **Is Abstract** property value of a class in the ancestor had the default value which is *false* and the same property value in a contributor has been changed to *true*, a modification change occurs.

There are three types of modification changes:

- **Addition modification change** that occurs when a value is added to a property.
- **Deletion modification change** that occurs when a value is removed from a property.

Copyright © 1998-2015 No Magic, Inc.

- **Replacement modification change** that occurs when one value is replaced with another. This type of modification change occurs only for properties that have multiplicity less or equal to 1.

Movement change

If an element owner has been changed in a contributor, a movement change occurs.



Let's say package A contains some class in the ancestor and package B contains the same class in a contributor. This means that the class has been moved from package A to package B in the contributor. This case is recognized as a movement change.



Another case of the movement change is when an attribute or an operation that has been owned by class A in the ancestor, becomes the attribute or an operation of class B in a contributor.

Order change

If elements order in a contributor has been changed an order change occurs. Order changes can occur on elements such as attributes, operations, and other ordered elements. Even if a single element in a collection has changed its place, the order change is applied to the entire collection.

Since an element can have several ordered collections, several order changes can occur on the single element.



Let's say class A has attributes a, b, and c in the ancestor. The attribute c has been moved up and placed above attribute a in a contributor. This means that the order of attribute collection in class A has changed in the contributor. This is a case of the order change.

Change states

Every change, whether it is addition, modification, deletion, movement, or order change, can be either accepted or rejected. It is change state.

All accepted changes will be incorporated into the target. Alternatively, they can be rejected and will not be applied to into the target.

Dependent changes

In some cases, other changes have to be accepted or rejected before accepting or rejecting a selected change. In other words, a selected change sometimes depends on other changes and it is called a dependent change.

For better understanding the concept of the dependent change, study the following examples.



Let's say a class attribute type has been changed to a type that had been created by another change. In the consequence, the attribute type change depends on the change that has created the type. This means that type creation change must to be accepted before accepting type modification change.

Copyright © 1998-2015 No Magic, Inc.



Let's suppose there is an attribute type change in a contributor. The old type has been deleted and a new type has been added to the contributor. In this case, three changes occur:

- deletion change (for the old type)
- addition change (for the new type)
- and modification change (for the property type)

These are also ownership changes, but they are accepted together with deletion and addition changes.

The modification change depends on the addition change and the deletion change depends on the modification change.Thus only accepting the deletion change means accepting the addition change, the modification change and the deletion change itself.

Conflicting changes

Conflict is an incompatibility of two changes, i.e. these changes cannot be accepted together. Conflicts can occur only in the 3-way merge.



Conflict can be as follows:

- Each contributor has changed a class name or any other element property value.
- One contributor has added an operation to a class and the other contributor deleted the class.
- One contributor has moved a class into one package while the other contributor moved it to another package.

Equivalent changes

Equivalent change is a pair of identical changes that are detected in both source and target, when performing the 3-way merge.



Let's say package A contains some class in the ancestor and both contributors contain the same class in package B. This means that the class has been moved from package A to package B in both source and target. This is a case of the equivalent change.



Another case of equivalent changes is when the same element A becomes renamed to B in both contributors.

Using Model Merge

No matter which type of the merge you use, the merge procedure generally includes the following phases:

1. Starting model merge described on page 439
2. Analyzing and managing merge results described on page 442
3. Finishing merge described on page 458

Preparing for merge

Read the following information and get informed about customizing the model merge. This is the optional phase of the model merge procedure. You may not need to perform it.

Turning off order changes detection

Order changes can be skipped while merging. For this you need to specify names of properties wherein order changes should not be detected.

Copyright © 1998-2015 No Magic, Inc.

To turn off the order changes detection in specific properties

1. From the **Options** menu select **Environment**. The **Environment Options** dialog will be opened.
2. Find the **Do Not Detect Order Changes for** option under the **Merge** category in the **General** options group.
3. In the option value cell, specify names of properties wherein order changes should not be detected while merging.



Property names must be written in camel case, for example, ownedAttribute, ownedElement, and so on.



For the information about editing this kind of option values, see "Editing long textual values" on page 300.

Starting model merge

You can start the merge either:

- From **Collaborate** > **Merge From**.
  This way of starting the merge can be applied explicitly for server projects. And you do not need to specify the ancestor, since it will be automatically identified straight after a source is selected. The 3-way merge will be applied for merging.
- From **Tools** > **Project Merge**.
  This way of starting the merge can be applied for both local and server projects, though used projects can be included in the merge only for server projects. You need to specify both a source and the ancestor as well as choose the way of merge.

Now let's go through either way of starting the merge procedure step-by-step.

To start the merge from **Collaborate** > **Merge From**

1. Open a server project that will be a target.
2. From the **Collaborate** menu, select **Merge From**.
3. In the opened dialog, select a server project that will be a source.
4. In the same dialog, click the **Advanced** button to see more merge options. Otherwise go to step #7.
5. In the **Optimize for** drop-down list, select
   - **Speed** if you need to merge the projects faster. Keep in mind that this will require more memory.
   - **Memory** if your need to decrease memory usage while merging the projects, since your computer does not have enough memory. Keep in mind that this will slow down the merge procedure.



You can also specify the **Optimize for** option in the **Environment Options** dialog. Find the option under the **Merge** category in the **General** options group.

6. In the **Used projects to include** list, select the used projects of the target you want to merge with appropriate used projects of the selected source. To select a used project, click to select the checkbox next to the used project name (see the following figure).



The list includes only the projects that

- Are used by the target in the read-write accessibility mode.
- Have the same composition in both contributors.
- Do not have private parts.

Copyright © 1998-2015 No Magic, Inc.

7. Click the **Merge** button. The project merge will be started. If changes are found, the **Merge** window will open. Proceed to Analyzing and managing merge results.

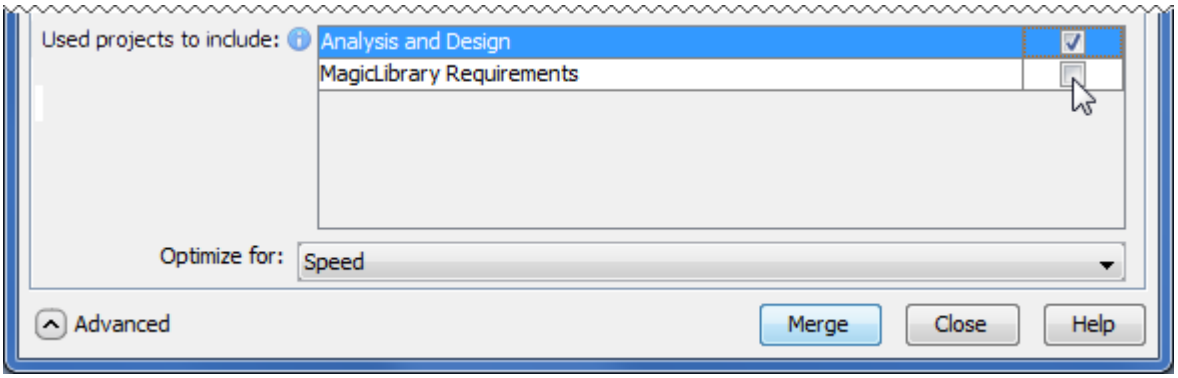


Figure 256 -- Selecting used projects for merge

To start the merge from **Tools** > **Project Merge**

1. Open a project that will be a target.
2. From the **Tools** menu, select **Project Merge**. The **Merge Projects** dialog will be opened within the active project selected as a target.
3. Select a merge type.

TIP!

Learn about the 3-way and the 2-way merge in "Choosing merge type" on page 435.

4. Select a project that will be a source.
5. If there is a case of the 3-way merge select a project that will be the ancestor.
6. Click the **Advanced** button to see more merge options. Otherwise go to step #9.
7. In the **Optimize for** drop-down list, select
   - **Speed** if you need to merge the projects faster. Keep in mind that this will require more memory.
   - **Memory** if your need to decrease memory usage while merging the projects, since your computer does not have enough memory. Keep in mind that this will slow down the merge procedure.

TIP!

You can also specify the **Optimize for** option in the **Environment Options** dialog. Find the option under the **Merge** category in the **General** options group.

8. In the **Used projects to include** list, select the used projects of the target you want to merge with appropriate used projects of the selected source. To select a used project, click to select the checkbox next to the used project name (see the following figure).

IMPORTANT!

The list is available only for server projects and includes only the projects that

- Are used by the target in the read-write accessibility mode.
- Have the same composition in both contributors.
- Do not have private parts.

Copyright © 1998-2015 No Magic, Inc.

9. Click the **Merge** button. The project merge will be started. If changes are found, the **Merge** window will open. Proceed to Analyzing and managing merge results.

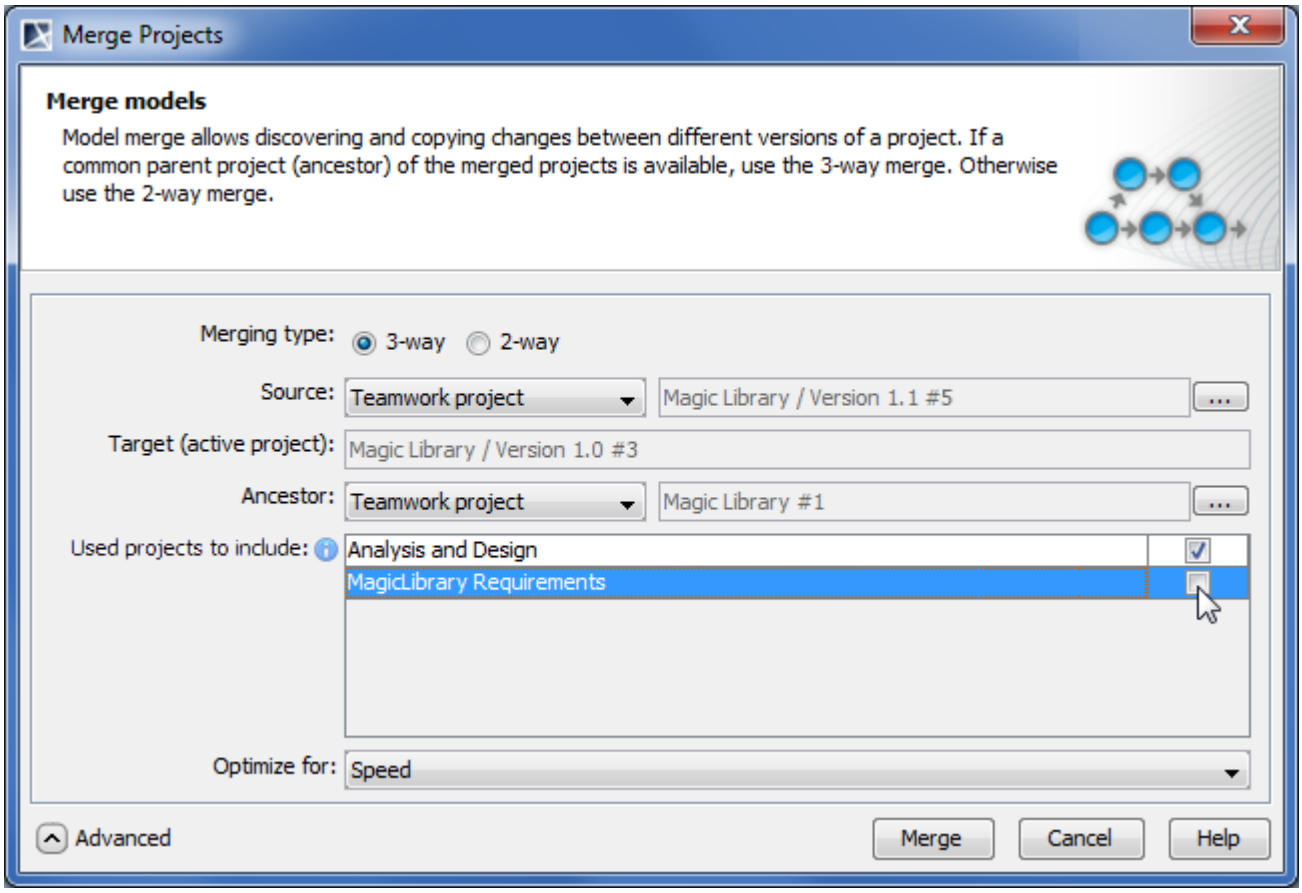


Figure 257 -- Merge Projects dialog with automatically selected target and list of target usages

Copyright © 1998-2015 No Magic, Inc.

Analyzing and managing merge results

The merge results are represented in the **Merge** window.

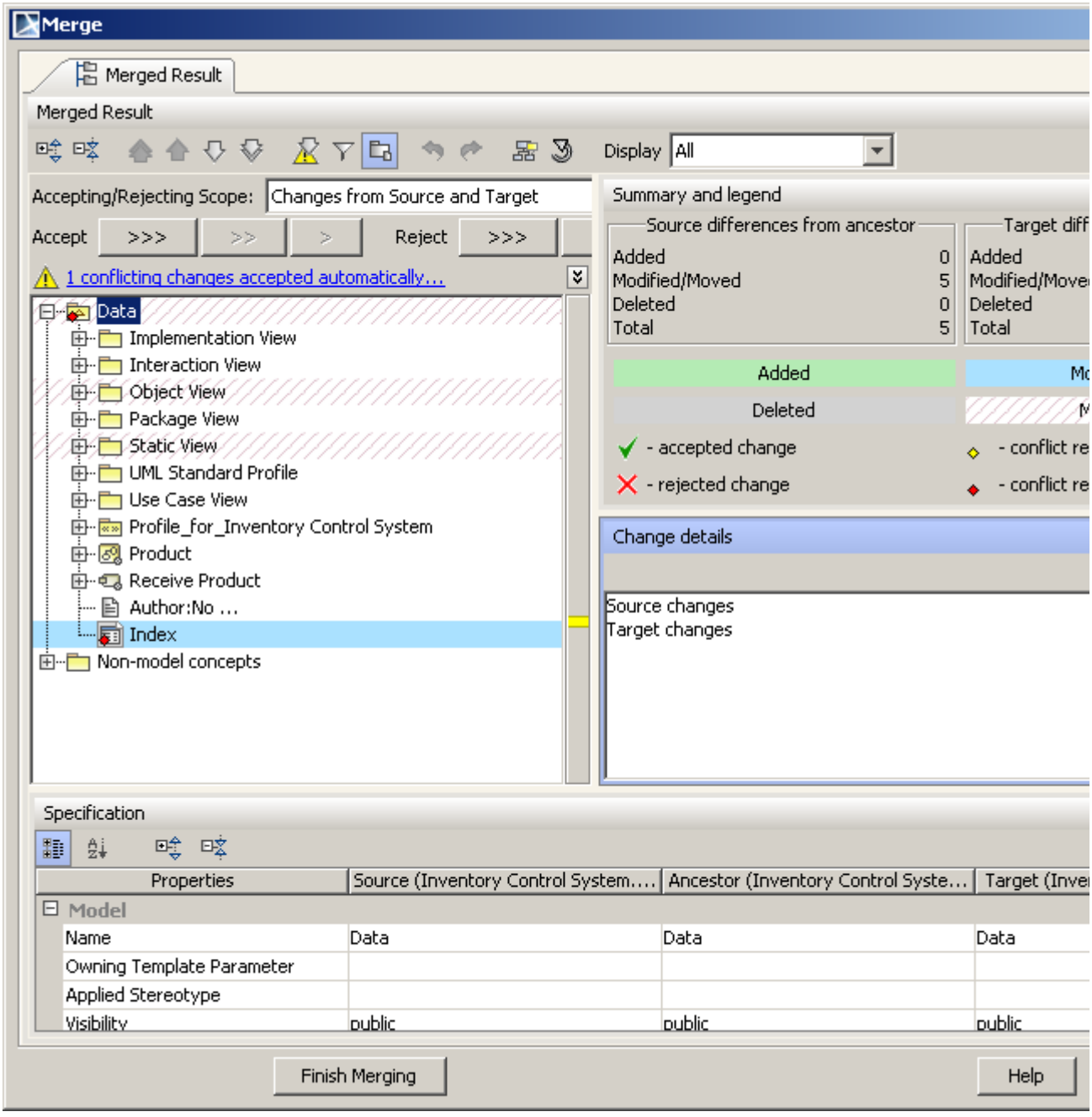


Figure 258 -- Merge window

Read about analyzing and managing model merge results in

- "Getting merge summary information and changes legend" on page 443
- "Understanding content of the Merged Result tree" on page 443
- "Inspecting element property changes in the Specification panel" on page 448
- "Inspecting changes in the Change details panel" on page 450
- "Displaying and navigating through changes" on page 451
- "Quickly navigating through conflicting changes" on page 453
- "Accepting and rejecting changes" on page 453

Copyright © 1998-2015 No Magic, Inc.

- "Viewing and analyzing changes of modified diagrams" on page 455

If you do not need to perform the analysis and management of the merge results, proceed to Finishing merge.

Getting merge summary information and changes legend

You can see how many differences have been detected between the ancestor and the source as well as between the ancestor and the target on the **Summary and legend** panel that is located at the top right of the **Merge** window. The panel also helps to understand the decorations and highlighting of elements in both the **Merged Result** tree and the **Specification** panel.

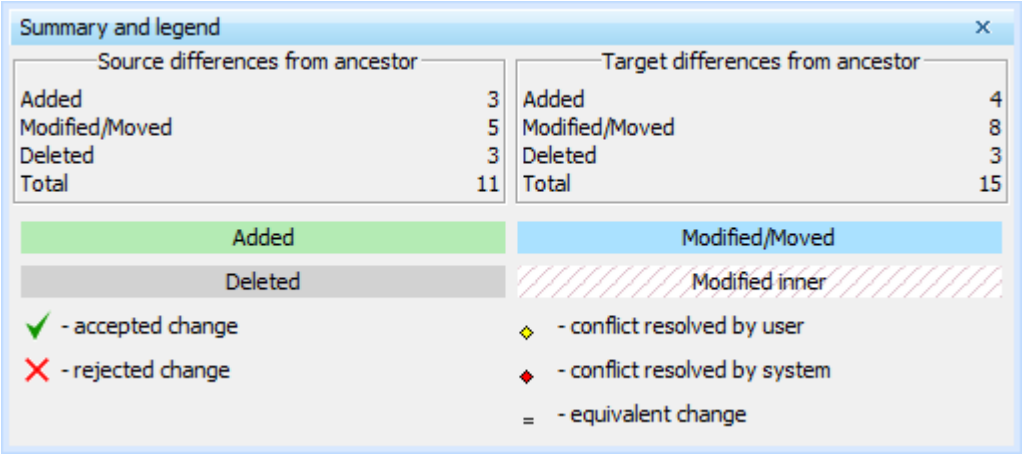


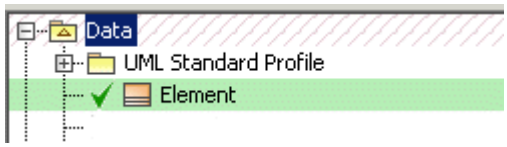
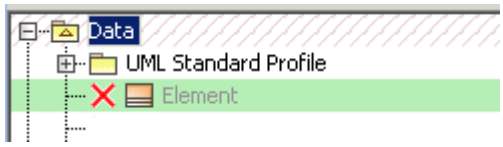
Figure 259 -- Summary and legend panel

Understanding content of the Merged Result tree

Elements with all change types except equivalent changes can be displayed in the **Merged Result** tree that is located at the top left of the **Merge** window.

Understanding element decorations

As a change occurs on a single element, the element icon indicates the state of the change which can be either accepted or rejected.

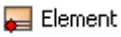
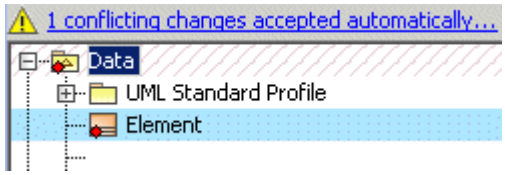
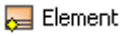
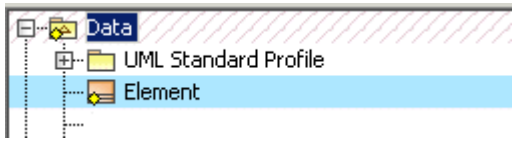
Decoration	Change state	Example
 (green tick before the element icon)	Accepted	 A new element had been created in one of the contributors and its addition was accepted.
 (red cross before element icon)	Rejected	 A new element had been created in one of the contributors and its addition was rejected.

Copyright © 1998-2015 No Magic, Inc.



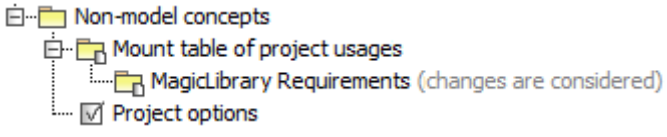
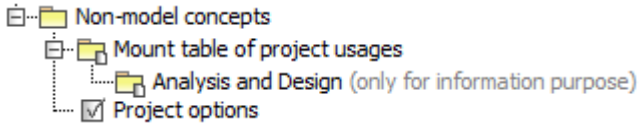
To understand the concept of change states, refer to "Change states" on page 437.

Elements with resolved conflicting changes are decorated as it is showed in the following table. Owners of these elements are also decorated.

Decoration	Change applied by	Example
 (red diamond on the lower-left corner of element icon)	System	 The element had a conflicting modification change which was resolved automatically.
 (yellow diamond on the lower-left corner of element icon)	User	 The element had a conflicting modification change which was resolved by the user.

Copyright © 1998-2015 No Magic, Inc.

Understanding used project representation

Text near module icon	Description	Example
(changes are considered)	A project used in the merged project as *read-write*. The used project changes that have been detected in both contributors are suggested for the merge.	
(only for information purpose)	A project that is either: • Used in the merged project as *read-only*. • Used in the merged project as *read-write* and has private parts. • Used in the merged project as *read-write*, but has different composition in both contributors. The contents of such used project is not inspected for changes.	

Recognizing addition changes



For the addition change description, refer to "Addition change" on page 436.

Added elements are highlighted in green background.

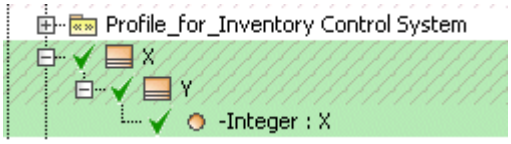


Figure 260 -- Example of accepted addition change

Recognizing deletion changes



For the deletion change description, refer to "Deletion change" on page 436.

Deleted elements are highlighted in grey background.



Figure 261 -- Example of accepted deletion change

Copyright © 1998-2015 No Magic, Inc.

Recognizing modification changes



For the modification change description, refer to "Modification change" on page 436.

Elements whose properties have been changed are highlighted in blue.

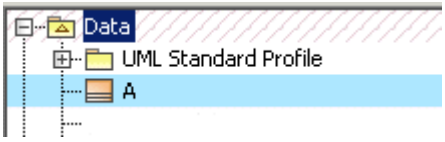


Figure 262 -- Element with modified properties

To see what properties have been modified, click the element and see changes in the **Specification** panel.



For more information about using the **Specification** panel, see "Inspecting element property changes in the Specification panel" on page 448.

Recognizing movement changes



For the movement change description, refer to "Movement change" on page 437.

Elements with movement changes are highlighted in the same blue background that is used to highlight modification changes, since movement changes are some kind of modification changes. Icon of an element with movement changes is represented with arrow symbol on the left.

The following figure illustrates a movement change, when element X has been moved from package A to package B. The movement change had been accepted and element X became owned by package B.



Figure 263 -- Example of accepted movement change

Copyright © 1998-2015 No Magic, Inc.

You can navigate from the element's original location to the new one (and vice versa) using the commands on the element shortcut menu.

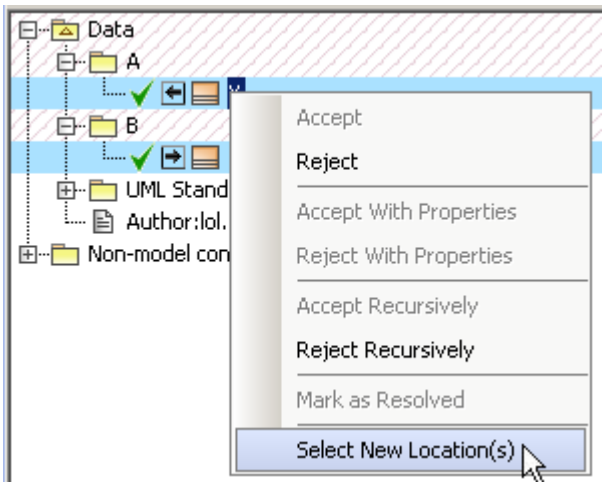


Figure 264 -- Navigating to new element location

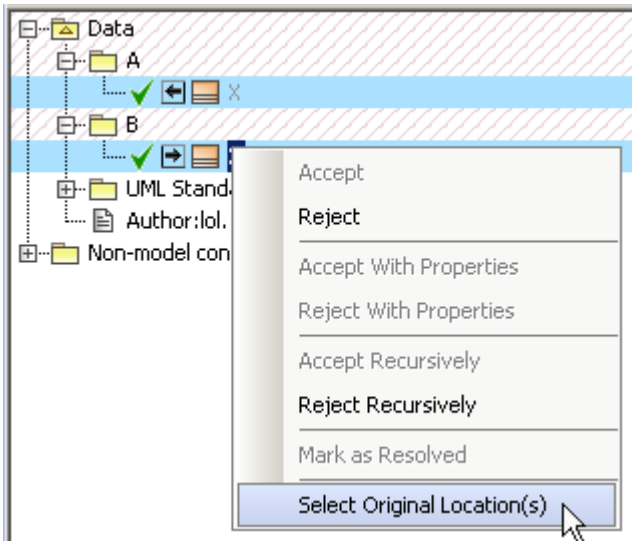


Figure 265 -- Navigating to original element location

Recognizing order changes



For the order change description, refer to "Order change" on page 437.

Elements with order changes are highlighted in the same blue background that is used to highlight modification changes, since order changes are some kind of modification changes.

Copyright © 1998-2015 No Magic, Inc.

After accepting the change, all the elements in the collection are reordered so that it would be the same as in one of the contributors.

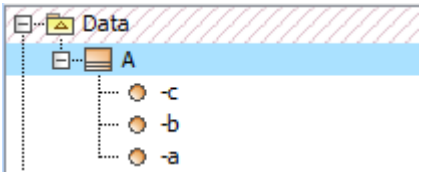


Figure 266 -- Example of accepted order change

TIP! You can turn off the order changes detection. For more about it, see "Turning on showing merged diagram annotations" on page 438.

Recognizing elements with changed owned elements

Elements whose owned elements (in any nesting level) have changes are highlighted using red diagonals.

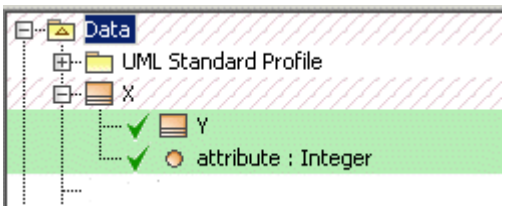


Figure 267 -- Element with owned elements added

Modified elements with changed owned elements are highlighted in blue with red diagonals.

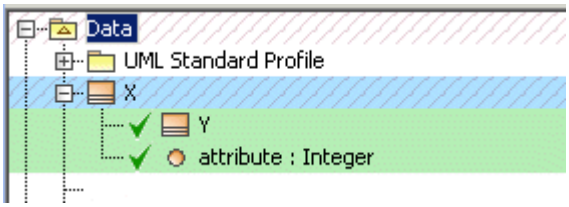


Figure 268 -- Element with owned elements added and modification change

Inspecting element property changes in the Specification panel

You can see all the details of an element property changes in the **Specification** panel that is located at the bottom of the **Merge** window. All what needs to be done is selecting the appropriate element in the **Merged Result** tree.

As you can see in the following figure, the first column of the panel contains a list of element property names and other columns display their value changes that have been detected in both contributors and the ancestor.

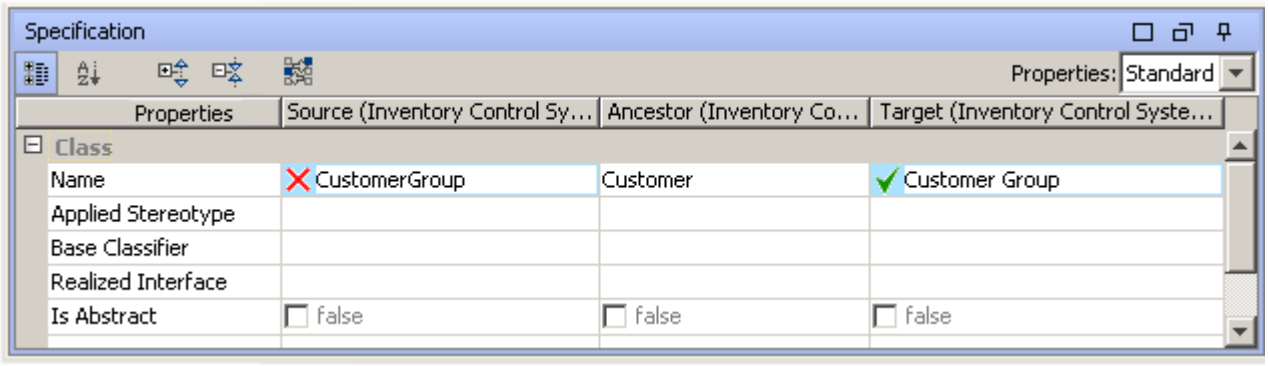


Figure 269 -- Displaying element property changes in Specification panel

Copyright © 1998-2015 No Magic, Inc.

To navigate from the **Specification** panel to the **Merged Result** tree

Do either:

- In the **Properties** column, right-click the name of a property, which references other elements. Then on its shortcut menu, point to **Select in Merged Result Tree** and choose an element to which you want to navigate.
- In the **Source**, **Ancestor**, or **Target** column, right-click the value of a property, which references other elements. Then on its shortcut menu choose an element to which you want to navigate.

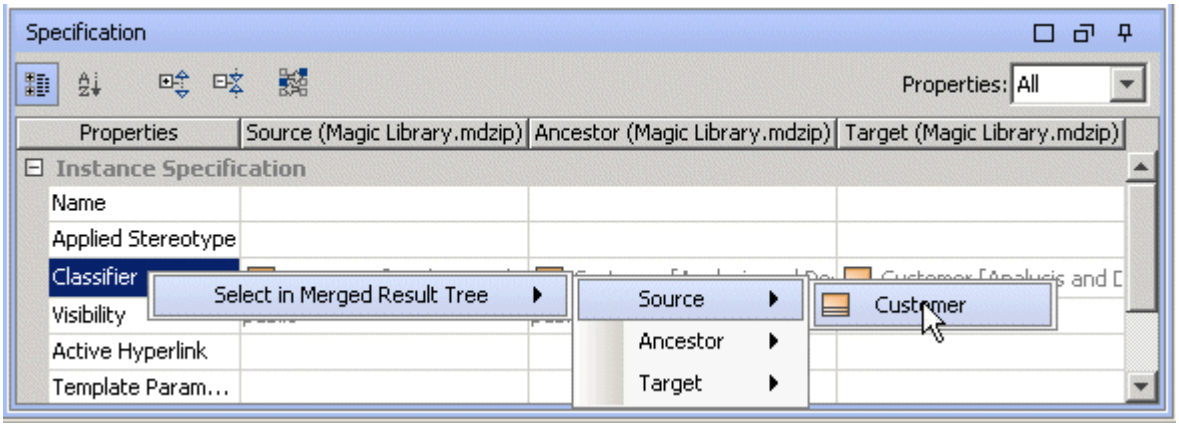


Figure 270 -- Using property name shortcut menu to navigate from Specification panel to Merged Result tree.

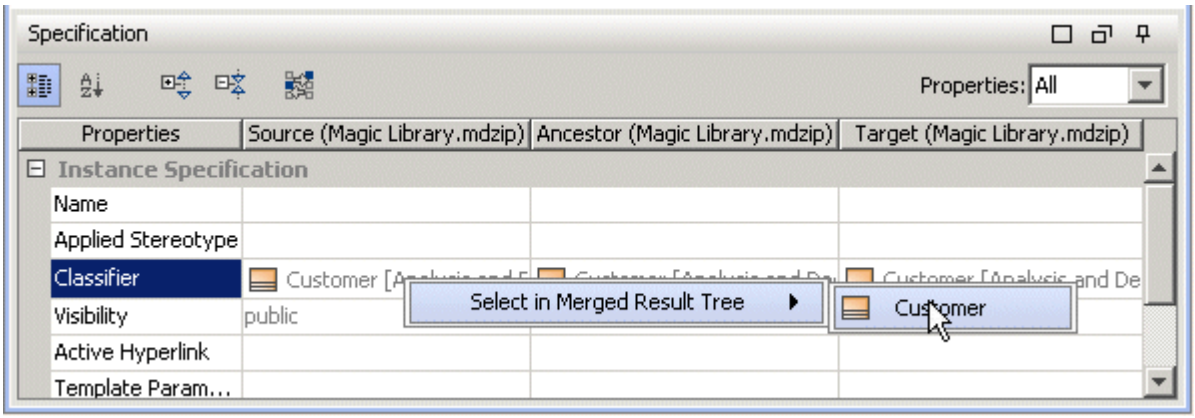


Figure 271 -- Using property value shortcut menu to navigate from Specification panel to Merged Result tree

Copyright © 1998-2015 No Magic, Inc.

Inspecting differences between textual values

You can see exact changes between textual values of element properties, such as element documentation, comments, pre and post conditions of a use case. Differences of compared texts can be displayed in a single dialog, where inserted and deleted parts of the text are appropriately highlighted.

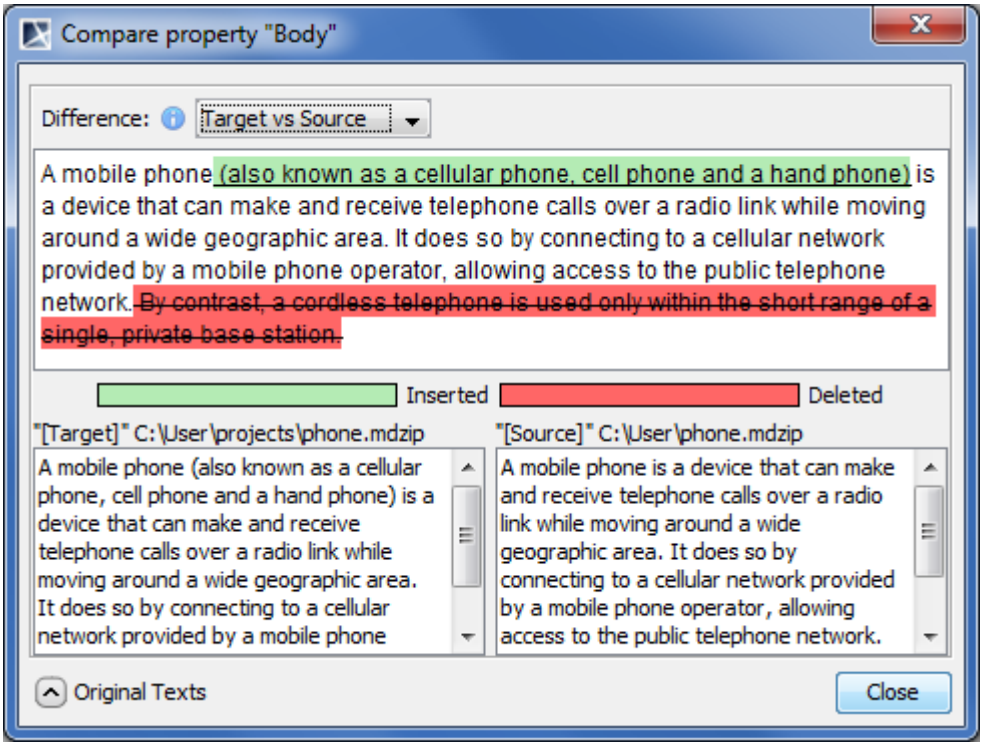


Figure 272 -- Displaying differences between long textual property values

To see differences of compared texts

Do one of the following:

- Select a property with a modified text on the **Specification** panel and then click on the toolbar.
- Double-click the property on the **Specification** panel.

The **Compare property** dialog opens.

Inspecting changes in the Change details panel

The **Change details** panel displays a tree structure reflecting the changes that occurred either on the element selected in the **Merged Result** tree or on the property selected in the **Specification** panel. If several elements or properties are selected, then all changes that have been detected in these elements or properties accordingly will be displayed in the **Change details** panel. The panel is located above the **Specification** panel on the **Merge** window.

The panel contains a tree structure that has two top-level branches:

- Source changes

Copyright © 1998-2015 No Magic, Inc.

- Target changes

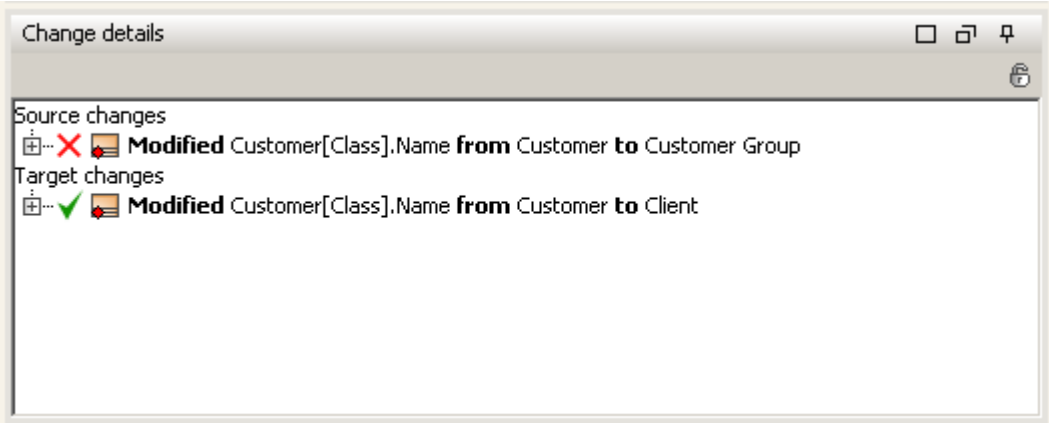


Figure 273 -- Change details panel

In this panel, you can see equivalent changes. In order to see them, expand the **Equivalent Changes** branch under an element with equivalent changes. A change, which is marked with is equivalent to a change in another contributor.

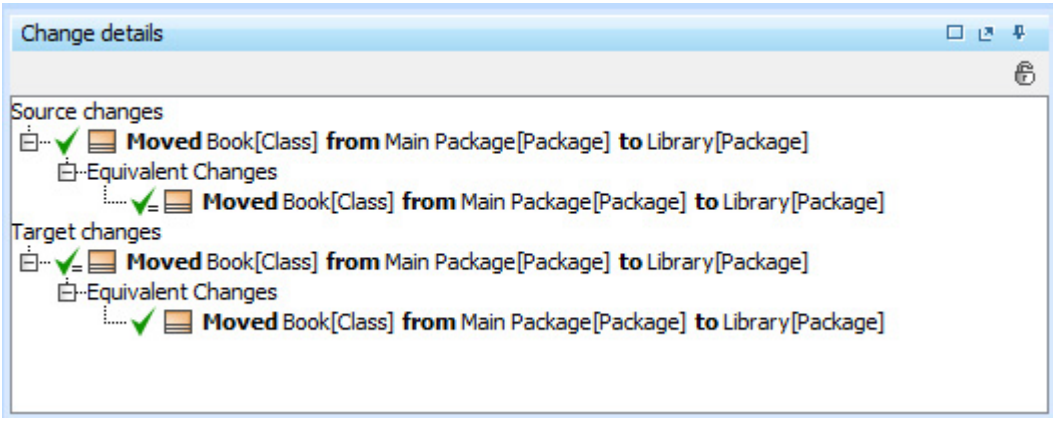


Figure 274 -- Displaying equivalent changes in Change details panel

TIP! For the equivalent changes description, refer to "Equivalent changes" on page 438.

Displaying and navigating through changes

You can customize the content of the **Merged Result** tree as well as navigate through changes in both the **Merged Results** tree and the **Specification** panel using the toolbar that is located at the top of the **Merge** window.

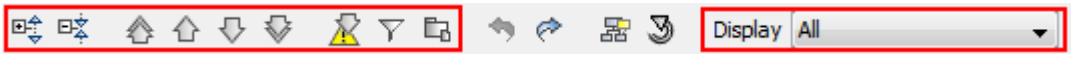


Figure 275 -- Toolbar items for displaying and navigating through changes

The toolbar buttons for customizing the **Merged Results** tree and navigating through changes are described in the following tables.

Button / Shortcut keys	Tooltip text	Description
	Expand	Click to expand all nodes in the **Merged Result** tree.

Copyright © 1998-2015 No Magic, Inc.

Button / Shortcut keys	Tooltip text	Description
	Collapse	Click to collapse all nodes in the **Merged Result** tree.
 ALT+HOME	Go To First Change	Click to select the first change. The button is not available, if the first change is already selected.
 ALT+UP ARROW	Go To Previous Change	Click to select the previous change.
 ALT + DOWN ARROW	Go To Next Change	Click to select the next change.
 ALT+END	Go To Last Change	Click to select the last change. The button is not available, if the last change is already selected.
 ALT+PAGE DOWN	Go To Next Conflict	Click to select the next conflict. The button is not available, if there are no conflicts. For more information about navigation through conflicting changes, see "Quickly navigating through conflicting changes" on page 453.
	Filter	Click to open the **Items Filter** dialog. Clear the checkbox next to an element that you need to hide in the **Merged Result** tree.
	Show Auxiliary Resources	Click to show or hide in the **Merged Result** tree the profiles and used projects with the «auxiliaryResources» stereotype applied (for example, UML Standard Profile).

You can also filter the elements displayed in the **Merged Results** tree by change type. Use the **Display** drop-down list to specify a change type. Be advised that filtering elements by a particular change type displays both elements with the particular changes and elements with particular owned element changes.

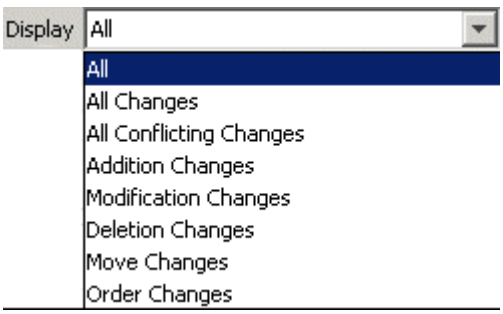


Figure 276 -- Choosing change type for displaying changes in Merged Results tree

Copyright © 1998-2015 No Magic, Inc.

Quickly navigating through conflicting changes

You can quickly navigate through the automatically accepted conflicting changes in the **Merged Result** tree. Use the panel on the right of the **Merged Result** tree that displays yellow buttons representing conflicting changes. Click a button to select the automatically accepted conflicting change.

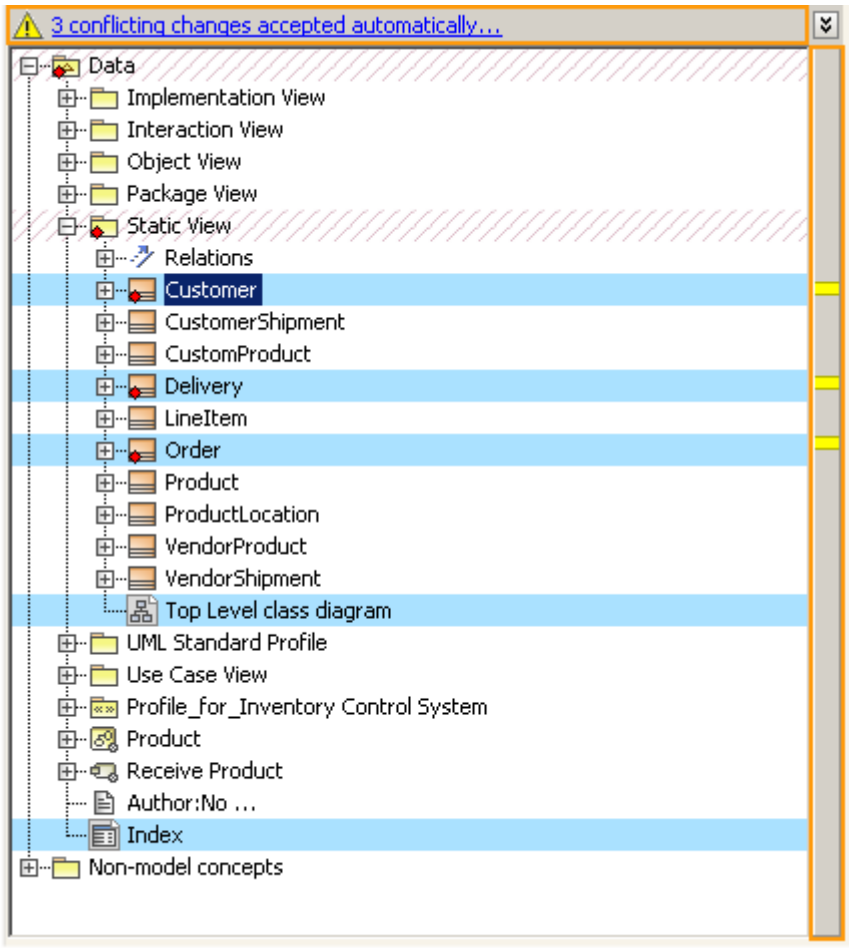


Figure 277 -- Buttons for quick navigation through conflicting changes

Accepting and rejecting changes

Accept and reject changes using the toolbar that is located above the **Merged Result** tree or using the shortcut menu.

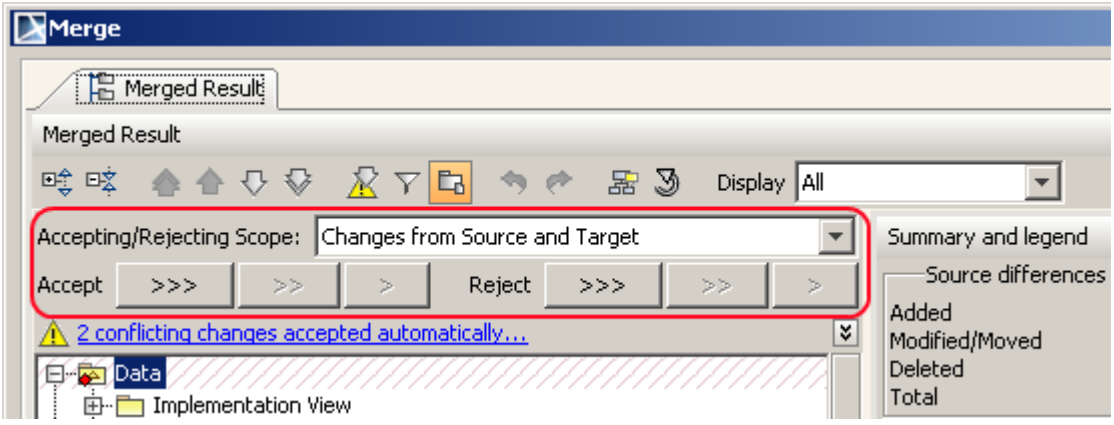
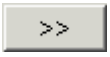
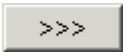
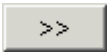


Figure 278 -- Toolbar for accepting and rejecting changes

Before starting to accept and reject changes, specify from what contributor changes will be accepted and rejected. Use the **Accepting/Rejecting Scope** drop-down list for this.

Copyright © 1998-2015 No Magic, Inc.

The toolbar buttons are described in the following table.

Button	Tooltip text	Description
Accept		
	Accept the selected change, its property changes, and all subelement changes	Click to accept all changes from both contributors starting from the selected element. For example, if Data, which is the top level model element is selected, then all changes for the whole project will be accepted.
	Accept the selected change its property change	Click to accept the selected element change together with the element property changes.
	Accept the selected change	Click to accept the selected element change.
Reject		
	Reject the selected change, its property changes, and all subelement changes	Click to reject all changes from both contributors starting from the selected element. For example, if Data, which is the top level model element is selected, then all changes for the whole project will be rejected.
	Reject the selected change its property change	Click to reject the selected element change together with the element property changes.
	Reject the selected change	Click to reject the selected element change.

The following table lists the commands of the shortcut menu that can be opened, when an element is selected in the **Merged Result** tree and the **Change details** panel.

Command	Description
Accept	Accepts the selected change. If the change has equivalent changes, they are accepted too.
Reject	Rejects the selected change. If the change has equivalent changes, they are rejected too.
Accept With Properties	Accepts the selected change and its property changes.
Reject With Properties	Rejects the selecting change, its property changes.
Accept Recursively	Accepts the selected change, its property changes, and all owned element changes.
Reject Recursively	Rejects the selected change, its property changes, and all owned element changes.
Mark as Resolved	Marks the change as resolved by the user, but not by the system. The conflicting change becomes marked as resolved by the user too.

Copyright © 1998-2015 No Magic, Inc.

Command	Description
Select in Merged Result Tree / Select in Specification panel	This command differs depending on the location of the selected change.

Viewing and analyzing changes of modified diagrams

A modified diagram can have two or three diagram difference viewers that are represented as separate tabs in the **Merge** window next to the **Merged Result** tree (see the following figure).

To open diagram difference viewers of a modified diagram

- Double-click the modified diagram in the **Merged Result** tree.
  Two or three new tabs will be opened in the **Merge** window. The ancestor's tab is always opened. The source's tab is opened, if changes have been detected in the source, and the target's tab is opened, if changes have been detected in the target.

All the diagram difference viewers are used to display the following changes:

- One or more symbols were moved or resized on the diagram.
- A new symbol was added to the diagram.
- A symbol was deleted from the diagram.

As you can see in the following figure, all these changes are highlighted in blue diagonals.



Be advised that symbols with property value changes that actually have no effect on their appearance are highlighted in a diagram difference viewer as well. For example, changing the **Show Stereotypes** property value in the **Symbol Properties** dialog of a class makes the class symbol highlighted as changed in a diagram difference viewer.

Copyright © 1998-2015 No Magic, Inc.

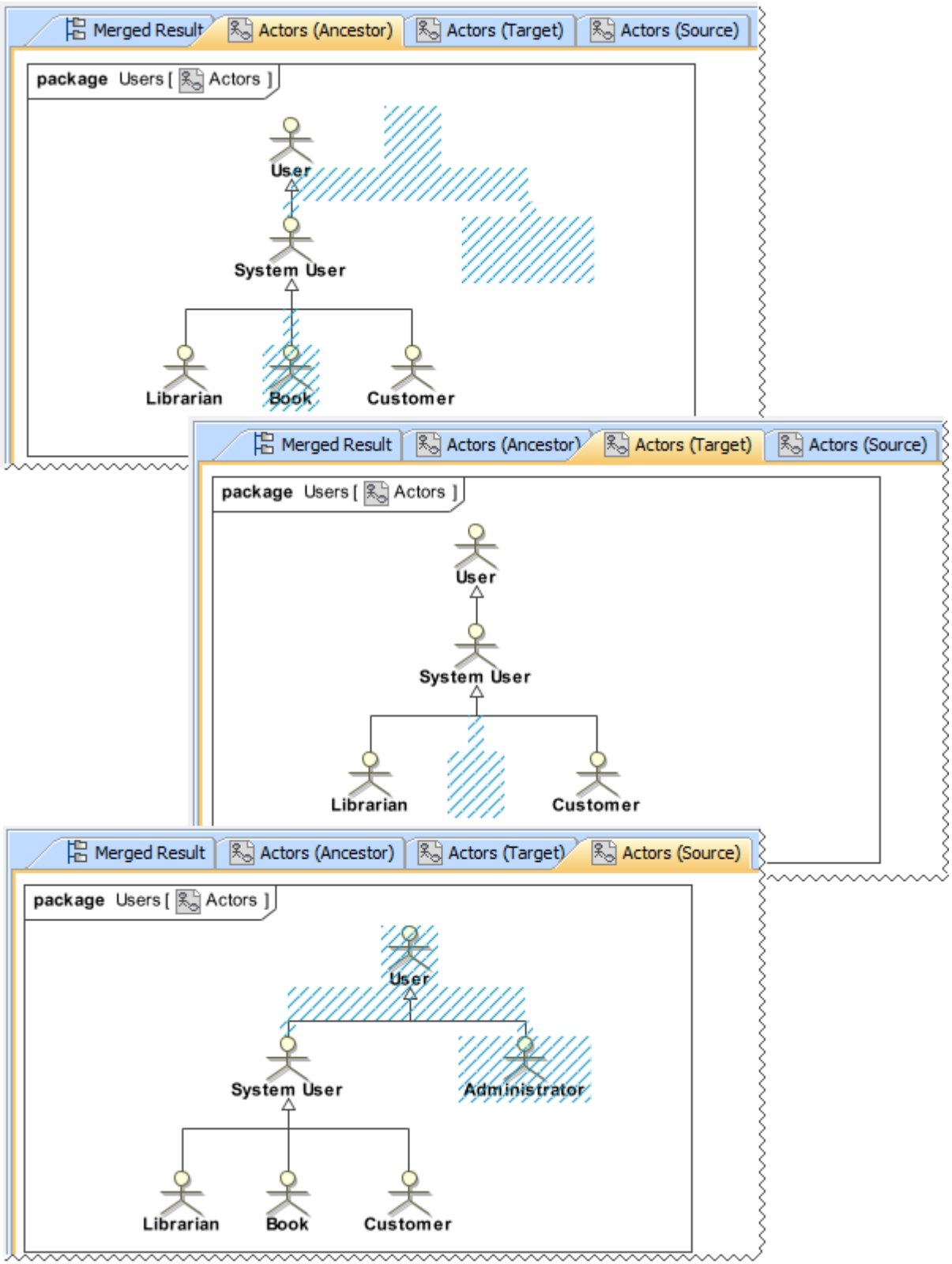


Figure 279 -- Ancestor's, target's, and source's diagram difference viewers

Copyright © 1998-2015 No Magic, Inc.

You can manage the content of a diagram difference viewer using the toolbar that is located below the diagram difference viewer.



Figure 280 -- Toolbar for managing content of diagram difference viewer

See the button descriptions in the following table.

Button	Tooltip text	Description
	Synchronize Zooming	Click to turn on or off the zooming synchronization in all difference viewers of the same diagram.
	Synchronize Scrolling	Click to turn on or off scrolling synchronization in all difference viewers of the same diagram.
	Mark Source Changes	Click to turn on or off the highlighting of the diagram changes that have been made in the source. This button is available in both ancestor's and target's diagram difference viewers.
	Mark Target Changes	Click to turn on or off the highlighting of the diagram changes that have been made in the target. This button is available in both ancestor's and source's diagram difference viewers.
	Print Diagram	Click the button to print a diagram. The diagram will be printed with highlighted areas on it, if the highlighting of the diagram changes is turned on.
	Zooming	Use the buttons to control the diagram size.
Return to Merged Result		Click to open the **Merged Result** tree with the appropriate diagram selected within.

Copyright © 1998-2015 No Magic, Inc.

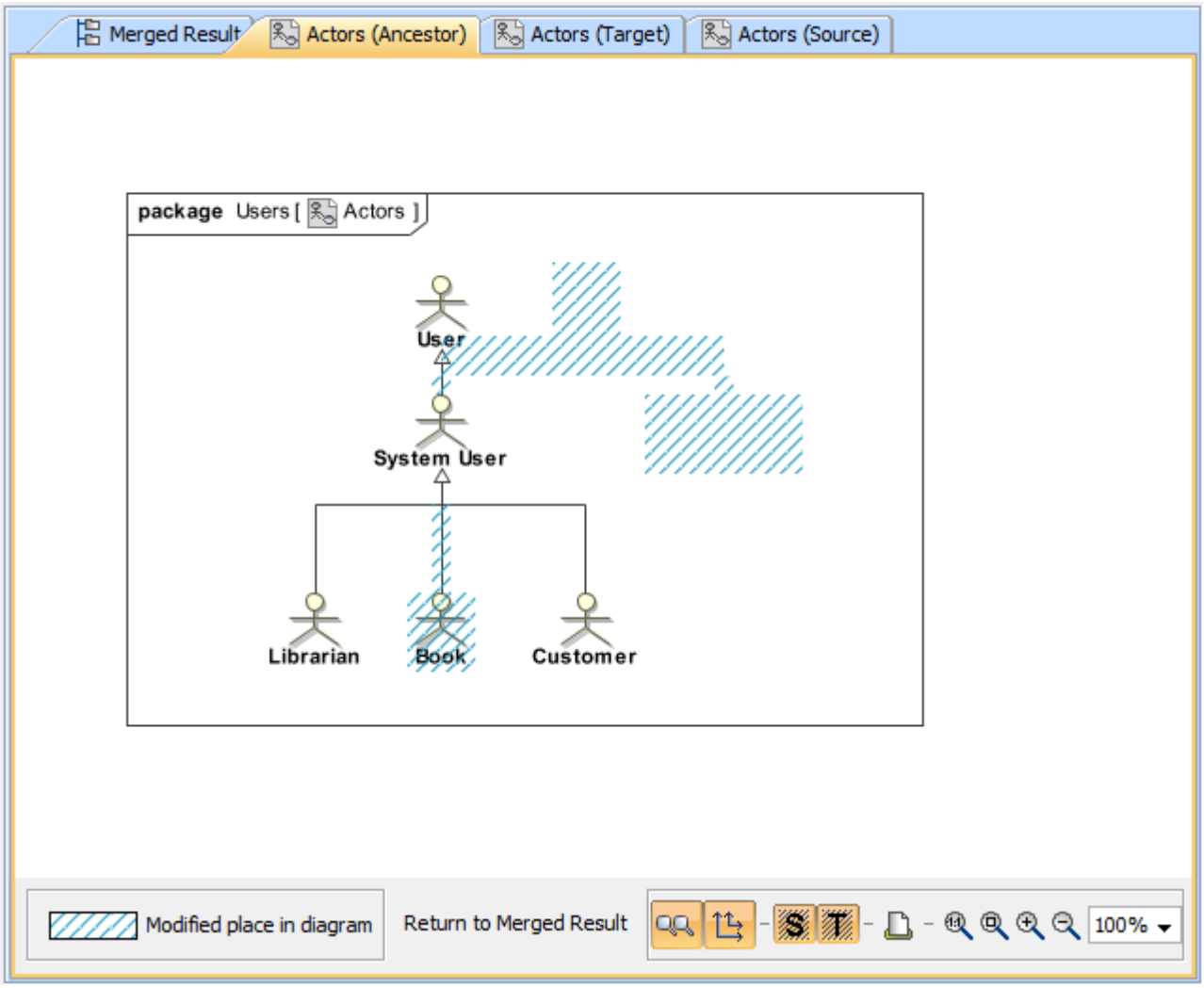


Figure 281 -- Marking both source and target changes in ancestor's diagram difference viewer

Finishing merge

To finish the merge procedure

1. On the **Merge** window, click the **Finish Merging** button. The question dialog box prompting to confirm changes that have been made to the target will open (see the following figure).
2. Do either:
   - Click **Yes**. All accepted changes will be copied to the target.

IMPORTANT! After the merge results are copied to the project, do not forget to save or to commit the project to the server.

Copyright © 1998-2015 No Magic, Inc.

- Click **No**. No changes will be copied to the target. The project remains unchanged.

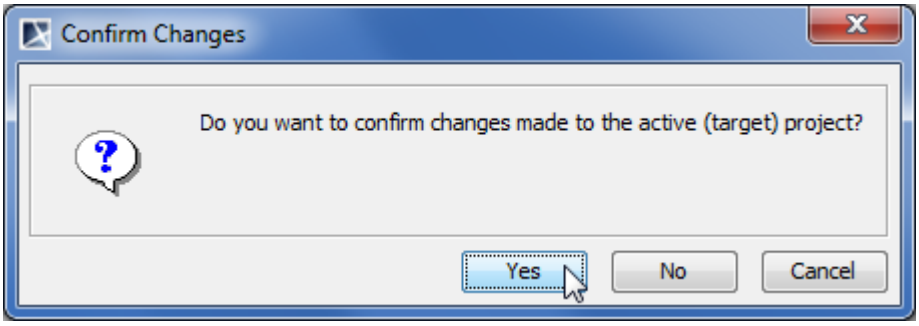


Figure 282 -- Dialog box prompting for changes confirmation

You can also click **Cancel** to close the dialog box and continue the merge procedure, that is go back to Analyzing and managing merge results.

Turning on showing merged diagram annotations

Before finishing the merge procedure, you can turn on showing merged diagram annotations. Click the Annotate Merged Diagram button on the toolbar that is located at the top of the **Merge** window for this.

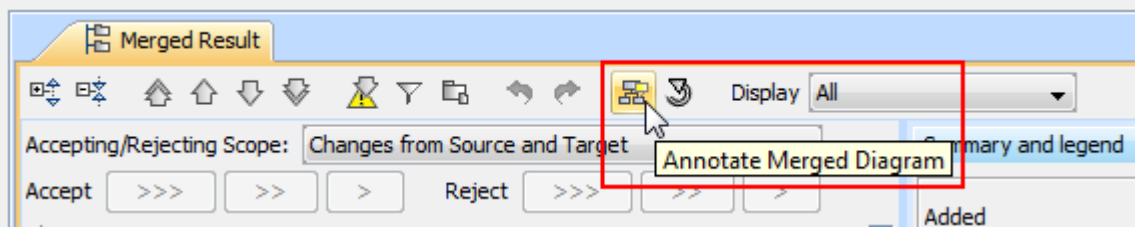


Figure 283 -- Turning on diagram annotations

Pattern Wizard

In MagicDraw, you can find various GOF, Java, Junit, CORBA IDL, and XML Schema design patterns.

NOTE This functionality is available in Standard, Professional, and Enterprise editions only.

You can also create new patterns and edit existing ones using Java code or JPython scripts. For a detailed description, see MagicDraw OpenAPI UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

To open the **Pattern Wizard**

- Select **Tools** from the class shortcut menu and then select the **Apply Pattern** subcommand.

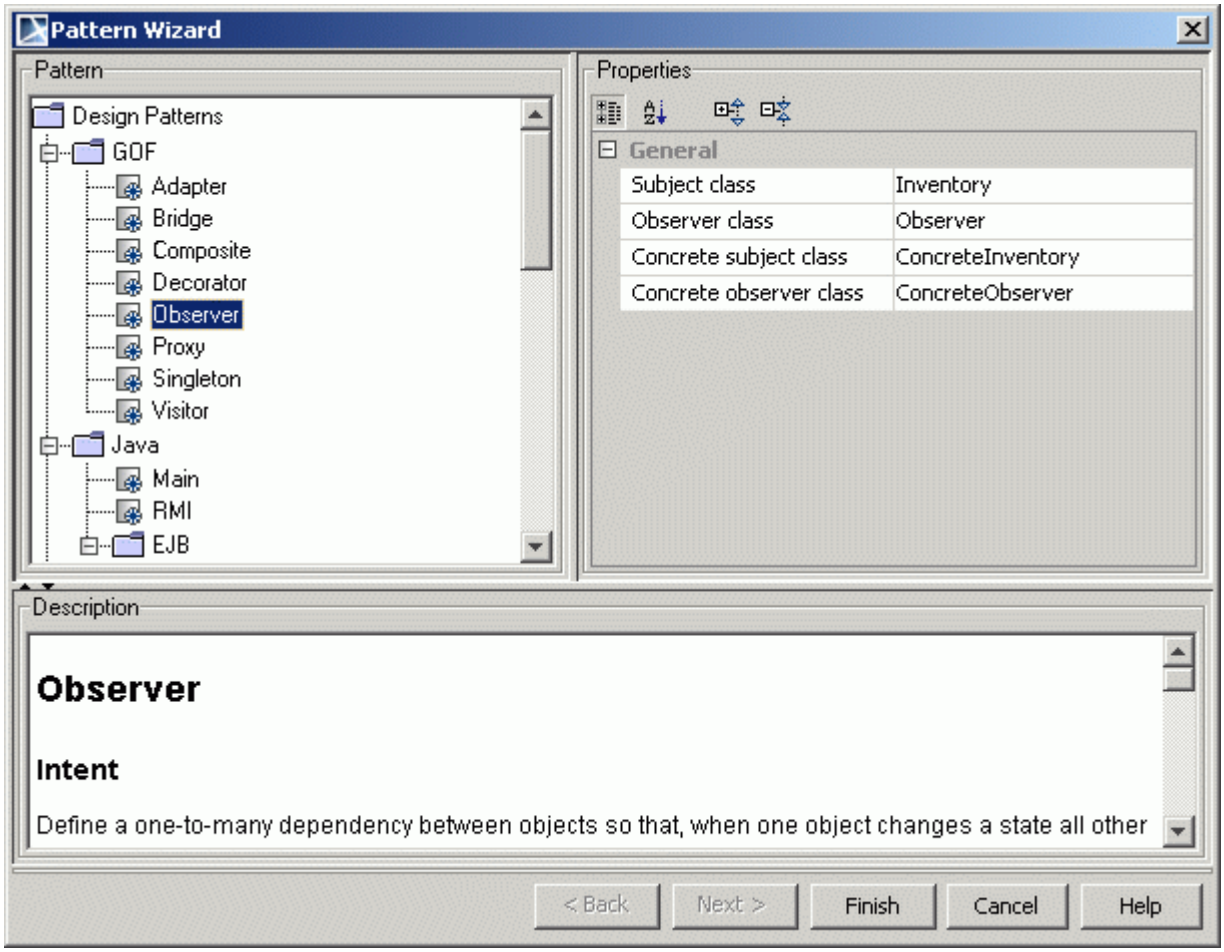


Figure 284 -- Pattern Wizard

Copyright © 1998-2015 No Magic, Inc.

The **Pattern Wizard** has three main collections of customizable options, which are represented by the hierarchy tree on the left side of the dialog box:

Category	Design Pattern	Properties
GOF Templates described in the Design Patterns of Reusable Object-Oriented Software	**Adapter**	Interface Class Adapter Class Adaptee Class **NOTE**: The **Next >** button is activated. In the **Adapter Operations** screen, add or remove operations you want to use.
	Bridge	Abstraction Implementor Implementor is: Abstract Class or Interface Name of Reference Suffix of the Concrete Implementor **NOTE**: The **Next >** button is activated. In the **Deriver Classes** screen, add or remove classes you want to use.
	Composite	Component Class Composite Class **NOTE:** The **Next >** button is activated. In the **Composite Operations** screen, add or remove operations you want to use.
	Decorator	Component Class Decorator Class Concrete Decorator Class **NOTE:** The **Next >** button is activated. In the **Decorator Operations** screen, add or remove operations you want to use.
	Observer	Subject Class Observer Class Concrete Subject Class Concrete Observer Class
	Proxy	Subject Class Proxy Class Real Subject Class **NOTE:** The **Next >** button is activated. In the **Proxy Operations** screen, add or remove operations you want to use.
	Singleton	Singleton class
	Visitor	Visitor Class Implementation Style: visit (MyClass), visitMyClass (MyClass) **NOTE:** The **Next >** button is activated. In the **Elements to Visit** screen, select classes and interfaces you want to visit.
Java Specific Java design patterns	**Main**	Main class

Copyright © 1998-2015 No Magic, Inc.

Category	Design Pattern	Properties
	RMI	Java RMI classes Remote Interface **NOTE:** The **Next >** button is activated. In the **Remote methods** screen, select methods from the **All** list to the **Selected** list.
JUnit JUnit is a regression testing framework. It is used by the developer who implements unit tests in Java. JUnit is Open Source Software. The provided templates allow the user to create the constructions implemented in the JUnit framework. For more information, go to http://www.junit.org.	**TestCase**	TestCase Class Create suite() Create Constructor TestCase(String) Create runTest() Create setUp() Create tearDown()
	Tested Class	Tested Class TestCase Class Create suite() Create Constructor TestCase(String) Create runTest() Create setUp() Create tearDown() **NOTE**: The **Next >** button is activated. In the **Tested Operations** pane, add or remove operations you want to use.
XML Schema Specific XML design patterns	**XSD complex Type**	Target Class Content: XSDcomplex content, XSDsimple content.
	XSD compositor	Target Class Compositor: XSDall, XSDchoice, XSDsequence Particle
	XSD simpleType	Target Class Content: XSDrestriction, XSDlist, XSDunion
	XSD simpleType (XSDlist)	Target Class Item Type
	XSD simpleType (XSDunion)	Target Class

Copyright © 1998-2015 No Magic, Inc.

Category	Design Pattern	Properties
	Simple XSD restriction	Target Class Stereotype: XSDsimpleType, XSDsimpleContent Base Min Exclusive Max Exclusive Max Inclusive Min Inclusive Total Digits Fraction Digits Length Min Length Max Length White Space Pattern
WSDL **Specific WSDL design pattern**	**Binding**	Use the **Binding** pattern when you want to create binding of some PortType.
CORBA IDL **Specific CORBA IDL design patterns**	**Interface**	Name Abstract Local Interface
	Value Type	Name Abstract Custom Value Type
	Type Definition	Name Type Definition Specifier: typedef, boxed value Base Type
	Sequence	Name Base Type Sequence Size Anonymous
	Array	Name Base Type Array Size Anonymous
	Fixed	Name Digits Scale Anonymous
	Union	Name Discriminator Type
	Enumeration	Name
	Struct	Name
	Exception	Name

Copyright © 1998-2015 No Magic, Inc.

Category	Design Pattern	Properties
< Back		Go back to the **Pattern** screen.
Next >		Go to the other appropriated screen.
Finish		Finish and implement the wizard. The appropriated classes and interfaces are created.
Cancel		Cancel the wizard without implementing your actions.
Help		The MagicDraw Help is displayed.

Creating Setters/Getters



This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

Setters and getters are common operations that contain almost every class. With the help of MagicDraw, set and get operations for class attributes can be generated automatically.

To create a setter or getter

- From the shortcut menu of the selected class, select **Tools**, and then select **Create Setters/Getters**. The **Select Attributes/Association Ends** dialog box opens.

Copyright © 1998-2015 No Magic, Inc.

- Add a tagged value "getter/setter for attribute=attribute_name" to the selected class.

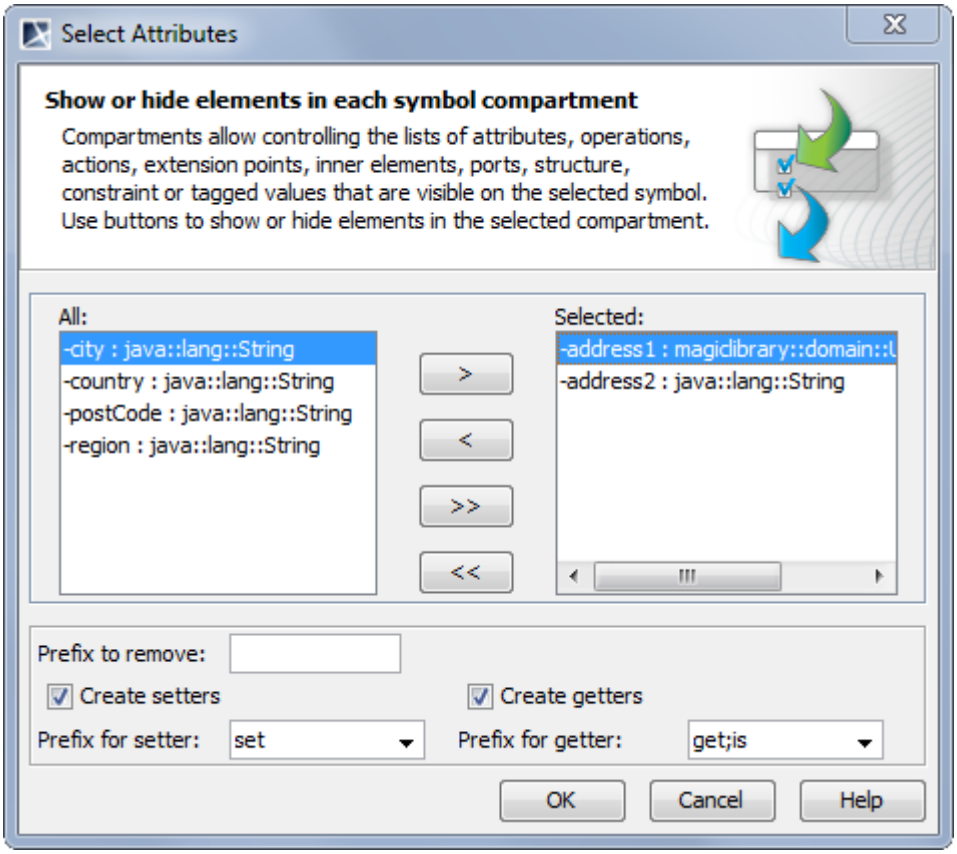


Figure 285 -- Select Attributes dialog

Box	Function
All	Contains names of all attributes defined within the selected class.
Selected	Contains the selected attributes.
>	Moves the selected attribute from the **All** list to the **Selected** list. Setter for that attribute will be generated.
<	Moves selected attribute from the **Selected** list to the **All** list.
>>	Moves all attributes from the **All** list to the **Selected** list. Setters for all attributes will be generated.
<<	Moves all attributes from the **Selected** to the **All** list.
Prefix to Remove	Type a prefix of an attribute (-, ….) you want to remove while generating setters or getters.
Create Setters	Generates setters for the selected attributes.
Create Getters	Generates getters for the selected attributes.
Prefix for Setter	Select a prefix for the generated setter (operation). Possible choices: set or Set.
Prefix for Getter	Select a prefix for the generated getter (operation). Possible choices: get;is, Get;Is; get, or Get. **NOTE:** “Get” is used for every getter, “is” is used if the type of an attribute is set as Boolean.
OK	Generates setters and/or getters for attributes that are in the Selected Items list.
Cancel	Exits the dialog box without any changes.

Copyright © 1998-2015 No Magic, Inc.

Box	Function
Help	Displays the MagicDraw Help.

The names of created operations (setters) are combined according to the following format:

```
public void set + <attribute name> (<attribute type> <attribute name>)
```



If you have an attribute called x of type int, then the generated setter will look this way:

```
public void setx (int x)
```

The names of created operations (getters) are combined according to the following format:

```
public <attribute type> get + <attribute name> ( )
```



If you have an attribute called x of type int, then the generated setter will look this way:

```
public int getx ();
```

Implementing or Overriding Operations



This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

When you inherit classes from the base class that has abstract functions, you have to redefine them in the inherited classes. The implement/override operations tool will help you generate operations that are defined as abstract in the base class.

The **Implement/Override Operations** command can be invoked in 2 cases:

- When one classifier inherits operations from the base classifier (Generalization relationship).
- When some classifiers implement an Interface (Realization relationship).

Copyright © 1998-2015 No Magic, Inc.

To start the **Implement/Override Operations** tool

From the shortcut menu of the selected class, select **Tools**. Then, select **Implement/Override Operations**. The **Select Operations to Implement/Override** dialog box opens.

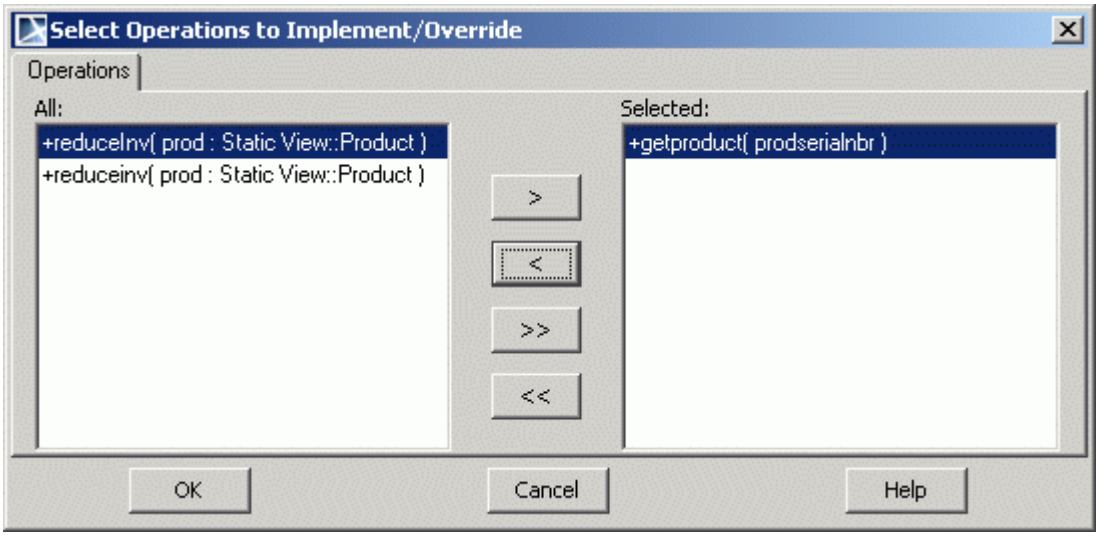


Figure 286 -- Select Operations to Implement/Override dialog box

Box	Function
All	Contains names of all operations defined within the selected class.
Selected	Contains the selected operations.
>	Moves the selected operation from the **All** list to the **Selected** list.
<	Moves selected operation from the **Selected** list to the **All** list.
>>	Moves all operations from the **All** list to the **Selected** list.
<<	Moves all operations from the **Selected** to the **All** list.
OK	Generates operations that are in the **Selected** list.
Cancel	Exits the dialog box without any changes.
Help	Displays the MagicDraw Help.



Double-click the item name and it will be moved to the opposite list.

Copyright © 1998-2015 No Magic, Inc.

Model Transformation Wizard



Transformations



Transformation engine itself is available in MagicDraw editions from Standard and up.

However, only MagicDraw Architect and Enterprise editions bring any particular transformations. So for MagicDraw Standard and Professional edition users, transformations are not available.

Plugins can bring additional transformations regardless of MagicDraw edition. For example, users, who have the Cameo Data Modeler plugin, can use the transformation engine to run ER to SQL(Generic/ Oracle) transformations, even if they do not have the MagicDraw Architect/Enterprise editions.

The Model Transformation Wizard enables running one of the predefined transformations. When using this wizard to run a chosen transformation, you have to perform the following steps:

1. Choose a transformation type.
2. Specify both the transformation source model (or a part of it) and destination package.
3. Select a type map.
4. Set custom transformation properties.

Each transformation converts the chosen model part into another model according to the rules of this transformation. Transformations are usually used for converting models between different modeling domains, e.g., UML and SQL.

All transformations follow a similar approach. They take a part of a model as the transformation source and copy it to the destination model, establishing traces between the transformation source and target elements. Then each transformation performs the specific model reorganizations, which are necessary for each transformation type according to the transformation options specified by the user in the transformation wizard. Transformation can also be performed in-place, i.e., the source model is not copied to the destination model, but transformation works directly on it instead.

Transformations also perform the so-called type remapping. During the transformation between the different modeling domains, such as UML and SQL, it is necessary to go through the data types used in the source model and change the types from the source domain into the equivalent types in the target domain, for example, changing String type usages in the UML model into the varchar type usages in the SQL model.

Available Transformations

Transformations are usually used for converting models between different modeling domains. Transformations are named by the types of their source models and their destination models. These are the available transformations:

- **Any to Any.** This transformation copies all your model or part of it to another package without making any changes. You can also remap types in the destination model by applying some type mapping rules.
- **Profile Migration.** Helper transformation for migrating models using one profile to models using another profile (usually - between different versions of the same profile - old and new).



UML to SQL(Generic/Oracle), SQL to UML, UML to XML Schema, XML Schema to UML transformations are available with the separately-installed Cameo Data Modeler plugin (which comes free of charge with MagicDraw Architect and Enterprise Editions and is separately purchseable for MagicDraw Standard and Professional editions).

Copyright © 1998-2015 No Magic, Inc.

NOTE

Additional custom types of transformations can be defined by MagicDraw plugins.

Working with Model Transformation Wizard

The wizard can be opened from several places.

To start the **Model Transformation Wizard**

Do either:

- From the **Tools** menu, choose **Model Transformations**.
- Right-click one or more packages and select **Tools** > **Transform**.

Selecting transformation type

In the first step of the Model Transformation Wizard, a list of the available transformation types is displayed.

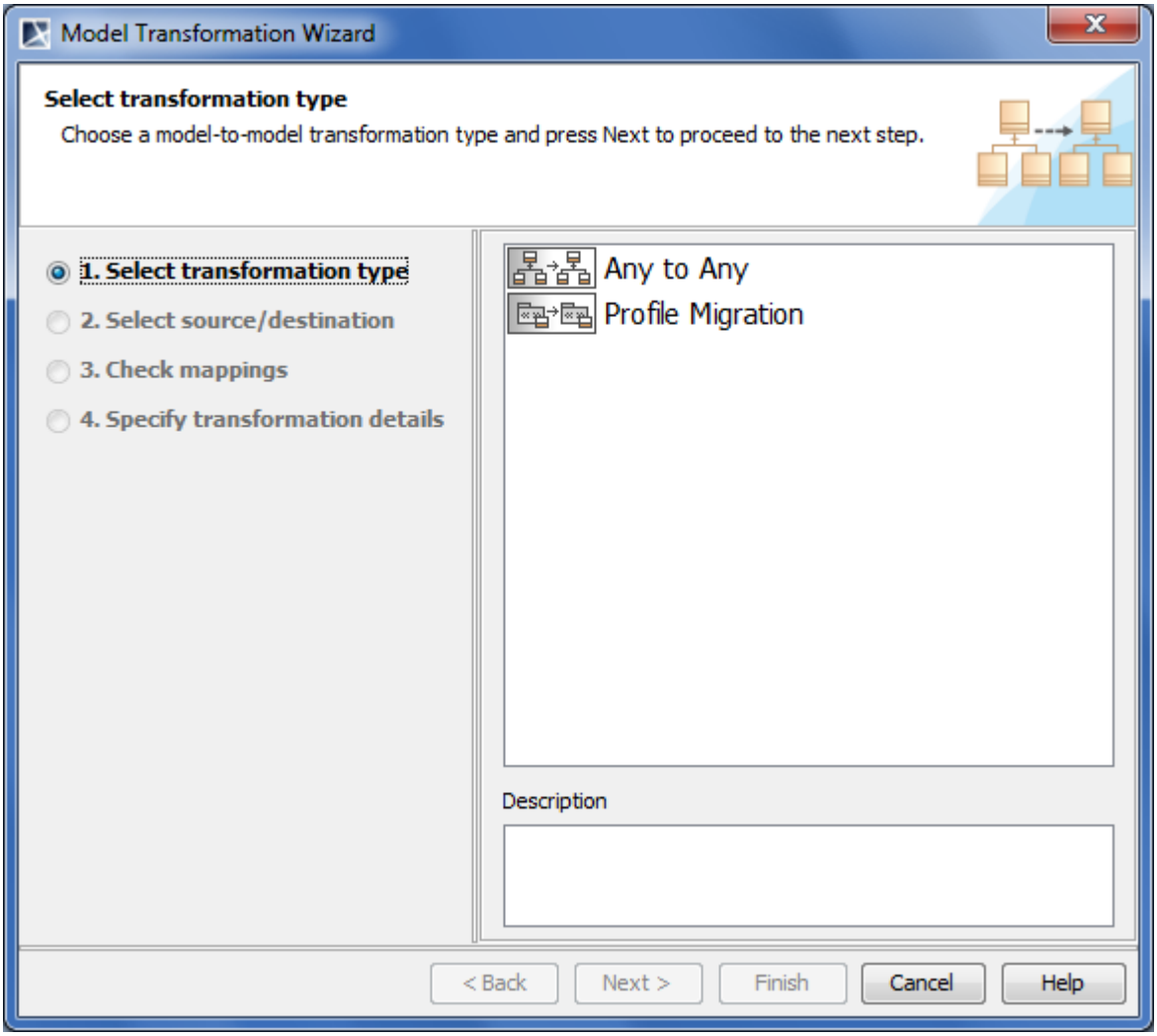


Figure 287 -- Model Transformation Wizard. Select transformation type

Transformation types are displayed in the list window.

Copyright © 1998-2015 No Magic, Inc.

The following operations are available in the **Select transformation type** window:

Button	Function
Next >	Proceed to the next step (in this case, Select source/destination).
Cancel	Cancel the wizard.
Help	Display the MagicDraw Help.

Selecting source and destination models

In the second step of the wizard, the **Transformation Source** tree displays all project data, i.e. the packages and their inner elements, that can be selected as a transformation source. Transformation will take the selected elements as input data.

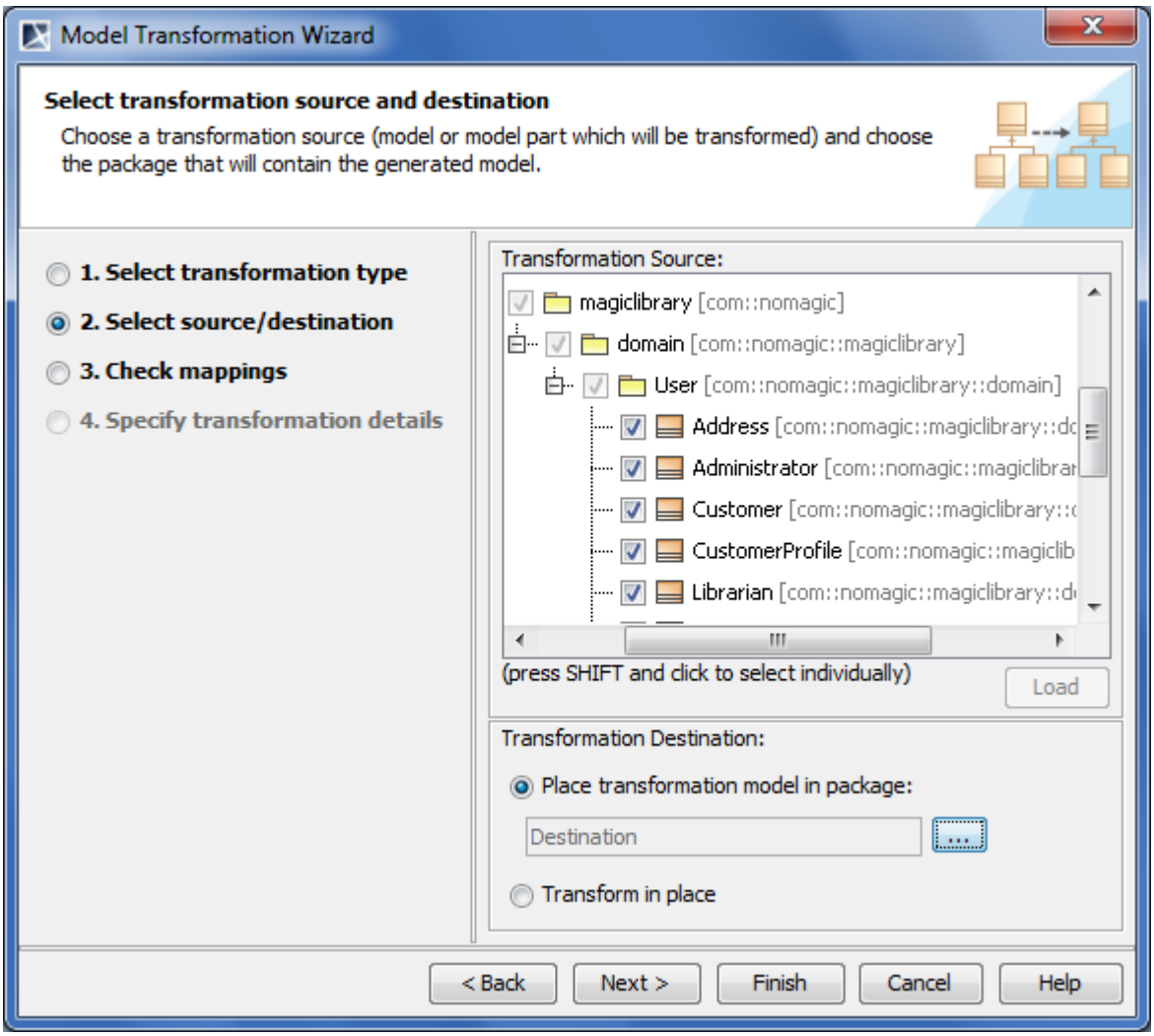


Figure 288 -- Model Transformation Wizard. Select source/destination

Select the **Place transformation model in package** option button to specify the package into which the source will be transformed. Click the **...** button to display the **Destination Package** dialog. Select an existing package from the **Packages** tree or create a new one.

Copyright © 1998-2015 No Magic, Inc.

Select the **Transform in place** option button, if you want the source model to be edited.



If you choose the in-place transformation, the model part selected as the transformation source, will be edited directly, and you will not retain your original model. So, please, be careful with this option.

If you choose the destination package, the source model will be copied to it and the transformation will be performed on this copy. Hence you will retain your source model and get a resulting model and traces will be established between elements in these model parts.

The following operations are available in the **Select source/destination** window:

Button	Function
< Back	Return to the previous dialog box.
Next >	Proceed to the next step (in this case, Select type mappings).
Finish	Finish the transformation configuration. All other options will be set by default. The **Model Transformations Wizard** exits and transformation results appear in the project.
Cancel	Cancel the wizard.
Help	Display the MagicDraw Help.

Copyright © 1998-2015 No Magic, Inc.

Selecting type mappings

The third step in the wizard allows for selecting a type map that will be applied during the transformation. Usually transformation has and brings in some predefined type map, but if you want, you can specify another type map.

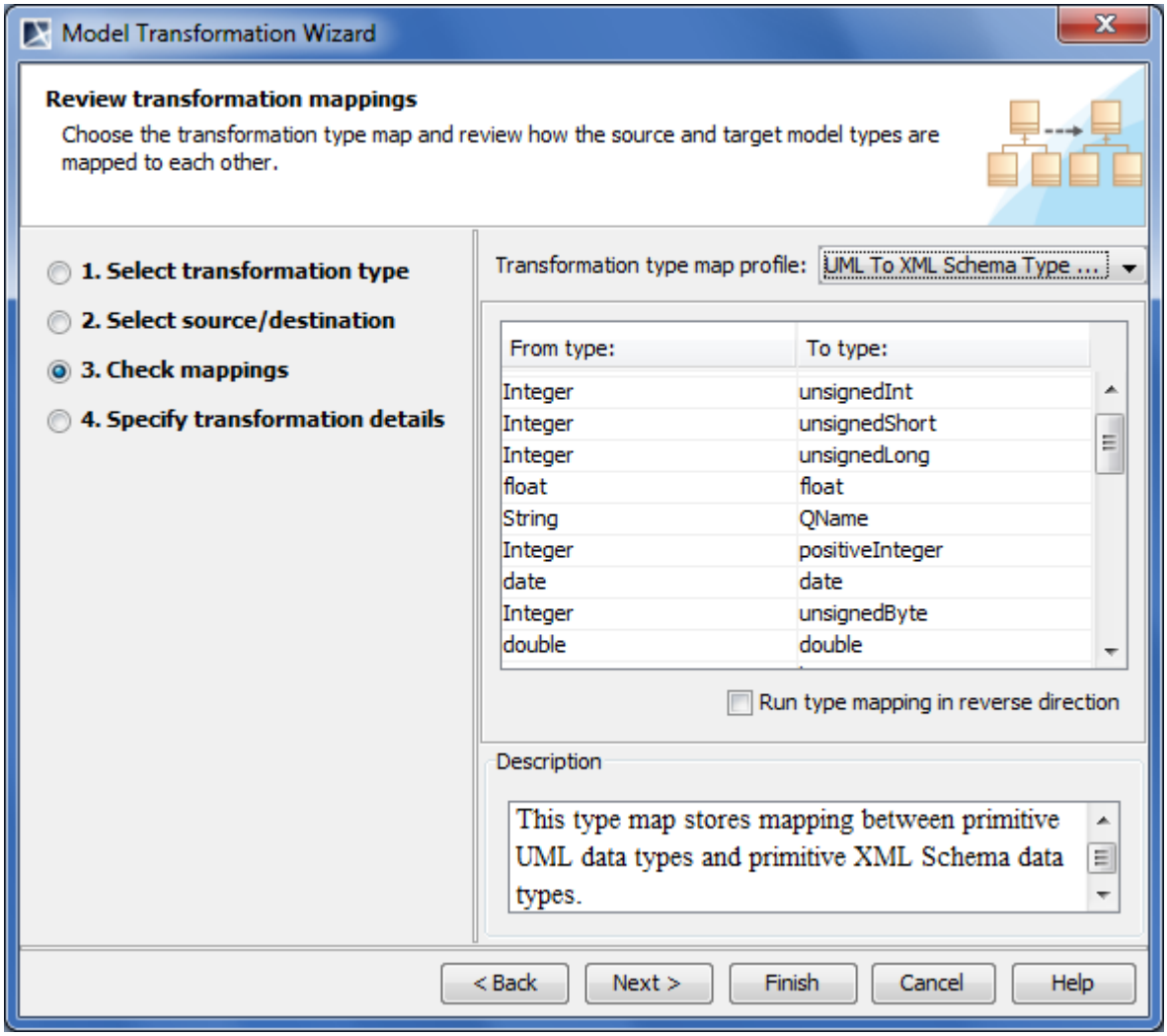


Figure 289 -- Model Transformation Wizard. Select type mappings

A type map can be regarded as a collection of rules of the form “Replace the usage of type X in the used project with the usage of type Y”.

A type map is a model object, i.e. a package with a collection of dependencies (for the details about modeling type maps, see section “Transformation Type Mapping” on page 474), hence all model manipulation operations can be performed on it. In particular, it can be refactored into a used project and mounted into any project, which needs it. It can be a simple package in your project as well, if you need a custom, one-off type map. A predefined type map can be taken from the MagicDraw module and edited.

To see a list of the type maps available in your model, click the down arrow in the **Transformation type map** combo box. These type maps specify the mapping rules that will be applied to the model during the transformation.

When you select a particular map, its contents are displayed in a table below. Each row in the table is a rule to remap one particular type to another. The **From type** and **To type** columns in the table show the source and target types.

Copyright © 1998-2015 No Magic, Inc.

The **Run type mapping in reverse order** check box creates the opposite type mapping. Type maps can be bidirectional, e.g., the same type map is reused both in the UML to XML schema and XML schema to UML transformations. This checkbox governs the direction in which the type map should be used.

The following operations are available in the **Select type mapping** window:

Button	Function
< Back	Return to the previous dialog box.
Next >	Proceed to the next step (in this case, Specify transformation details). **NOTE:** This button is disabled during the **Any to Any** transformations.
Finish	Finish the configuration of the transformation. The **Model Transformations Wizard** exits and the transformation results appear in the project.

Setting custom transformation properties

The **Transformation Details** table displays the various properties of a specific transformation, selected in the first step of the **Model Transformation Wizard.** Each transformation type has its own set of options, which govern functionality of that transformation.

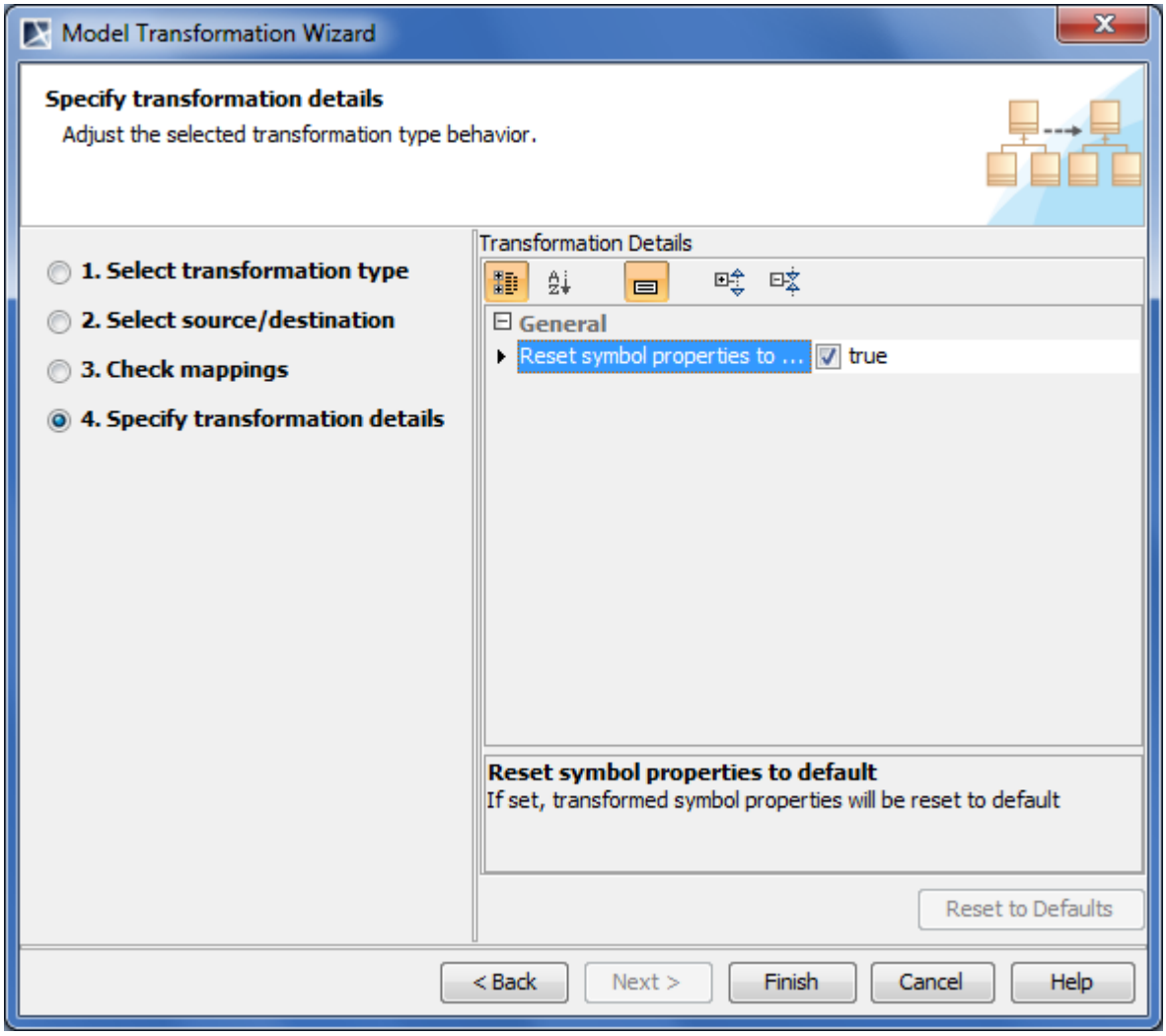


Figure 290 -- Model Transformation Wizard. Specify transformation details

To change the transformation properties to the default values, click **Reset to Defaults**.

Copyright © 1998-2015 No Magic, Inc.

The following operations are available in the **Specify transformation details** window:

Button	Function
< Back	Return to the previous dialog box.
Finish	Finish the transformation configuration. The **Model Transformations Wizard** exits and the transformation results appear in the project.

Transformation Type Mapping

During the transformation between the different modeling domains, such as UML and SQL, it is necessary to go through data types used in the source model and change the types from the source domain into the equivalent types in the target domain, for example, changing String type usages in the UML model into varchar type usages in the SQL model. This is achieved by establishing a type map and then supplying it for the transformation (many transformations have default, predefined type maps).

A type map can be regarded as a collection of rules of the form “Replace the usage of type X with the usage of type Y”. A type map is created by modeling means and is a model object, hence all model manipulation operations can be performed on it. In particular - it can be refactored into a module and mounted into any project, that needs it. It can be a simple package in your project as well, if you need a custom, one-off type map. Predefined type map can be taken from the MagicDraw module and edited.

Copyright © 1998-2015 No Magic, Inc.

A type map is a stereotyped package, holding a collection of stereotyped dependencies. Stereotypes for building type maps are stored in the *Model Transformation Profile*.

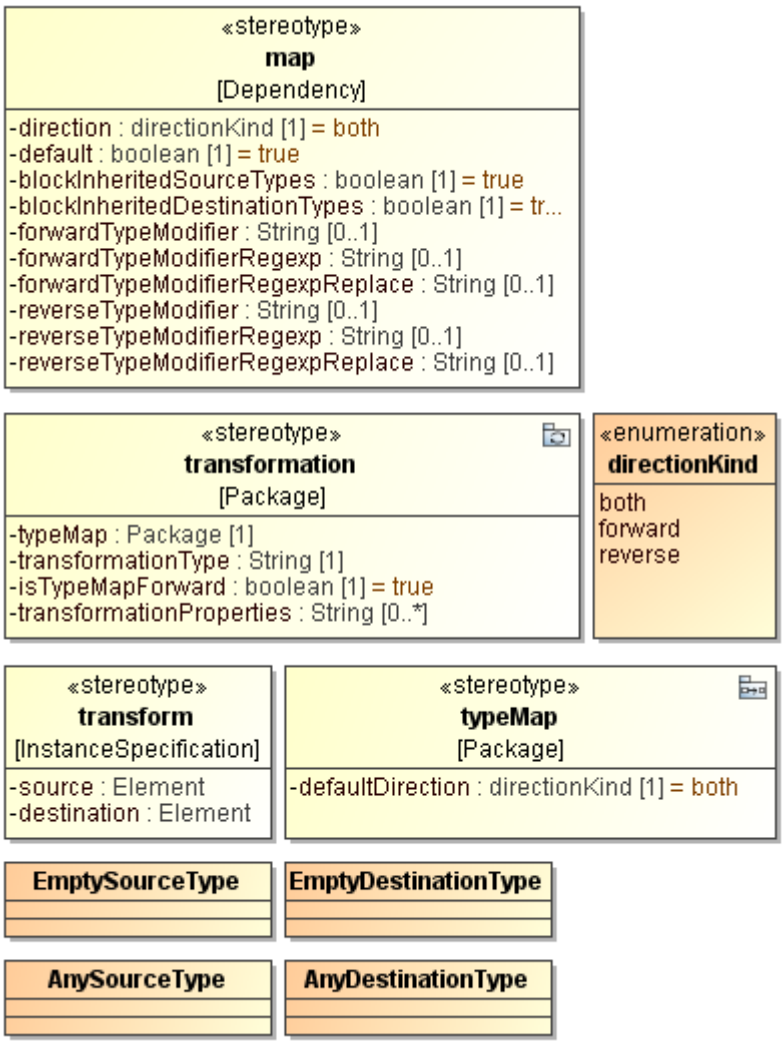
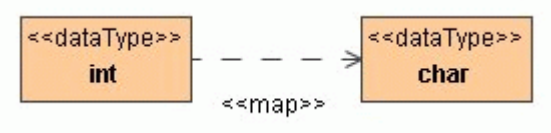


Figure 291 -- Stereotypes used for creating mapping rules

To create a transformation type map

1. Use or import *Model_Transformation_Profile.xml.zip.*
2. Create a package, which will represent your type map. Apply a stereotype «*typeMap*» to it.
3. Choose types (data types, classes, enumerations) in your source domain and their corresponding types in your target domain. Create the desired dependency relationships between the corresponding types. Apply a stereotype named «*map*» to these dependencies.



IMPORTANT!

Be sure to place dependencies in the type map package (MagicDraw is prone to placing dependencies in or near the dependent model element, so you may need to relocate them).

In the example above, after the transformation, all *int* types will be transformed to *char.*

Copyright © 1998-2015 No Magic, Inc.

Each of the thus created dependencies represents one type remapping rule. The package represents the complete type map.

Type mapping rule behavior can be further customized by setting various tags on the rules (see "Controlling Type Mapping Rule Behavior" on page 476).



Transitive type mapping (of the form type1->type2->type3) is not supported.

Controlling Type Mapping Rule Behavior

There are several different tags that can be set on a type map or an individual rule to change its behavior slightly.

Controlling direction

By default, the same type map can be applied in two directions: forward and backward. The backward direction can be set by selecting the **Run type mapping in reverse order** check box in the third step of the Model Transformation Wizard. This is useful, when there are two related opposite transformations for some domain; for example, the same type map is used for both UML to XML schema and XML schema to UML transformations.

If you want to limit the directions, in which type map can be used, you can set the **defaultDirection** tag for your type map package. Possible values are **forward**, **reverse**, and **both** (default).

The direction can also be limited on a per-rule basis. This is controlled by setting the **direction** tag on the type map dependency. Then the mapping rule is excluded from the rule set when the type map is run in a different direction than specified for this rule.

Multiple rules for the same source type

There can be multiple mapping rules for the same source type. For example, String -> varchar and String -> nvarchar. In this case, one of the rules must be marked as default by setting the **default** tag value on it to true.



The type map having several rules for the same type and without any one set as default cannot be used.

During the initial transformation, only the default rules for each source type come into play. E.g., if the user has a property with the String type, this will be transformed to property having the varchar type set.

However, during the transformation update, all rules come into play. If the destination type is one of the acceptable types according to the map, it is not changed. Otherwise it is replaced with the default mapping.

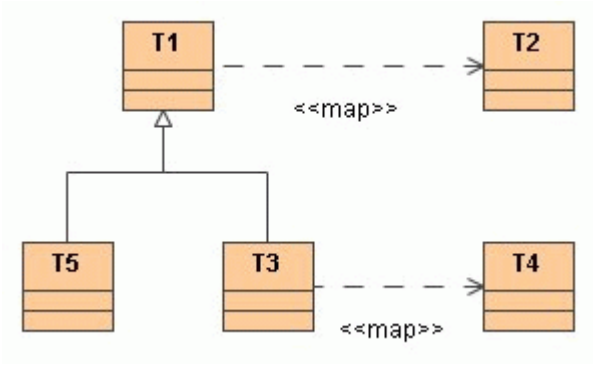
Regarding the example above, let's say that after the initial transformation, the user changes the type of the property in the destination model from varchar to nvarchar (as a post-transformation refinement process). If the user now runs a transformation update, this change will not be overwritten, since nvarchar is an acceptable type as there is a String -> nvarchar mapping in the type map as well. If on the other hand the user sets the type of this property to number, this would be reset the during transformation update, and the type will be forced back to varchar, as there is no String -> number mapping.

Controlling type inheritance, any, and empty types

You can also control mapping behavior for the type inheritance. By default, derived subtypes are also mapped by the rule governing the parent type (unless, of course, they have their own rules for mapping). If the

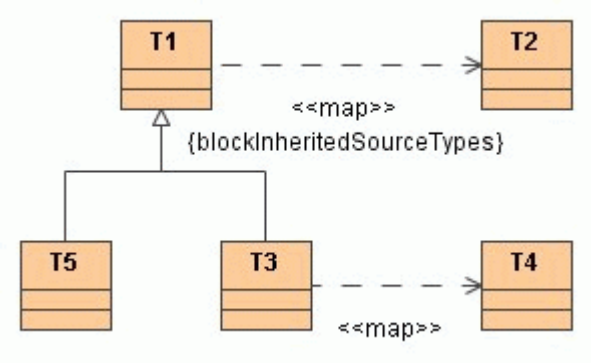
Copyright © 1998-2015 No Magic, Inc.

blockInheritedSourceTypes tagged value is set, derived types are not affected by this rule. Let's review the following example:



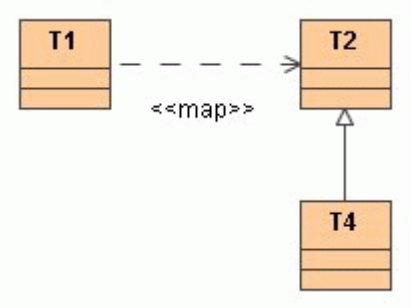
Here T1, T2, and T5 are types in the source domain, while T2 and T4 are types in the destination domain. Given these two mappings (T1 -> T2 and T3 -> T4), the following statement is true: T1, and all types derived from it (such as T5), are mapped to the T2 type, except T3 and any of the types derived from it. These types are mapped to T4.

Now consider an example where blockInheritedSourceTypes is set:



In this case, T3, along with the types derived from it, are still mapped to T4. T1 is still mapped to T2. However, unlike the previous example, T5 and all the types inherited from T1 are NOT mapped to T2.

You can also control the mapping behavior of the type inheritance in the destination model. This is only effective on the transformation updates, the second (and successive) reapplications of the transformation. By default, derived subtypes in the destination model are not overwritten, since they are considered suitable substitutes of their parent. Let's review the following example:



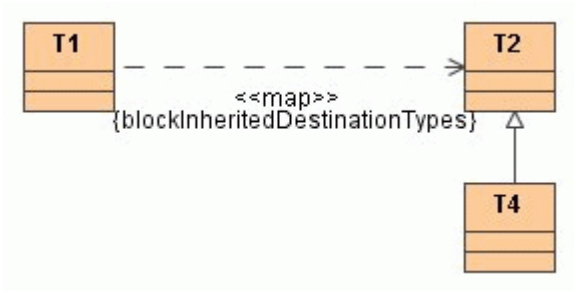
Here T1 is a type in the source domain, while T2 and T4 are types in the destination domain. Given this mapping (T1 -> T2), on the first application of the transformation, type T1 residing in the source model will be mapped to type T2 in the destination model.

Now let us look at a case, where the user refines the destination model by changing the type on the destination model attribute from T2 to T4. This situation is quite common, for example, the user refines an attribute type from string to basic URI in the XML schema, or from Integer to nonNegativeInteger, and so forth. The essence

Copyright © 1998-2015 No Magic, Inc.

is that the mapping for inherited types of T2 is performed as if there was a mapping T1 -> T2 (default), T1 -> T4, T1 -> <any_other_type_inherited_from_T2>.

Now consider what happens, when we apply the **blockInheritedDestinationTypes** tagged value:



In this case, type T4 has no special treatment. If the user applies the transformation, T1 is mapped to T2. Afterwards the user refines the destination model, changing the attribute type from T2 to T4. If the user now updates the transformation, the attribute type is overwritten: T4 is reset back to T2.

When the user loads the type map in the reverse direction, the roles of the **blockInheritedSourceTypes** and **blockInheritedDestinationTypes** are transposed (unless of course the **direction** tag mandates that this mapping is not used in the reverse direction).

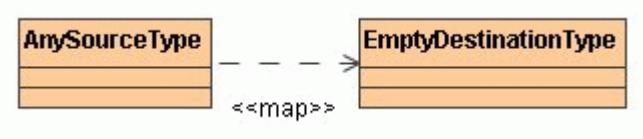
The special type **EmptySourceType** (residing in the *Model_Transformation_Profile.xml.zip*) is used in type maps to indicate that the attributes with no type should be mapped with this dependency.

The special type **EmptyDestinationType** (residing in the *Model_Transformation_Profile.xml.zip*) is used to indicate that the attributes in the destination classes should have no type after remapping (type removal).

The special type **AnySourceType** is a template that matches any type in the source model (see mapping rules for type inheritance). By using this type, together with the inheritance mapping rules, the user can specify that any other types not defined by the mapping should be interpreted by this mapping.

The special type **AnyDestinationType** is a template that matches any type in the destination model (see mapping rules for type inheritance).

Here is an example of template type usage:



According to this rule, any types in the source model for which there are no other mapping rules should be stripped in the destination model.

Type modifiers

Type mapping rules can also affect type modifiers during the type replacement. Type modifier is a small string, which modifies type usage in the typed element. They are used, for example, for specifying arrays during the modeling (e.g., property type = char and type modifier = [30] gives property:char[30]). Type modifiers are extensively used in SQL models for specifying number field widths and varchar field lengths. For example, phone:varchar”(100)”, where varchar is a type of phone property and “(100)” is a type modifier.

Each type mapping rule can carry a triple <modifier, regexp, replacementregexp> for setting type modifiers during the type replacement. These are specified in the tags on the mapping rule <**forwardTypeModifier**, **forwardTypeModifierRegexp**, **forwardTypeModifierRegexpReplace**> triple for controlling modifiers during the forward application of type map and correspondingly the <**reverseTypeModifier**,

Copyright © 1998-2015 No Magic, Inc.

reverseTypeModifierRegexp, **reverseTypeModifierRegexpReplace**> triple for controlling modifiers during reverse application of type map.

Any of the components of the triple can be missing, i.e., not specified.

If no tags are specified, then type modifiers are not changed during the type remapping operation (whatever modifier was in the source model, it will be copied into the target model).

If just the modifier is specified for the mapping rule, then modifiers are set during the application of this type rule. This can be used for setting the fixed type modifiers. For example, mapping boolean in the UML model to number(1) in the SQL models (in this case the modifier=”(1)” is used in the type map).

If all three are specified, a modifier, regexp, and regexp replacement, modifier remapping is performed as follows: during the transformation, the existing type modifier is matched against the given regexp. If it does not match, the type modifier is overwritten with the value, specified in the modifier field of the rule. If it does match regexp, the replacement is run on the match result and produces a type modifier to be set as a result. This allows quite complex rules to be written and executed, however this mandates good knowledge of regexp.

Let’s review the following live example: in the char -> varchar type mapping rule for the UML to SQL transformation, the following triple can be used: modifier="(255)", modifierRegexp="^[\(\[]([0-9]*)[\)\]]$", and modifierRegexpReplace="($1)". This causes the char[20] type usages (type=char, modifier=”[20]”) in the source be changed to varchar(20); char (without modifier) would be remapped to varchar(255).

If regexp replacement is not specified, it is treated as if “$0” was specified: the type modifier is copied from the source, if it does match the regexp.

Transformation Traces and Update

When a transformation is performed, it establishes traces between the transformation source model elements and the transformation result model elements. These traces carry information of what was transformed into what. Traces are stored in an auxiliary package under the transformation destination package carrying the «transformation» stereotype and containing a lot of instance specifications (since this is implementation specific data, please, do not edit the internals). If you want to remove trace information, simply delete this package.

Trace information can be used for navigating between the model layers. This is done with the traceability features of MagicDraw. To navigate in the forward direction, i.e., from the transformation source model element to the destination model element, right-click that element and choose **Go To** > **Traceability** > **Model Transformations** > **Transformed To** > <element>. To navigate in the backward direction, i.e., from the transformation destination model element to the source model element, right-click that element and choose **Go To** > **Traceability** > **Model Transformations** > **Transformed From** > <element>.

Traceability information is also visible in the element’s Specification window, the **Traceability** tab; in the Properties panel, **Traceability** tab; it can also be depicted in the Relation Map diagram or in the custom dependency matrix.

Traces can be used for running the transformation update. The transformation update reapplies the transformation with the same source and target for the purpose of carrying additional changes from the source (which occurred after transformation was made) into the destination.

During the transformation update, presence of unmapped model elements in the source model indicates that these are newly added elements. Usual rules and the same behavior for the transformation are used for these elements as if this were the first application of the transformation.

During the transformation update, if model elements already contain mapping, and the source and destination does not match, the question arises - which properties to use. This is fundamental problem for all updating operations. The general solution is to have some kind of merge between the source and target. But merge is clumsy and expensive. MagicDraw implements a simplistic approach meaning that during the transformation update a user can choose, wherever he/she wants the source model element properties to win (destination

Copyright © 1998-2015 No Magic, Inc.

model element properties are overwritten) or wherever he/she wants the destination model properties to win (destination model properties are not changed).

To update a transformed model

- Right-click the destination package and select **Tools** > **Update Transformed Model**.

The **Model Refresh Options** dialog opens.

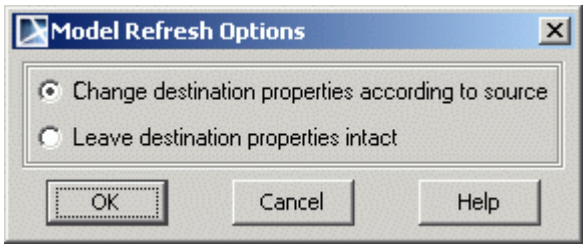


Figure 292 -- Model refresh options dialog box

The **Change destination properties according to source** option causes overwriting of element properties in the destination model with properties from the source model (only for the elements connected with mapping dependencies).

The **Leave destination properties intact** option leaves the destination model properties unchanged but different from the source model, in other words retains changes made to the destination model while it ignores changes made to the source model.

In Oracle DDL transformation updates, only changing of destination properties according to the source is available. It means, all changes are overwritten when updating a transformed Oracle DDL model, and the **Model Refresh Options** dialog does not appear in this case.

NOTE

If new elements are added to the source, once the update occurs, copies of the new elements will be created in the destination model. If an element is deleted from the source, it will not be removed from the destination after the update.

Profile Migration Transformation

Transformation provides a way to migrate your model according specified mapping rules.

Profile migration transformation search-and-replaces the following elements:

- Applied stereotypes
- Tagged values
- Usages of one type with another type

Element symbols are updated according to model changes.

Before starting transformation you have to define transformation mapping.

Profile Migration Transformation mapping

Before starting transformation you have to create transformation mapping rules.

To create mapping rules you need to create Dependency relationship between elements you want to transform. Mapping rules can be created (dependency relationship can be created) between the following elements:

1. Stereotypes
2. Tags

Copyright © 1998-2015 No Magic, Inc.

3. Types

Creating mapping rules for Stereotype transformation

This type of transformation is used to replace Stereotype. To create Stereotype transformation mapping rule:

1. Create Dependency relationship between Stereotypes which you want to transform.
2. Apply *ReplaceStereotype* stereotype to Dependency.
3. Perform transformation.

Tag values of old stereotype are preserved when tag name and type of tag value is the same. For tag values with different names create mapping rule for tag transformation.

ReplaceStereotype stereotype has the following tags:

- **disableNewTypeCreation** tag. By default false. Set this tag to true if you do not want to perform transformation when target and source metaclasses are not compatible. For example, if you do not want that Class would be changed to Use Case.
- **disableReplaceWhereSaveAsElementValue** tag. By default false. Set this tag to true if you do not want that stereotype would be changed where it is used as Tag value (tag value is stereotype itself, for which you perform transformation).

Creating mapping rules for Tag transformation

This type of transformation is used to replace Tag (when tag names differs). For example, source stereotype has *author* Tag and target stereotype has *name* Tag.

To create Tag transformation mapping rule:

1. Create Dependency relationship between Tags which you want to transform.
2. Apply *ReplaceTaggedValue* stereotype to Dependency.
3. Perform transformation.



To create mapping rule correctly, you have to create Dependency relationship not only between Tags, but also between Stereotypes of these tags (see mapping between stereotypes, which is described above).

Creating mapping rules for types transformation

This type of transformation is used to replace type. For example, to replace type of Attribute.

To create types transformation mapping rule:

1. Create Dependency relationship between Types which you want to transform.
2. Apply *ReplaceType* stereotype to Dependency.
3. Perform transformation.

Starting Profile Migration Transformation

To start the **Model Transformation Wizard**:

Do either:

- From the **Tools** menu, choose **Model Transformations**.
- Right-click one or more packages and select **Tools** > **Transform**.

Copyright © 1998-2015 No Magic, Inc.

To start the Profile Migration Transformation

- In the Model Transformation Wizard, select the **Profile Migration** transformation.

NOTE

The **Next** step is disabled in the wizard, if there are no mapping defined.

Sample of the Profile Migration Transformation

This sample describes step-by-step instructions how to create profile migration mapping rules and perform transformation. In this sample we will change one stereotype to other.

1. Create *Book* stereotype with Class metaclass.
2. Create your model, for example, create *Source* package with *Source* Class diagram. Draw *Sample* Class and apply *Book* stereotype.

To change *Book* stereotype to other, for example to *Magazine* stereotype, you have to create profile migration transformation mapping rules. Follow next steps for creating mapping.

3. Create stereotype *Magazine* with Class metaclass.
4. From the *Magazine* stereotype to *Book* stereotype draw Dependency relationship.
5. To the Dependency relationship apply *ReplaceStereotype* stereotype (see Figure 1).

Profile Migration transformation mapping rule is created. Now you can start transformation.

6. To start transformation select **Model Transformations** from the **Tools** menu. The **Model Transformation Wizard** will open.
7. Select the **Profile Migration** transformation and click **Next**.
8. Select the *Source* package in the *Select source/destination* step (Figure 3). Click **Next**.
9. The **From** and **To** fields display the mappings of the selected transformation profile in the *Check mappings* step (Figure 4). Click **Finish**.

After this transformation stereotype of *Sample* Class will be changed to *Magazine* stereotype.

TIP!

For more information about Model Transformation Wizard, see "Working with Model Transformation Wizard" on page 469.

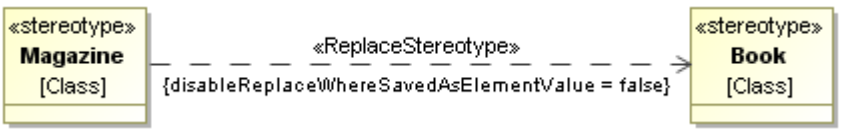


Figure 293 -- Profile Migration Transformation mapping

Copyright © 1998-2015 No Magic, Inc.

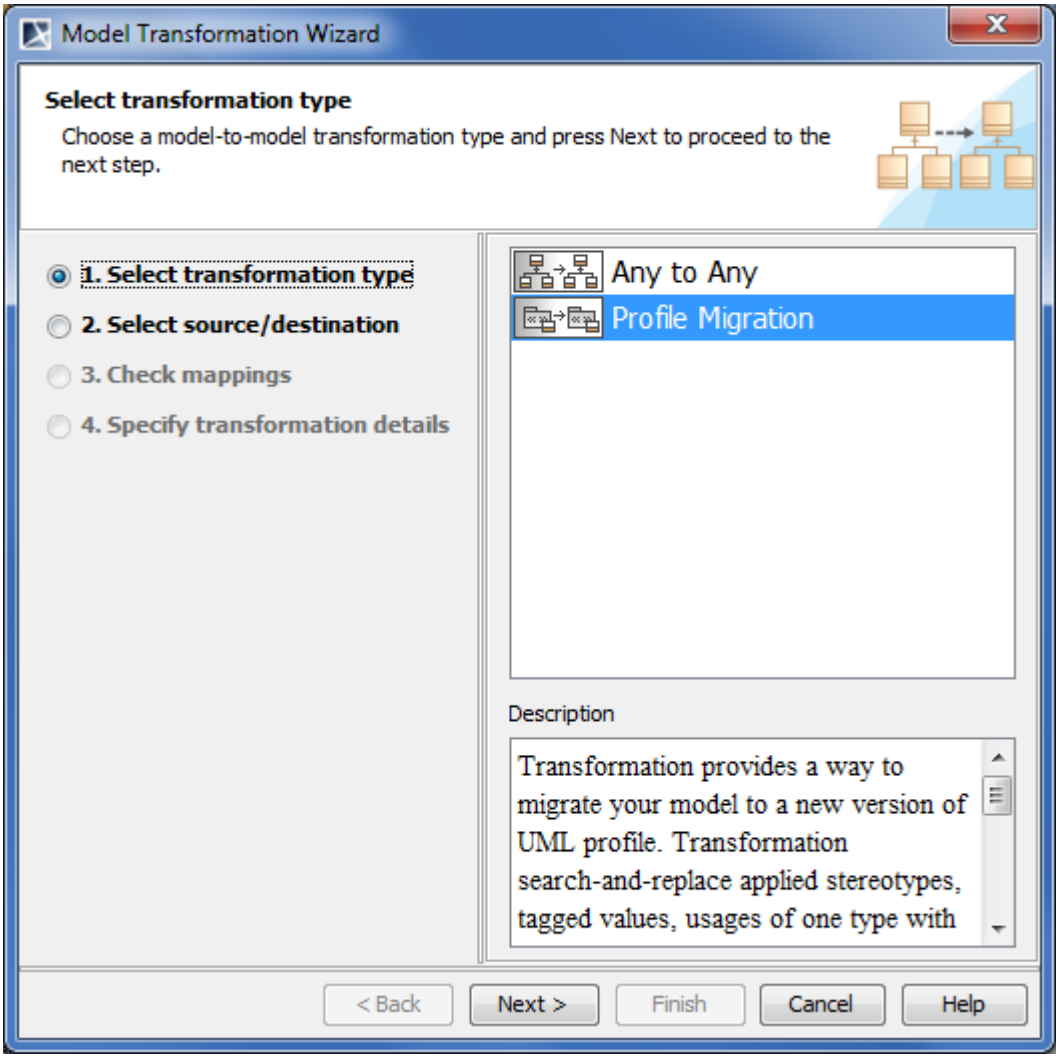


Figure 294 -- The Model Transformation Wizard, the Select transformation type step

Copyright © 1998-2015 No Magic, Inc.

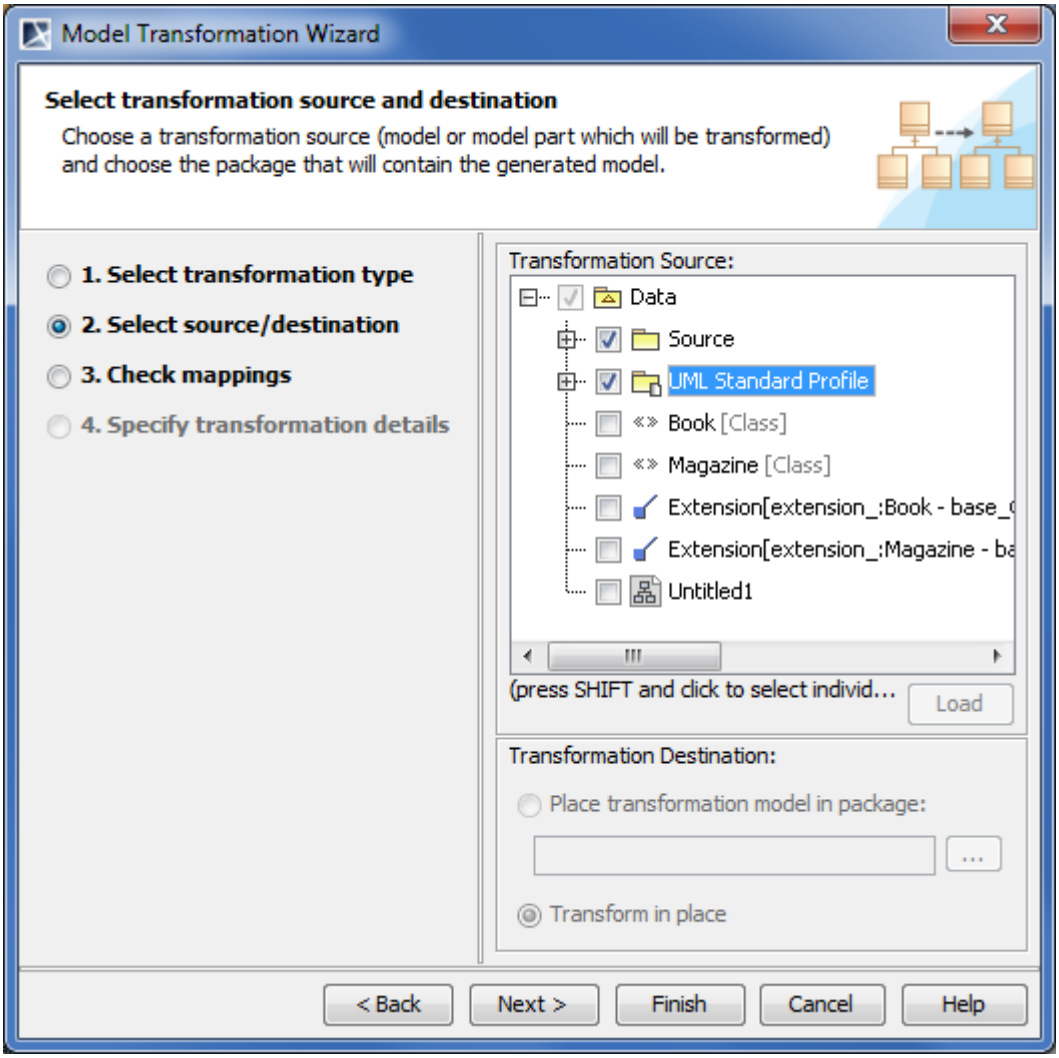


Figure 295 -- The Model Transformation Wizard, the Select source/destination step

Copyright © 1998-2015 No Magic, Inc.

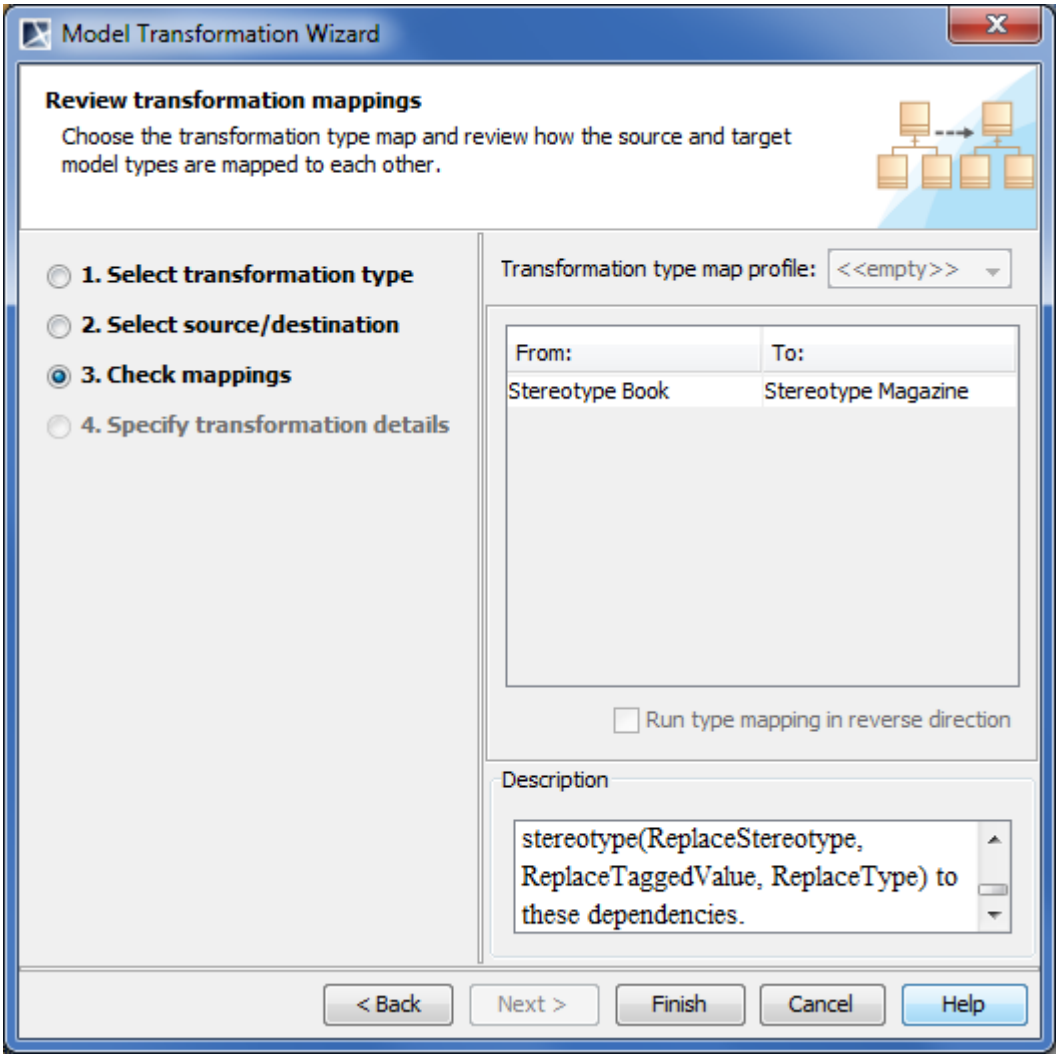


Figure 296 -- The Model Transformation Wizard, the Check mappings step

Automatic Instantiation Wizard

The Automatic Instantiation Wizard allows for creating instances of various entities with just a few clicks. It is especially useful when working with complex models or assembling large systems from parts. While creating instances automatically, you can:

- Select properties for which the instances will be created
- Change types of instantiated properties
- Assign default values for instantiated properties

For properties having the multiplicity more than 1, you can create as many instances of the same type as you need, since parallel parts can be added while creating instances. Previously, this feature had been supported by the SysML plugin only.

The Automatic Instantiation Wizard consists of three steps:

- Step #1. Parts selection
- Step #2. Package selection

Copyright © 1998-2015 No Magic, Inc.

- Step #3. Diagram creation (optional)

The Automatic Instantiation wizard analyzes the structure of the selected element and collects all available information about that element from the model (attributes, properties, and so on). All this information is presented in the first step of the wizard.

The Automatic Instantiation wizard evaluates collected information and provides a suggestion of all possible instances to be created.

If you are satisfied with default values provided in the Automatic Instantiation wizard, you can finish the wizard in the first step without making any changes. The instances will be created with the default values and in your working package. But if you want to change default values, create several different configurations, see the Working with Automatic Instantiation Wizard and the Case study: sections on how to do that.

To open the Automatic Instantiation wizard

1. Select the elements for which you want to create instances.
2. From the shortcut menu, select **Tools** > **Create Instance**. The **Automatic Instantiation** wizard opens.

Related references

Working with Automatic Instantiation Wizard

Working with Automatic Instantiation Wizard

Following all steps of the Automatic Instantiation wizard, you can create several different sets of instances. You can add each set in a different package and even display on the newly created diagrams.



After finishing the wizard, a very large number of instance specifications can be created. That depends on the size of your system. That is why we strongly recommend to select a new package wherein all created instance specifications will be stored.

Step #1. Parts selection

In this step, you can see all available instance specification values (parts) of your selected element. Here you can select which parts or properties to instantiate, create or select values for parts and properties, as well as add or remove parallel parts.

Parts are displayed if the multiplicity is greater than 1. The number of parts displayed depends on the multiplicity.

If the multiplicity is a particular value, for example 2, then you will see 2 parts displayed in then Automatic Instantiation wizard. Also in this case, you will not be able to add parallel parts or remove existing.

If the multiplicity is defined as an interval (for example, 0..*, 1..10, 1..*) or *, only one part will be created by default. If you need more parts, you can create them by pressing Insert or selecting **Add parallel part** form the selected property's shortcut menu. If the multiplicity is 1, parts will not be created.

The parts and properties are displayed with the suggested types and values. In this step of the wizard, you can change them.

You can change type for the part only in that case, if there is an available sub element for that part in your model (see the Case study:).

To select a value for a part

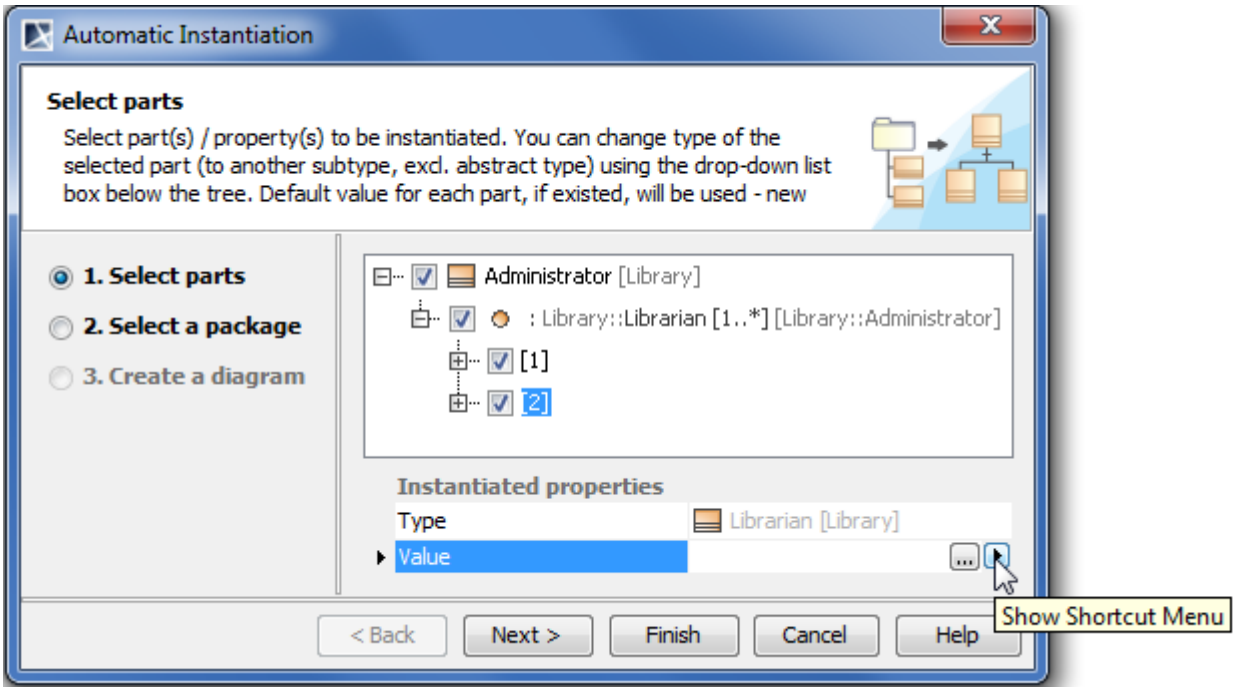
1. Select a part for which you want to add a value.

Copyright © 1998-2015 No Magic, Inc.

2. In **Instantiated properties**, click **Value** specification cell and then click the **...** button. The Select Instance Specification dialog opens.
3. In the opened dialog, select the instance specification you want to add for the selected part.
4. Click the **OK** button.

To create a value for a part

1. Select a part for which you want to create a value.
2. In **Instantiated properties**, click **Value** specification cell and then click the Show Shortcut Menu button.



3. In the opened shortcut menu, point **Value Specification** and then select on of the available value types.
4. In the **Value** property specification cell, type the desired value (see the following figure).

Copyright © 1998-2015 No Magic, Inc.

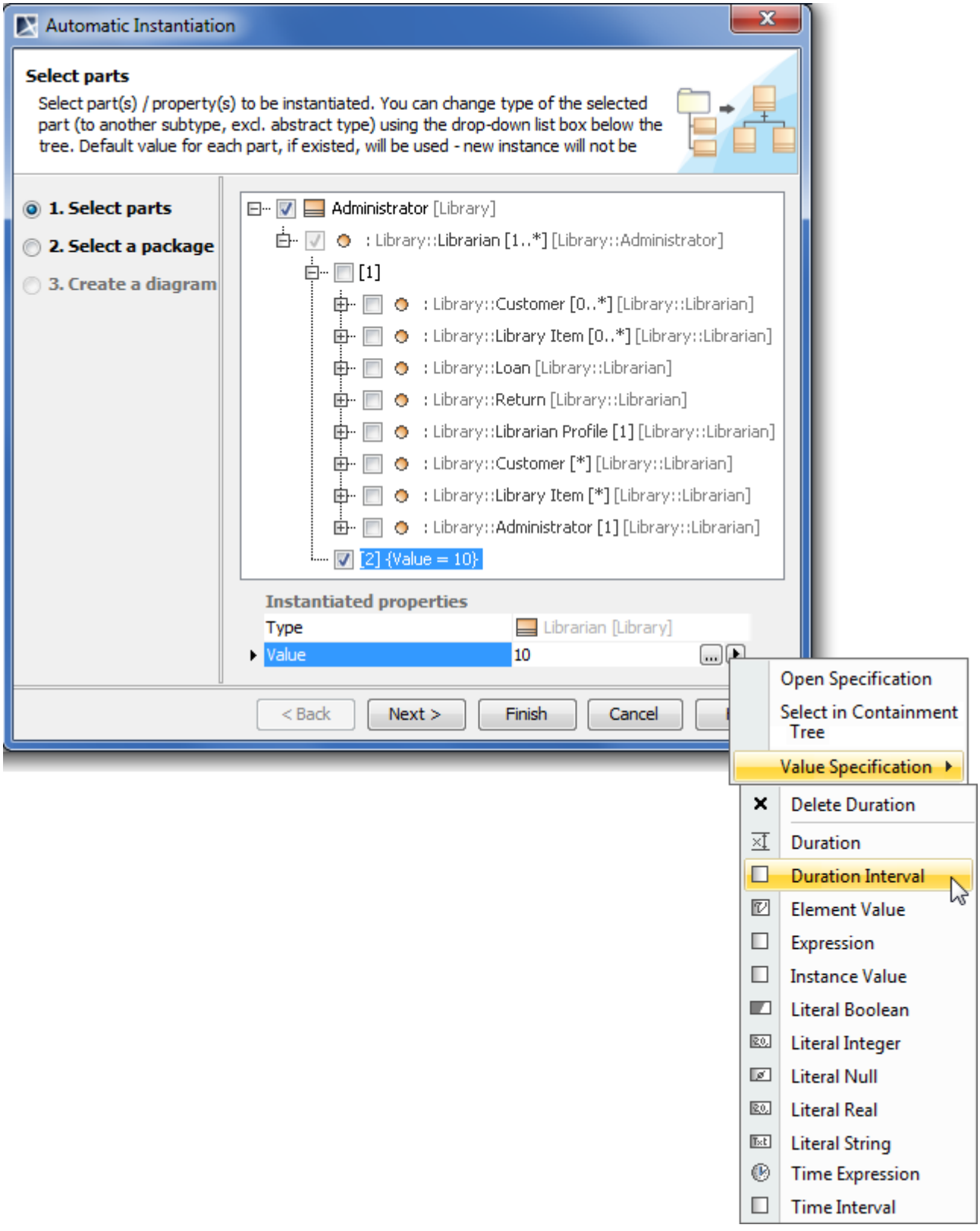


Figure 297 -- Selecting value

To add a parallel part

- Select a property for which you want to add a parallel part and do one of the following:
  - From the shortcut menu, select **Add parallel part**.

Copyright © 1998-2015 No Magic, Inc.

- Press **Insert** button.

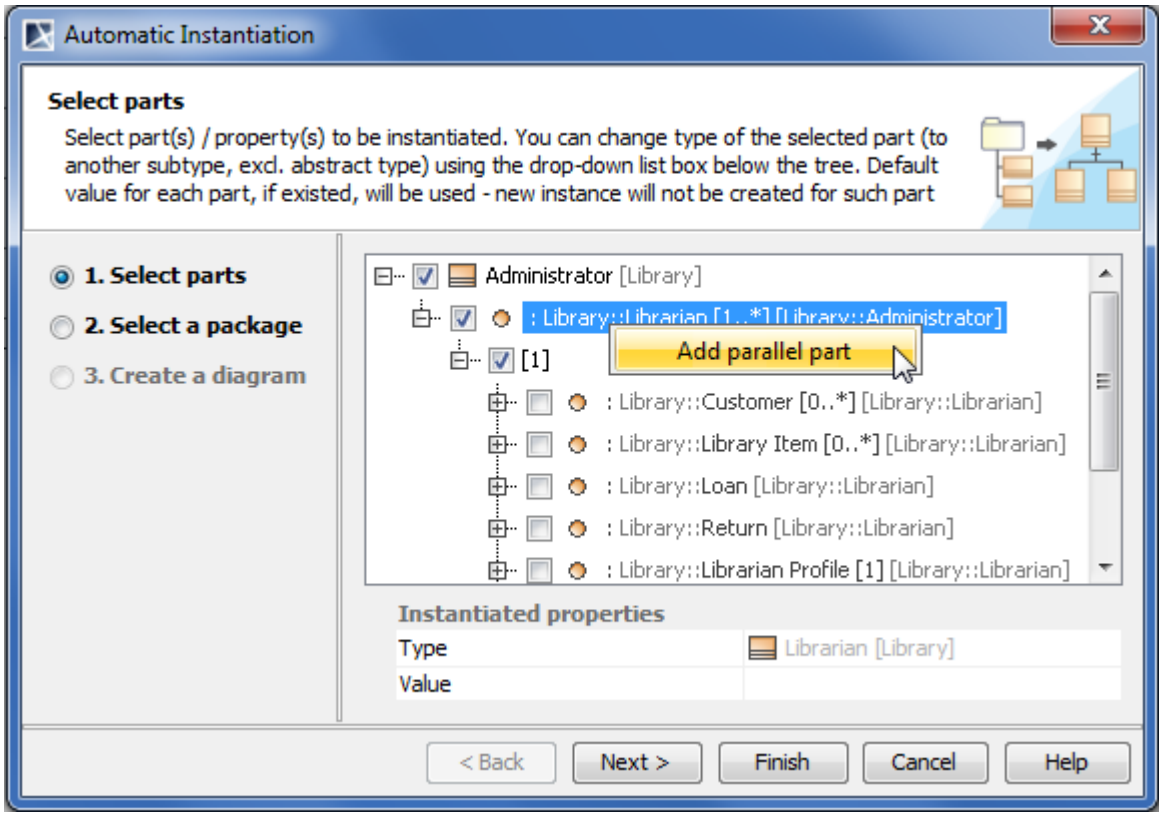


Figure 298 -- Adding parallel part

To remove a parallel part

- Select a part you want to remove and do one of the following:
  - From the shortcut menu, select **Remove parallel part**.
  - Press **Delete** button.

After this step, if you do not wish to change the default package or specify diagram creation settings, click the **Finish** button.

Copyright © 1998-2015 No Magic, Inc.

Step #2. Package selection

In this step, you can select the existing package or create a new one by clicking the **Create** button, or copy the selected package with all its content by clicking the **Clone** button.

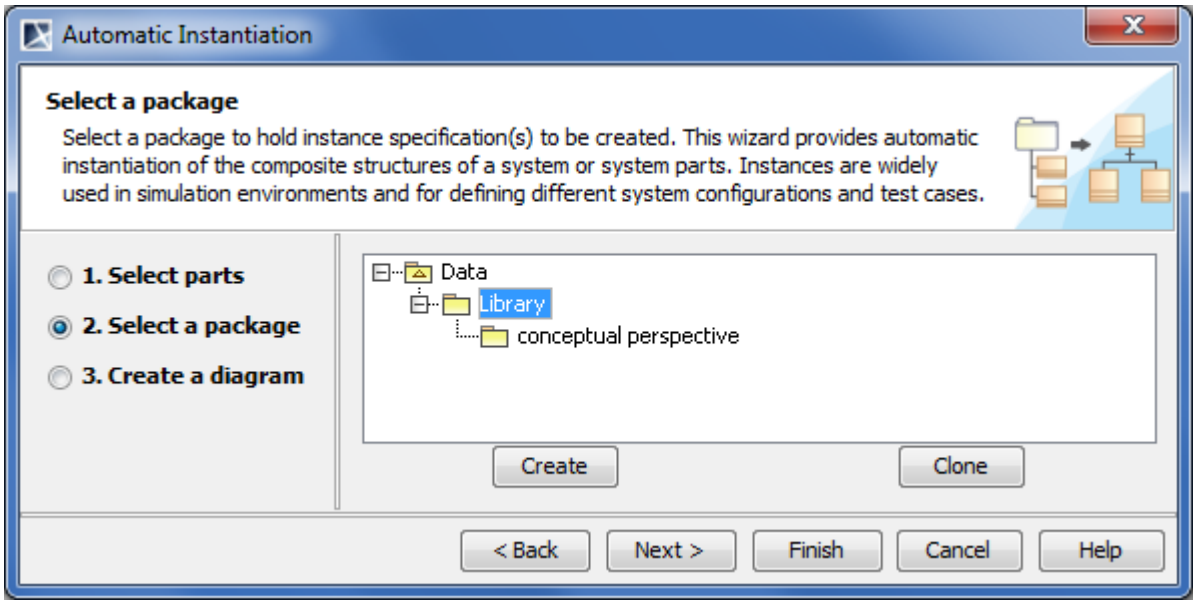


Figure 299 -- Selecting package

After this step, if you do not wish to display your created instance specifications, click the **Finish** button.

Copyright © 1998-2015 No Magic, Inc.

Step #3. Diagram creation (optional)

In this step, you can choose wether you want to show your created instance specifications in a new diagram or not. If you do not want to create a new diagram, you can simply skip this step.

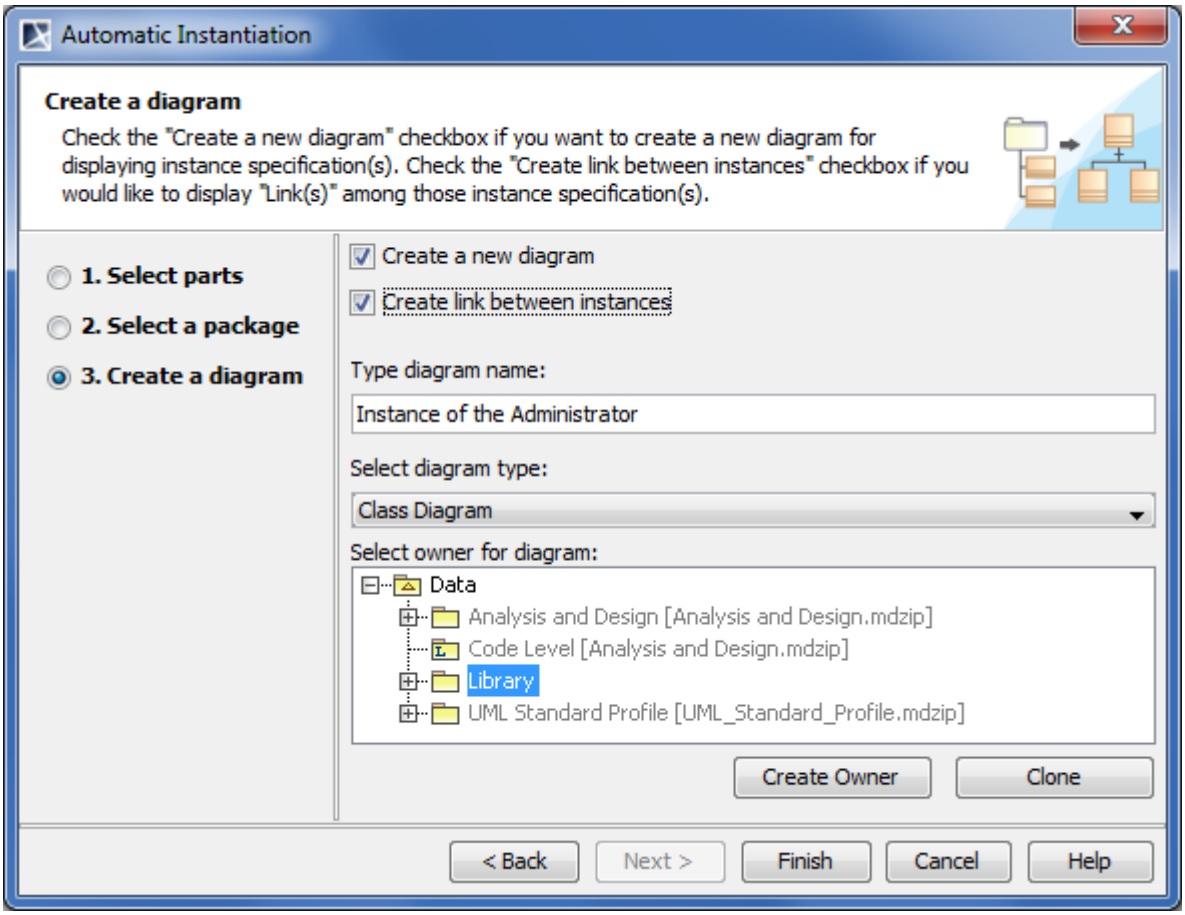


Figure 300 -- Creating new diagram

If you choose to create a new diagram, select the **Create a new diagram** check box. If you want to show links among the created instance specifications, select the **Create link between instances** check box as well.

After that, specify the diagram name. By default the diagram name is *Instance of the <element name>*.

From the **Select diagram type** list, you can select one of the diagram types.

The final step is to select, create new, or clone an owner for your new diagram.

After clicking the **Finish** button, a new diagram with instance specifications is displayed.

Case study:

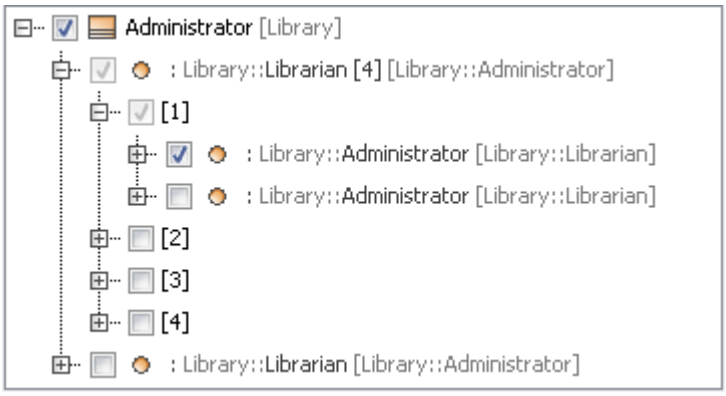
Let's say we have three classes named *Administrator*, *Librarian*, and *Personnel*. These classes are connected in the following way:



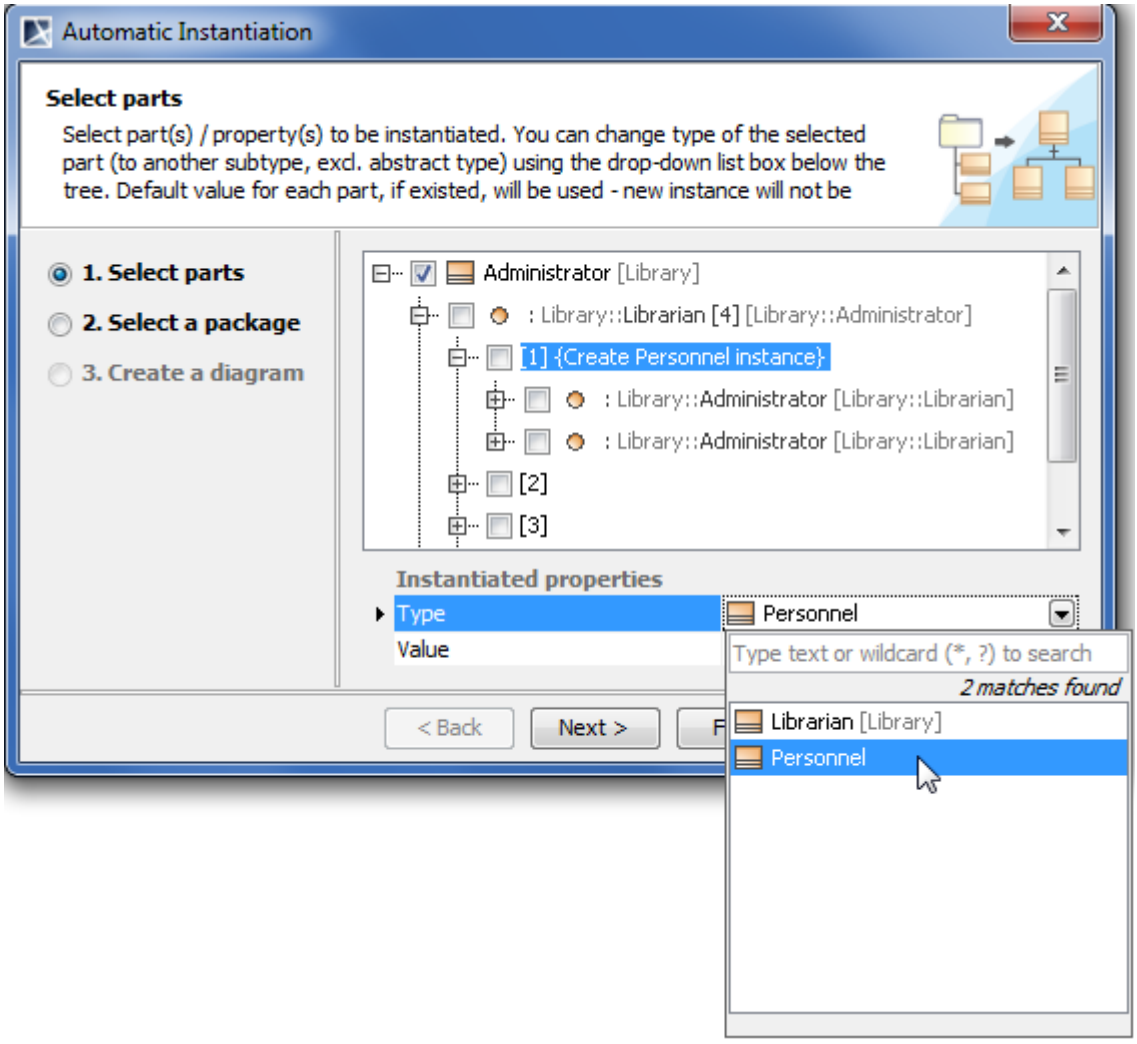
The multiplicity in the association end near the *Librarian* is 4. In this case, the **Automatic Instantiation** wizard suggests four parts. In the following figure, these values are displayed as [1], [2], [3], and [4]. Each of them

Copyright © 1998-2015 No Magic, Inc.

have a set of properties inherited from the type of the parts. In this case, we cannot add more parts because the multiplicity points to the particular number of available parts.

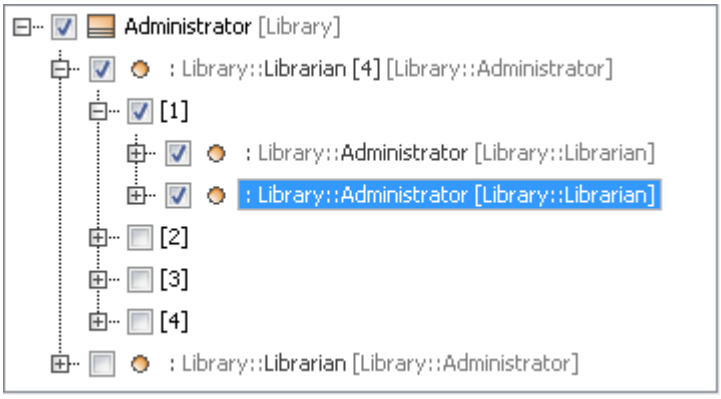


The type of the part can be changed, if there is a sub element in the model. In this case, class *Librarian* has a sub class *Personnel*, so the type of part [1] can be changed to *Personnel* as you can see in the following figure.

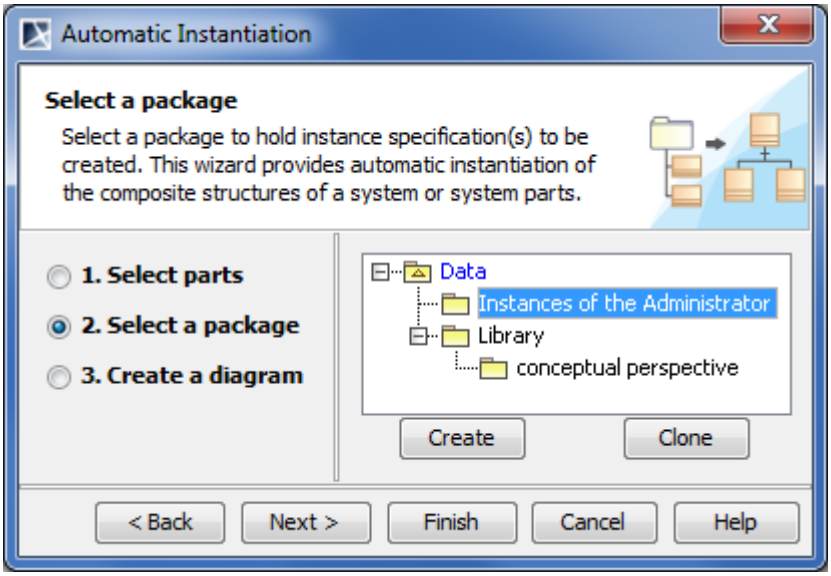


Copyright © 1998-2015 No Magic, Inc.

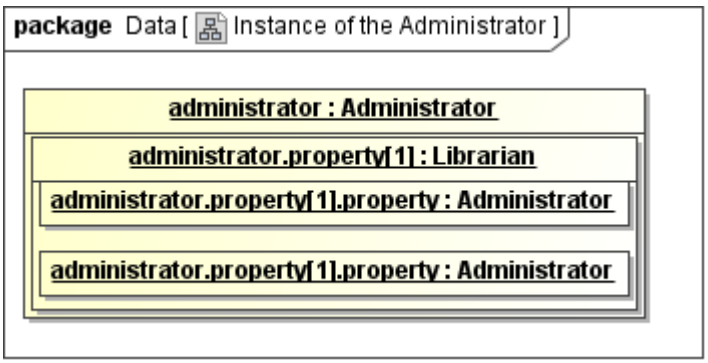
Let's select parts and properties for instantiation:



The next step is to select a new package, wherein our created instance specifications will be stored. For that purpose, we move to the second step of the wizard and create a new package *Instances of the Administrator*.



The final step is to create a new diagram, wherein our created instances will be displayed. After selecting the diagram type (in this case, we will select to create a class diagram) and owner, let's click the **Finish** button. See the result in the following figure.



Related references

- Instance Specification
- Link
- Value Specification

Copyright © 1998-2015 No Magic, Inc.

Resource Manager

MagicDraw Resource Manager functionality allows you to manage local resources (installed with MagicDraw, downloaded) and resources available on the web.

With Resource Manager you can manage different types of resources, such as, Profiles, Plugins, Templates, Language resources, Case studies/examples, Custom diagrams, and others.

The Resource Manager functionality allows:

- finding needed resources and download or update them
- seeing the resource descriptions
- creating your own resources and share them
- checking resource dependency in the **Resource/Plugin Manager** window, the **Required** table



The Resource Manager functionality is included in all MagicDraw editions, except, Reader.

Copyright © 1998-2015 No Magic, Inc.

Resource/Plugin Manager window

To open the Resource/Plugin Manager window

- From the **Help** menu, select **Resource/Plugin Manager**. The Resource/Plugin Manager window appears.

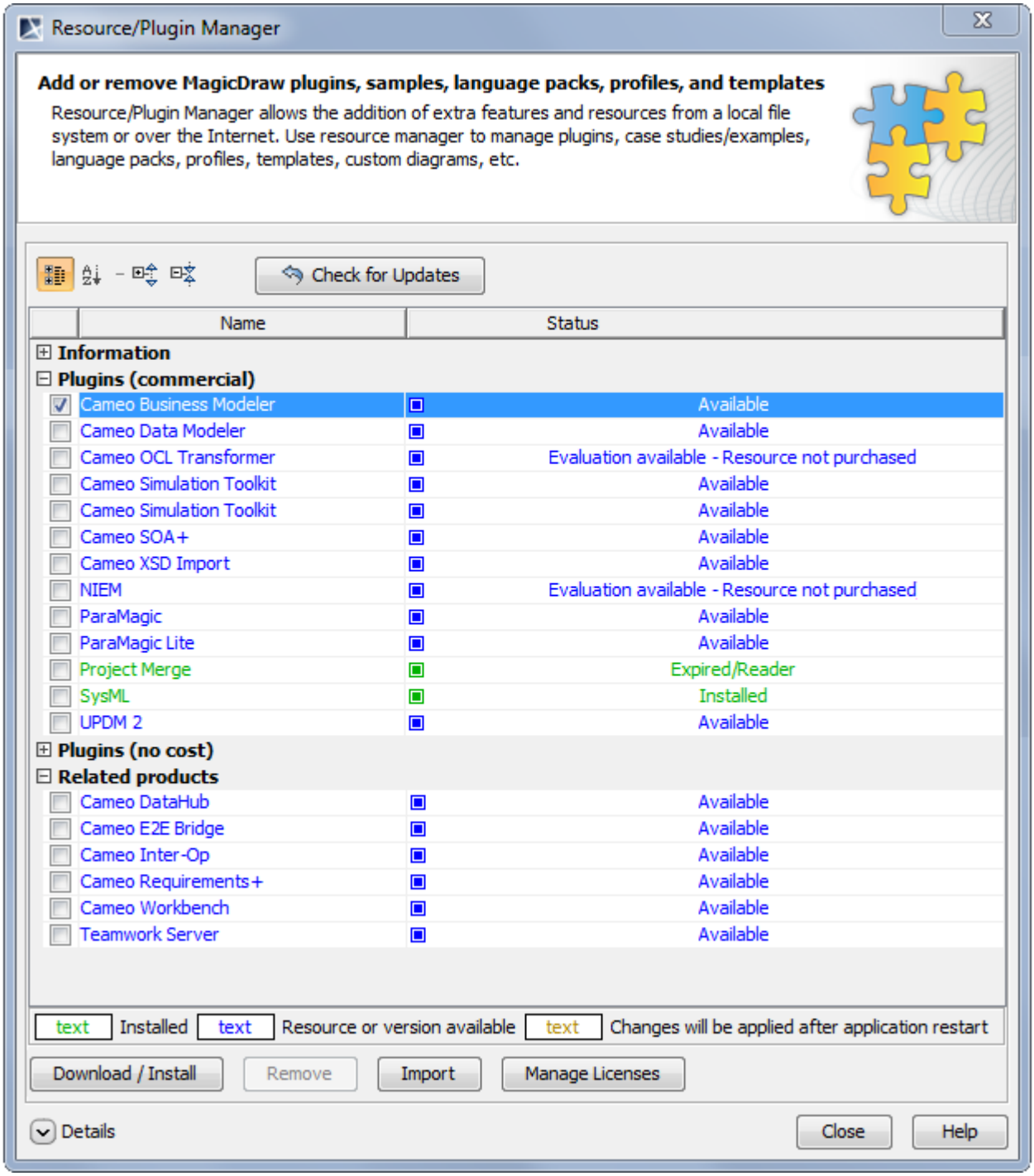


Figure 301 -- Resource/Plugin Manager window

The **Resource/Plugin Manager** window contains the following elements.

Element name	Type	Description
Check for Updates	Button	Click to check for updates.

Copyright © 1998-2015 No Magic, Inc.

Name	Column in the table	Different types of resources are listed in separate nodes. There are the following resources types: model libraries, plugins (commercial), plugins (no-cost), profiles, samples, templates. In front of each resource is a check box. Select this check box to manage the resource.
Status	Column in the table	The resource status can be one of the following: • Evaluation expires <*expiration date*> • Not installed (downloaded) • Not installed (available) • Will be installed after restart • Active • Will be removed after restart • Not downloaded • Removed (if resource does not exist on the web).
Text colors	Explanatory text	The row lists the meanings of the text colors: •  - installed •  - resource or version available •  - changes will be applied after application restart
Download/Install	Button	Click to download or install the selected resources. **NOTE:** The **Download/Install** button is unavailable if no resources are selected.
Remove	Button	Click to remove the selected resource or plugin. Button is active if at least one resource is selected with state “not installed” or “installed”.
Import	Button	Click to import the locally saved resource/plugin. The **Open** dialog opens.
Manage Licenses	Button	Click the button to apply the license for the selected resource or plugin. The **License Manager** window opens. For more information about licensing, see "Licensing Information" on page 44.
Details	Button	Click to expand or collapse the **Resource/Plugin Manager** window. When the Resource/Plugin Manager window is expanded, the following additional fields are displayed: • Name, Resource home page, Provider, and Description. • The Required table (with Name, Required, Status columns). The table represents the requirements for the newer resource/plugin version. **NOTE**: The Required table is displayed when the resource is installed and a newer version is available on the web.

Related external resources

“Distributing Resources” in MagicDraw OpenAPI UserGuide.pdf

Spelling Checker

Spelling Checker enables you to:

Copyright © 1998-2015 No Magic, Inc.

- Check spelling as you type. A shortcut menu provides spelling options. Right click the word underlined in red to enter the shortcut menu. Spelling options will be displayed. Words can also be entered into a customized dictionary using **Add to Dictionary** (see "Spell checking as you type" on page 497).
- Check the spelling of a whole project or of a selected part. You can list all the spelling errors found in a project and correct them easily (see "Spell checking for the whole project or the selected scope" on page 499).
- Set Spelling Checker options. You can set spelling checker options, such as skipping numbers, upper case words in the **Environment Options** dialog box (**Spell Check** option) (see "Setting the spell checking options" on page 503).
- Add a spell checking dictionaries. All "Open Office" supportive spelling languages can be added additionally to the existing ones (see "Spell checking dictionaries" on page 505).

Spell checking as you type

On typing spell checker checks if typed word is correct. In case the word is incorrect it is underlined with red winding line. Right-click on underlined word invokes context menu with suggested editions and capability to add word to dictionary.

Spell checking as you type is performed in the following locations in MagicDraw:

- symbol names on diagram pane,
- for properties in the specification dialog boxes,
- in the documentation pane,
- in the Containment tree,
- In various location on log messages, names, typing and comment fields.

Right-click on underlined word to invoke the shortcut menu in the following MagicDraw window locations:

1. Diagram pane (see Figure 302 on page 497).
2. Containment tree (see Figure 303 on page 498).
3. Dialog boxes (see Figure 304 on page 499).

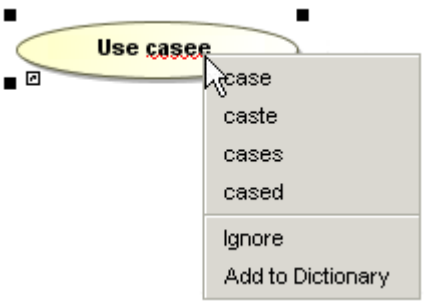


Figure 302 -- Shortcut menu of the spelling error on the diagram pane

Copyright © 1998-2015 No Magic, Inc.

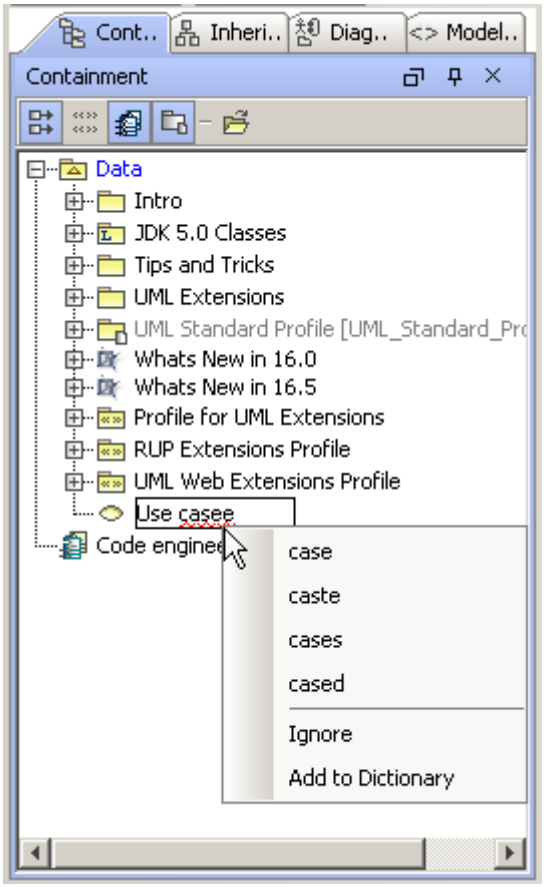


Figure 303 -- Shortcut menu of the spelling error in the Containment tree

Copyright © 1998-2015 No Magic, Inc.

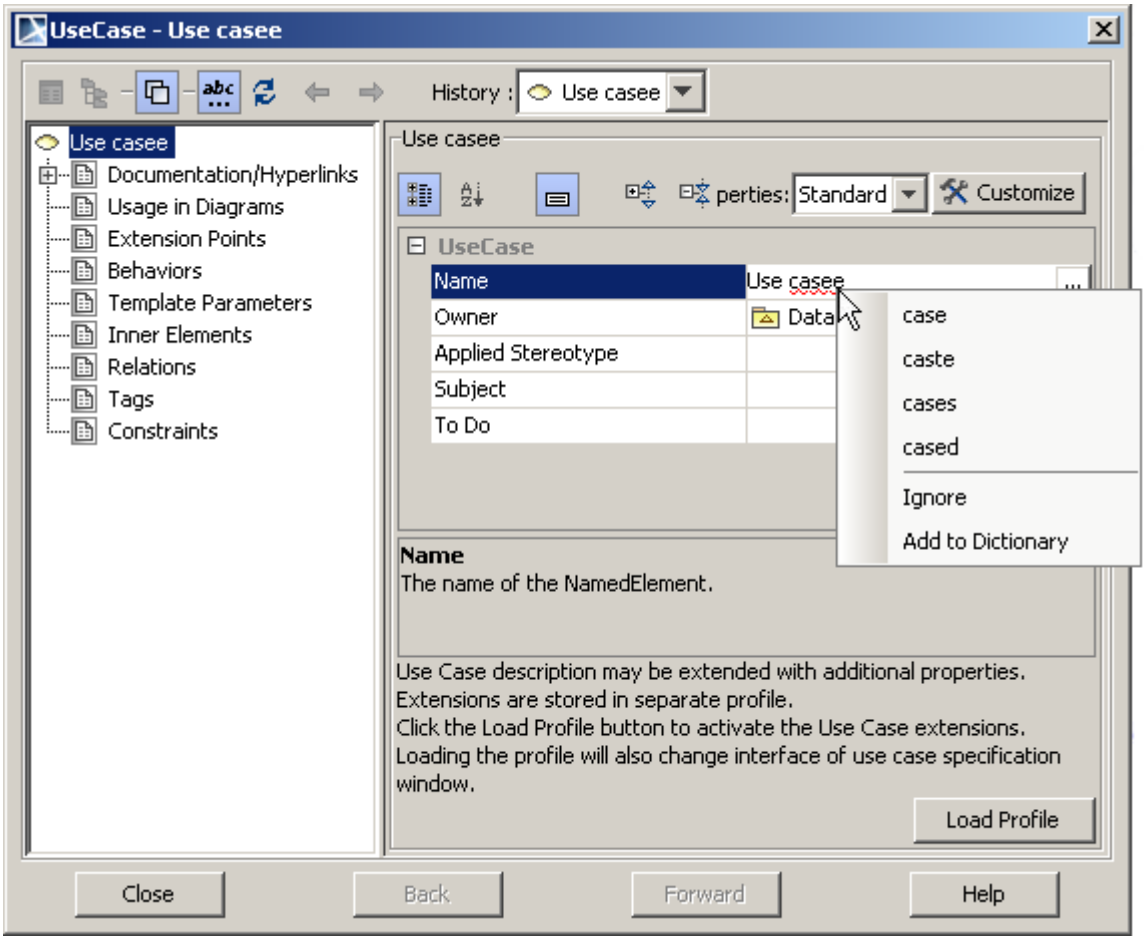


Figure 304 -- Shortcut menu of the spelling error in the dialog box

There are three ways to correct the spelling error:

1. Wrongly typed word can be changed by typing or by selecting provided suggestion that is always correct syntactically, but not always correct semantically.
2. The **Ignore** command. Select the **Ignore** menu item and the wrongly spelled word is treated as correct for this time, but will be discovered as wrongly spelled in the other case.
3. The **Add to dictionary** command. Select the **Add to dictionary** menu item in pop-up menu. By pressing this item the wrongly spelled word will be added to custom dictionary. Next time it will not be treated as wrongly typed.

NOTE

Wrongly spelled words are underlined only in edit mode of particular component. If edit mode has been left, underline disappears and vice versa.

Spell checking for the whole project or the selected scope

Checking spelling for the whole project

1. From the **Tools** main menu, select the **Check Spelling** command or press the **Check Spelling** button in the main toolbar. The **Check Spelling** dialog box opens.

Copyright © 1998-2015 No Magic, Inc.

2. Click the **Check** button. The **Validation Results** dialog box opens.

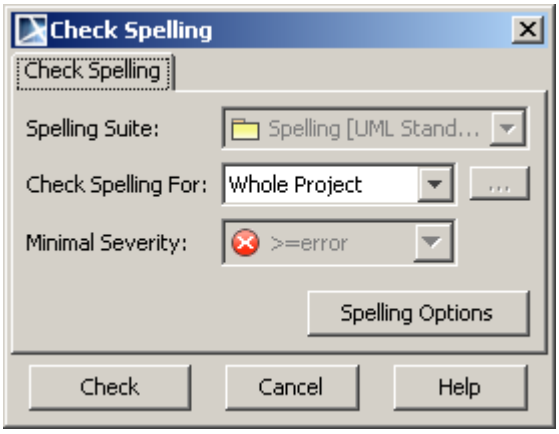


Figure 305 -- The Check Spelling dialog box

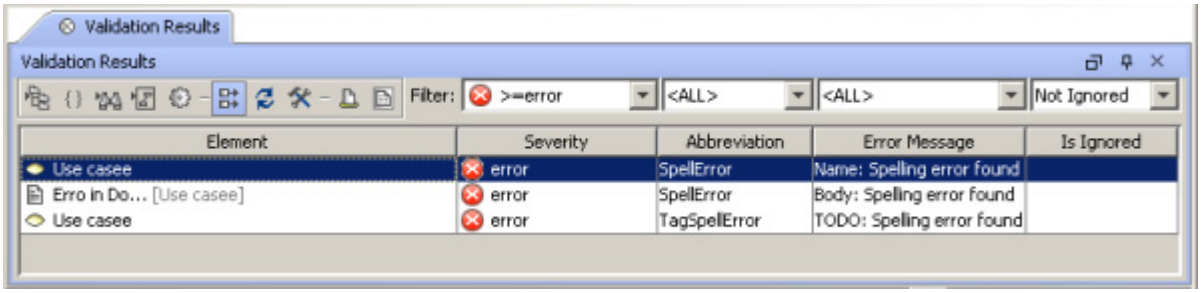


Figure 306 -- The Validation Results dialog box

Checking spelling for the selected scope

1. From the **Tools** main menu, select the **Check Spelling** command. The **Check Spelling** dialog box opens.
2. In the **Check Spelling For** combo box select the **Validation Selection** option.
3. Click the **...** button. In the element Selection dialog, select the scope to check spelling.
4. Click **OK**.
5. In the **Check Spelling** dialog box, click **Check**. The **Validation Results** dialog box opens.

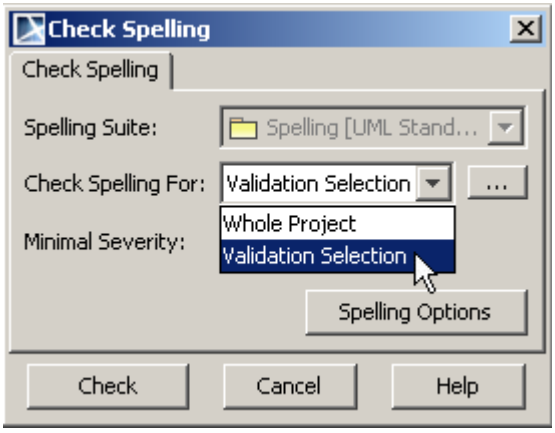


Figure 307 -- The Check Spelling dialog box, the Check Spelling For option

Copyright © 1998-2015 No Magic, Inc.

Analyzing the Check Spelling (the Validation Results window)

The **Validation Results** window provides with all spelling errors. You can choose either to correct particular error or to ignore it.



For more information about correcting the spell checking error, see "Solving the spell checking errors" on page 501.

Spelling error in the **Validation Results** window has description of construct as follows:

Column title	Description
Element	Element with contains spelling error. Spelling error can be found in the element name, or inner properties, such as documentation, the To Do property and others.
Error Message	<Element property name>: <Spelling error found>

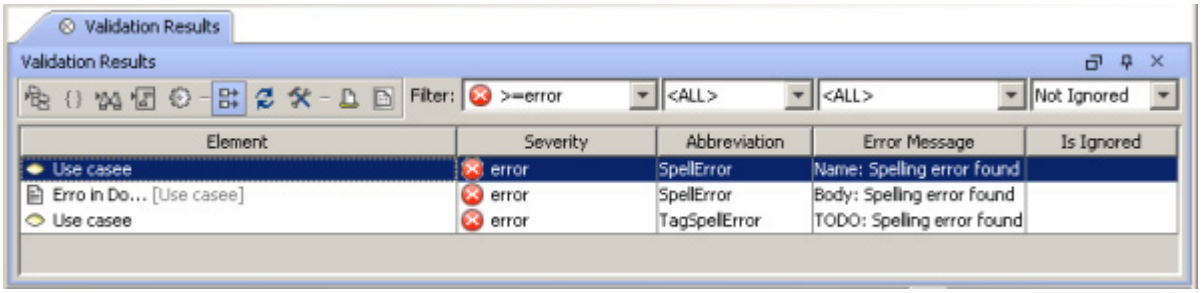


Figure 308 -- The Validation Results window

For more information about validation functionality, see “Validation” on page 612. Here the **Validation Results** window is described in more details.

Solving the spell checking errors

To solve the error:

1. In the **Validation Results** window select the error and with right click invoke its shortcut menu.
2. In the Containment Tree select the element with error (marked with red circle and white cross inside) and with right-click invoke its shortcut menu.
3. On the diagram pane, select the element with error (highlighted with red border) and with right-click invoke its shortcut menu.

In the shortcut menu of element with error, select the **Correct** command to invoke the **Spell Checker** dialog box.

In the **Spell Checker** dialog box the **Element** property shows the name of the element. The **Property** shows the name of the element's property that has spelling error in its value. In the **Value** property all wrongly spelled words are underlined.

Press the **Next** button to go to the next spelling error found during validation. To close spelling dialog and save changes press **OK**. To close dialog without saving changes press **Cancel**.

Copyright © 1998-2015 No Magic, Inc.

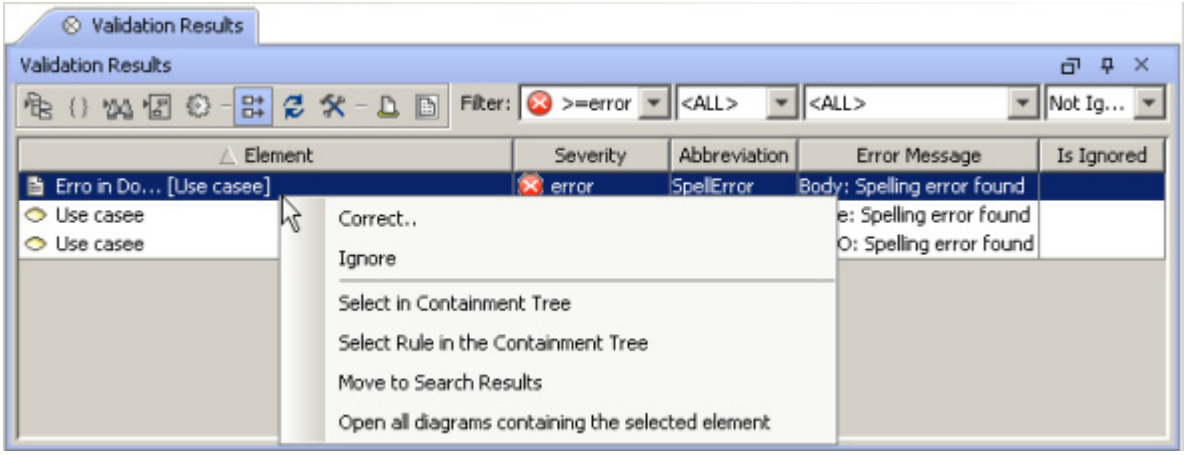


Figure 309 -- Solving the spell checking errors in the Validation Results window

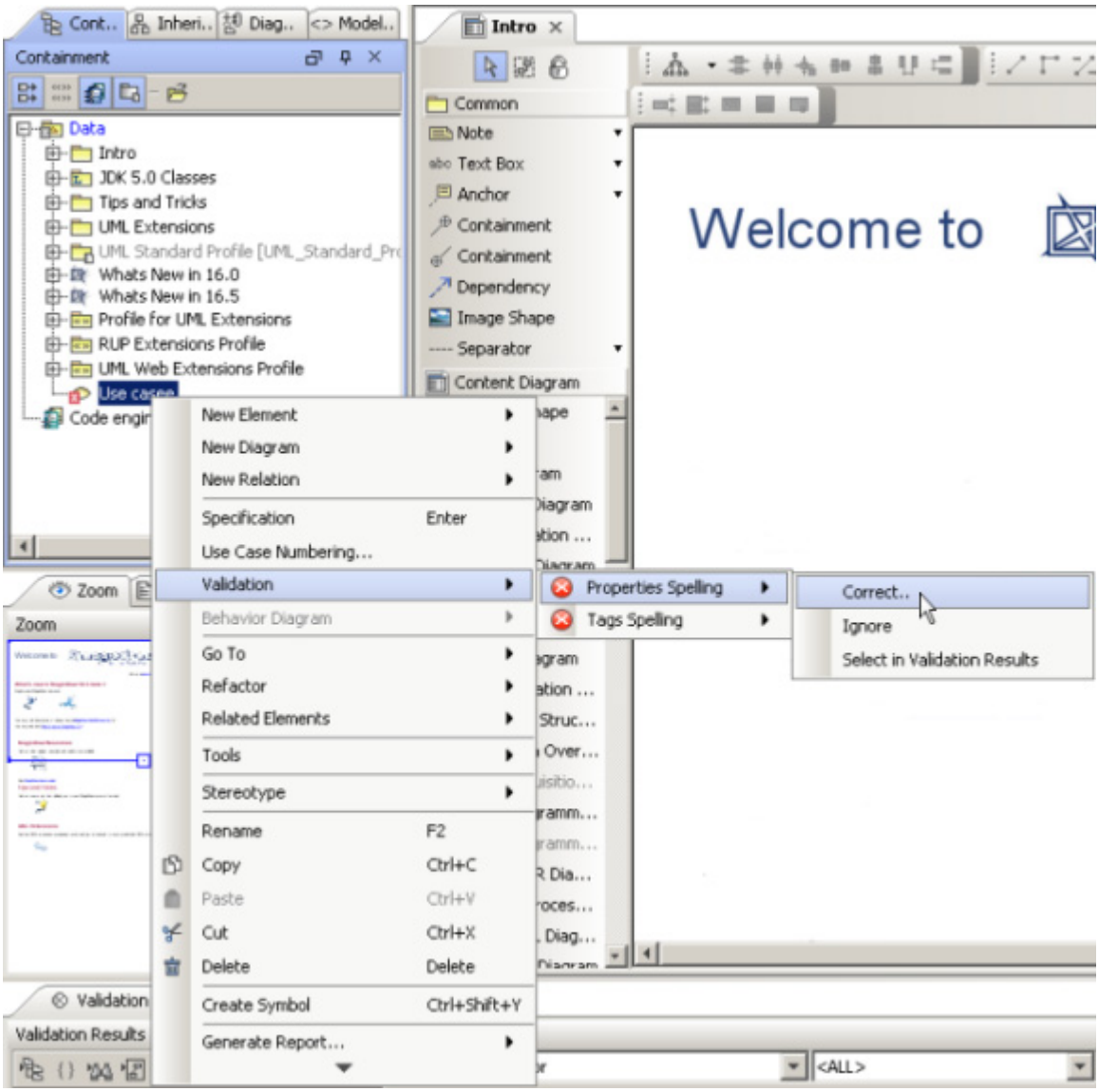


Figure 310 -- Solving spell checking errors in the Containment tree

Copyright © 1998-2015 No Magic, Inc.

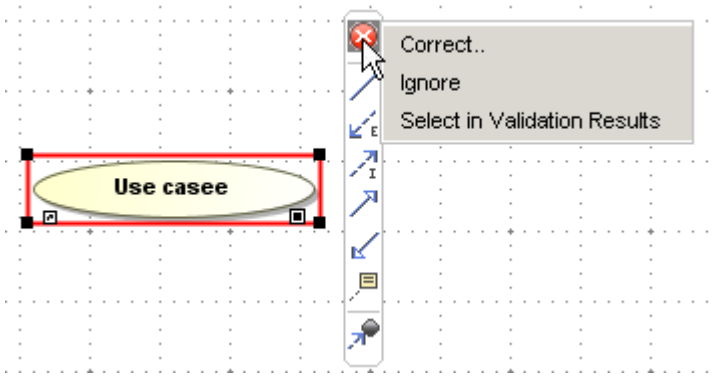


Figure 311 -- Solving spell checking errors in the diagram pane

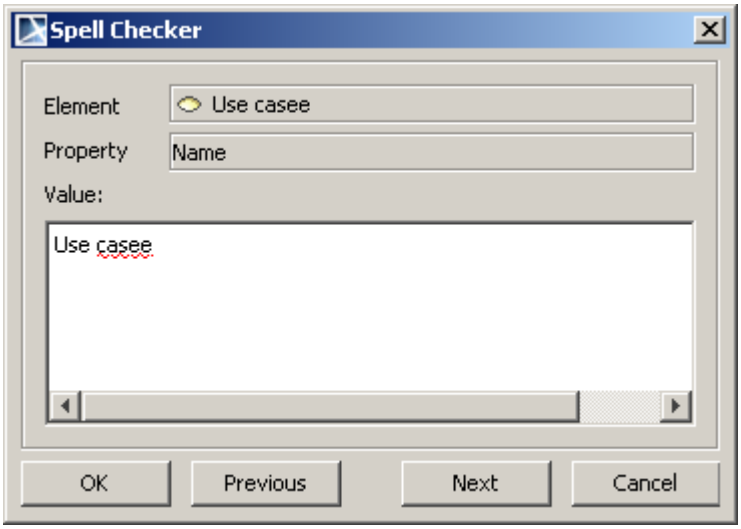


Figure 312 -- The Spell Checker dialog box

Setting the spell checking options

1. From the **Options** main menu, select **Environment**. The **Environment Options** dialog box opens.
2. Select the **Spelling** branch. Define the spelling options.

Copyright © 1998-2015 No Magic, Inc.

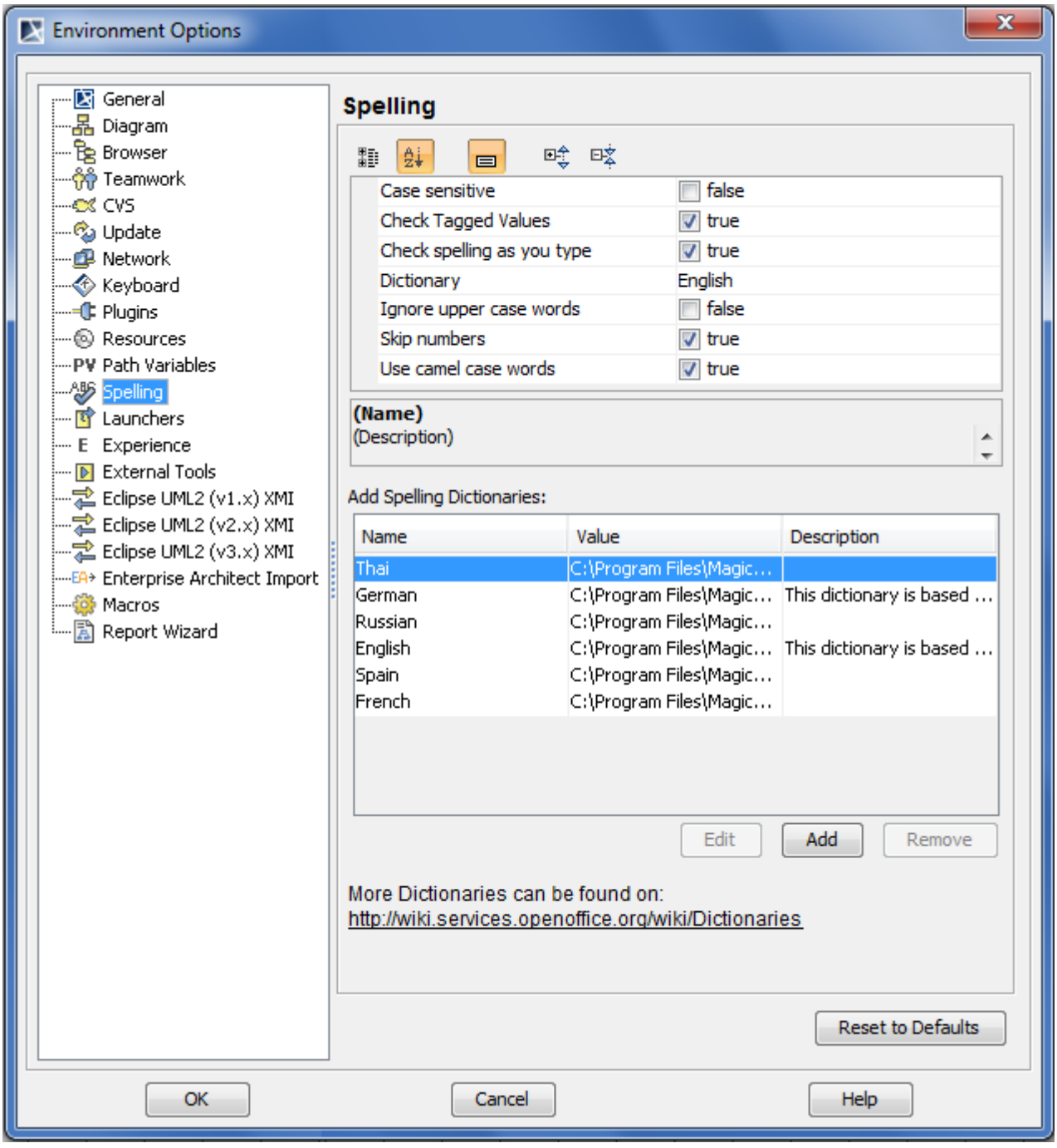


Figure 313 -- The Environment Options dialog box, the Spelling branch

Spell checking options

See the spelling options in the table below.

Property name	Function
Check Spelling as you type	Underlines wrongly spelled words and provide the suggestions list of possible corrections in context menu. For more information about spell check on type, see "Spell checking as you type" on page 497.

Copyright © 1998-2015 No Magic, Inc.

Check Tagged Values	Checks all tagged values that is Type of string.
Dictionary	Select language for spelling. All "Open Office" supportive spelling languages can be added additionally to the existing ones. For more information about spelling dictionaries, see "Spell checking dictionaries" on page 505.
Case Sensitive	If true, words will differ in meaning based on differing use of uppercase and lowercase letters.
Use camel case words	If true, compound words or phrases in which the words are joined without spaces and are capitalized within the compound-as in BackColor, will be spelled as separate words.
Ignore upper case words	If true, words with uppercase words only are not to be spell checked.
Skip numbers	If true, numbers are not to be spell checked.

Spell checking dictionaries

All Apache OpenOffice supportive spelling languages are available.

To import spelling dictionaries:

1. Click the **Add** button in the **Environment Options** dialog box > the **Spelling** branch > the **Add Spelling Dictionaries** group. The **Dictionary** dialog box opens.
2. Type the name of a new spelling dictionary in the **Name** text box.
3. Click the **...** button and select the OpenOffice zip file location.
4. Type the description of a new spelling dictionary in the **Description** text box.

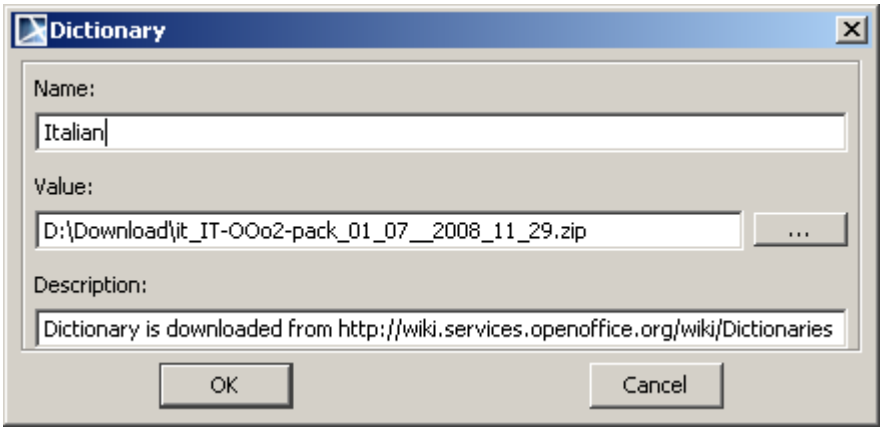


Figure 314 -- The Dictionary dialog box

NOTE More dictionaries can be found at http://wiki.services.openoffice.org/wiki/Dictionaries

Defining properties of the customized element to be spell checked

DSL customization classes and their properties can be checked either. You can choose what properties of the customized element (Class with «Customization» stereotype) you want to be spelling checked.

Copyright © 1998-2015 No Magic, Inc.

Defining customization class tag "checkSpelling" value can do it. "checkSpelling" tag can be found in properties tag group. By creating value for this tag you can choose String properties to check spelling for. By default there are no properties marked as checkable.



For more information about DSL, see MagicDraw UMLProfiling&DSL UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

8 MODEL ANALYSIS

MagicDraw provides the following capabilities for your model analysis:

- "Displaying Related Elements" on page 508 - displays paths among shapes that have already been created in the model data, use the quick and simple Display related elements functionality.
- "Analyzing Usages and Dependencies" on page 509 - the feature allows you to track and view element dependencies in UML models, explore how model elements are used by other elements, and understand the relationships between used and dependent elements.
- "Traceability" on page 514 - allows for tracking, visualizing, navigating, and analyzing the elements involved in traceability relationships.
- "Derived Properties" on page 537 - you can create derived properties in the generic table and in the element's Specification window.
- "Relation Map" on page 544 - the Relation Map diagram allows you to rapidly review and analyze relationships between the elements of the model.
- "Specifying Criteria for Querying Model" on page 551 - allows for defining the criteria that can be used to associate row and column elements in a dependency matrix, represent relations between elements in a relation map, gather the contents of a smart package, or calculate the value of a derived property.
- "Symbol Usage in Diagrams" on page 576 - displaying the list of diagrams in which symbol of current element is represented.
- "Projects Comparison" on page 579 - compares two local projects or two versions of the same project, as well as two diagrams.
- "Metrics" on page 587 - allows measuring a project by different viewpoints.
- "Analyzing Package Dependencies" on page 604 - checks and analyzes package dependencies of the whole project or when exporting and sharing packages.
- "Validation" on page 612 - a facility for evaluating completeness and correctness of the models created by the user.
- "Active Validation" on page 632 - instantly checks model for correctness and completeness, displays errors in the model and suggests solutions.
- "Model Visualizer":
    - "Class Diagram Wizard" on page 642 - helps to create and customize new class diagrams.
    - "Package Overview Diagram Wizard" on page 649 - generates the package dependency diagram for packages in your project.
    - "Package Dependency Diagram Wizard" on page 644- generates diagrams containing packages (created within a project) and shows the relationships between them.
    - "Hierarchy Diagram Wizard" on page 653 and "Realization Diagram Wizard" on page 656 - prepares diagrams and report documents of the relationships between classes in the UML model.
    - "Activity Decomposition Hierarchy Wizard" on page 659 - converts activity into class and/or SysML Block Definition Diagram.
    - "Content Diagram Wizard" on page 662 - generates content of diagrams that are used in the project.

Copyright © 1998-2015 No Magic, Inc.

- "Sequence Diagram from Java Source Wizard" on page 665 - creates a sequence diagram of Java method implementation.

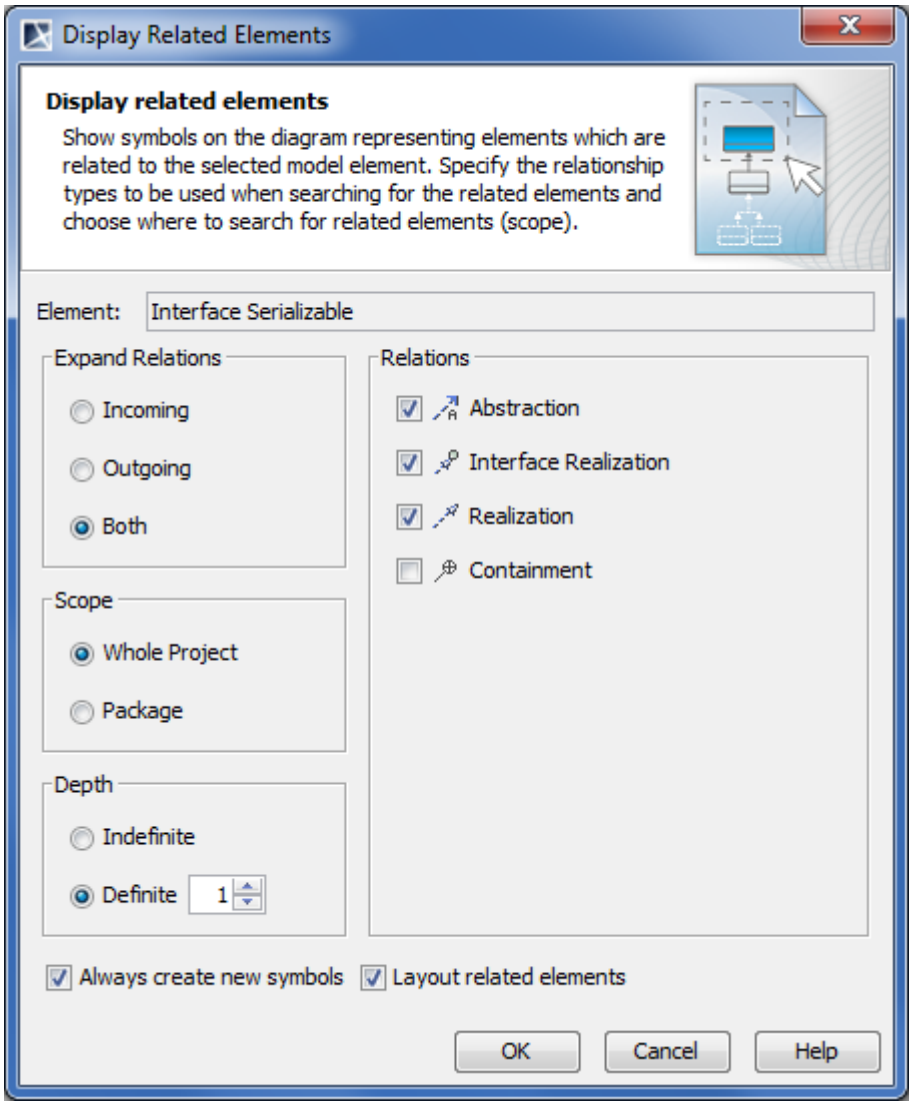
Displaying Related Elements

On a diagram pane, you can display shapes of elements related to any selected element.

To display shapes of related elements

1. Select an element shape on diagram pane.
2. Do one of the following:
   - Right-click the shape and from the shortcut menu, select **Related Elements** > **Display Related Elements**.
   - From the **Edit** menu, select **Symbol** > **Related Elements** > **Display Related Elements**.

The **Display Related Elements** dialog opens.



Copyright © 1998-2015 No Magic, Inc.

3. On the right side of the dialog, choose relationship types that you need to use when searching for the related elements by selecting the check box in front of each desired relationship type.
4. On the left side of the dialog, choose the search scope and specify other options for displaying the related elements. The options are described in the following table.

Option name	Description
Expand Relations	Select what kind of relationships should be displayed on the diagram: • **Incoming**, in which the element participates as client • **Outgoing**, in which the element participates as supplier • Both
Scope	Select in what scope the elements related to the selected model element will be found: • Whole Project • Package
Depth	Specify the range for searching for selected relationships: • **Indefinite**. All possible relationships are involved. • **Definite**. Specify the level of hierarchy for the relationships involved.
Relations	Select the relationships you want to display on the diagram.
Always create new symbols	Select to create a symbol even if a related element already has a symbol in the diagram.
Layout related elements	Select to layout new symbols by using the default diagram layout. Note that position of the selected element can be changed a little bit.

To display paths among shapes that already exist in the model

- From the shape shortcut menu, select **Related Elements** > **Display Paths**.
- Select a symbol and from the **Edit** menu, select **Symbol** > **Related Elements** > **Display Paths**.

Analyzing Usages and Dependencies



This functionality is available in Standard, Professional, Architect, and Enterprise editions.

The Used By and Depends On features allows you to track and view element dependencies in UML models, explore how model elements are used by other elements, and understand the relationships between used and dependent elements.

It is also useful for analyzing associations between elements or searching for diagrams where these elements are represented.

Copyright © 1998-2015 No Magic, Inc.

Understanding Used By

If you want to find all the elements that reference the current element, use the Used By functionality.



If element1 references element2, this means that element1 uses element2. Conversely, element2 is also used by element1. In the **Attribute Specification** dialog, add class2 in the **Type** box. This means that class2 is used by that attribute.

Containing other elements is not considered usage. For example, when a package contains an inner element class, this does not mean that the package uses the class. The class is categorized as only a container of the package.

Understanding Depends On

If you want to find all elements that current element depends on, use the Depends On functionality.



If element1 contains a reference to element2, this means that element1 depends on element2.

Searching for Usages / Dependent Elements

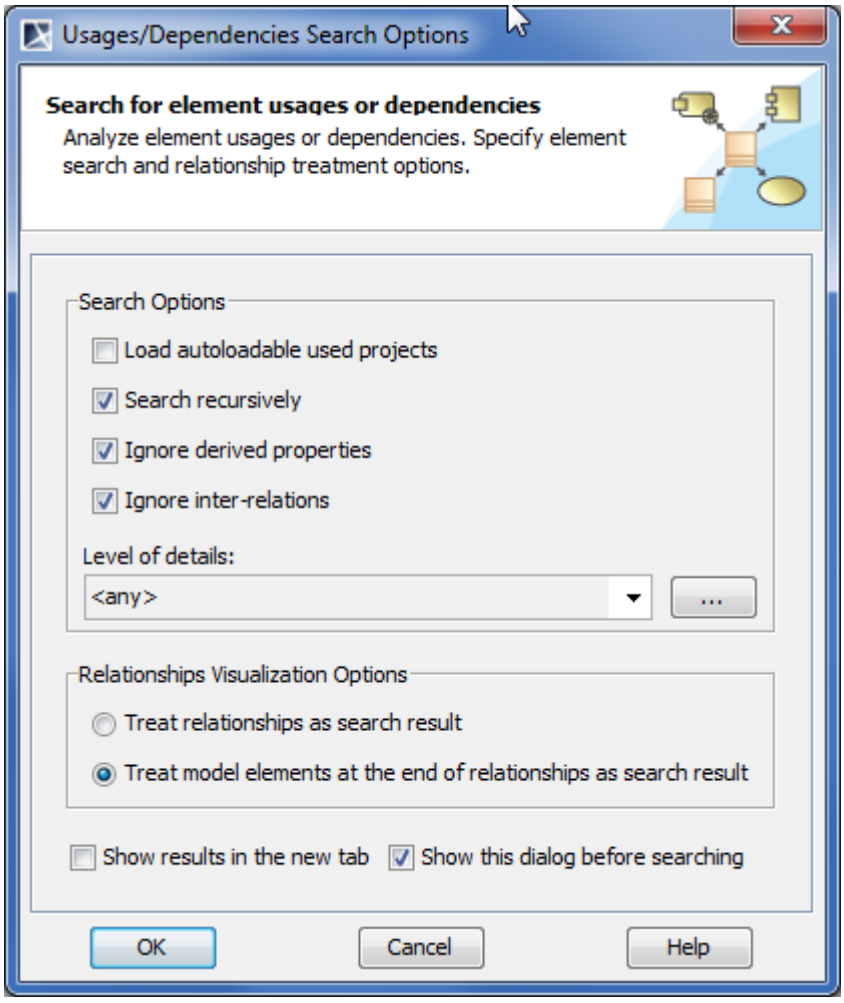
To search for usages / dependent elements

1. Select an element in the Model Browser or on the Diagram pane.
2. From the element shortcut menu, select the **Related Elements** command and then select either the **Used By** or **Depends On** command.
3. The **Usages/Dependencies Search Options** dialog appears.

The **Usages/Dependencies Search Options** dialog covers the spectrum of usages and dependent element functionality. This means that if you clear or select any check boxes in the

Copyright © 1998-2015 No Magic, Inc.

Usages/Dependencies Search Options dialog, the next time you search for dependencies, the values for these check boxes remain the same.



Element	Description
Load autoloadable used projects	If the model has unloaded used projects, select this check box to load all elements to be included in performing the usages/dependencies search. **NOTE:** This element is available if the **Used By** command was selected.
Search recursively	If selected, usages/dependencies of inner elements (beneath the level of the current element) are listed in the search result list. If unselected (default value), the usages/dependencies table lists these elements that are using the current element. For example, let's say, element1 contains element2. When you search non-recursively, only elements that use element1 are listed. When you search recursively, elements using element1 are listed, while the other branch lists these elements that use element2.
Ignore derived properties	If selected, derived properties of the element are not included when searching for elements usages/dependencies. A derived property is the one that is automatically calculated from the other properties. **NOTE:** It is strongly recommended to keep this option selected. Otherwise all derived properties will be included into the usages/dependencies searching scope, and this may cause a severe performance downgrade when searching for usages/dependencies.

Copyright © 1998-2015 No Magic, Inc.

Element	Description
Ignore inter-relations	If unselected, only usages/dependencies outside the current element (above the level of current element and in the same level) are listed. You may use this option when you export the package as new project. You may analyze the dependencies of a package inner element to its outer elements (note that you cannot export a package that contains inner elements with dependencies to the outer elements according to this package). For example, let's create package p1 and package p2. Connect these packages using the dependency relationship. In the Model Browser, drag this dependency to p1. In the **Usages/Dependencies Search Options** dialog, select the **Ignore inter-relations** check box. Search for the p1 dependency. The dependency to the dependency relationship will not be found because this element is a child of p1.
Level of details	Select predefined types (Classifiers, Packages) from the combo box or click the **...** button, which opens **Select Element/Symbol Type** dialog and allows the selection of custom types. Only elements of these types will be displayed in the search results. For example, *Package* selected as *"*level of details*"*. Class B depends on Class A (owned in package p2), because it uses A as type of one operation parameter. In this case package p2 will be displayed as a result, class A will be added under p2 node.
Treat relationships as search result	If selected, search results provide a complete list of used or dependent relationships together with other results.
Treat model elements at the end of relationships as search results	If selected, relationships are skipped and only usages/dependent elements between the model elements connected with these relationships are displayed.
Show results in the new tab	If selected, a new Element Usages/Dependencies window is displayed for every new search.
Show this dialog before searching	If selected, the **Usages/Dependencies Search Options** dialog will open before searching for element usages or dependencies next time. If unselected, the dialog will not open before searching for element usages or dependencies next time. **TIP!** You can open the **Usages/Dependencies Search Options** dialog by clicking the Usages/Dependencies Search Options button on the toolbar in the Element Usages/Dependencies window.

4. Depending on which command you selected on the shortcut menu, either the Element Usages or Dependencies window opens.

   The Elements Using/Dependencies window lists the results of the usages/dependencies. In the Elements Using/Dependencies window you can analyze results, search for an element location (in a diagram, for example, or in a browser), and filter results.

   Because the Usages/Dependent Elements Results window is not synchronized with the model,

Copyright © 1998-2015 No Magic, Inc.

any changes made to the model elements will not show in the results window until you click Refresh.

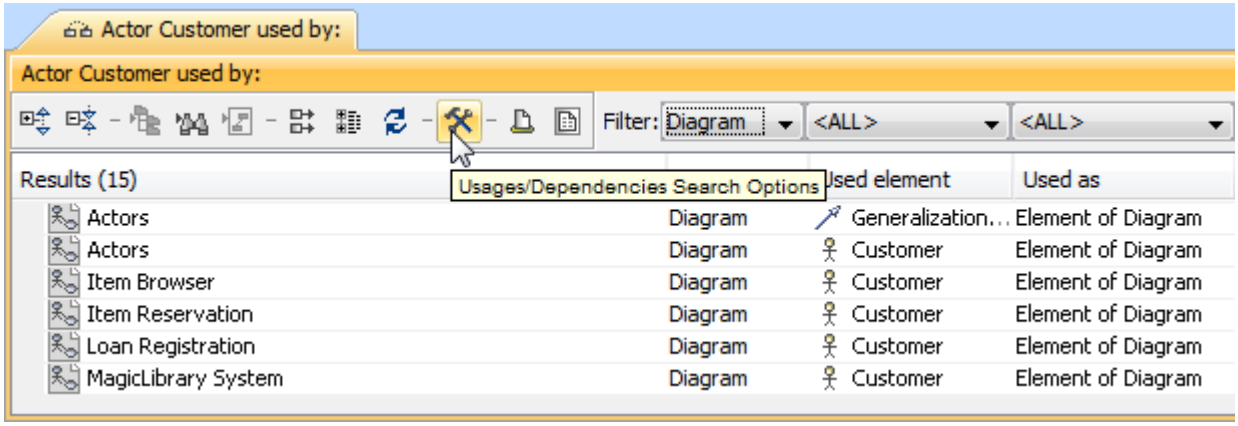


Figure 315 -- Fragment of Actor Customer used by window

Element	Description
Expand	Expands records listed in groups. Click the plus sign next to the group name to display the contents.
Collapse	Collapses records listed in groups.
Select in Containment Tree	In the usages/dependencies table, select the record. Click the **Select in Containment Tree** button. The Element is selected in the Browser, and the **Containment tree,** and the symbol of this element is selected in the diagram pane. Click the **Select in Containment Tree** button to open any closed and previously loaded diagrams. You can also select the element in the Containment Tree by double clicking it. **NOTE:** The **Select in Containment Tree** button is not available for multiple element selections.
Move to Search Results	In the usages/dependencies table, select the record. Click the **Move to Search Results** button. In the usages/dependencies table results are moved to the Browser, Search Results tree. In the Search Results tree, you will see all the results in one list and organized in two groups: From diagram and From model.
Open all diagrams that contain current Usages/ Dependencies	Diagrams, which are referenced in the usages/dependencies table, are opened. In the open diagrams, the view is focused on used/dependent elements. **NOTE:** The **Open all diagrams that contain current Usages/ Dependencies** button is inactive when there are no elements that are used in diagrams.
Show/hide the Full Path Names	The full path is displayed next to the element name. **NOTE:** For a symbol this button is not valid.
Refresh	The usages/dependencies table should be refreshed after: • Elements or symbols are deleted. • New dependencies/usages are created for the particular model element.
Show Usages/Dependencies Search Options	The **Find Usages/Dependent Elements Options** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

Element	Description
Filter	**Filter** boxes in the **Elements used by and depends on** windows are placed above the column, which will be filtered. In the **used by** window, you can filter by element type or usage type. In the **depends on** window, you can filter according to element type and dependency type. When you expand the **Filter** box, you will see items listed in the usages/dependencies columns. **NOTE:** In the **Element Type Filter** box within the **used by** window, combo box, additional **Show All but Symbols** filtering options is added.

Traceability



Traceability



The traceability solution is available in Architect and Enterprise editions.



You can also read about the traceability feature and analyze the examples in the Traceability sample.

To open the sample, do any of the following:

- On the Welcome screen, select **Samples** > **Product Features** > **Traceability**.
- Go to the folder *<MagicDraw installation directory>\samples\product features* and open the *Traceability.mdzip* file.

All the examples, given in this section, are based on the data from this sample.

As of version 16.8, MagicDraw supports the traceability feature that allows you to track, visualize, navigate, and analyze the elements involved in the traceability relations.

The traceability relations are used to relate the elements representing the same concept in different UML models at different levels of abstraction or from different viewpoints. Different levels of abstraction may contain, include, or even correspond to different stages of system development process (starting from requirements analysis and finishing with implementation). The higher level of abstraction (e.g., requirements analysis), contains models with specifying elements, and the lower level of abstraction (e.g., implementation) includes models with realizing ones.

The traceability relations help to determine how your requirements or the other model artifacts are satisfied. As they may change, you can use traceability relations to monitor the impact of these changes.

Multiple types of custom and extended UML relationships (e.g., realization, abstraction), tags (e.g., **Alternative Flow of Events Diagrams**), and properties (e.g., **Owned Behavior**) are used to represent traceability relations between the specifying and realizing elements for showing traceability from requirements analysis to implementation and deployment models.

Relation Map and Dependency Matrix

You can visualize the traceability relations of your project in order to analyze them using the MagicDraw features, such as Relation Map (for the analysis of traces among multiple levels of abstraction) and Dependency Matrix (for the analysis of traces between any two levels of abstraction).

Element's Specification window, Properties panel, Go To, and Notes

You may track and navigate the elements, that are directly or indirectly related to a specific element through the traceability properties that will be represented in a special property group of the element's Specification window

Copyright © 1998-2015 No Magic, Inc.

and **Properties** panel, or on the element's shortcut menu under the **Go To** menu. The traceability properties can also be visualized on a diagram using the standard MagicDraw mechanism for displaying property values in notes.

Traceability Report

A Traceability Report is particularly useful when there is a need to visualize and verify that requirement analysis, design, and implementation model elements are all covered in higher or lower levels of abstraction, for example, all use cases should be covered with design classes realizing them.

For more detailed information about the MagicDraw traceability solution, read the following subsections:

1. "Creating Traceability Relations" on page 515.
2. "Traceability Relations Representation" on page 516.
3. "Navigating between Different Levels of Abstraction" on page 522.
4. "Analyzing Traceability Relations" on page 523.
5. "Predefined Traceability Rules" on page 525.

Creating Traceability Relations

Multiple relationships can be used for traceability representation, depending on the element type and method used.

The traceability relations can be represented by the following types of element dependencies:

- UML relationships (such as abstraction, realization, derivation)
- UML properties (such as **Owned Behavior**)
- UML tags (such as **Alternative Flow of Events Diagrams**)

The traceability relations can be **single level** and **multilevel**.

- When elements are related directly by using a custom UML relationship, property, or tag, the traceability relation is considered to be a single level relation.
- When elements are related indirectly by usually using multiple types of the above mentioned dependencies, the traceability relation is considered to be a multilevel relation.

The traceability relations can be **specifying**, **realizing**, or **other**. This depends on which direction a relation is analyzed and which element is considered as a basis.

- A relationship between a specific element and an element that is the realization of this specific element, from the point of view of the specific element, is considered to be **realizing traceability** (for forward traceability).
- A relationship between a specific element and an element that is the specification of this specific element, from the point of view of the specific element, is considered to be **specifying traceability** (for backward traceability).

Copyright © 1998-2015 No Magic, Inc.

- All other traceability relations.

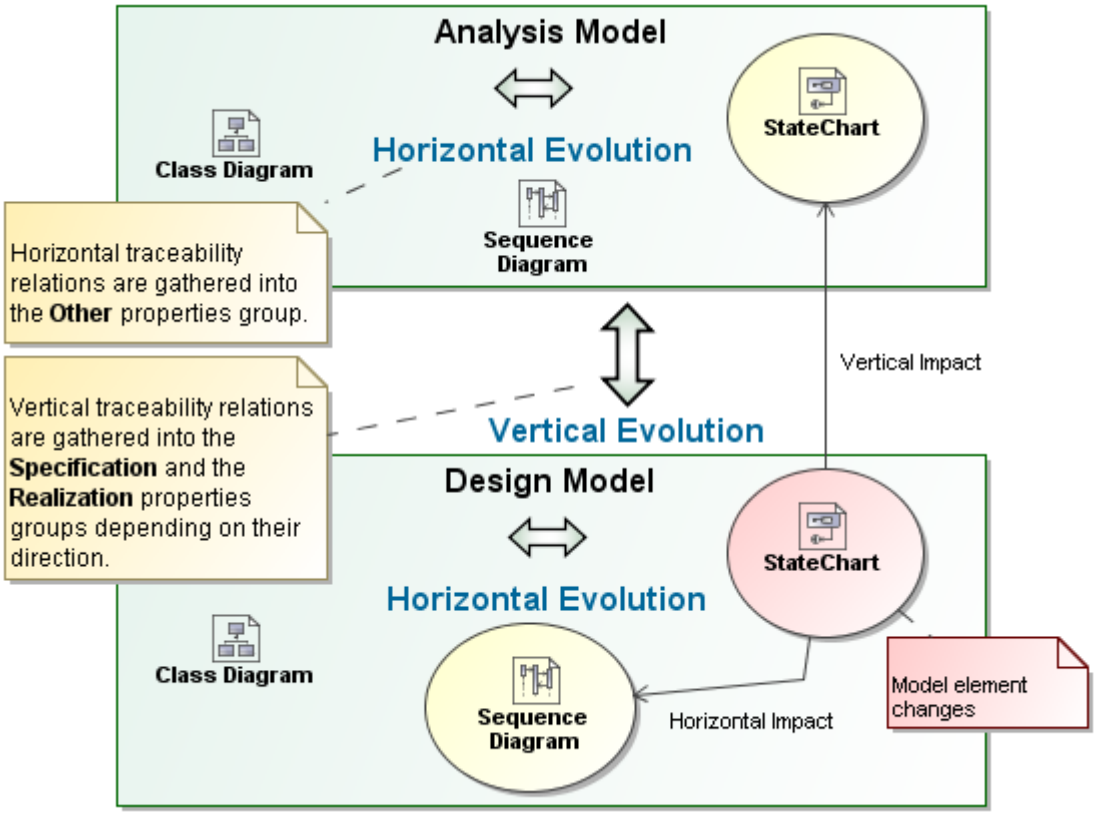


Figure 316 -- Horizontal and vertical traceability representation through the realizing, specifying, and other traceability relations

TIP!

For more information about the traceability feature, see "Traceability" on page 514.

Traceability Relations Representation

The traceability relations are represented through the so-called traceability properties that have been predefined for each element type according to the traceability method supported in MagicDraw.

A single traceability property shows an element or a set of elements that are related to a particular element through the relationships which are specified by some traceability rule. The predefined traceability rules are stored in the Traceability customization profile used by the MagicDraw Profile.

The properties for the traceability relations are grouped into the realization, specification, and other groups, owning both direct (single level) and indirect (multiple levels) traceability relations. The indirect traceability relations are represented by the so-called **All** properties, for example, **All Specifying Elements**, **All Realizing Elements**, and **All Specific Classifiers**.

Copyright © 1998-2015 No Magic, Inc.

The relations for traceability creation, visualization and navigation is also possible without using traceability properties. However, the customizable model driven traceability properties, which represent the traceability relations in a single place, can greatly help in traceability information visualization and access.

TIP!

You can customize the predefined traceability properties according to your needs or create your own traceability properties and specify your own rules.

You can also customize the grouping of the traceability properties according to your needs.

For more information, please, refer to the sections “Extending metamodel with derived properties” and “Creating your own property groups and subgroups” from the chapter “DSL Customization engine” in “UML Profiling and DSL UserGuide.pdf”.

Element’s traceability properties are represented in the following places:

- Specification window — See "Traceability properties in Specification window" on page 517.
- **Properties** panel — See "Traceability properties in Properties panel" on page 520.
- Note on a diagram — See "Traceability properties in notes" on page 520.
- **Go To** submenu — See "Traceability properties in Go To submenu" on page 521.
- Traceability Report — See "Traceability properties in Traceability Report" on page 521.

TIP!

- For a list of the predefined traceability rules, see "Predefined Traceability Rules" on page 525.
- For more information about traceability feature, see "Traceability" on page 514.

Traceability properties in Specification window

The Traceability property group in element’s Specification window is one of the places, wherein the element’s traceability properties, showing its realizing and/ or more specific elements, are represented.

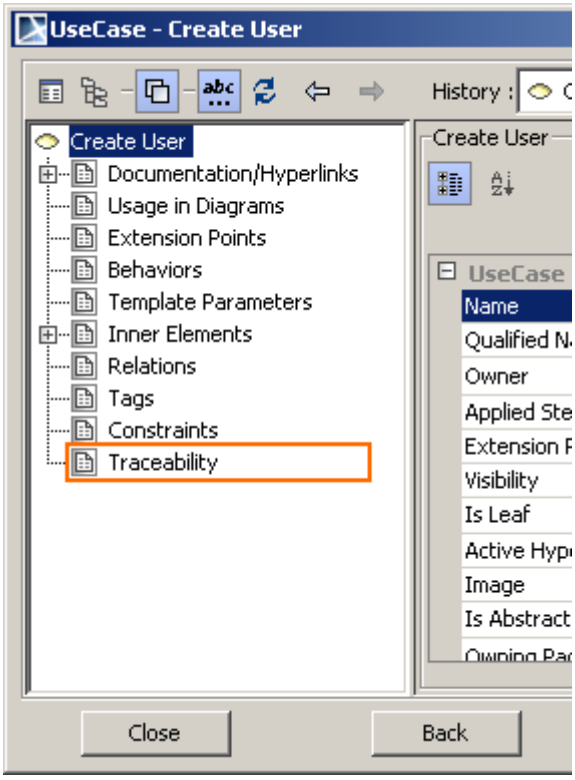


Figure 317 -- Traceability property group in the Specification window

Copyright © 1998-2015 No Magic, Inc.

To view an element’s traceability properties in its Specification window

1. Select an element and open the element’s Specification window by using one of the ways given in section "Specification Window" on page 273.
2. Click the **Traceability** property group. Now you can view element’s traceability properties.

The following picture gives an example of the traceability relations between the elements from different abstraction levels of the same project.

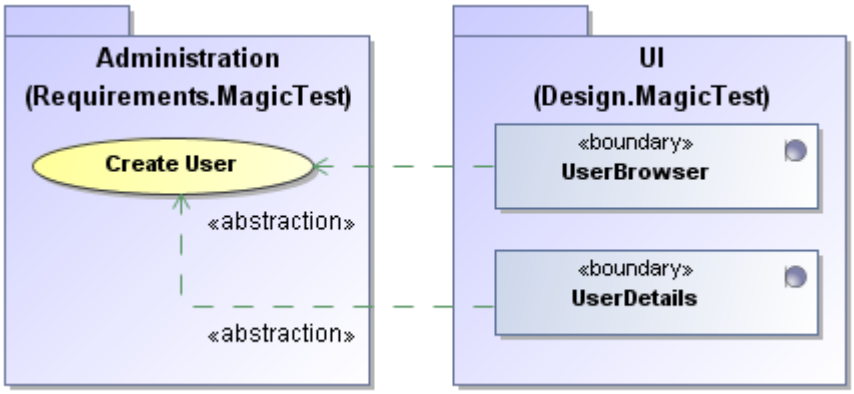


Figure 318 -- An example of single level traceability relations

As you see, the “Create User” use case from the “Requirements” package is covered by two design classes, and these classes are accordingly specified by this particular use case.

Therefore, the traceability properties such as **Realizing Class** and **Realizing Element** represent both design classes as realizing elements of this use case (see Figure 319 on page 519). Accordingly the traceability properties such as **Specifying Use Case** of each design class represent the “Create User” use case as a more specific element (see Figure 320 on page 519).

Copyright © 1998-2015 No Magic, Inc.

Note that the same related element can be represented through different traceability properties.

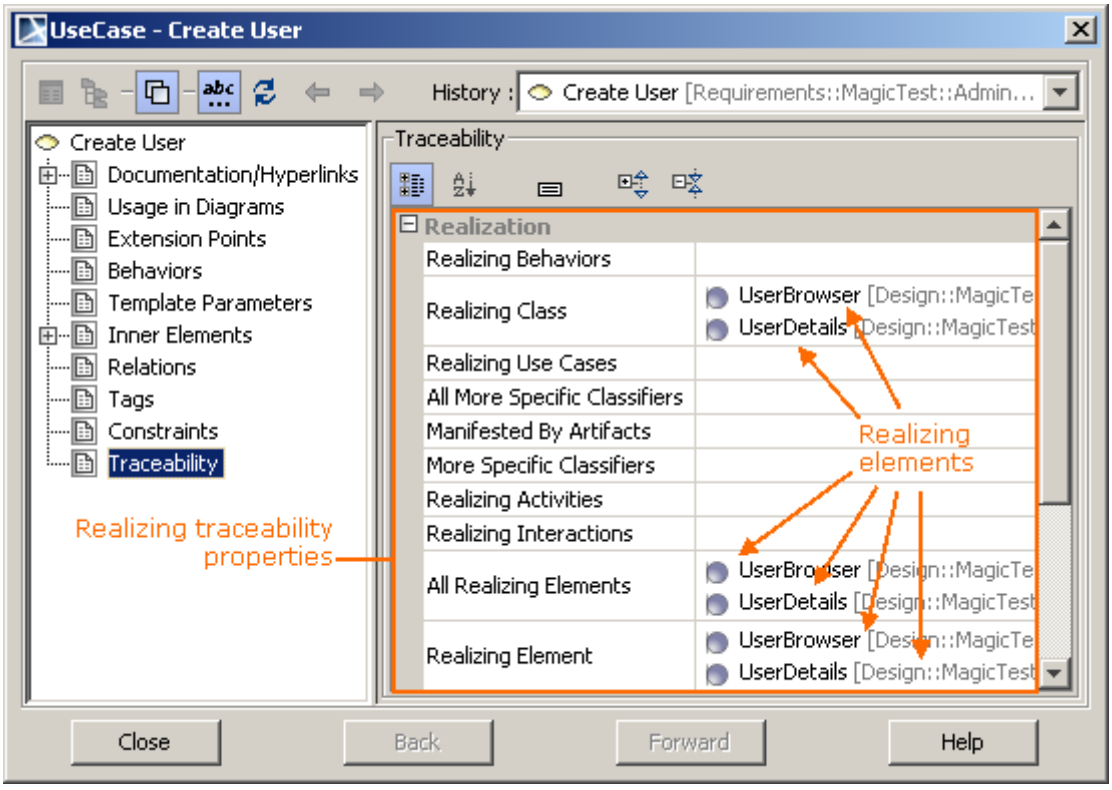


Figure 319 -- Realizing traceability properties in Specification window

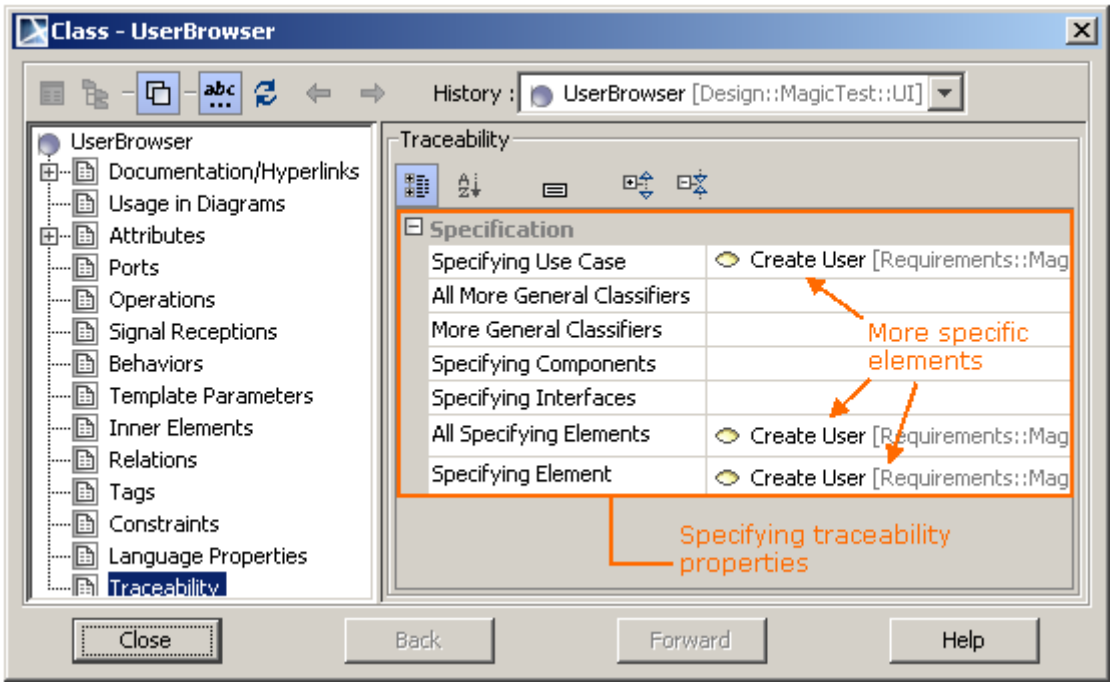


Figure 320 -- Specifying traceability properties in Specification window

TIP!

- For more information about element's Specification window, see "Specification Window" on page 273.
- For more information about traceability properties representation, see "Traceability Relations Representation" on page 516.

Copyright © 1998-2015 No Magic, Inc.

Traceability properties in Properties panel

The **Traceability** tab in the element's **Properties** panel (at the bottom of the Browser window) is one of the places, wherein the element's traceability properties, showing its realizing and/ or more specific elements, are represented.

To view an element's traceability properties in its **Properties** panel

1. Select the element in the Containment tree or its shape on the diagram.
2. In the Model Browser, click on the **Properties** panel > **Traceability** tab. Now you can view element's traceability properties.

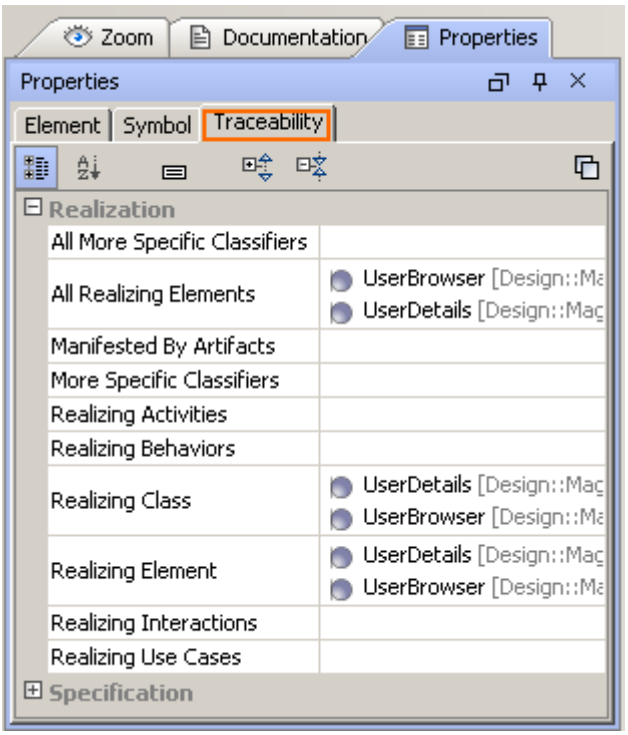


Figure 321 -- Traceability tab in Properties panel

TIP!

- For more information about element's **Properties** panel, see "Properties tab" on page 95.
- For more information about traceability properties representation, see "Traceability Relations Representation" on page 516.

Traceability properties in notes

The traceability properties can be visualized on a diagram using the standard MagicDraw mechanism displaying the property values in notes.

To visualize an element's traceability with other elements in the note on a diagram

1. Create a note for the element, whose traceability properties you want to visualize.
2. From the note's shortcut menu, select **Edit Compartment** > **Element Properties**.
3. In the list on the left-hand side, click the traceability property of you choice (e.g., **Realizing Class** or **Specifying Use Case**) and then click the **>** button.

Copyright © 1998-2015 No Magic, Inc.

4. Click **OK**. Now you can view the element's traceability with other elements.

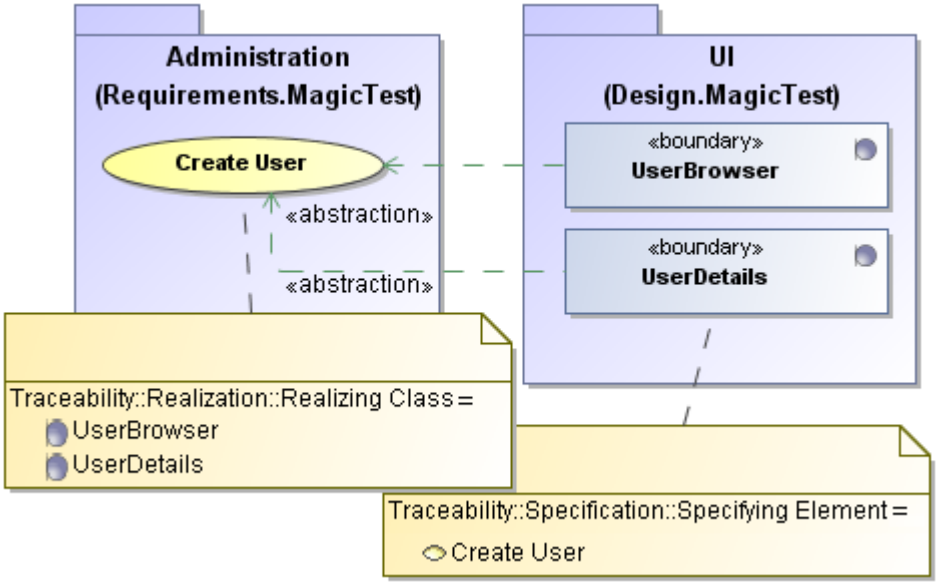


Figure 322 -- Traceability properties in the notes on a diagram

TIP!

- For more information about notes and comments on a diagrams, see "Note" on page 943 and "Comment" on page 859.
- For more information about traceability properties representation, see "Traceability Relations Representation" on page 516.

Traceability properties in Go To submenu

The submenu of the **Go To** menu on the element's shortcut menu allows you to easily find and navigate to the related elements through the traceability relations.

To select an element from the higher/lower level of abstraction in the Containment tree

1. Select the element in the Containment tree or its shape on the diagram.
2. From the element's shortcut menu, select **Go To** > **Traceability** > **Specification/ Realization/ Other**, choose a property, and then select an element. The element will be selected in the Containment tree.

TIP!

If there are more than 10 values, the scroll bar is shown and the text box for typing the keyword is available.

TIP!

For more information about traceability properties representation, see "Traceability Relations Representation" on page 516.

Traceability properties in Traceability Report

The Traceability Report feature supports the coverage analysis and publishes elements that are related to the selected elements through the traceability relations. You can generate a report either for the whole project or for a selected part of the project. This feature is the output of the coverage analysis.

Coverage analysis provides the visibility of each element's related artifacts, indicated as realizing (lower level of abstraction) and/ or specifying (higher level of abstraction) ones.

The main objective of the Traceability Report is to visualize and verify that Analysis, Design, and Implementation model elements are all covered.

Copyright © 1998-2015 No Magic, Inc.

The Traceability Report provides the ability to:

- Find the areas of uncovered parts.
- According to the report information, create additional artifacts to increase coverage.
- Measure the coverage quantitatively.
- Identify the redundant artifacts.

To generate a traceability report of the selected scope

1. On the **Tools** menu, click **Report Wizard**.
2. In the **Select Template** area, select **Traceability** > **Traceability** and then click **Next**.
3. Click **Next**.
4. In the **Select Element Scope** area, define the scope of the report by using the buttons placed between two lists, and then click **Next**.
5. In the **Output Options** area, define the appropriate options.
6. Click **Generate**. Your traceability report will be generated (generation time depends on the selected scope).

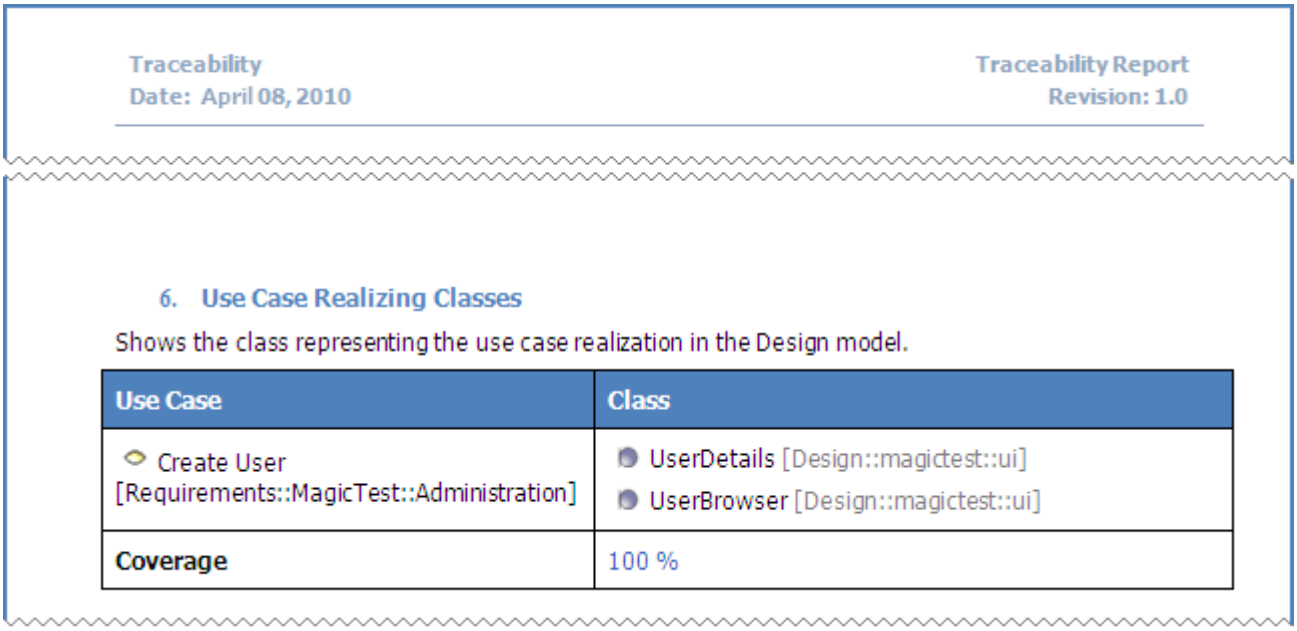


Figure 323 -- A fragment of Traceability Report

For more information about traceability properties representation, see "Traceability Relations Representation" on page 516.

Navigating between Different Levels of Abstraction

During the development process and in order to understand the system fast, navigating between elements from different levels of abstraction is necessary. Navigating from one element to another is easy using the MagicDraw GUI capabilities of the traceability feature, if there is at least one traceability relation between them.

Copyright © 1998-2015 No Magic, Inc.

To navigate between different levels of abstraction, use the following features

•Element's Specification window	Open the element's Specification window and select the **Traceability** property group, and then right-click on the property. Choose **Open Specification** and select an element (if there is more than one related element) as it is depicted in Figure 324 on page 523. The Specification window of this element will be opened.
•Element's **Properties** panel	Open the element's **Properties** panel and select the **Traceability** tab, and then right-click on the property. Choose **Open Specification** and select an element (if there is more than one related element) as it is depicted in Figure 324 on page 523. The Specification window of this element will be opened.
•**Go To** submenu on element's shortcut menu	From the element's shortcut menu, select **Go To** > **Traceability** > **Specification/ Realization/Other**. Choose a property and then select an element. The element will be selected in the Containment tree.

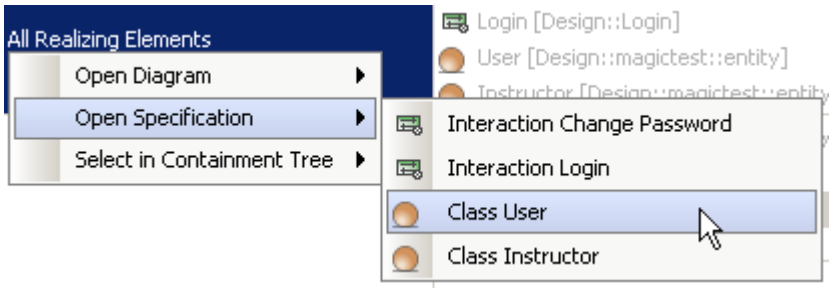


Figure 324 -- Selecting an element to open specification

Analyzing Traceability Relations

You can visualize the traceability relations among the elements in your project in order to analyze them, using the following MagicDraw features:

- **Relation Map** (for the analysis of traces among multiple levels of abstraction).
  See "Analysis using Relation Map" on page 523.
- **Dependency Matrix** (for the analysis of traces between any two levels of abstraction).
  See "Analysis using Dependency Matrix" on page 523.

Analysis using Relation Map

The Relation Map feature allows you to rapidly review and analyze multilevel relationships among elements starting from the requirements to implementation all the way through different levels of abstraction (analysis, design, and so on).



For more information about the Relation Map feature, see "Relation Map" on page 544.

Analysis using Dependency Matrix

You can perform the impact and/or gap analysis in your project using the Dependency Matrix feature, which is a powerful way for representing traceability relations between multiple elements from different packages, levels of abstraction, views, or other relations that cannot be represented on diagrams, for example, relations through UML tags.

Copyright © 1998-2015 No Magic, Inc.

You can create your own dependency matrices or even custom dependency matrix types for visualization of various traceability relations.You only need to define a relevant traceability property as **dependency criteria** for this.

TIP!

For a list of the predefined traceability rules,
see "Predefined Traceability Rules" on page 525.

The following figure depicts an example of traceability relations between several model elements.

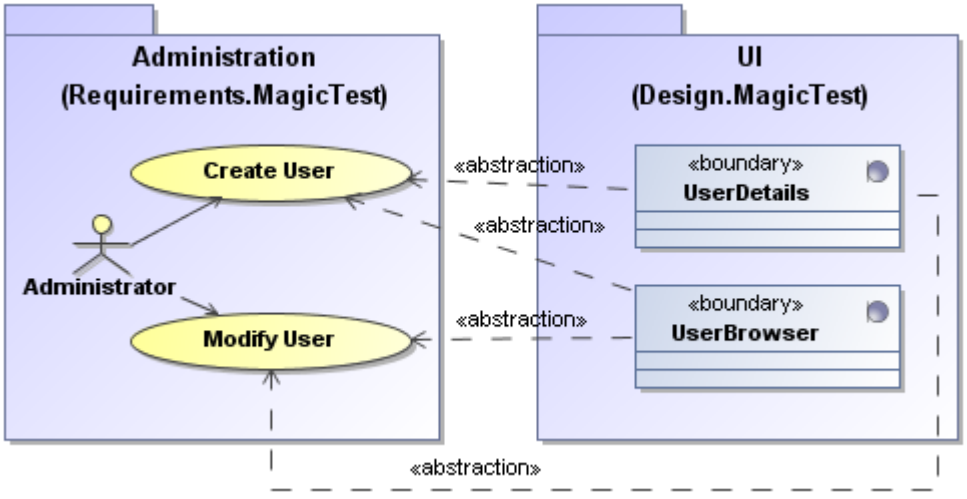


Figure 325 -- Example of traceability relations

These relations can be represented by the following traceability properties:

1. **Realizing relations (use cases → classes):**
   - Realizing Class
   - Realizing Element
   - All Realizing Elements
2. **Specifying relations (classes ← use cases):**
   - Specifying Use Case
   - Specifying Element
   - All Specifying Elements

You can create a dependency matrix to visualize these traceability relations.

The following steps will show you how to create a dependency matrix for the **Realizing Class** predefined property.

To create a dependency matrix for the **Realizing Class** predefined property

1. Create a Dependency Matrix diagram.
2. Define row and column element types as follows:
   2.1 For the row type, select **"UseCase"**.
   2.2 For the column type, select **"Class"**.
3. Define row and column scopes as follows:
   3.1 For the row scope, expand **Data** > **Requirements** > **MagicTest** and select **Administration**.
   3.2 For the column scope, expand **Data** > **Design** > **MagicTest** and select **UI**.
4. Define the **Realizing Class** property as dependency criteria.
5. Rebuild the matrix.

Copyright © 1998-2015 No Magic, Inc.

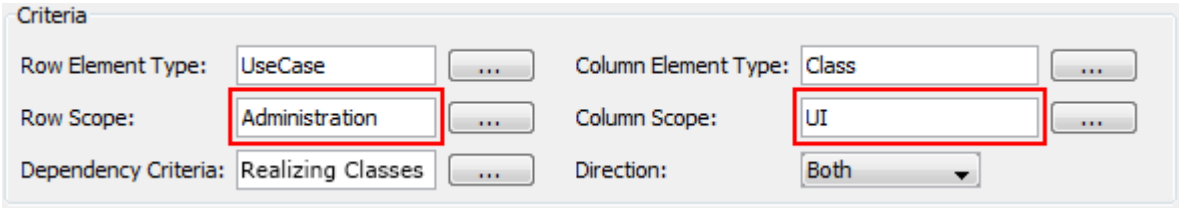


Figure 326 -- Defining criteria for visualization of traceability relationships that can be represented by Realizing Class predefined property

The following figure depicts the created dependency matrix showing the traceability relationships between use cases and them realizing design classes (the highlighted intersections mark the traceability relations depicted in Figure 325 on page 524).

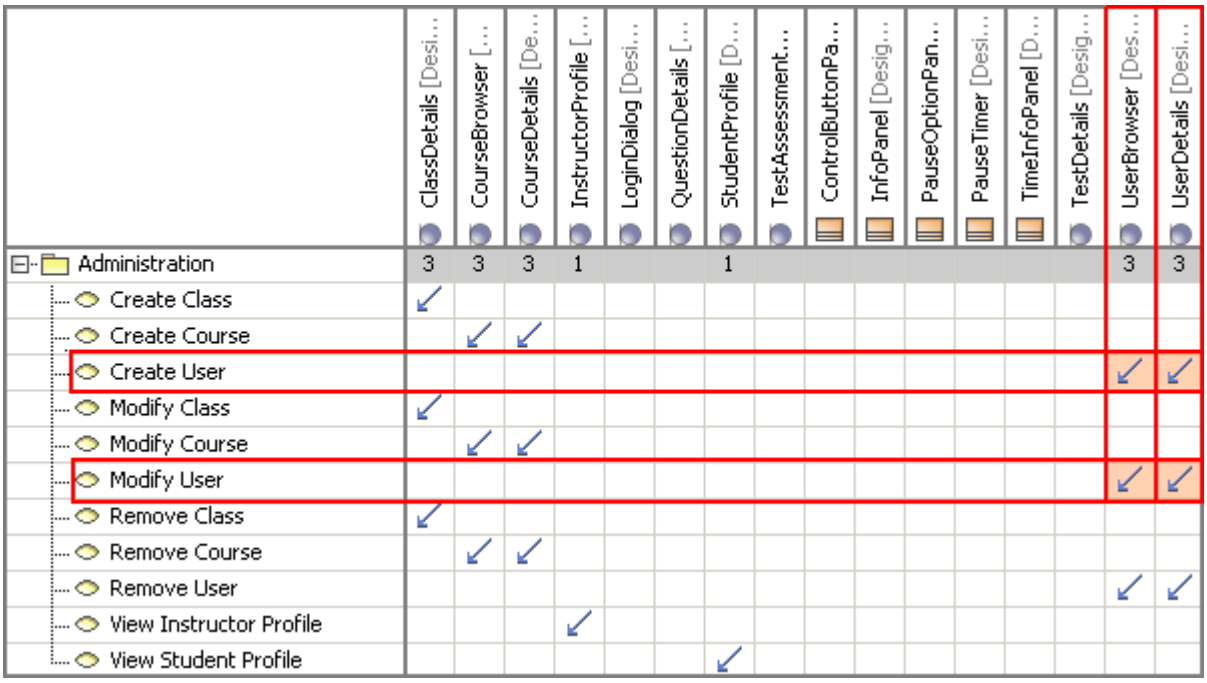


Figure 327 -- Visualization of Realizing Class traceability relations on dependency matrix

TIP!

For more information about the Dependency Matrix feature, see "Dependency Matrix" on page 764.

Predefined Traceability Rules

Besides allowing you to create your own traceability rules, MagicDraw allows you to use a set of predefined element properties for traceability purposes. The traceability solution also includes a report template used for coverage analysis. You can customize this template or create your own templates for producing your own traceability analysis reports.

The main objective of the predefined traceability rules is to allow the use of the traceability engine instantly (out-of-the-box) for a typical project. The predefined rules allow for analyzing and verifying links between elements residing in Analysis, Design, and Implementation layers of your models. Elements in these layers are extended with a set of properties, pointing to elements from the higher abstraction layer (backward traceability) and the lower abstraction layer (forward traceability). Overall, there are three groups of predefined traceability properties, which will be detailed in subsequent sections:

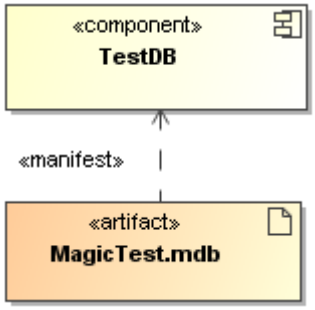
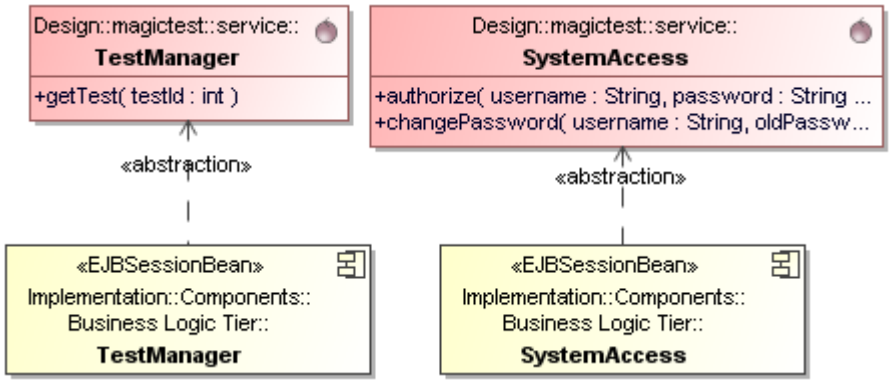
- "Forward Traceability - Realization" on page 526
- "Backward Traceability – Specification" on page 529

Copyright © 1998-2015 No Magic, Inc.

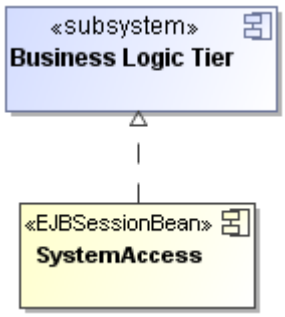
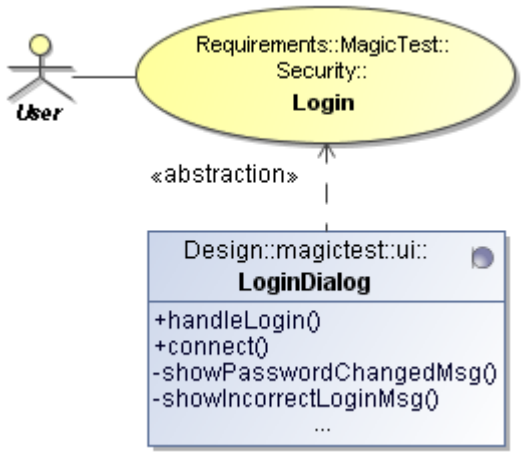
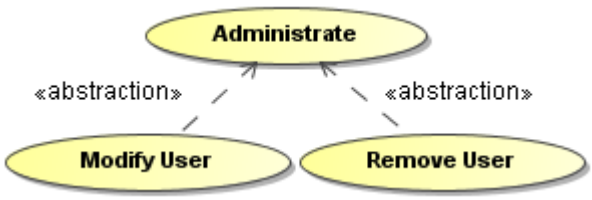
- "Other traceability relations" on page 532

Forward Traceability - Realization

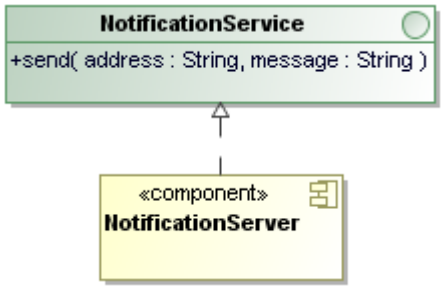
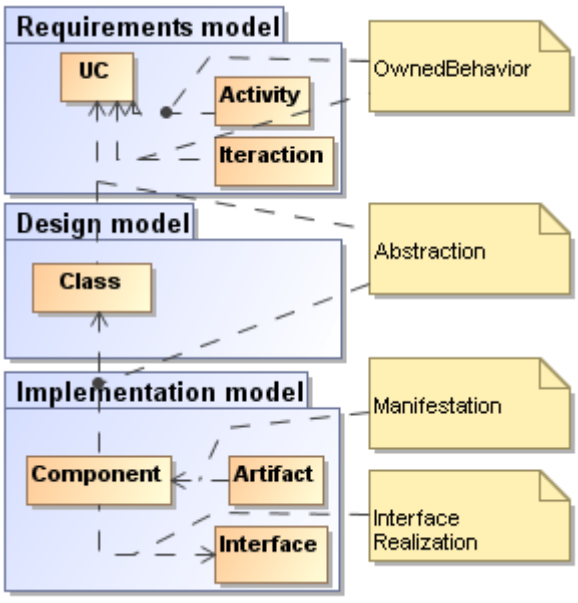
The forward traceability ensures that all specified artifacts are covered by elements from the lower abstraction level.

Property name	Description	Applied for	Reference through...	Value elements type
Manifested By Artifacts	The property shows artifacts that physically render the current component.	Component	Relationships: Manifestation	Artifact
	Example: 			
Realizing Component	The property shows the components representing the class realization in the Implementation model.	Class	Relationships: Abstraction	Component
	Example: 			
Realizing Classifier	The property shows classifiers that realize components through component realization.	Component	Relationships: Component Realization, Realization	Classifier

Copyright © 1998-2015 No Magic, Inc.

Property name	Description	Applied for	Reference through...	Value elements type
	Example: 			
Realizing Class	The property shows the classes representing the use case realization in the Design model.	Use Use Case	Relationships: Abstraction	Class
	Example: 			
Realizing Use Case	The property shows the realizing use cases of the current use case in the lower level of abstraction thus demonstrating how the use case is implemented. For example, the Requirements Use Case realizes the Business Use Case.	Use Use Case	Relationships: Abstraction	Use Case
	Example: 			

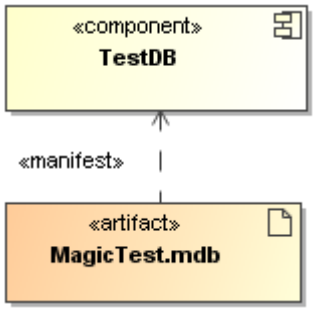
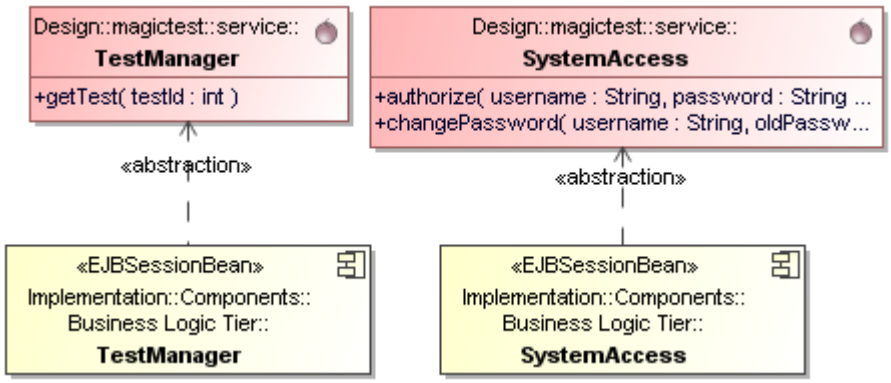
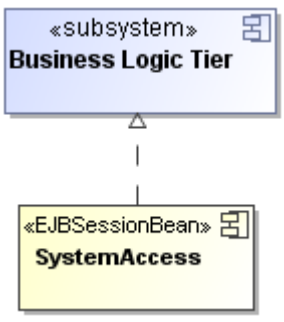
Copyright © 1998-2015 No Magic, Inc.

Property name	Description	Applied for	Reference through...	Value elements type
Realizing Classifier	The property shows the classifiers conforming to the contract specified by the interface and related with this interface through the interface realization relationship.	Interface	Relationships: Interface Realization	Classifier
	Example:  			
Realizing Element, All Realizing Elements	The **Realizing Element** property gathers realizing elements from the lower abstraction level. The **All Realizing Elements** property transitively gathers realizing elements from all lower abstraction levels.	Element	Relationships: Abstraction, Component Realization, Interface Realization.	Element
	Example:  			

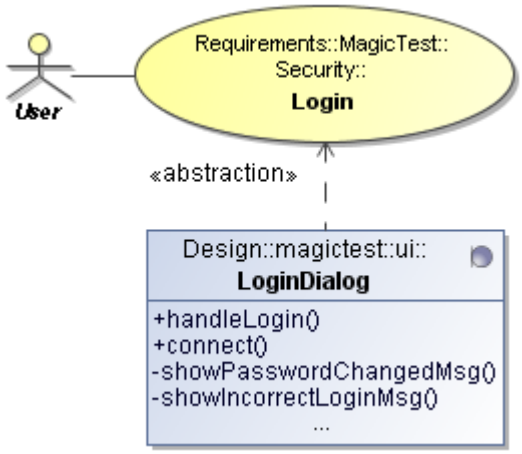
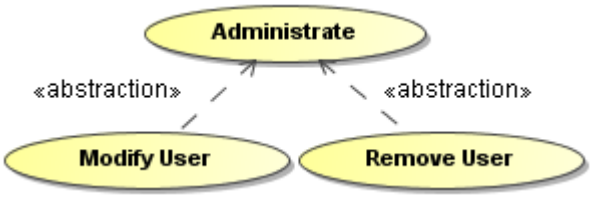
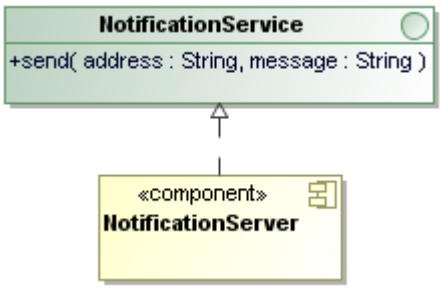
Copyright © 1998-2015 No Magic, Inc.

Backward Traceability – Specification

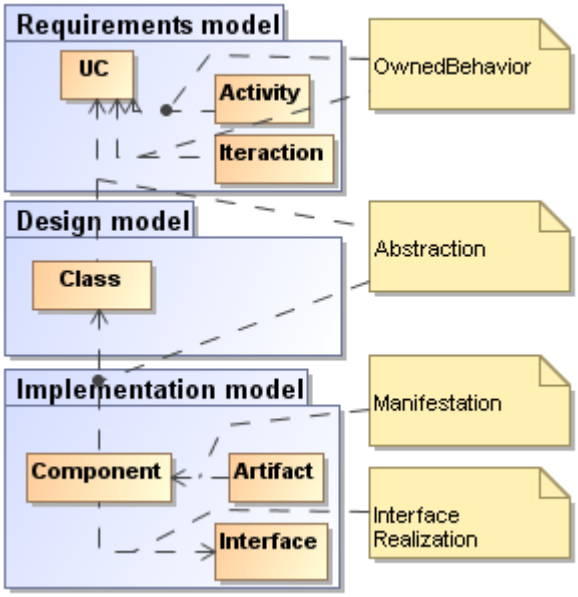
The backward traceability ensures that nothing but the intended functionality is covered.

Property name	Description	Applied for	Reference through...	Value elements type
Manifested In Artifacts	The property shows which components are manifested in an artifact.	Artifact	Relationships: Manifestation	Component
	Example: 			
Specifying Class	The property shows the classes representing the component specification from the Design model.	Component	Relationships: Abstraction	Class
	Example: 			
Specifying Component	The property shows the components that are realized by classifiers through component realization.	Classifier	Relationships: Component Realization, Realization	Component
	Example: 			

Copyright © 1998-2015 No Magic, Inc.

Specifying Use Case	The property shows the use cases (from the Requirements model) representing the class specification.	Class	Relationships: Abstraction	Use Case
	Example: 			
Specifying Use Case	The property shows the use cases that specify the given use case in the higher level of abstraction. For example, the Business Use Case specifies the Requirements Use Case.	Use Case	Relationships: Abstraction	Use Case
	Example: 			
Realized Interface	The property shows the interfaces specifying the contract, in which the related classifier conforms to.	Classifier	Relationships: Interface Realization	Interface
	Example: 			

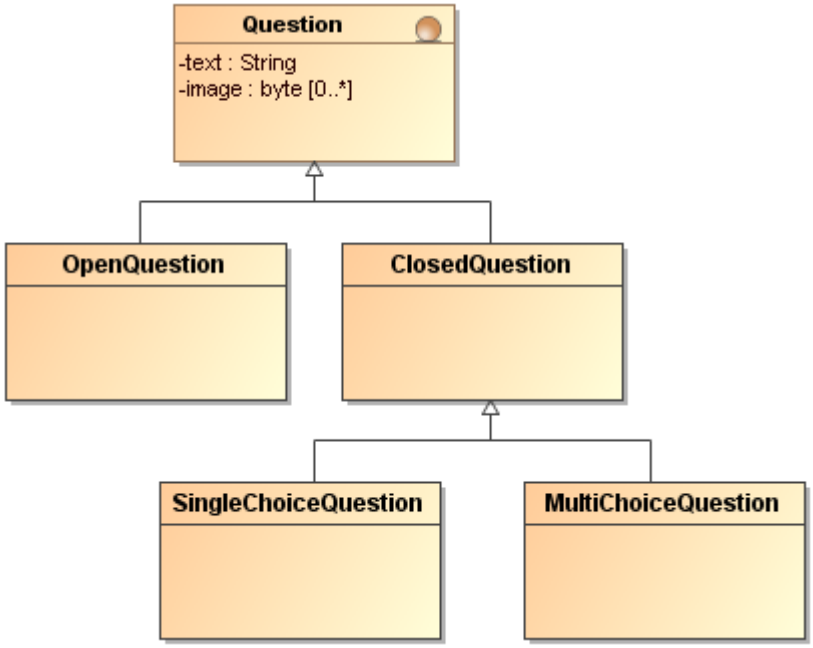
Copyright © 1998-2015 No Magic, Inc.

Specifying Element, All Specifying Elements	The **Specifying Element** property gathers specifying elements from the upper abstraction level. The **All Specifying Elements** property transitively gathers specifying elements from all upper abstraction levels.	Element	Relationships: Abstraction, Component Realization, Interface Realization.	Element
	Example:  			

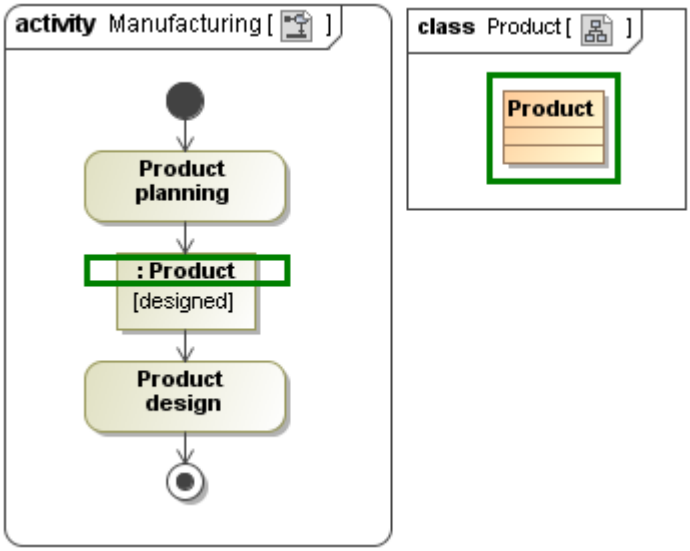
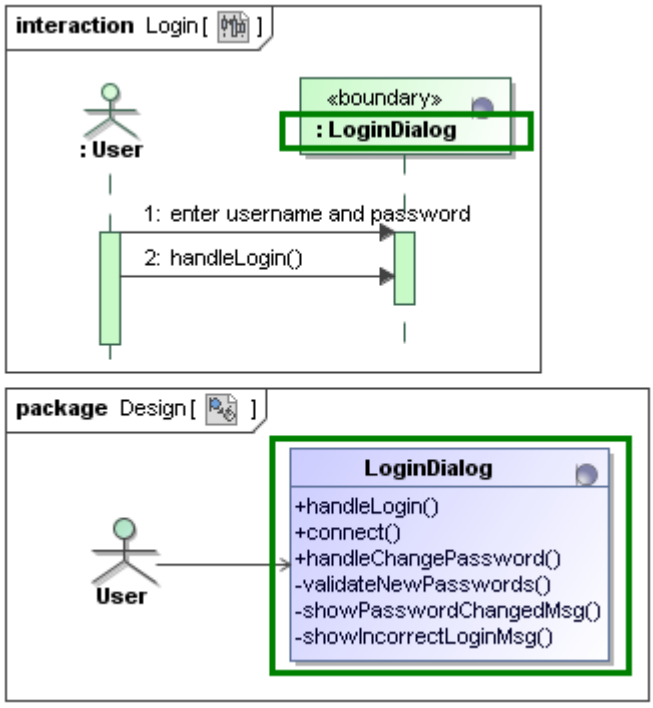
Copyright © 1998-2015 No Magic, Inc.

Other traceability relations

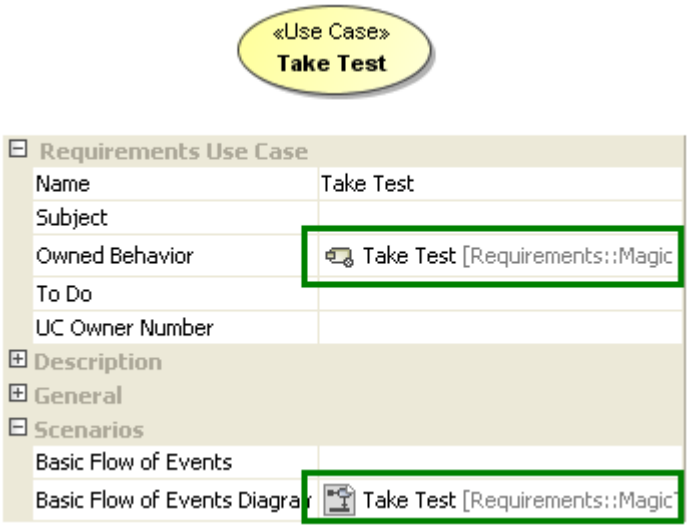
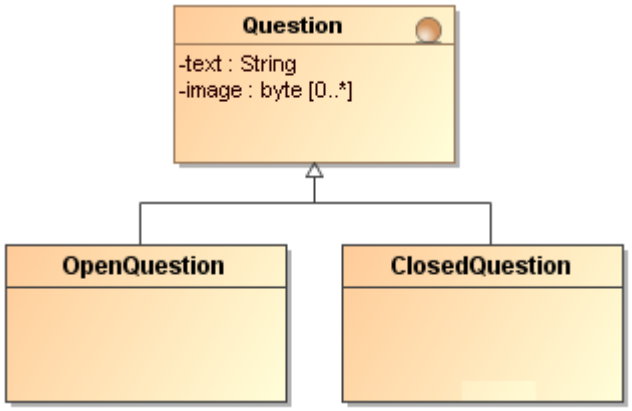
This group contains properties enabling traceability between elements of the same abstraction level.

Property name	Description	Applied for	Reference through...	Value elements type
Base Classifier, All General Classifiers	The **Base Classifier** property gathers classifiers from which the current element directly inherits. The **All General Classifiers** property gathers classifiers from which the current element directly or indirectly inherits.	Classifier	Relationships: Generalization.	Classifier
	Example:  			
Participates in Activity	The property shows either the activities wherein the classifier is used as an object type, or the activities that are owned by the classifier.	Classifier	Property chain	Activity

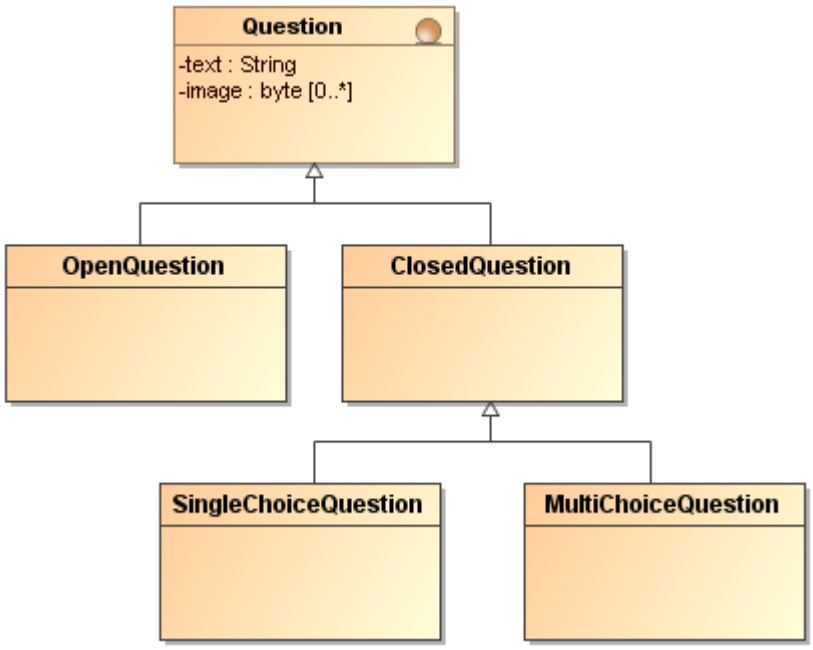
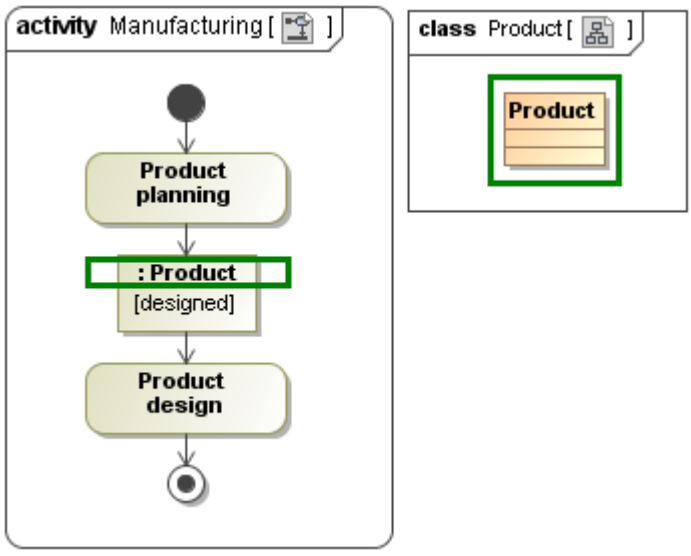
Copyright © 1998-2015 No Magic, Inc.

Property name	Description	Applied for	Reference through...	Value elements type
	Example: 			
Participates in Interaction	The property shows either the interactions wherein the classifier is used as an object type, or the interactions that are owned by the classifier.	Classifier	Property chain	Interaction
	Example: 			

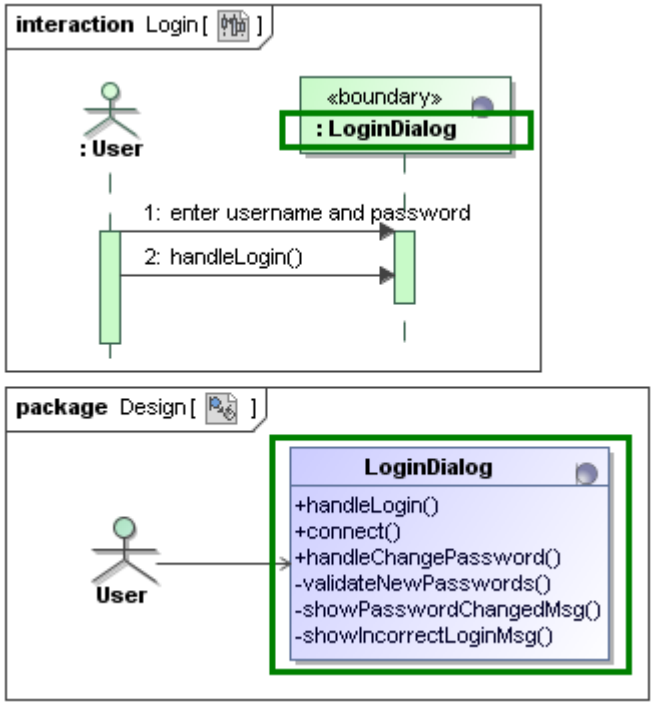
Copyright © 1998-2015 No Magic, Inc.

Property name	Description	Applied for	Reference through...	Value elements type
Describing Behavior	The property gathers behavior diagrams (state, activity, sequence, communication, and interaction) that describe the use case.	Use Case	Properties: ownedBehavior Tags: Exceptional Flow of Events Diagrams, Basic Flow of Events Diagrams, Alternative Flow of Events Diagrams.	State, Activity, Sequence, Communication, and Interaction Diagrams
	Example: 			
Specific Classifier	The property shows directly inherited classifiers.	Classifier	Relationships: Generalization.	Classifier
	Example: 			
All Specific Classifiers	The property gathers directly or indirectly inherited classifiers.	Classifier	Relationships: Generalization.	Classifier

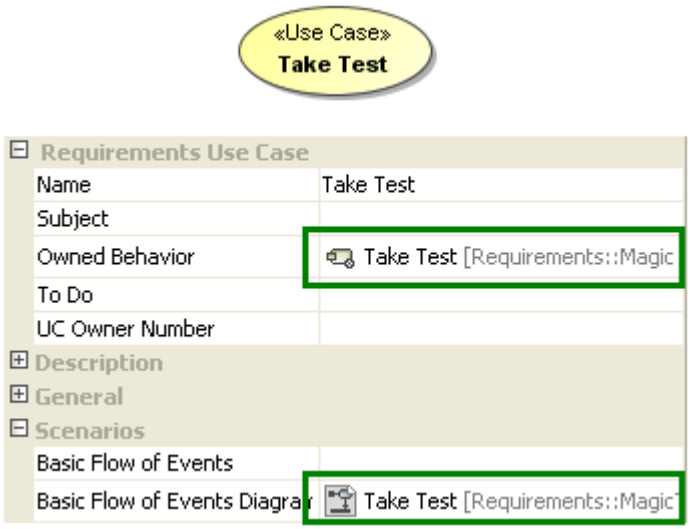
Copyright © 1998-2015 No Magic, Inc.

Property name	Description	Applied for	Reference through...	Value elements type
	Example:  			
Used Classifier	The property gathers what classifiers are used in the activity as object types, or the classifier that owns the activity.	Activity	Property chain	Classifier
	Example:  			
Used Classifier	The property gathers what classifiers are used in the activity as object types, or the classifier that owns the activity.	InterAction	Property chain	Classifier

Copyright © 1998-2015 No Magic, Inc.

Property name	Description	Applied for	Reference through...	Value elements type
	Example:  			
Described Use Case	The property shows the described use cases by behavior or behavior diagram.	State, Activity, Sequence, Communication, and Interaction Diagrams	Properties: ownedBehavior Tags: Exceptional Flow of Events Diagrams, Basic Flow of Events Diagrams, Alternative Flow of Events Diagrams.	Use Case

Copyright © 1998-2015 No Magic, Inc.

Property name	Description	Applied for	Reference through...	Value elements type
	Example: 			

Derived Properties

While working with your models, sometimes in the same diagram or table, you need to see information that is not directly related to the element. You can display that information in the following ways:

- In the note connected to an element
- In the table, as a new column

To display the information indirectly related to an element, you need to create derived properties. This is a property whose values are calculated.

MagicDraw gives you a possibility to create, edit, or delete derived properties in

- The Specification window of an element.
- A generic table.

You can manage derived properties using the **Expression** dialog.

To create a new derived property

1. Open the **Expression** dialog by doing one of the following:
   - In the selected element's Specification window, click the **Traceability** property group and then click the **Create** button.
   - In the toolbar of an active table, click **Show Columns** and then select **New Derived Property**.
2. In the open dialog, specify the name and expression of the derived property.



For more information about the **Expression** dialog, see "Specifying Criteria for Querying Model" on page 551.

Copyright © 1998-2015 No Magic, Inc.

3. Click **OK**.



If you want to create a derived property with the result other than an element, you need to change the derived property type manually.

To change a type of the derived property manually

1. On the **Containment** tab toolbar of the **Containment** tab on the Model Browser, click the Filter button. The **Items Filter** dialog opens.
2. In the open dialog, click to select the **Derived Properties Suite** check box and then click **OK**.
3. In the Containment tree, expand the Derived Properties package and find the customization that contains the derived property you want to modify.
4. Expand the customization and select the property which type you want to change.
5. Open the selected property's Specification window.
6. Change the property type from *Element* to the desired one (for example, *String*).
7. Save and reopen the project when you are finished.

To edit a derived property

1. Do one of the following:
   - In the selected element's Specification window, click the **Traceability** property group, select a derived property you want to edit, and then click the **Edit** button.
   - In the active table, from the derived property column heading's shortcut menu, select **Edit Expression**.
2. In the **Expression** dialog, make changes and click **OK**.

NEW! To edit a derived property value

- In the selected element's Specification window, in the **General** or **Traceability** property group, click a derived property value cell, and then click the **Edit** button .



You can edit the value of the derived property when the operation type is simple navigation and you had specified only one criterion.

For more information about simple navigation operations, refer to "Using Simple Navigation Operations to Specify Criteria" on page 554.

To remove a derived property

1. Do one of the following:
   - In the selected element's **Specification** window, click the **Traceability** property group, select a derived property you want to edit, and then click the **Delete** button.
   - In the active table, from the derived property column heading's shortcut menu, select **Remove Derived Property**.

To display derived properties on a note shape

1. Draw a note for the selected element or select an existing one.
2. Do one of the following:
   - Select a note and click Edit Compartment.



Copyright © 1998-2015 No Magic, Inc.

- On the note's shortcut menu, click **Edit Compartment** > **Element Properties**. The **Compartment Edit** dialog opens.

3. In the **All** list, move properties you want to display on a note shape to the **Selected** list by doing one of the following:
   - Select one or more properties and click the **>** or **>>** button.
   - Double-click to select properties one by one.
4. Click **OK**. Selected derived properties are displayed in the note.

To display derived properties in a table

1. In the toolbar of an active table, click **Show Columns** and then click the **Select Columns** command.
2. In the opened **Select Columns** dialog, select properties you want to display.
3. Click **OK** when you are done.

Case study:

Let's say, we have *LibraryItem* class with operations and we need to display only those operations that have a «setter» stereotype applied.



Let's create the **Setters** derived property for the class *LibraryItem*. Open the *LibraryItem* **Specification** window and in the **Traceability** property group click the **Create** button. In the **Expression** dialog, select the **Meta Chain** tab. Then specify the derived property name, expression for the meta chain as it is shown in the following figure.

Copyright © 1998-2015 No Magic, Inc.

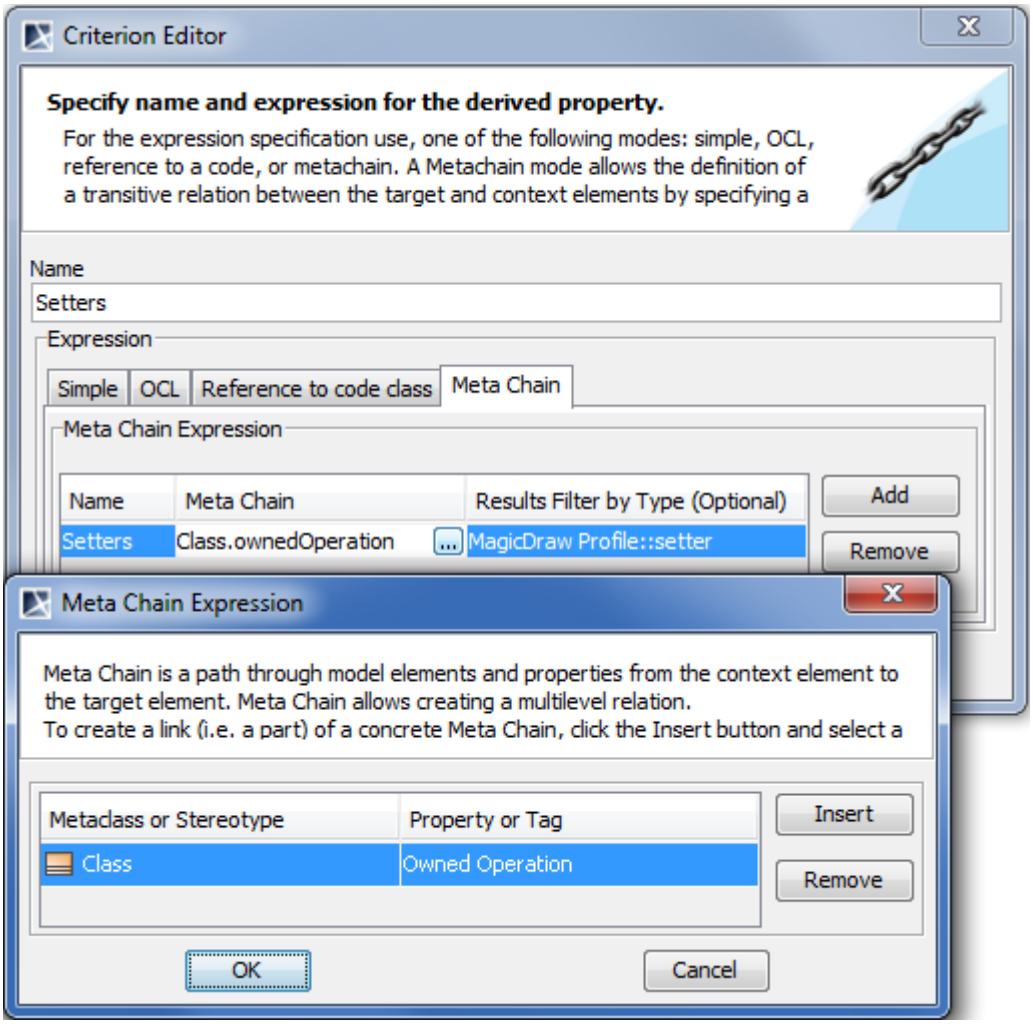


Figure 328 -- Specification of derived property name and expression

In the following figure, you can see that our created **Setters** derived property is placed under this subgroup **Other** in the class **Specification** window. The **Setters** derived property has a number of operations as its values.

Copyright © 1998-2015 No Magic, Inc.

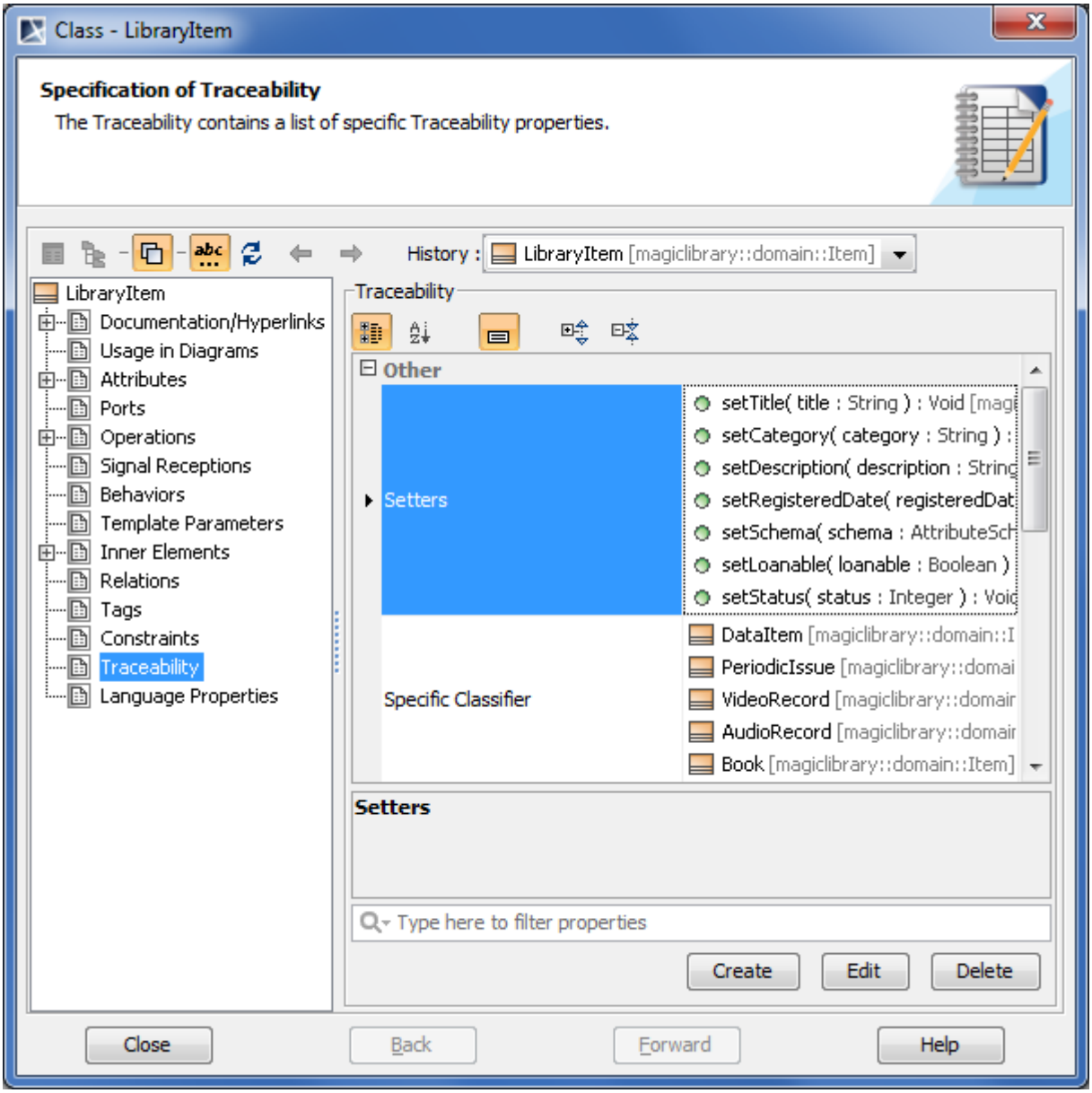


Figure 329 -- Example of newly created derived property

Next, we need to display values of the **Setters** derived property for the *LibraryItem* class in the diagram.



A derived property is applicable to an element type, not the specific element. In this particular example, our created derived properties are applicable to all classes, not only to *LibraryItem*.

We can use a Note element wherein we can display only those operations of the *LibraryItem* class that have the «setter» stereotype applied. Let's draw a Note and connect it to the *LibrartItem* class. Then click the Edit Compartment as is shown in the following figure. In the **Compartment Edit** dialog, select the **Element Properties** tab. From the properties list, select to display the **Setters** property. After clicking the **OK** button, all operations with the «setter» stereotype applied are displayed in the Note (see the following figure).

Copyright © 1998-2015 No Magic, Inc.

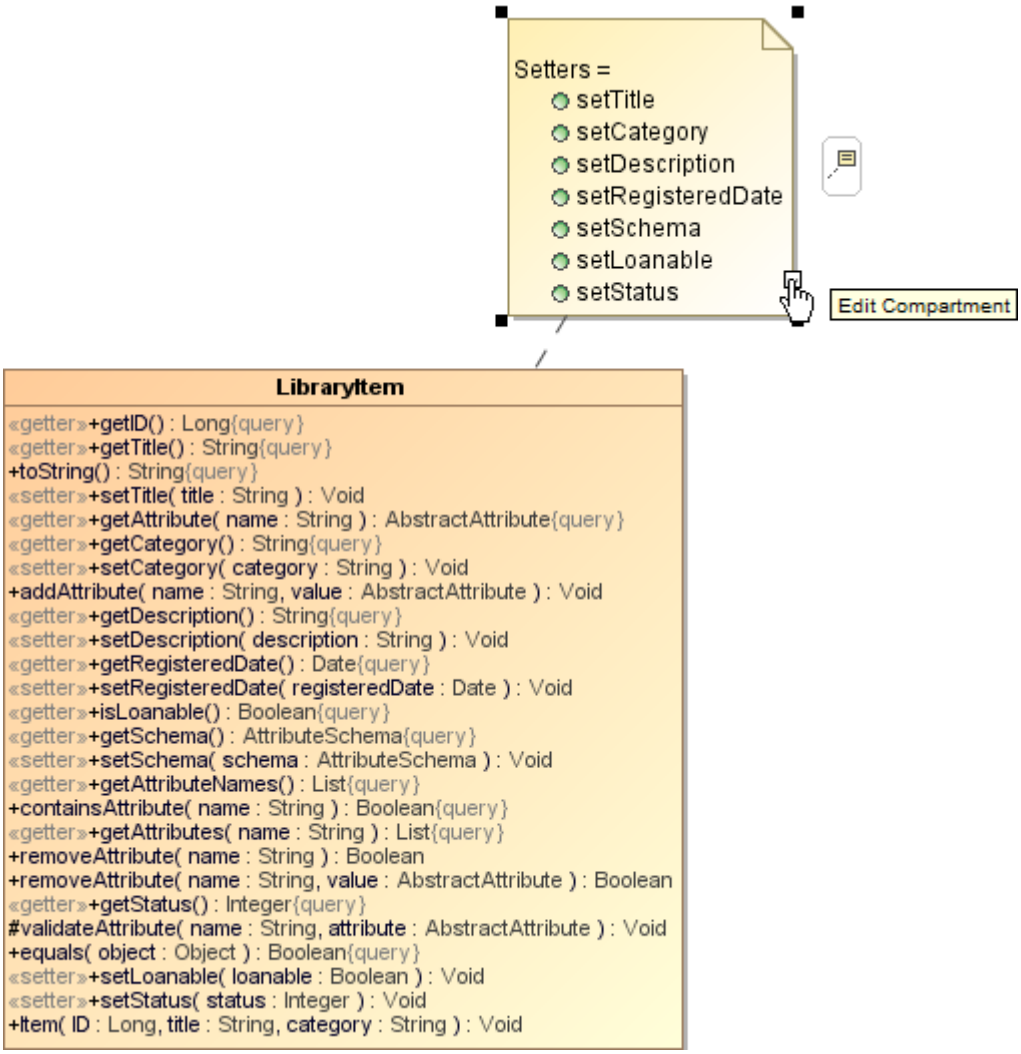


Figure 330 -- Operations with «setter» stereotype are displayed in Note

Now, let's create a generic table wherein we can also display operations with the «setter» stereotype for the LibraryItem class, as well as for other classes. For that purpose, we select Class as an Element Type and add existing classes to the table. Then we select which information to display as new columns. Previously created Setters derived property is also available to add as a new column through the Show Columns menu (see the following figure).

#	Name	Setters
1	LibraryItem	setTitle(title : String) : Void setCategory(category : String) : Void setDescription(description : String) : Void setRegisteredDate(registeredDate : Date) : Void setSchema(schema : AttributeSchema) : Void setLoanable(loanable : Boolean) : Void setStatus(status : Integer) : Void
2	AttributeSchema	setName(name : String) : Void

Figure 331 -- Operations with «setter» stereotype are displayed in new column

Copyright © 1998-2015 No Magic, Inc.

Now, let's create a new derived property directly in the generic table. This property will show which operations have «getter» stereotype applied. We can do that using the same **Expression** and doing the same actions as described previously. In the following figure is the example of the **Getters** derived property expression.

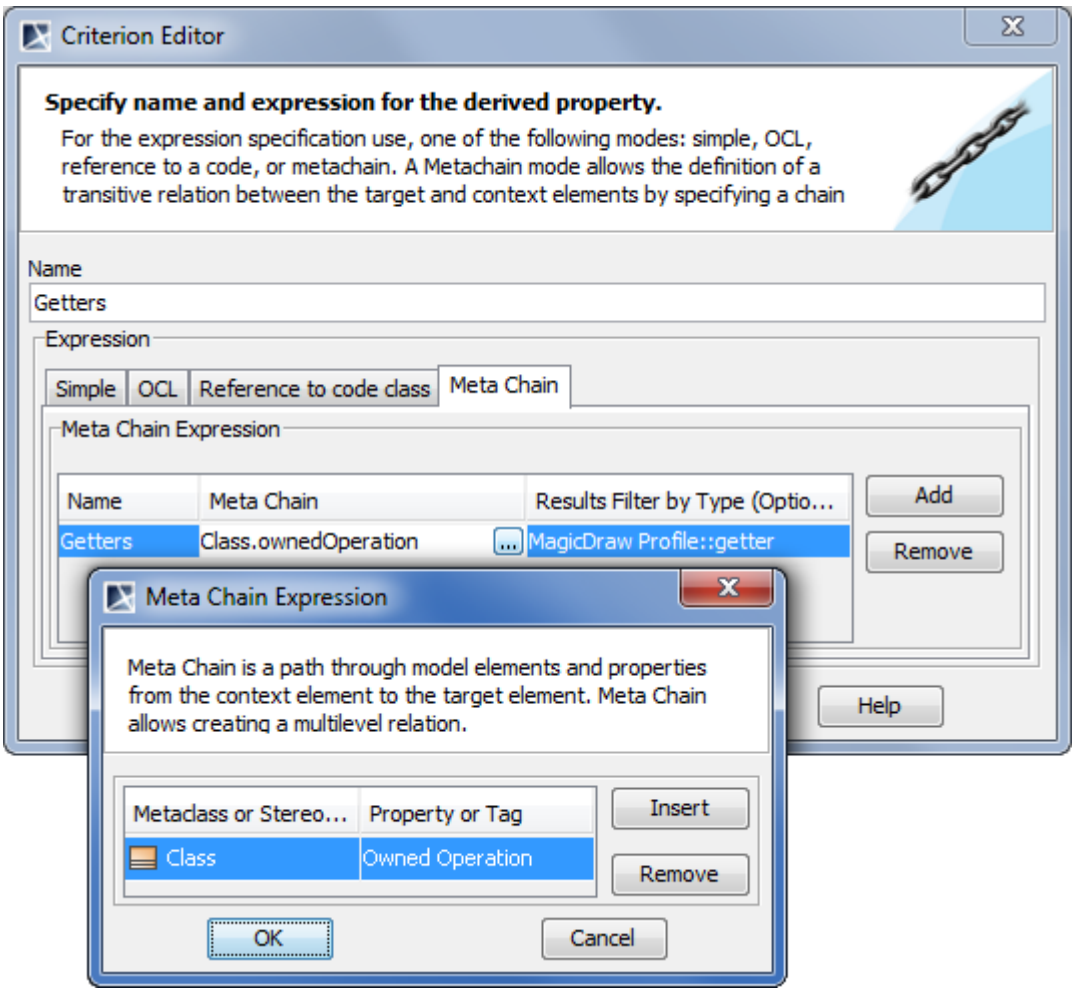


Figure 332 -- Specification of derived property name and expression

When the expression is specified, the newly created derived property is displayed as a new column (see the following figure).

#	Name	Setters	Getters
1	LibraryItem	setTitle(title : String) : Void setCategory(category : String) : Void setDescription(description : String) : Void setRegisteredDate(registeredDate : Date) : Void setSchema(schema : AttributeSchema) : Void setLoanable(loanable : Boolean) : Void setStatus(status : Integer) : Void	getID() : Long getTitle() : String getAttribute(name : String) : AbstractAttribute getCategory() : String getDescription() : String getRegisteredDate() : Date isLoanable() : Boolean getSchema() : AttributeSchema getAttributeNames() : List getAttributes(name : String) : List
2	AttributeSchema	setName(name : String) : Void	getName() : String getAttributeType(name : String) : String isAttributeMultiple(name : String) : Boolean getID() : Long

Figure 333 -- Operations with «getter» stereotype are displayed in new column

Copyright © 1998-2015 No Magic, Inc.

Related references

Generic Table

Specification Window

Related concepts

Note

Relation Map

NOTE

This feature is available in Standard, Professional, Architect, and Enterprise editions.

The Relation Map diagram allows you:

- Review and analyze relations among the elements of the whole model.
- **NEW!** Create new elements directly in the relation map.

The model structure in the diagram can be over-viewed in two different layouts: tree or radial.

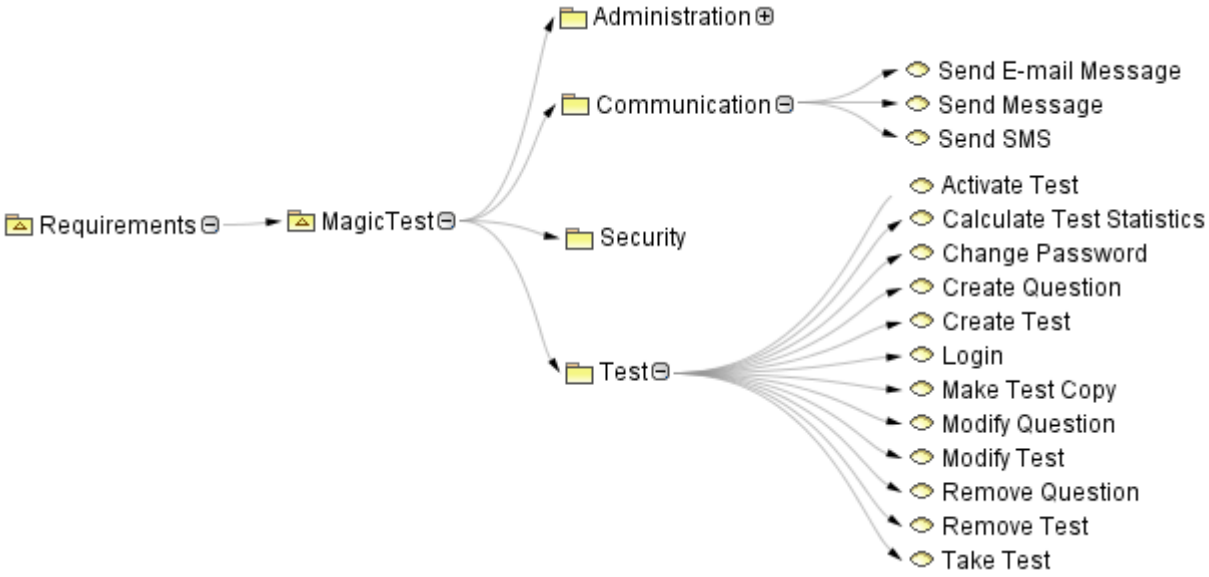


Figure 334 -- Tree-style layout of Relation Map

Copyright © 1998-2015 No Magic, Inc.

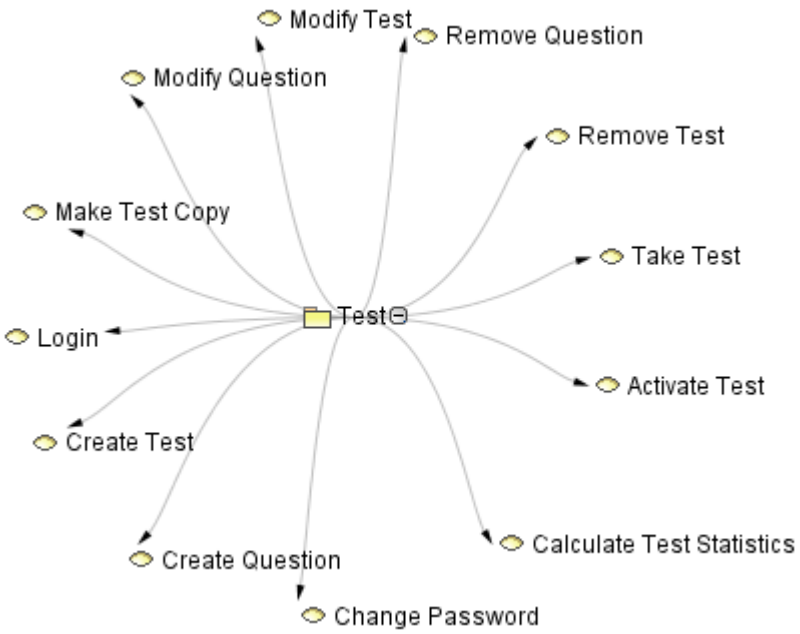


Figure 335 -- Radial-style layout of Relation Map

Relation Map makes it easier for you to

- Discover the existing relationships of the nodes (elements) in a unique and highly usable feature giving a fast project overview in two different rendering types:
  - Dynamic – holding only needed information at a time by centering the diagram to the selected node and visualizing related elements to the set depth.
  - Static – representing and holding all the step-by-step discovered structure in a diagram. This rendering type is good for publishing, viewing structure and presentation purposes.
- Have a radial or tree layouts for visualizing multilevel relationships that are suitable for hierarchical and compact representation of structures.
- Observe traceability from requirements to implementation all the way through different levels of abstraction (analysis, design, and so on).
- Have a visualized representation of relationships between elements from different views.
- Have a usable and good-looking project context map.
- Make fast analysis of UML model.

Creating Relation Maps

Relation map is one of analysis diagram types. A newly created relation map appears in the Model Browser as a model element. The actions relevant for diagrams are valid for relation maps as well.

To create an empty relation map

1. In the Containment tree, select an element, for which you need to create the relation map. This element is the *context* of the relation map.
2. Do one of the following:
   - From the main menu, select **Diagrams** > **Create Diagram**. Type “rel” and press **Enter**.

Copyright © 1998-2015 No Magic, Inc.

- On the main toolbars, click the **Create Diagram** button. Type “rel” and press **Enter**.
- Press Ctrl+N. Type “rel” and press **Enter**.
- On the **Analyze** menu, click **Create Relation Map**.
- Right-click the element and from the shortcut menu select **Create Diagram** > **Analysis Diagrams** > **Relation Map**.
- Right-click the element and from the shortcut menu select **Related Elements** > **Create Relation Map**.

The newly created relation map opens on the right side of the application window.



The context of the relation map automatically becomes the owner of it when using any of the last two approaches.

When using other approaches, you can specify another element as the owner.

Specifying Relation Map Criteria, Layout, and Depth

After creating a relation map, you need to specify criteria, layout, and depth for it.

To change the relation map properties

- Right-click the diagram pane and from shortcut menu, select **Specification**, or click on the Options toolbar and then select **Specification**. Change the property values in the Specification window of the relation map.
- Specify the most popular property values by using the relation map environment capabilities, such as toolbars and the **Criteria** area.

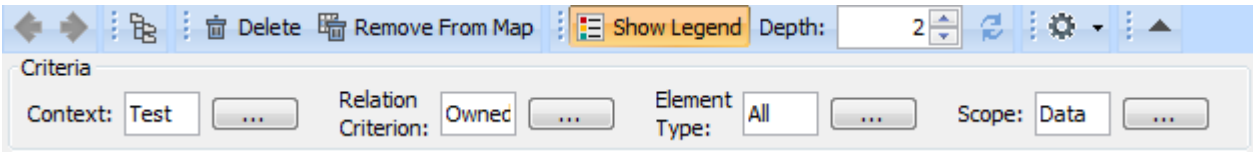
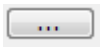
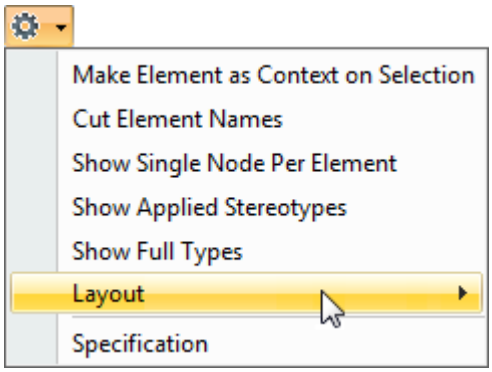
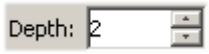


Figure 336 -- Relation Map toolbars and Criteria area

Element Name	Element Type	Description
Context	Text box with ...	Click to select the main element from which the structure is started, that is, the context of the structure. For more information, see "Selecting and changing the context" on page 547.
Relation Criterion	Text box with ...	Click to select relation criteria that will be displayed on the relation map. For more information, see "Specifying Criteria for Querying Model" on page 551.
Element Type	Text box with ...	Click to select element types that will be shown on the relation map.

Copyright © 1998-2015 No Magic, Inc.

Element Name	Element Type	Description
Scope	Text box with 	Click to select packages from which the relation map structure will be shown.
	Command on the Options button menu	Select a the relation map layout: • **Tree** – see the example in Figure 334 on page 544. • **Radial** – see the example in Figure 335 on page 545.
Depth 	Spin box	Type or select the level of the relation map branches that will be automatically expanded.

Selecting and changing the context

Context is the element, from which a relation map is drawn and the analysis is started. You can set any element as the context.

To set an element as the context of the relation map

- Drag the selected element from the Model Browser to the relation map.
- In the **Criteria** area, click the **...** button next to the **Context** box and then select the element in the open dialog.



For more information about the element Selection dialog, see "Selecting an Element" on page 352.

- If the **Make Element as Context on Selection** option is turned on, click any node to make it the new context of the of the structure. To turn this option on, click  on the toolbars and select **Make Element as Context on Selection**.

Specifying Other Options

Cutting element names

The element names can be cut if they are too long, which makes the relation map difficult to read.

To cut element names

- Click  on the toolbars and then select **Cut Element Names**.



To see the entire name of an element, move the pointer over the element name. A ToolTip showing the entire name appears.

Copyright © 1998-2015 No Magic, Inc.

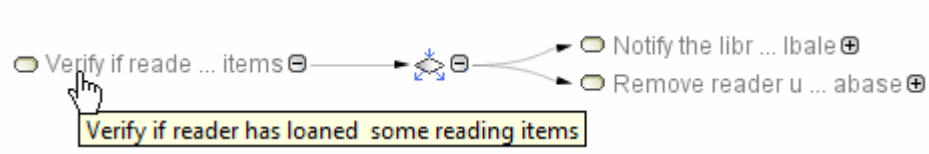
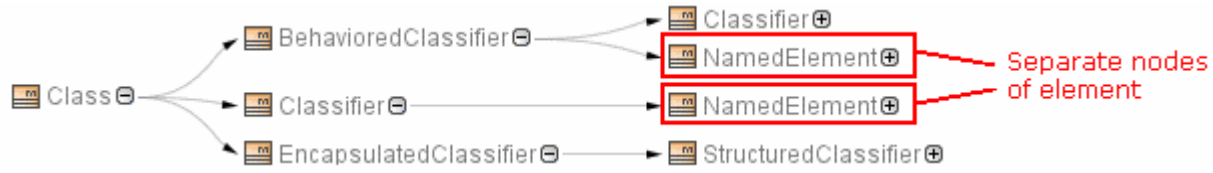


Figure 337 -- Example of ToolTip showing element's entire name

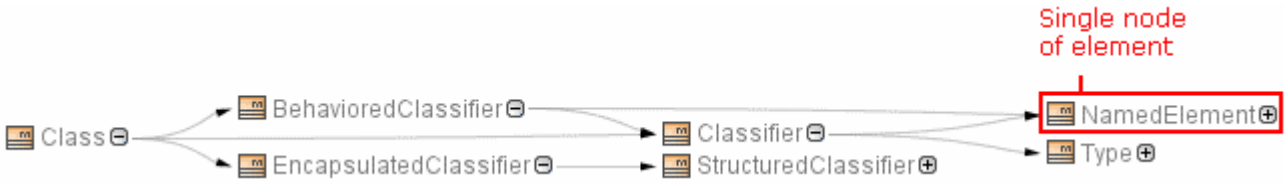
Showing single node per element

There are two ways of element node representation in relation map:

- Turn the Show Single Node Per Element mode off, if you need to see separate nodes of the same element representing every occurrence of the element in different branches of the graph.



- Turn the Show Single Node Per Element mode on, if you need to see only one node of the element representing all occurrence of the element in the whole graph.



Colors and Legend in Relation Map

You can specify colors to represent different relation criteria in a relation map (see the following figure). Select the colors for each relation criteria in the **Relation Criterion** dialog when specifying the criteria.

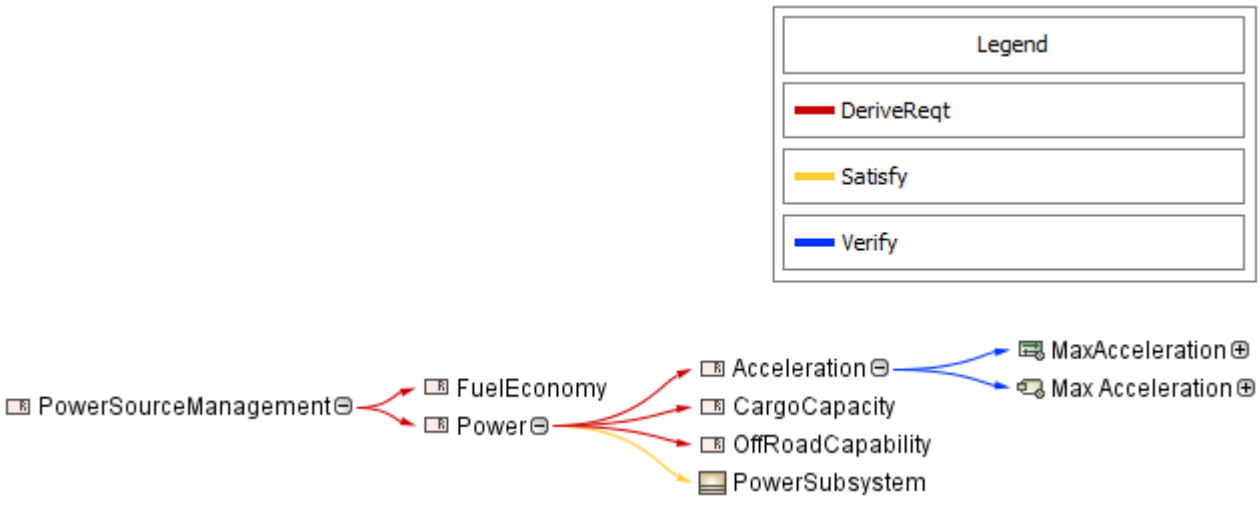


Figure 338 -- Using colors for performing coverage analysis

Copyright © 1998-2015 No Magic, Inc.

The legend on the diagram pane will help you identify relation criteria by colors. Double-click the legend to open the **Relation Criterion** dialog and change the selected colors or even the selected relation criteria.

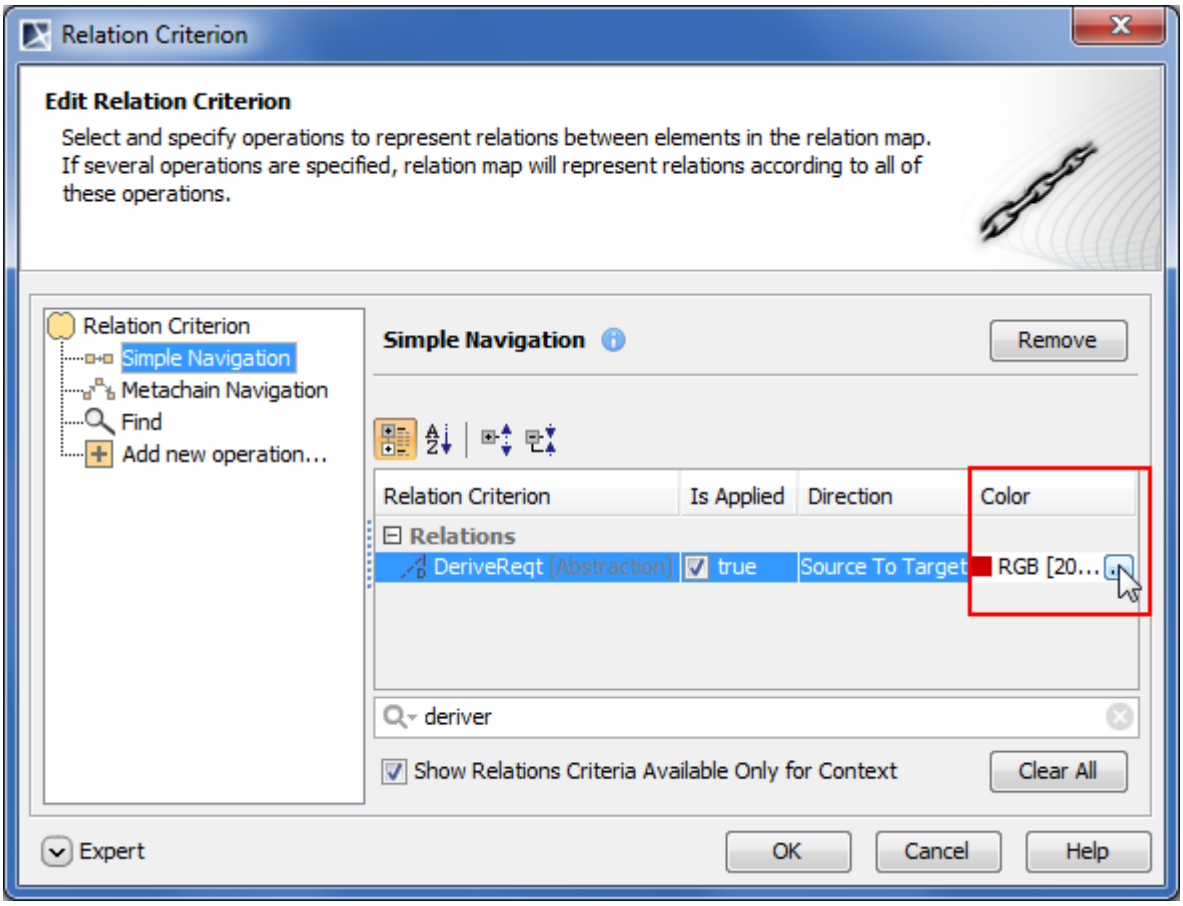


Figure 339 -- Changing color of selected relation criterion

TIP!

For more information about using the **Relation Criterion** dialog, see "Specifying Criteria for Querying Model" on page 551.

The legend is displayed by default, but you can hide it.

To hide the legend

- Click Show Legend on the View toolbar.

Manipulations in Relation Map

You can manipulate and analyze your relation map diagram in the following ways:

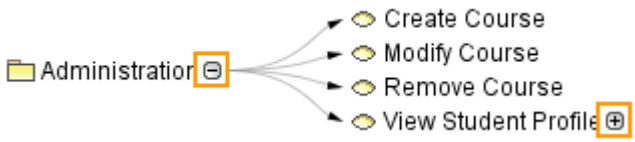
- Expand/suppress branches according to the current filters configuration.
- Restore the layout of manual suppressed/expanded branches, moved, removed and hidden symbols.
- See invalid elements marked by on demand or active validation (for more information about validation, see "Validation" on page 612 and "Active Validation" on page 632).
- Move the whole structure.
- Move the selected Node.
- Zoom in/ zoom out.

Copyright © 1998-2015 No Magic, Inc.

- Select an element in the Containment tree.
- Open all diagrams containing the selected element.
- Remove an element from the model.
- Hide an element in the diagram.
- **NEW!** Create elements directly in the Relation Map.

To expand/suppress branches

- Click the smart manipulator near the Node.



To move the relation map structure

- Click on an empty place in the Relation Map and drag it. The whole structure will be moved.
- Click a Node and drag it. The selected Node will be moved.

To zoom in or zoom out the relation map structure

- Press Ctrl + mouse wheel scroll up to zoom in.
- Press Ctrl + mouse wheel scroll down to zoom out.

TIP!

The traditional Zoom In, Zoom Out, Fit in Window, Zoom 1:1 actions are available. For more information on zooming, see Section “Zooming” on page 254.

To remove/hide an existing element

- Element removing and hiding will be initiated and work in same way as for other diagrams. Press Delete, to hide an element. Press Ctrl+D, to delete and element from the model.

To restore the layout of the manually suppressed/expanded branches, moved, removed and hidden symbols

- Click  on the View toolbar.

NOTE

If the **Preserve manually suppressed/expanded branches and hidden elements** option in the **Environment Options** dialog > **Diagrams** > **Relation Map**, is selected, the layout of moved elements will be restored.

To see invalid elements marked by on demand or active validation

1. Run validation against model. On the **Analyze** menu, click **Validation** > **Validation**.
2. In the **Validation** dialog, select **Validation Suite** and click **Validate**.
3. Invalid elements will be marked on the Relation Map.

To select an element in the Containment tree

- Select the element and click  on the Navigation toolbar.
- On the selected element shortcut menu, click **Select in Containment Tree**.

Copyright © 1998-2015 No Magic, Inc.

NEW! To create an element

1. Select an element that you want to decompose.
2. From the selected element shortcut menu, select **Create Related Element**.
3. Select element type and relationship.

For quick and easy element creation, use shortcut keys.

NEW! To decompose an element

1. Select an element.
2. Click  or press Insert (Cmd + I for Mac users).

NEW! To create new element of the same type

1. Type element name.
2. Press Ctrl+Enter while in the name edit mode.

Specifying Criteria for Querying Model

Specifying criteria is necessary to

- Associate row and column elements in a dependency matrix.
- Represent relations between elements in a relation map.
- Gather the contents of a smart package.
- Calculate/gather the value of a derived property.

A structured expression can also be the body of the executable opaque behavior, metric definition or validation rule, when the StructuredExpression language is selected for specifying that body.

The criteria can be as simple as a UML relationship or as complex as an OCL 2.0 expression.

To specify the criteria, you can use one of the following operation types:

- Simple navigation
- Metachain navigation (indirect relations through chains of properties)
- Find
- Union
- Exclude
- Filter
- Various scripts

Also, you can use custom operation types. Moreover, a criterion can be an expression, which combines several operations by passing the results of one operation call to another operation.

If multiple criteria (operations) are specified, the result will be calculated according to all these criteria (operations).

Concepts

For better understanding the further material, first of all read the following description.

Copyright © 1998-2015 No Magic, Inc.

Contextual element

The element, from which begins the calculation of the result, according to the specified criteria. In a dependency matrix, the context is row or column element. In a relation map, this is every node of the structure. A smart package is the context itself, and the context of a derived property is the target of the customization element that owns this derived property.

Getting Started with Specifying Criteria

The dialog for specifying criteria can be opened from several different places.

To open the dialog for specifying criteria

Do either:

- In the **Criteria** area of a dependency matrix, click the **...** button next to the **Dependency Criteria** box. The **Dependency Criteria** dialog opens.



Learn more about dependency matrices in "Dependency Matrix" on page 764.

- In the **Filters** area of a relation map, click the **...** button next to the **Relation Criterion** box. The **Relation Criterion** dialog opens.



Learn more about relation maps in "Relation Map" on page 544.

- Open the Specification window of a smart package, under the **Content** category on the right, click the cell of the **Query** property value, and then click . The **Query** dialog opens.



Learn more about smart packages in "Smart Packages" on page 402.

- Open the Specification window of a selected element, click **Traceability** on the left, and then click **Create** on the right. The **Expression** dialog opens.



Learn more about derived properties in "Derived Properties" on page 537.

- On the toolbar of a generic table, click the **Show Columns** button, and then select **New Derived Property**. The **Expression** dialog opens.



Learn more about generic tables in "Generic Table" on page 785.

- Open the Specification window of an opaque behavior, click the cell of the **Body and Language** property value, and then click . When the **Body and Language** dialog opens, select **StructuredExpression** from the **Language** drop-down list.



Learn more about executable opaque behaviors in "Creating executable opaque behaviors" page 952.

Copyright © 1998-2015 No Magic, Inc.

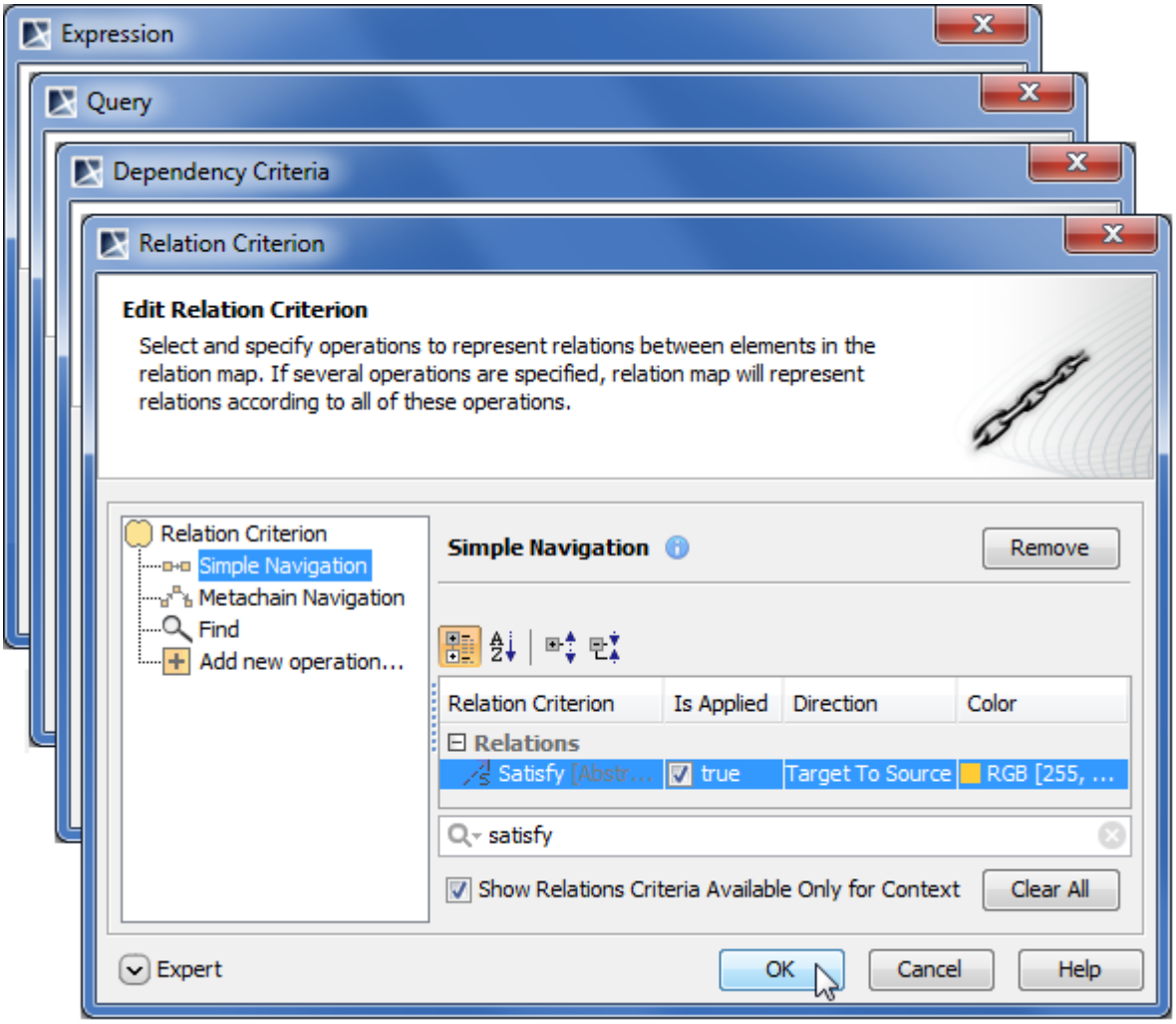


Figure 340 -- Variety of dialogs for specifying criteria

The layout of the dialog is the same in all the described cases. On the left side of the dialog, you can select an operation type that best suits for specifying the needed criterion. Selecting an operation type opens the criteria specification panel on the right side of the dialog.

The dialog has two modes: Standard and Expert. *The Expert mode offers more options for specifying criteria and allows for creating combinations of several operations.*

To switch to the Expert mode for specifying criteria, and back

- Click **Expert** at the bottom of the dialog. The dialog is switched to the Expert mode. Click **Standard** to get back to the Standard mode.

Copyright © 1998-2015 No Magic, Inc.

Also, be advised that before starting to specify criteria it is useful to read the tips and warnings, which appear after the pointer is over the button or the button appropriately.

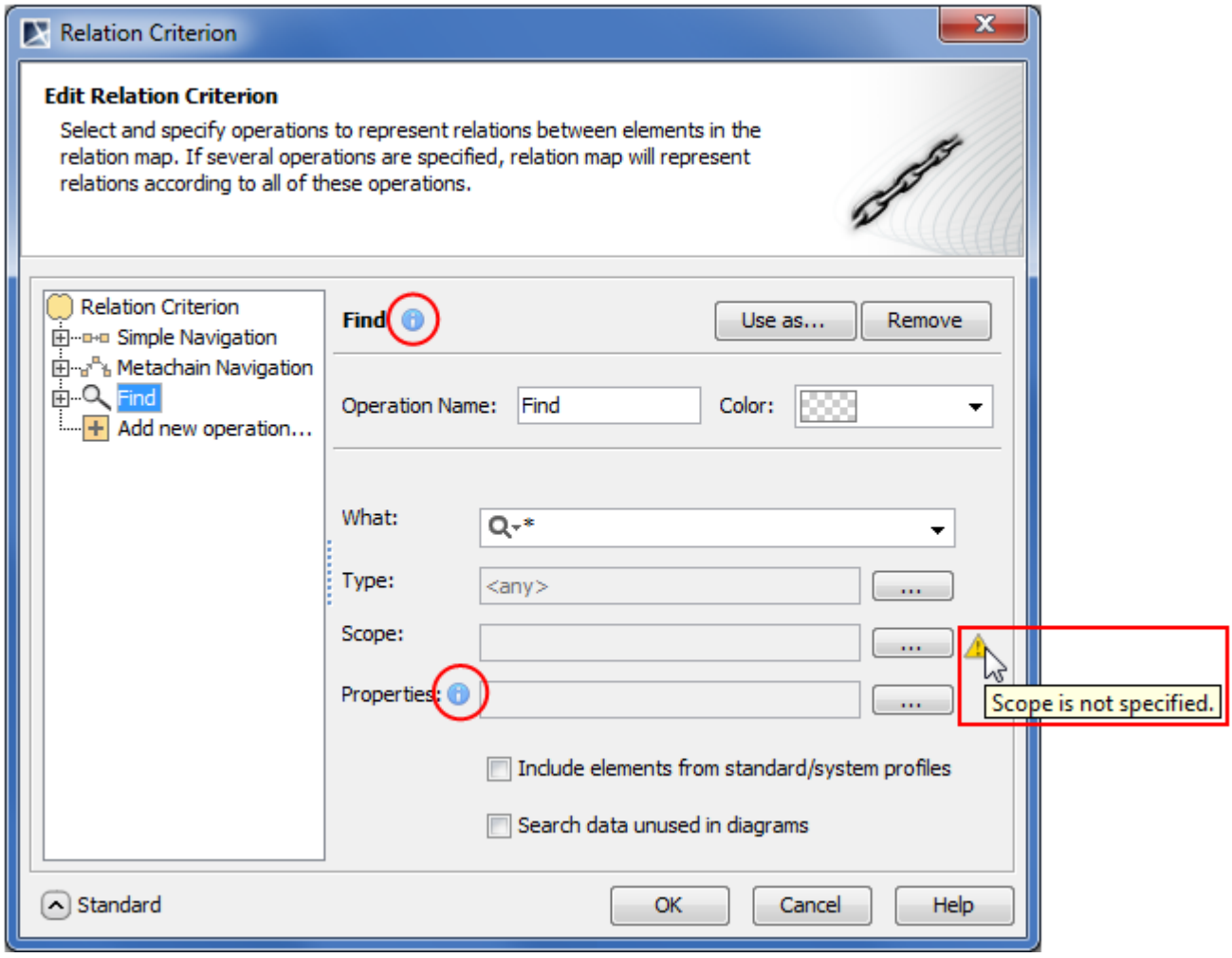


Figure 341 -- Tips and warnings helping to specify find options in Relation Criterion dialog

Learn more about specifying the criteria in the following sections:

- "Using Simple Navigation Operations to Specify Criteria" on page 554
- "Using Metachain Navigation Operations to Specify Criteria" on page 557
- "Using Find Operation to Specify Criteria" on page 560
- "Creating New Operations" on page 562

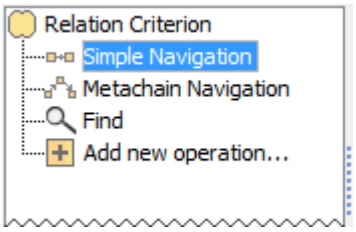
Using Simple Navigation Operations to Specify Criteria

Use the simple navigation operation type, if you need to specify a direct relation between the elements through model relationships, properties, or tags.

Copyright © 1998-2015 No Magic, Inc.

To specify a simple navigation as criterion

1. Click **Simple Navigation** on the left side of the dialog.



2. Select a relationship, property, or tag and specify other options described in the following table.

Column	Mode	Description
Relation Criterion	Standard/ Expert	Name of a relationship, property, or tag. The whole list is by default filtered and shows only these relationships, properties, and tags that are available for the Contextual element type. To see the full list, click to clear the check box below the list.
Is Applied	Standard/ Expert	Select the check box to define the appropriate relationship, property, or tag as criteria for the calculation. To clear all the selected criteria, click the **Clear All** button.

Direction	Standard/ Expert	Select from the drop-down list the direction of the appropriate relationship, property, or tag.
		A source is the Contextual element.
		A target is the result of the calculation.
		
		If the direction is defined for relationships, then
		• The **Source To Target** direction means that only the outgoing relations, which are pointing from the source element to the target element, will be treated as a result of this criterion.
		• The **Target To Source** direction means that only the incoming relations, which are pointing from the target element to the source element, will be treated as a result of this criterion.
		• If the **Any** direction is chosen, the both above described cases will be treated as a valid result.
		If the direction is defined for properties or tags, then
		• The **Source To Target** direction means that only the properties, which exist in the source element, will be treated as a result of this criterion.
		• The **Target To Source** direction means that only the properties, which exist in the target element, will be treated as a result of this criterion.
		• If the **Any** direction is chosen, the both above described cases will be treated as a valid result.

Include Subtypes	Expert	Select the check box to include subtypes of the selected relationship.
Filter by Property Value	Expert	Click the button and in the open dialog select one or more properties of the selected relationship and specify their values for more specific filter.

Copyright © 1998-2015 No Magic, Inc.

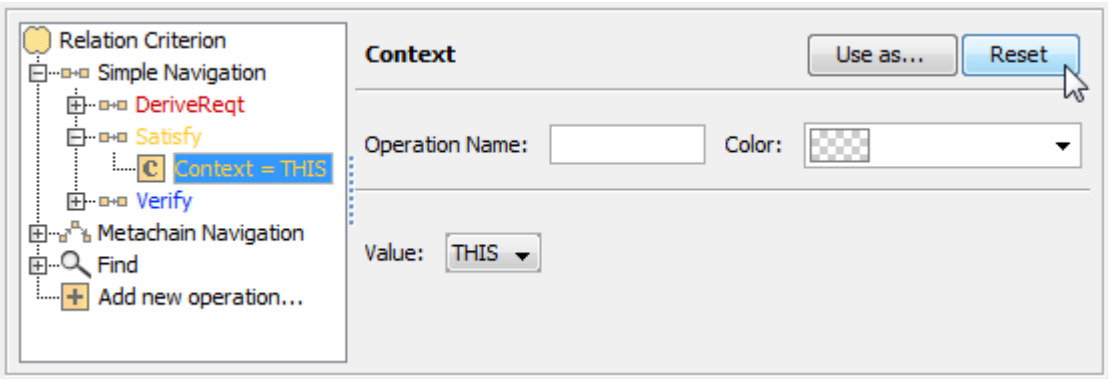
Column	Mode	Description
Result Type	Expert	Click the ... button and in the open dialog select one or more result element types. The column is available only in the **Query** dialog and the **Expression** dialog. How to define element types for representation on a dependency matrix, see "To specify a row/column element type" on page 766. How to define the element types for representation on a relation map, see the description of the Element Type filter in "Specifying Relation Map Criteria, Layout, and Depth" on page 546.
Color	Standard/ Expert	Click the ... button and in the open dialog select the color to represent the appropriate relationship, property, or tag. The column is available only in the **Relation Criterion** dialog.

Manipulations in the Expert mode

The Expert mode allows for changing the Contextual element of a simple navigation criterion as well as creating a new operation, where the result of the simple navigation operation is used as an argument.

To change the Contextual element of a simple navigation criterion

1. On the left side of the dialog, expand the simple navigation criterion and select the Context parameter.
2. On the **Context** specification panel, click the **Reset** button.

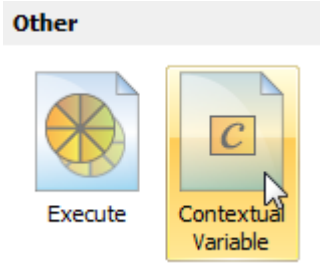


The value of the Context parameter becomes null.

3. Select a new context from the list of operations and element types.



To restore the original context, expand the **Other** group and click **Contextual Variable**.

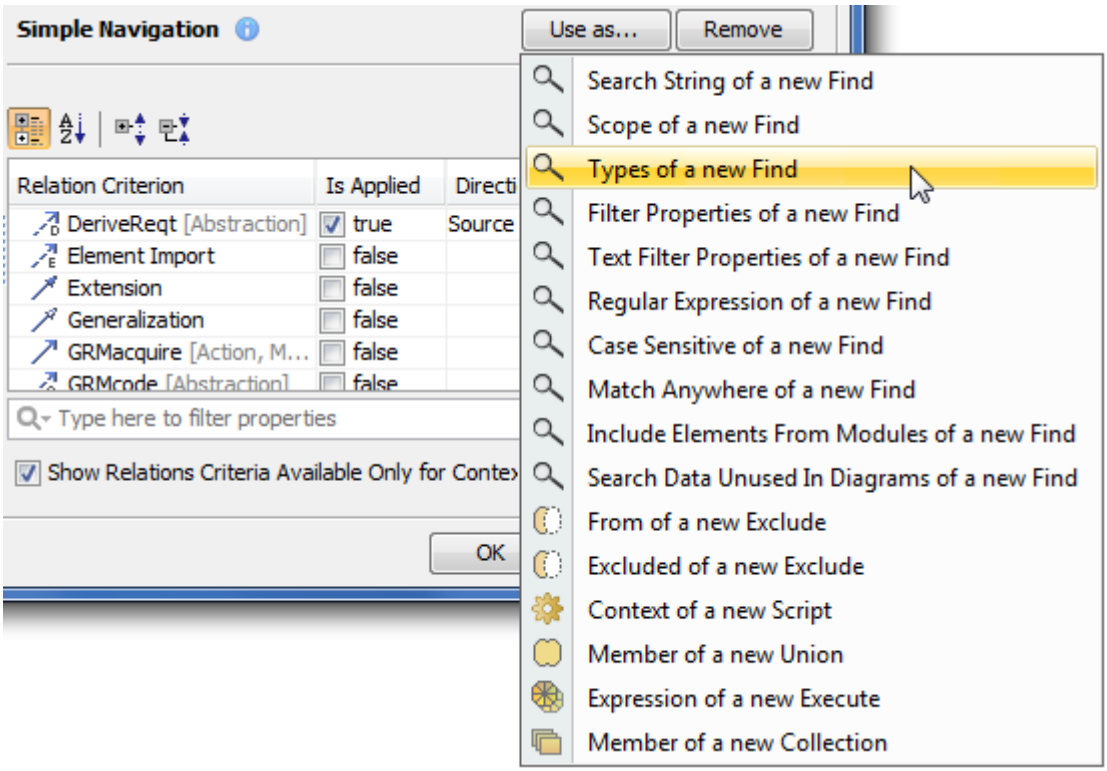


To use the expression of a simple navigation operation as a parameter of another operation

1. On the left side of the dialog, click **Simple Navigation**.

Copyright © 1998-2015 No Magic, Inc.

2. On the criteria specification panel, click the **Use as** button and select the appropriate command.



3. Specify other parameters of the new operation.

Using Metachain Navigation Operations to Specify Criteria

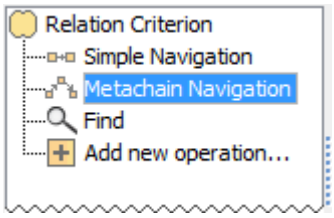
Use the metachain navigation operation type, if you need to specify indirect (multilevel) relations between elements through the chains of properties.



Metachains allow searching for the indirectly related elements, thought they do not support loops and recursive relations.

To specify a metachain navigation as criterion

1. Click **Metachain Navigation** on the left side of the dialog.



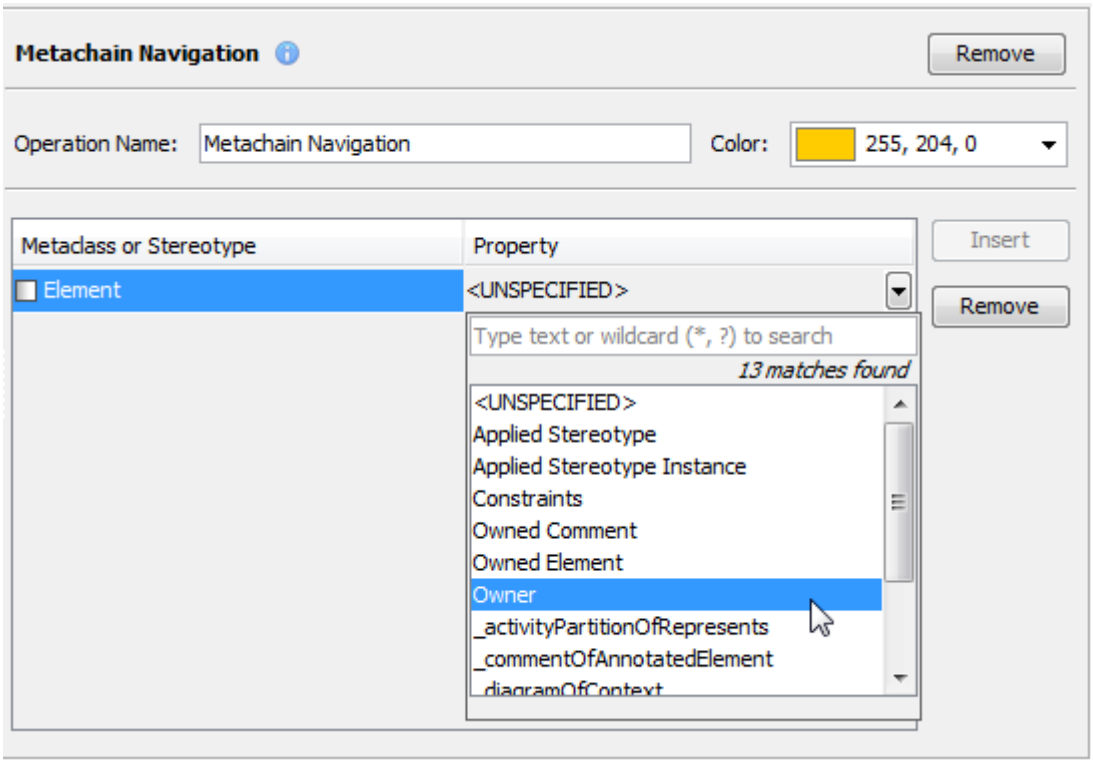
2. Change the name of the operation, if there is a need.
3. Select a color to represent the specified metachain navigation criteria.



The color selection is available only in the **Relation Criterion** dialog.

Copyright © 1998-2015 No Magic, Inc.

4. Click the **Insert** button to create a new step of the metachain. A new row is added to the list, which is described in the following table.



5. Repeat the previous step to create as many steps of the metachain as you need for specifying the criteria.

NOTE To remove a step from the metachain, click the **Remove** button below the **Insert** button.

Column	Description
Metaclass or Stereotype	Either a metaclass or a stereotype of a metachain step. The first step, when created, already has a metaclass or a stereotype selected by default. This value is the context element type. The default metaclass or stereotype of the first step can be changed. The second step and all further steps are more specific: the values suggested for selection are limited according to the values selected in the first step, since you will not be able to add a new step until the current step is not specified.
Property	A property of the selected metachain or stereotype.

TIP! How to add a new metachain operation, see "Creating New Operations" on page 562.

Manipulations in the Expert mode

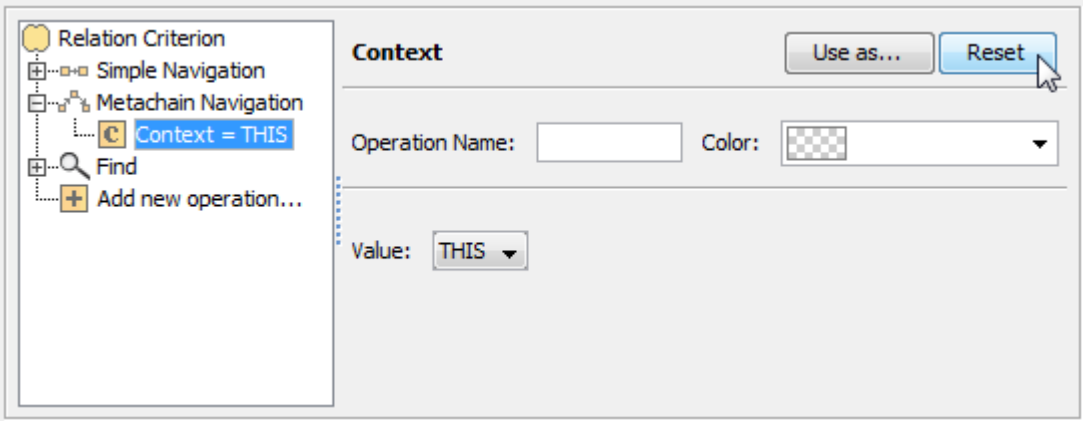
The Expert mode allows for changing the Contextual element of a metachain navigation operation as well as creating a new operation, where the result of the metachain navigation operation is used as an argument.

To change the Contextual element of a metachain navigation criterion

1. On the left side of the dialog, expand the metachain navigation criterion and select the Context parameter.

Copyright © 1998-2015 No Magic, Inc.

2. On the **Context** specification panel, click the **Reset** button.

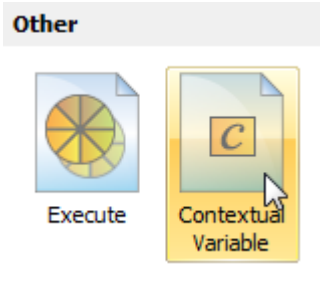


The value of the Context parameter becomes null.

3. Select a new context from the list of operations and element types.



To restore the original context, expand the **Other** group and click **Contextual Variable**.

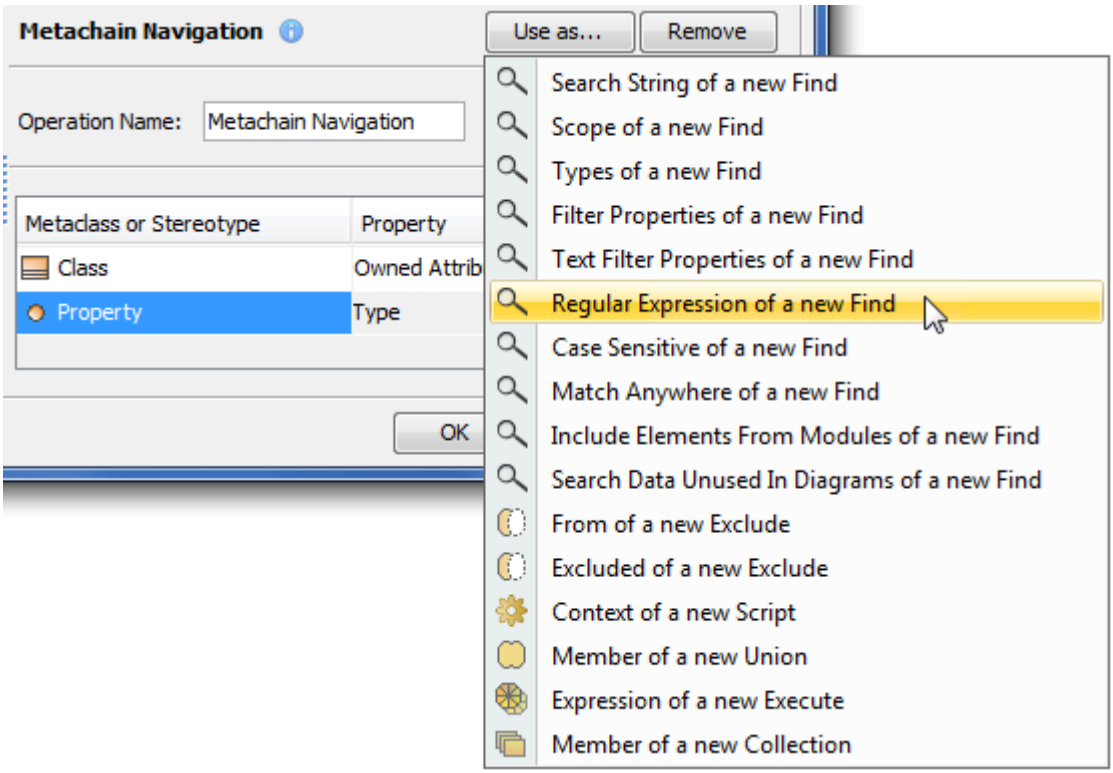


To use the expression of a metachain navigation operation as a parameter of another operation

1. On the left side of the dialog, click **Metachain Navigation**.

Copyright © 1998-2015 No Magic, Inc.

2. On the criteria specification panel, click the **Use as** button and select the appropriate command.



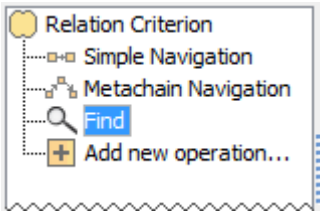
3. Specify other parameters of the new operation.

Using Find Operation to Specify Criteria

Use the find operation type, if you need to search for elements by name, documentation contents, property or tag value, and applied stereotype name. The find operation works similarly to the MagicDraw search, when the search parameters are defined in the **Find** dialog.

To specify a find operation as criterion

1. Click **Find** on the left side of the dialog.



2. Change the name of the operation, if needed.
3. Select a color to represent the specified find criteria.



The color selection is available only in the **Relation Criterion** dialog.

4. Specify find options.



For the information about specifying the find options, refer to "Find dialog" on page 132.

Copyright © 1998-2015 No Magic, Inc.



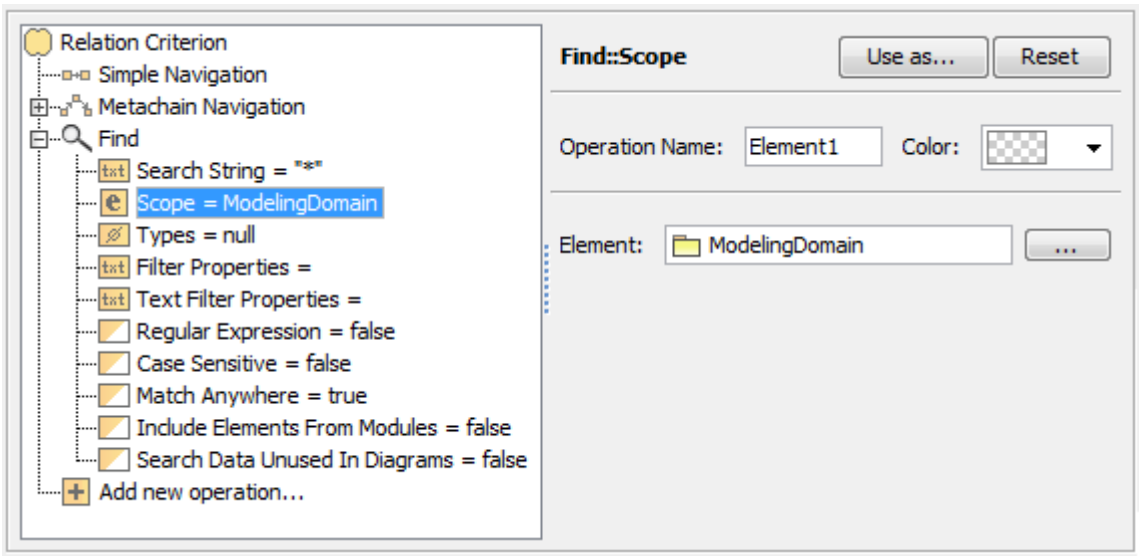
How to add a new find operation, see "Creating New Operations" on page 562.

Manipulations in the Expert mode

The Expert mode allows for changing the options of a find operation (including the Contextual element) as well as creating a new operation, where the result of the find operation is used as an argument.

To specify options (parameters) of a find operation

1. On the left side of the dialog, expand the find operation.
2. Select the option (parameter) you need to specify and on the option specification panel, select the value from the list of operations and element types.

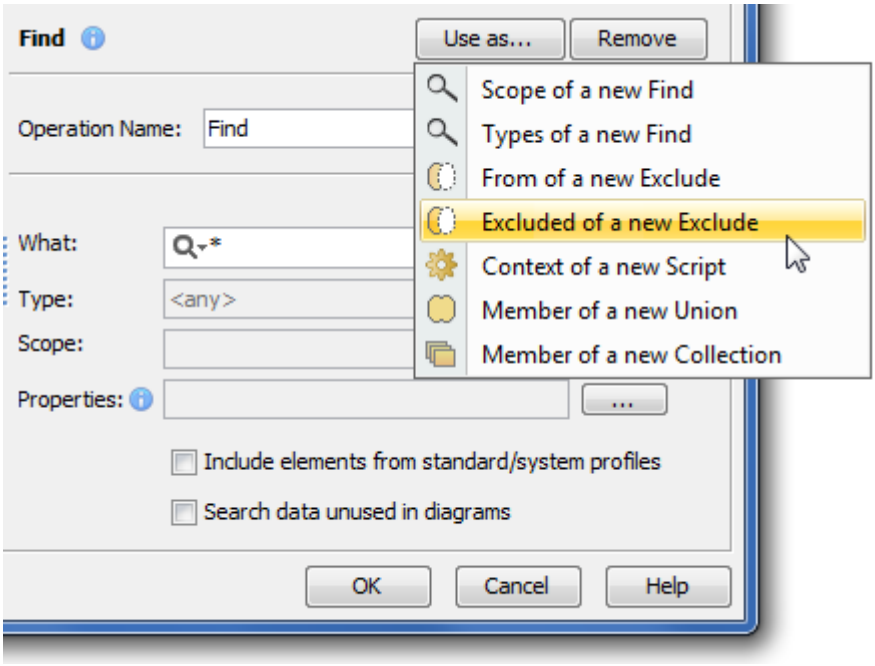


To use the expression of a find operation as a parameter of another operation

1. On the left side of the dialog, click **Find**.

Copyright © 1998-2015 No Magic, Inc.

2. On the criteria specification panel, click the **Use as** button and select the appropriate command.

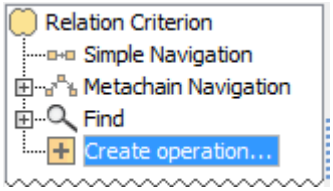


3. Specify other parameters of the new operation.

Creating New Operations

To create a new operation

1. Click **Create operation** on the left side of the dialog.

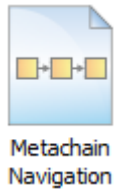
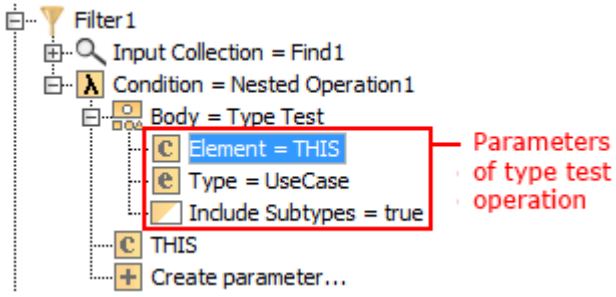


2. Click the appropriate icon on the right side of the dialog to select a new operation type (see the descriptions in following table).

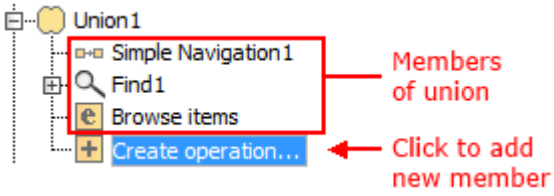
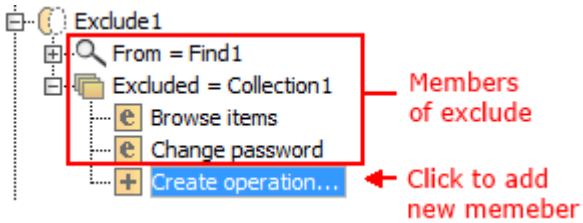
NOTE If the Standard mode is on, switch to the Expert mode to make more operation types available.

Icon	Mode	Descriptions
Operations group		
 Simple Navigation	Expert	Click to create a new simple navigation operation. This icon is available only when specifying members of a union or exclude operation, input collection for a filter operation, scope and type parameters for the find operation. For more information, refer to "Using Simple Navigation Operations to Specify Criteria" on page 554.

Copyright © 1998-2015 No Magic, Inc.

Icon	Mode	Descriptions
 Metachain Navigation	Standard/ Expert	Click to create a new metachain navigation operation. For more information, refer to "Using Metachain Navigation Operations to Specify Criteria" on page 557.
 Find	Standard/ Expert	Click to create a new find operation. For more information, refer to "Using Find Operation to Specify Criteria" on page 560.
 Filter	Expert	Click to create a new filter operation and then specify both the input collection that you need to filter and the filter criteria. For more information, refer to the description of the Filter built-in operation.
 Type Test	Expert	Click to specify a new type test operation as a condition for selected filter operation. In other cases, the icon is not available. The operation tests, whether the type of the element matches the given type or stereotype. If the types matches, it returns *true*, and if they not – *false*. You can also use this operation to check, if the element is an instance of the given classifier. For this operation, you must specify the following parameters: • **Element** – a model element, which type you need to test. • **Type** – a type, stereotype, or classifier for testing the element. • **Include Subtypes** – *true*, if the inherited types, stereotypes, or classifiers of the selected Type parameter value should be included in the test; *false*, if not. 
 Nested Operation	Expert	Click to specify a new nested operation. Nested operation is an operation, whose body is a structured expression. It is inline equivalent to defining an executable operation in the model as an Opaque Behavior with the StructuredExpression language and then using it. Nested operation can be used as an argument to higher order operation calls, such as Filter. In simple operation calls it is not available.

Copyright © 1998-2015 No Magic, Inc.

Icon	Mode	Descriptions
 Union	Expert	Click to create a new union operation and then specify as many members for it as you need. 
 Exclude	Expert	Click to create a new exclude operation and then specify both **From** and **Excluded** members for it. 
 Operation from Model	Expert	Click to add a new operation from the model. You can select either one of the built-in operations, which are stored in standard/system profiles of your project, or a custom operation. For more information, refer to "Calling operations from the model" on page 565.
 Script	Expert	Click to create a new script operation, which language can be one of the following: • BeanShell • Groovy • Jython • JRuby • JavaScript • OCL 2.0 It can also be a reference to a java class (Binary). For more information, refer to "Creating script operations" on page 570.
Values group		
 Element	Expert	Click to select an element from the model.
 String	Expert	Click to create a String value.

Copyright © 1998-2015 No Magic, Inc.

Icon	Mode	Descriptions
 Boolean	Expert	Click to create a Boolean value.
 Integer	Expert	Click to create a Integer value.
 Null	Expert	Click to create a Null value.
 Collection	Expert	Click to create a collection of values.
Other group		
 Execute	Expert	Click to add an operation executing a specified expression. Execute operation takes the supplied expression fragment and grafts it into the current expression tree for executing. Expression fragment is an XML string that can be dynamically fetched from any source, such as another tag. The unique value of this operation is that the specified expression can be calculated dynamically instead of being fixed. As a result, the calculation to be executed can depend on the Contextual element.
 Contextual Variable	Expert	Click to add an expression retrieving the contextual variable specified by name. Most common case is accessing the *THIS* variable, which stores the current Contextual element. This is usually the starting point – argument for other operations – such as Simple Navigation or Metachain Navigation.

Calling operations from the model

In a structured expression, you can call an operation defined in the model. An operation from the model can be either built-in or custom. The built-in operations, such as Union, Find, or Filter, are stored as executable opaque behaviors in the *Built-In Operations* package, which is owned by *UML Standard Profile::MagicDraw Profile*.

Copyright © 1998-2015 No Magic, Inc.

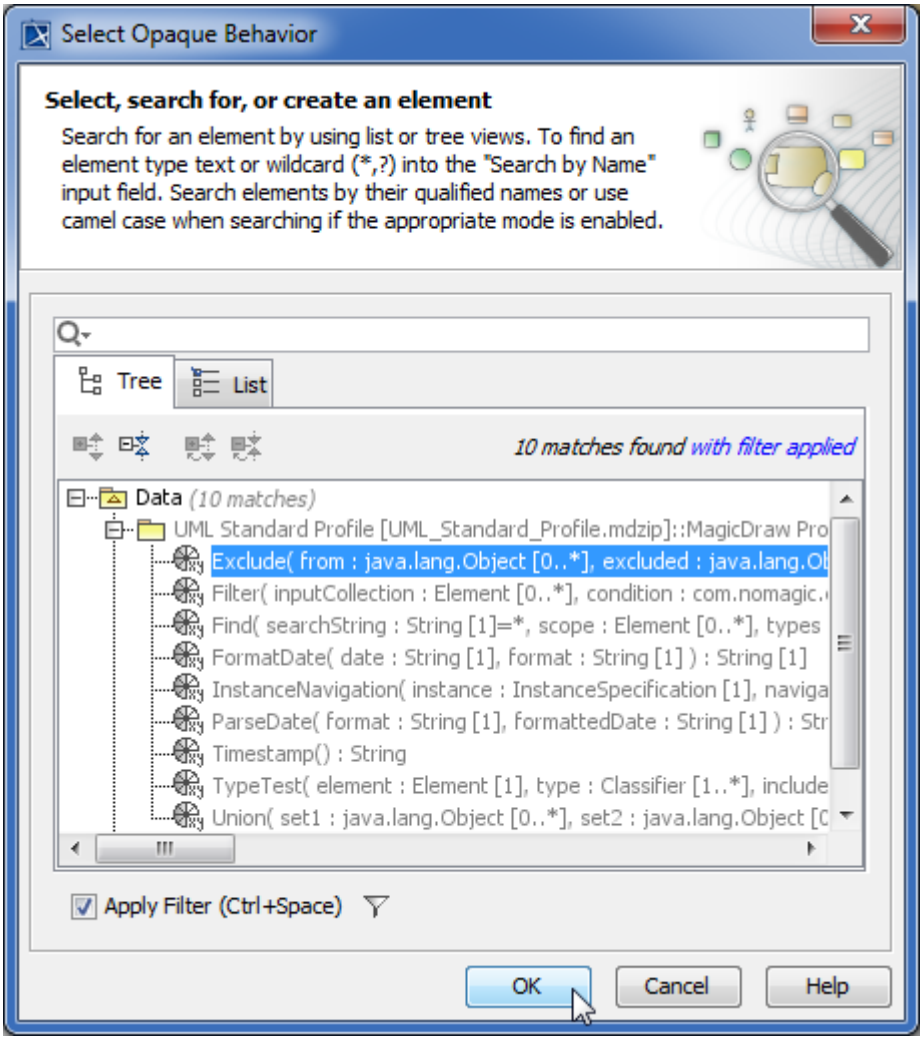


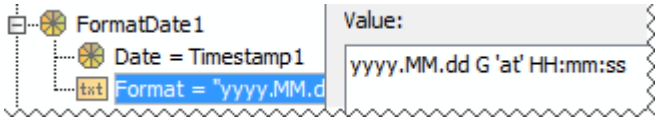
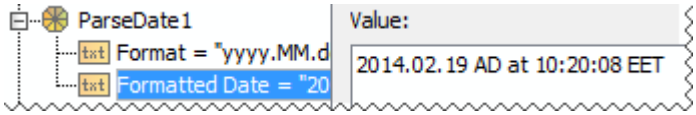
Figure 342 -- Built-in operation types. Selecting to create new Exclude operation

IMPORTANT!

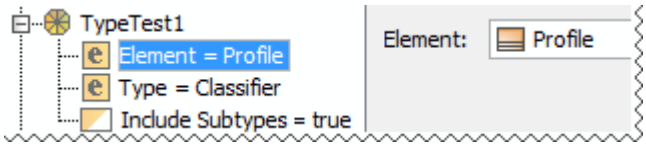
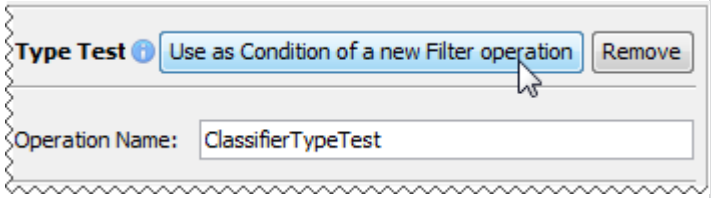
Be sure the **Exclude Used Projects** command on the Filter Options button menu is unchecked!

Copyright © 1998-2015 No Magic, Inc.

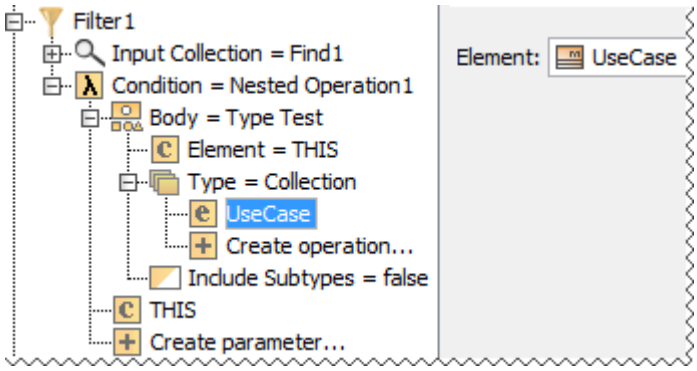
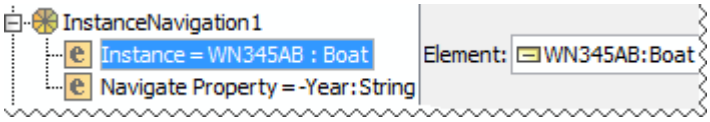
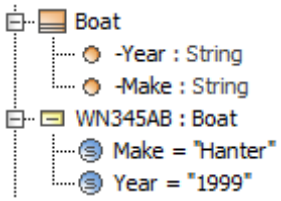
Built-in operations

Operation name	Description	Example
Timestamp	The operation returns current system time in milliseconds as a string type value. It takes no parameters.	Result = “1392798008000”
FormatDate	The operation converts a date and time given in milliseconds to a human readable format. The operation takes two parameters: • **Date** – a date and time in milliseconds that should be converted to a human readable format. It must be a string type value and can be the result of a Timestamp operation. • **Format** – a date and time format for the conversion. It must be a string type value. For the date and time formats, refer to the SimpleDateFormat page.  The result of this operation is a string type value.	Date = “1392798008000” Format = “yyyy.MM.dd G 'at' HH:mm:ss z” Result = “2014.02.19 AD at 10:20:08 EET”
ParseDate	The operation converts a date and time in a human readable format to milliseconds. In other words, the operation reverses the result of the FormatDate operation (it returns the value that can be the Date parameter for a FormatDate operation). The operation takes two parameters: • **Format** – a current format of the date and time that should be converted to milliseconds. It must be a string type value. • **Formatted Date** – a date and time that should be converted to milliseconds. It must be a string type value.  The result of this operation is a string type value.	Format = “yyyy.MM.dd G 'at' HH:mm:ss z” Formatted Date = “2014.02.19 AD at 10:20:08 EET” Result = “1392798008000”

Copyright © 1998-2015 No Magic, Inc.

Operation name	Description	Example
TypeTest	The operation tests, whether the type of an element matches a given type or stereotype. If the types matches, it returns *true*, and if they not – *false*. You can also use this operation to check, if an element is an instance of a given classifier. The operation takes three parameters: • **Element** – a model element, whose type you need to test. • **Type** – a type, stereotype, or classifier for testing the element. • **Include Subtypes** – *true*, if the inherited types, stereotypes, or classifiers of the selected Type parameter value should be included in the test; *false*, if not.  The TypeTest operation can be used in the Filter operation as the condition. For this, click the **Use as Condition of a new Filter operation** button on the Type Test operation specification panel. 	Element = class *Profile* Type = Class Include Subtypes = *false* Result = *true* The element type matches the given type. Element = class *Profile* Type = Classifier Include Subtypes = *true* Result = *true* Though the element type does not match the given type, it is its subtype, and in this case subtypes are included in the test. Element = class *Profile* Type = Classifier Include Subtypes = *false* Result = *false* Thought the element type is a subtype of the given type, in this case subtypes are not included in the test.

Copyright © 1998-2015 No Magic, Inc.

Operation name	Description	Example
Filter	The operation analyses given collection of elements (one by one) and returns only those elements that meet the specified filter criteria. The operation takes two parameters: • **Input Collection** – an arbitrary collection of elements of arbitrary types. • **Condition** – an operation determining filter criteria. Keep in mind that *this operation must have exactly one input parameter and return a result of boolean type!* Otherwise, the Filter operation fails and returns no value. Hence, if you need to use the script operation with many parameters as the filter criteria, you must wrap it inside the nested operation (the TypeTest operation or any other built-in operation is wrapped automatically).  The result of this operation is collection of elements as well.	Input Collection = use case *Log in* use case *Log out* actor *User* use case *Change password* Condition = TypeTest: Element = Input Collection Type = UseCase Include Subtypes = *false* Result Collection = use case *Log in* use case *Log out* use case *Change password*
InstanceNavigation	The operation finds the needed slot of the given instance and returns one or more values of that slot. The operation takes two parameters: • **Instance** – an instance specification (instance), whose slot value you need to access. • **Navigate Property** – a property, whose slot you need to find.  The result type of this operation matches the type of the property specified as the Navigate Property parameter.	Instance = *WN345AB* Navigate Property = *Year* Result = “1999” 

Creating custom operations

You can also create custom operations. As well as the built-in ones, they can be defined in the model as executable opaque behaviors. Parameters of the operation can be declared as parameters of the opaque behavior.

Copyright © 1998-2015 No Magic, Inc.

To create a custom operation

1. Create a package.



We recommend storing the custom operations in a separate package for the following purposes:

- To keep your model well-formed.
- To reuse the operations in different projects.

2. Create an executable opaque behavior.



For more information, see "Creating executable opaque behaviors" on page 952.

Creating script operations

Use the script operation, when you need to specify an inline expression in OCL 2.0, as Java binary, or in any of MagicDraw supported script languages. MagicDraw supports any script language that has a JSR-223-compliant engine.

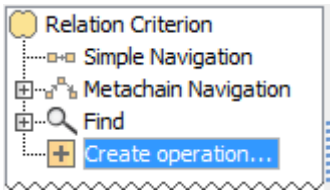
Adding a script operation directly into a structured expression is the same as defining an operation in the model and then calling it from the structured expression.



For more information, see "Calling operations from the model" on page 565.

To create a script operation

1. Click **Create operation** on the left side of the dialog.



2. Click the **Script** icon on the right side of the dialog.



If you do not see the **Script** icon, switch to the Expert mode.

3. If needed, change the operation name.
4. If you are specifying the criteria for displaying relations between elements in a *relation map*, you can *select a color*, which will represent these relations. Otherwise, the color selection is not available.
5. From the **Language** drop-down list, select a language.



If the engine of the language you need is not built into MagicDraw by default, you can add it to the application yourself.

Make sure the Automaton plugin is enabled!

For more information, refer to “3.5 Adding a Scripting Language to MagicDraw” in MagicDraw MacroEngine UserGuide.pdf.

6. Type in the **Body** box. Use the syntax of the selected language.

Copyright © 1998-2015 No Magic, Inc.

7. In the **Listeners Configuration** box, type a reference to the java class, which specifies when the result of the script operation must be recalculated.



The listeners configuration is not necessary, but using it can improve the performance.

For the instructions on how to create your own listener using MagicDraw API, please refer to “Diagram Events” in MagicDraw OpenAPI UserGuide.pdf.

8. Click the **...** button next to the **Results Filter By Type** box and select one or more element types to filter the result of the script operation.

The script operation is created. Now you can create a new one, where the result of the previously created operation is used as an argument.

Specifying parameters

The number of parameters a script operation can have depends on the script operation language:

- An OCL expression can have a single parameter, which by default specifies the Contextual element. Though the parameter cannot be removed, you can easily reset the Contextual element, if necessary.
- A binary expression inherits parameters from the Java class to which the expression body references. You cannot create additional parameters.
- Other script operations, such as JavaScript or Groovy, can have as many parameters as you need (see the following figure).

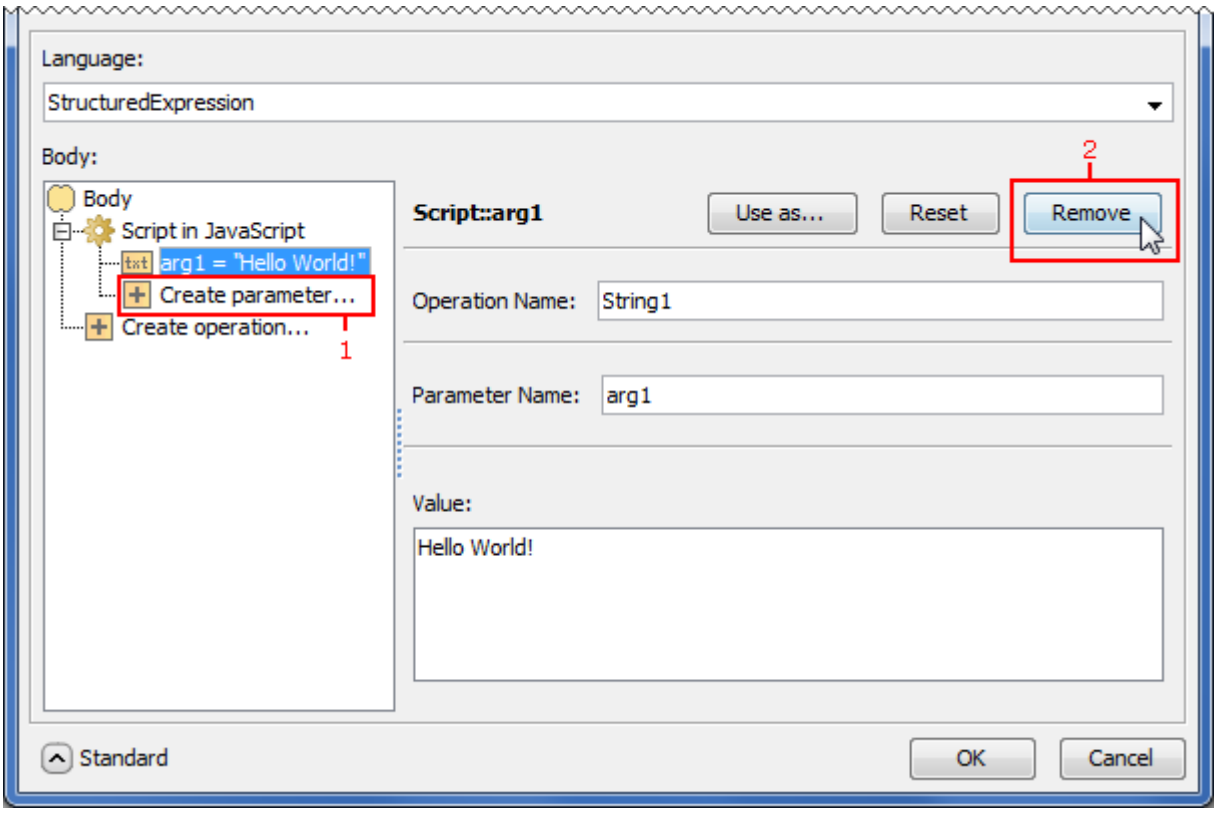


Figure 343 -- Means for managing parameters of JavaScript operation: 1 - creating; 2 - removing

Copyright © 1998-2015 No Magic, Inc.

Writing scripts

NOTE The following instructions applies to BeanShell, Groovy, JRuby, JavaScript, and Jython scripts only.

How to access the arguments and other values from the script body?

To access an argument from the script body, you should refer to the corresponding parameter name.

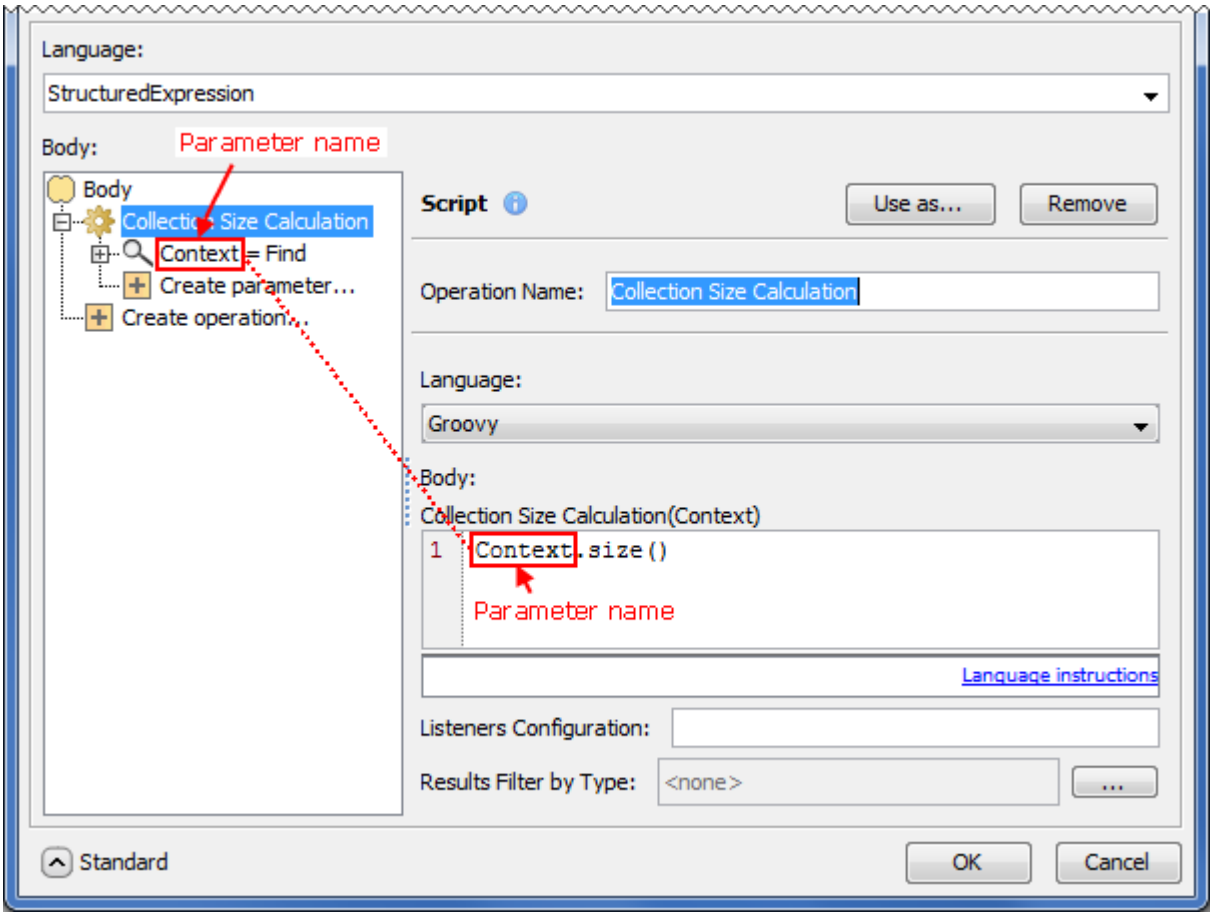


Figure 344 -- Accessing arguments from script body

A script body can access the following values:

- Arguments passed to this script as parameters, such as the Context parameter in the preceding figure.

Copyright © 1998-2015 No Magic, Inc.

- Values passed to the structured expression, inside of which the script operation is defined.

EXAMPLE

If you have an opaque behavior with a couple of parameters, you can refer to them from a script operation, which is inside of the structured expression - the body of this opaque behavior (see Figure 345 on page 573).

EXAMPLE

If you have a structured expression specifying a derived property, validation rule, relation criterion, etc., use *THIS* to refer to the Contextual element.

- Arguments passed to a nested operation as parameters, if the script operation is defined inside of that nested operation (see Figure 346 on page 574).
- Globally defined values:
  - *project* (current project)
  - *application*

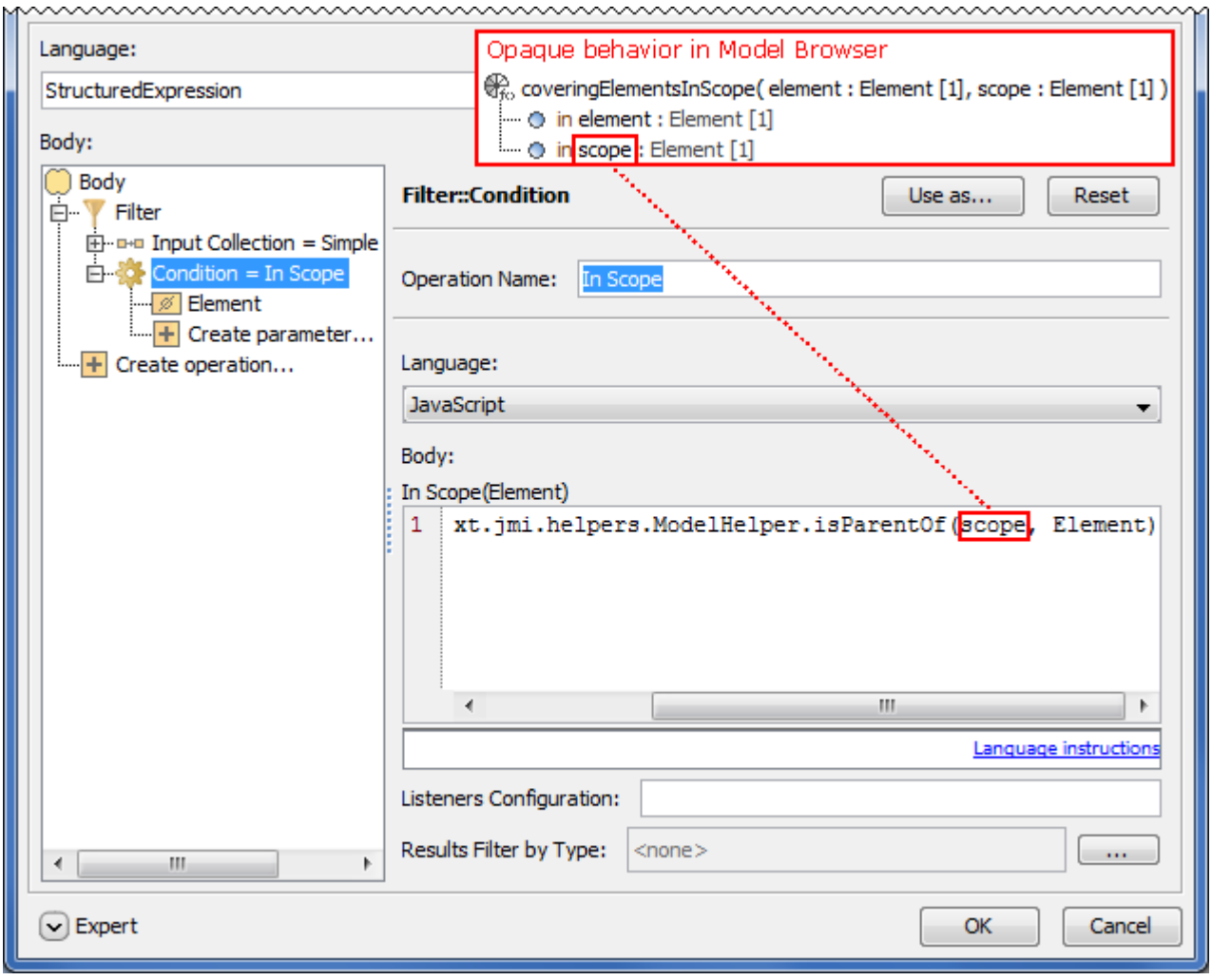


Figure 345 -- Accessing argument passed to parameter of opaque behavior

Copyright © 1998-2015 No Magic, Inc.

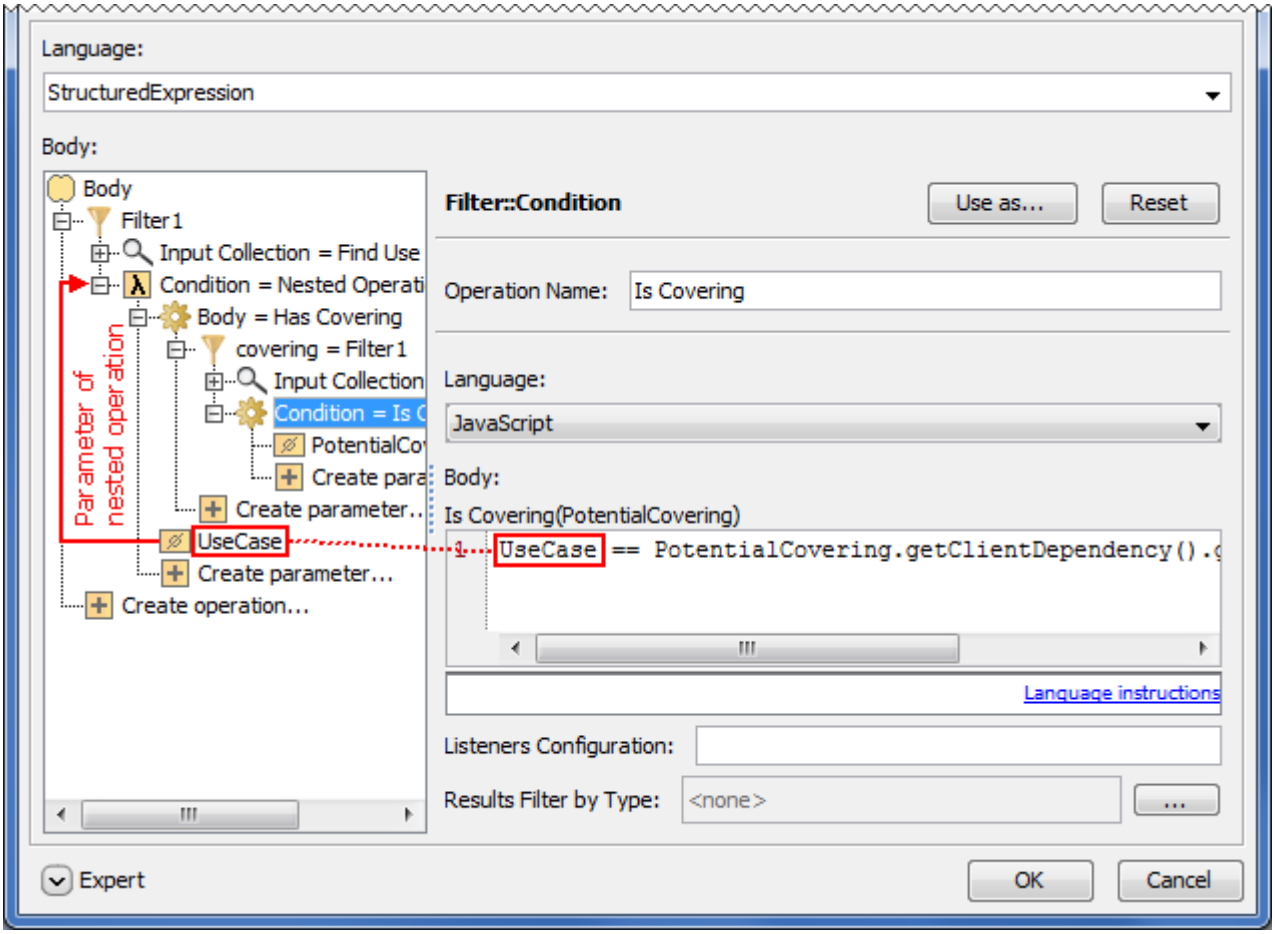


Figure 346 -- Accessing argument passed to parameter of nested operation

How many statements can a script have?

The script can have multiple statements. In this case the result of the entire script is the result of the last statement.

What MagicDraw functionality can a script use?

The script can call MagicDraw Open API.



For the list of available MagicDraw Open API methods, refer to *javadoc.zip*, which can be found in *<MagicDraw installation directory>\openapi\docs*.

For the instructions how to use these methods, see MagicDraw OpenAPI UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

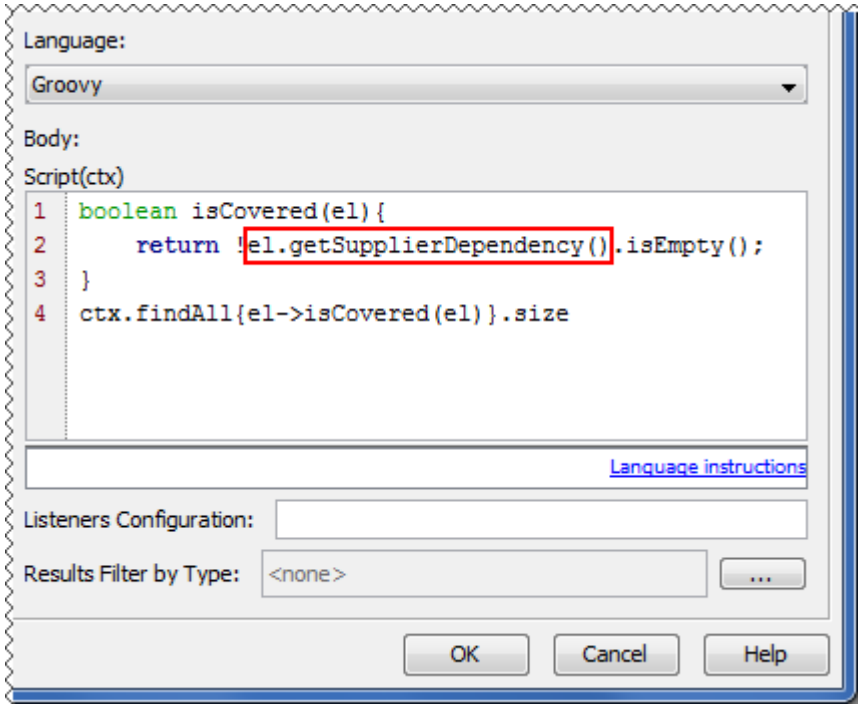


Figure 347 -- Using UML metamodel Open API to navigate between UML elements

More complex model access operations are available in *ModelHelper* and *StereotypesHelper*.



Use *import* statements to shorten java class names as shown in the following figure.

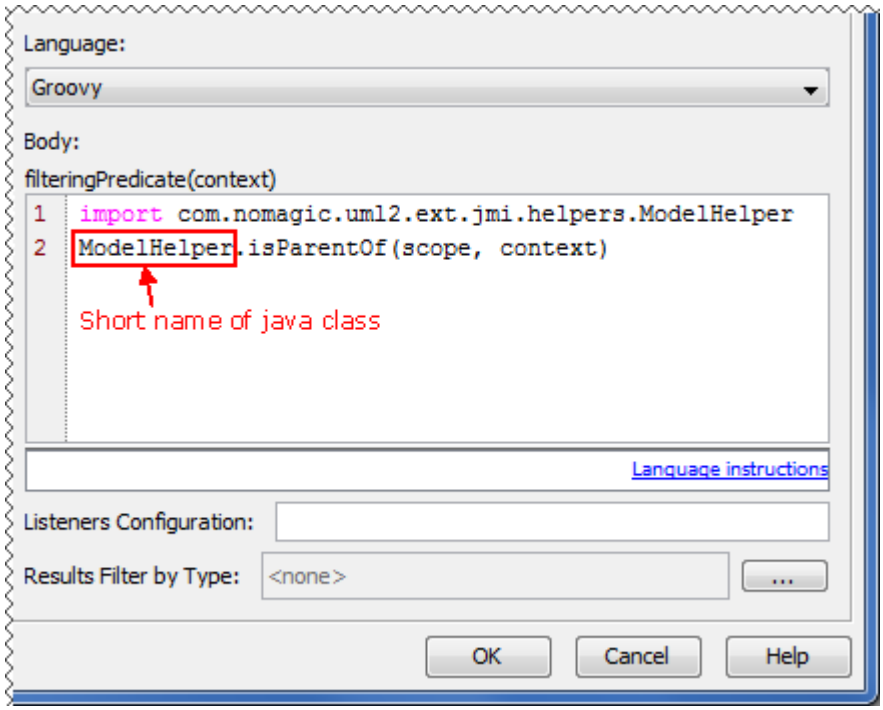


Figure 348 -- Using ModelHelper to define filtering condition script

Copyright © 1998-2015 No Magic, Inc.

How to simplify scripts?

To simplify the script, move complex sub calculations out of the script. Use other operations of the structured expression to specify these sub calculations and then pass the results to the script through parameters.

EXAMPLE

Let's say we need a script operation, which counts the number of nested packages. You can use MagicDraw Open API to navigate through the model and find all the nested packages, but it would be complicated.

To simplify the script:

1. Use the Find operation to list these nested packages.
2. Use the script operation for nothing else than counting the number of the list items. Pass the result of the Find operation to this script through the PackageList parameter as the following figure shows.

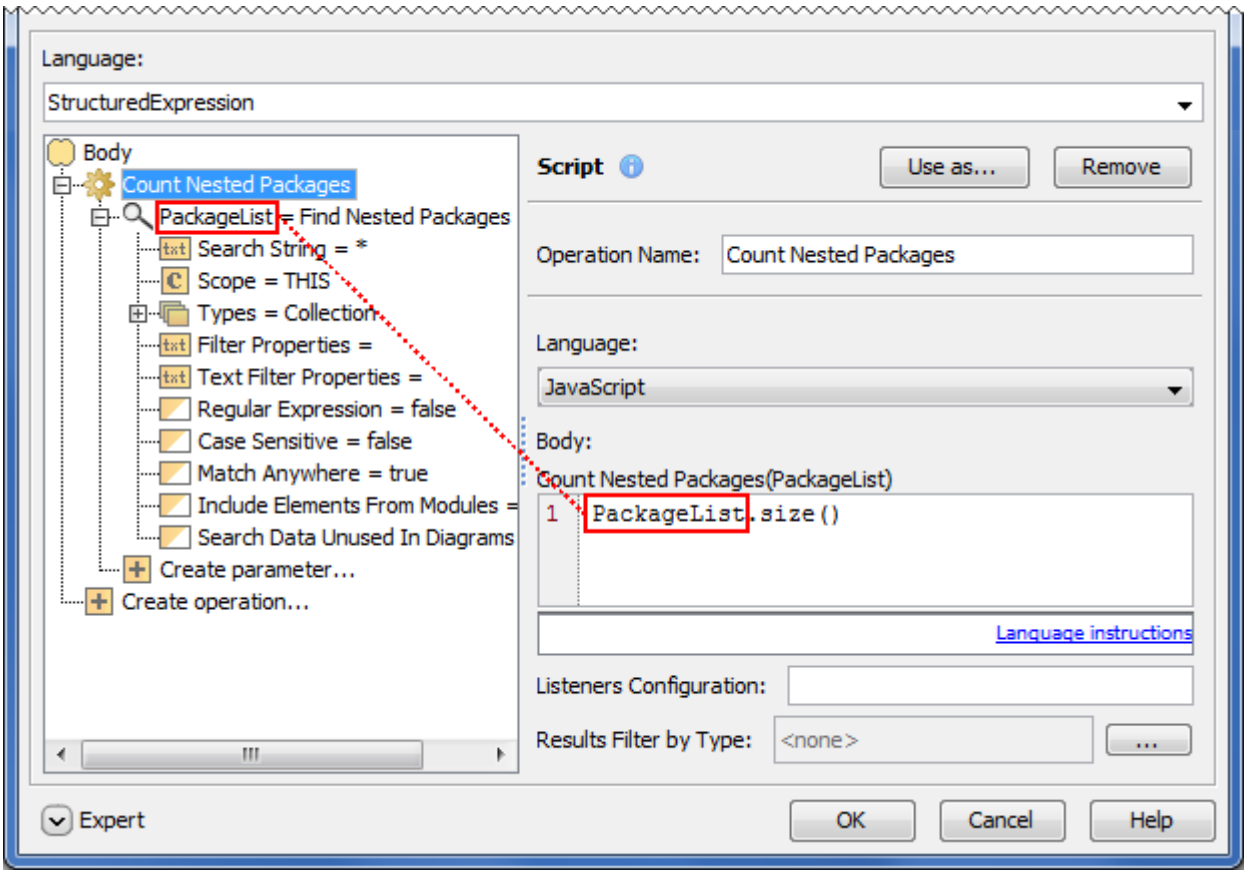


Figure 349 -- Using Find operation to simplify script body

Symbol Usage in Diagrams

Concepts

Symbol

A visual representation of some model element in diagrams. Symbols are further subdivided into shapes and paths (these are lines used in the model for representing various relationships).

Copyright © 1998-2015 No Magic, Inc.

The symbol usage in diagrams functionality allows the user to see the usage of a symbol throughout the diagrams of a project. You may search for symbol usage in diagrams from the element Specification window and from element shortcut menu.

To search for diagrams in which symbol is used from the element Specification window

1. Open the element's Specification window and click **Usage in Diagrams** on the left. The table shows all diagrams in which the symbol is represented.
2. Select the diagram you want to open.
3. Click the **Open** button. The diagram is opened and the symbols of the element are selected. If the diagram includes more than one of the same element symbol, all symbols of the same element are selected in the diagram.

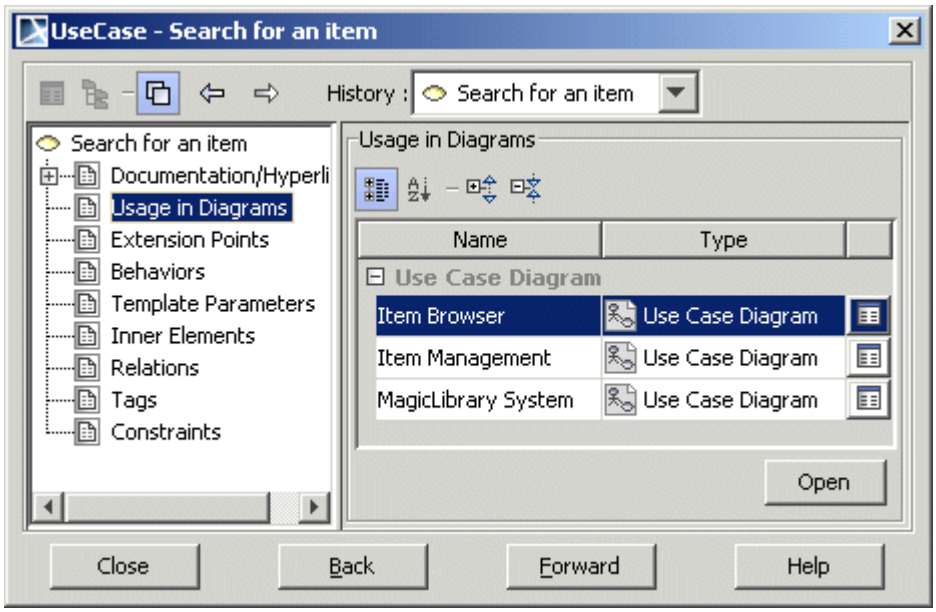


Figure 350 -- Element Specification window, Usage in Diagrams pane

To open the diagram specification dialog, click the  button near the diagram in the list.



For more information about working with the element's Specification window, see "Specification Window" on page 273.

To search for diagrams in which symbol is used from the element's shortcut menu

1. Select the element in the Model Browser or select the symbol on a diagram.
2. From the element's shortcut menu, select **Go To** > **Usage in Diagrams**.

Copyright © 1998-2015 No Magic, Inc.

3. Select a diagram whereon you want to see the selected element symbol. The diagram is opened and symbols of the element are selected on it.

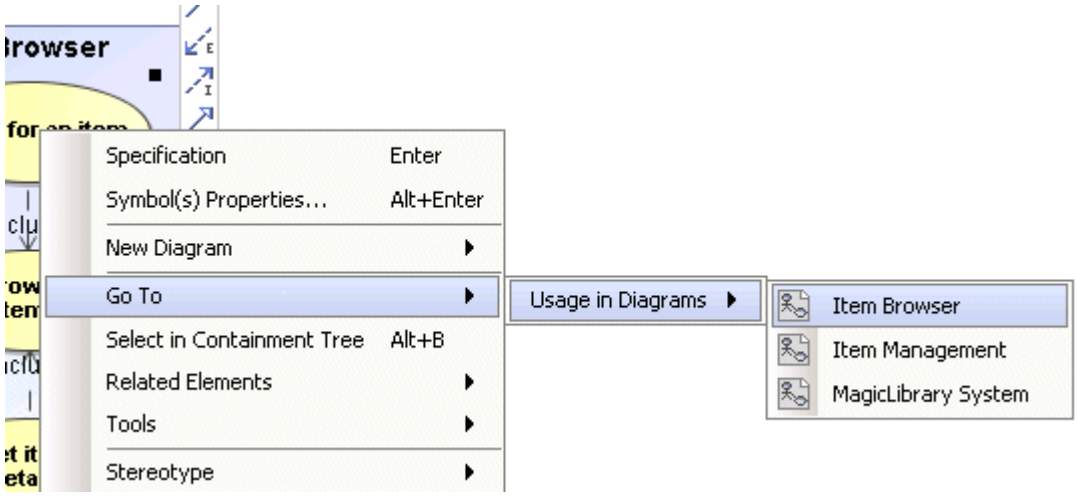


Figure 351 -- Searching for symbol usage in diagrams from element's shortcut menu

To search for diagrams in which symbol is used from the main menu

1. Select the element in the Model Browser or select the symbol on a diagram.
2. From the **Analyze** menu, select **Go To** > **Usage in Diagrams**.
3. Select a diagram whereon you want to see the selected element symbol. The diagram is opened and symbols of the element are selected on it.

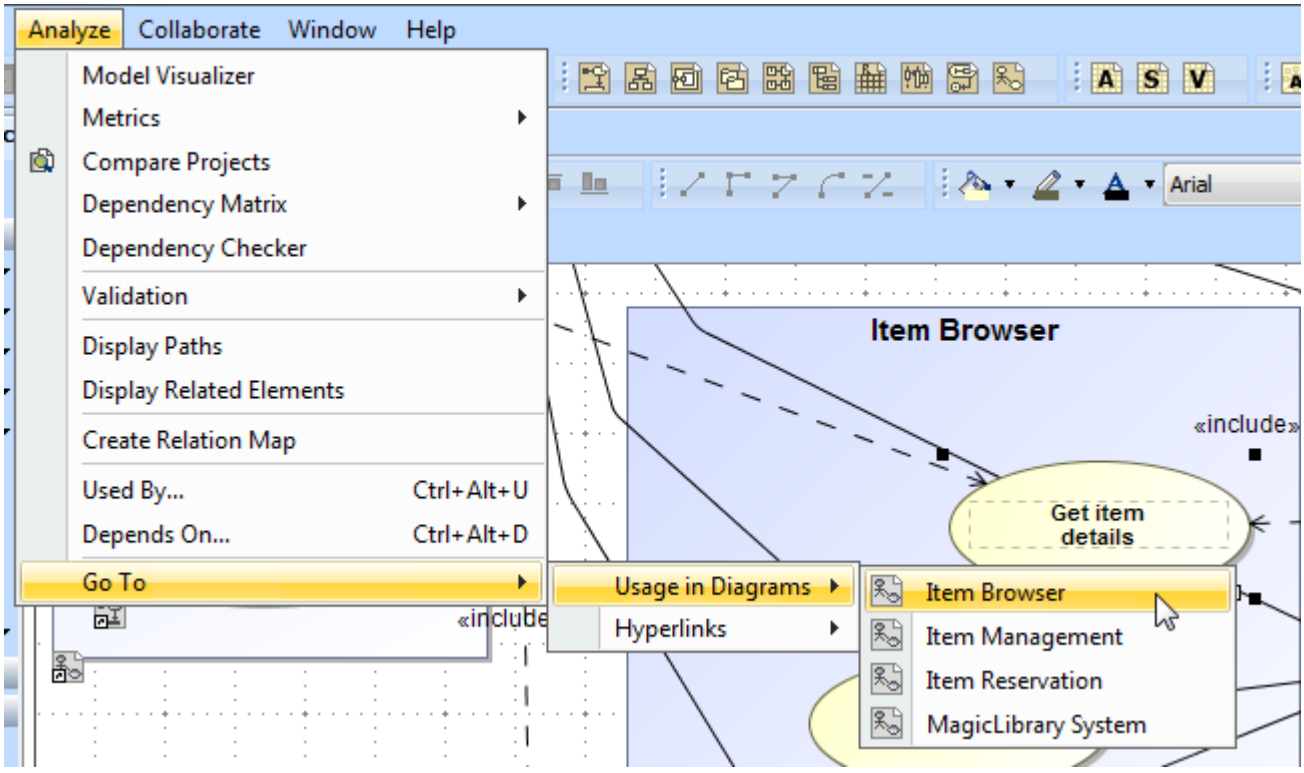


Figure 352 -- Searching for symbol usage in diagrams from main menu.

NOTE

- All symbols of the same element are selected in the opened diagram.
- If more than one symbol exists in the same diagram, then the diagram is zoomed out, to fit the view in screen.

Copyright © 1998-2015 No Magic, Inc.



For information about the Usages/Dependencies functionality for analyzing associations between elements, see "Analyzing Usages and Dependencies" on page 509.

Projects Comparison



Visual Model Differencing



This functionality is available in Architect, and Enterprise editions only.

The use of projects comparison functionality allows you to compare two local projects or two versions of the same server project, as well as two diagrams. Model elements are compared by the element ID.

You can compare:

- current project with locally saved project.
- current project with open project.
- two server project versions.
- current project with server project version.
- local project and server project version.
- diagrams.

Related references

Comparing Projects
Diagrams Comparison
Text comparison

Comparing Projects

First of all you have to select the projects you want to compare. Differences of the compared projects will be displayed as two model trees. Differences are marked using colors and highlighting.

When comparing two projects the following data changes will be reflected:

- New model elements.
- Deleted model elements.
- Model elements with modified data.
- Model elements that changed location.
- Inner elements changes.

To select the projects you want to compare

1. Open a project you want to compare and from the **Analyze** menu, select **Compare Projects**.

Copyright © 1998-2015 No Magic, Inc.

2. The **Compare Projects** dialog appears.

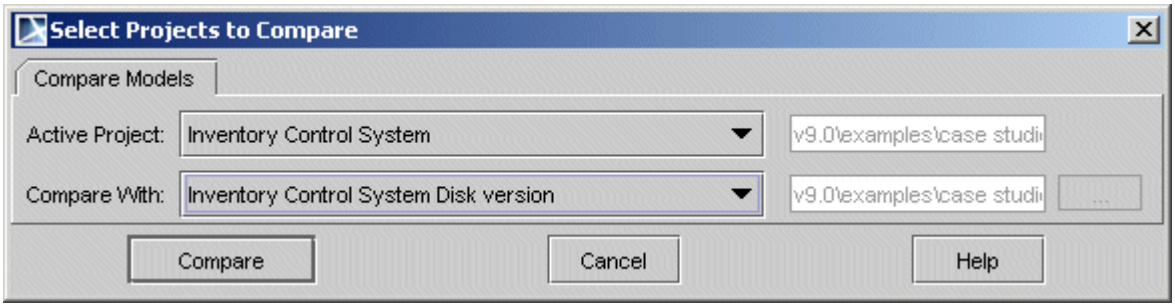


Figure 353 -- Select Projects to Compare dialog

3. From the **Active Project** box, select a project you want to compare with another project. The list contains names of all open projects in MagicDraw.
4. From the **Compare With** box, select a project with which you want to compare the first project. The following options are available:
    - open project names are listed, except the project, which is selected in the **Active Project** box.
    - **Local Project**. Click the **...** button and select a project version you want to compare the current project with.
    - **Server Project**. Click the **...** button and from the **Open Server Project** dialog, select a project and its version you want to compare with.
    - <current project>Disk version. By selecting this option you will be able to compare currently changed project version with the unchanged project version on the disc without current changes.

      By default the first opened project should be selected. If there are no more projects open, Local Project is selected as the value.

Copyright © 1998-2015 No Magic, Inc.

Understanding model differences

All differences of the compared projects are displayed in the **Difference Viewer** dialog.

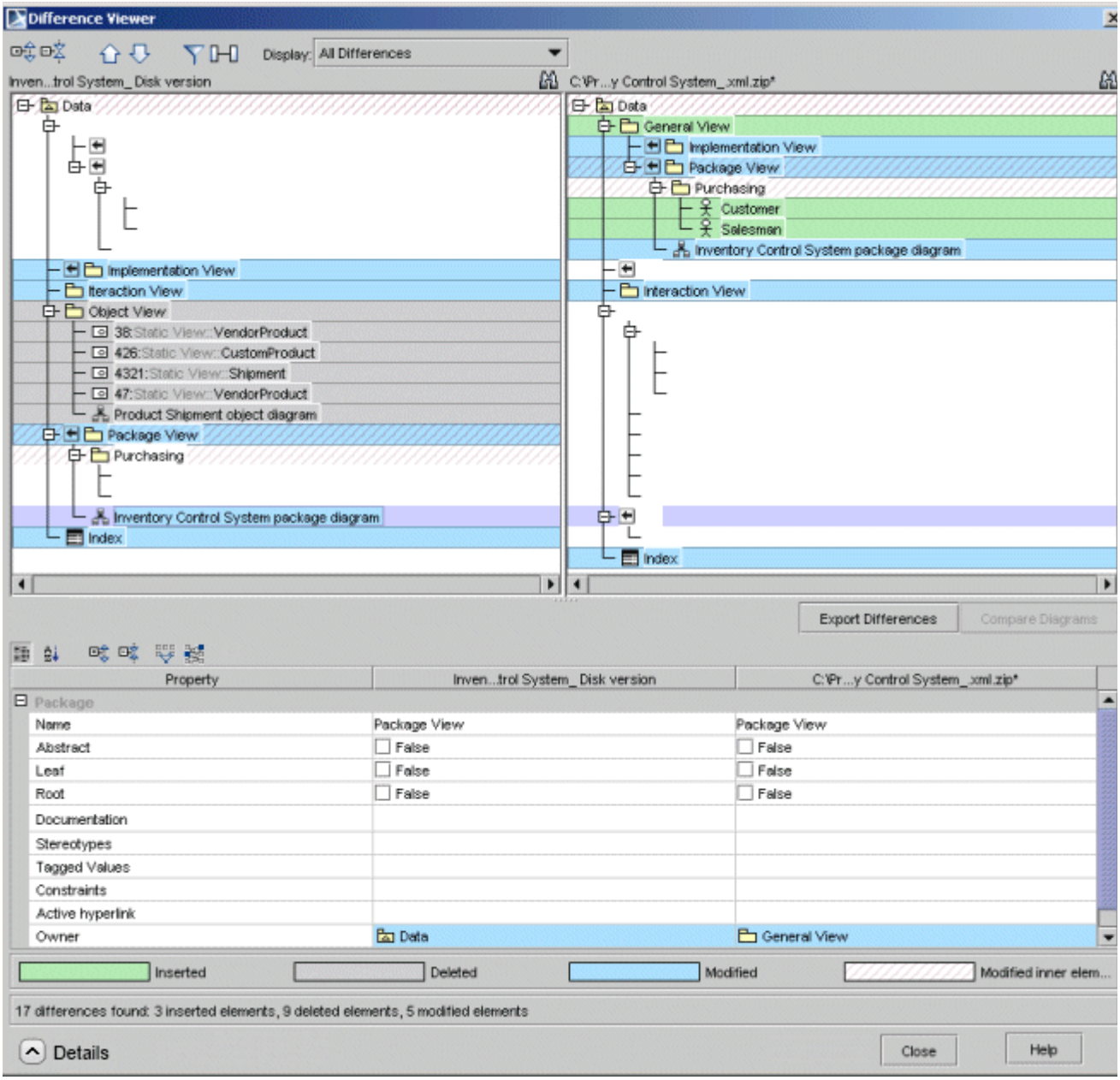


Figure 354 -- Difference Viewer dialog

In the **Difference Viewer** dialog, two compared projects are displayed. The number of differences is displayed at the bottom of the dialog. The number of differences number includes inserted, deleted, and modified elements. The number does not include elements with changes to inner elements. Differences number is displayed in following order:

To mark changes in the model elements several colors are used:

- *Elements that do not exist in the other model (inserted elements)*. Element is displayed only in the right-hand tree. New element is highlighted in light green.
- *Elements that exist in the other model, but do not exist in current model (deleted elements)*. Elements are always displayed in the left-hand tree. The deleted element is highlighted in grey.
- *Elements with modified element data (modified elements)*. The modified element is highlighted as changed in both trees. The modified element is highlighted in light blue.

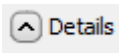
Copyright © 1998-2015 No Magic, Inc.

- *Elements that changed location (parent has changed)*. Element is marked as a modified element. Empty nodes are displayed in the opposite tree where the element does not exist. On the moved element and on its former position, a button is displayed. Pressing the button on the former position, selects the place where the element has moved to. Pressing the button on the moved element position, selects the former element place. Also, you can perform these operations using the shortcut menu commands **Go to former position** or **Go to moved element**.
- *Elements that have inner elements that changed*. The element is marked in both trees and is highlighted in light grey dashes. An element with modified element data and changed inner elements is marked as modified and as element with changed inner elements. Element is marked as modified when:
    - Element specification properties have changed. Element specification properties include all properties, which are not displayed in the Browser as separate elements. Model element specification properties are treated as changed only if the element property can be changed from the element specification. If an element specification has changed because of changes made to other elements, the element should not be treated as changed. Example: typed values in the tagged values specification, attribute links in an object and instance specification, etc.
    - Element parent has changed.
    - Path is drawn from/to element.

Buttons available in the **Difference Viewer** dialog:

Button	Function
 **Expand All**	Both trees are expanded.
 **Collapse All**	Both trees are collapsed.
 **Go to Previous Difference**	Select the difference listed prior to the current one.
 **Go to Next Difference**	Select the difference listed next from the current one.
 **Filter**	The **Filter** dialog opens. Show/hide the elements you want to analyze.
 **Include Relation Ends**	Relations are displayed in the elements specifications. A relation added to the element means that the element is marked as modified.
Display:	**All** - Shows all elements of the projects.
	All Differences - Shows only differences made between the projects.
	Deleted Elements - Shows only elements that were deleted from the projects.
	Inserted Elements - Shows only elements that were inserted in the projects.
	Modified Elements - Shows only elements that were modified.

Copyright © 1998-2015 No Magic, Inc.

Button	Function
 **Find Next Element**	The **Find** dialog opens. Search for elements within the corresponding project.
Export Differences	Creates .html or .txt differences report. In the **Save Difference Report As** dialog, select the directory where you want to save this report. **NOTE:** *.html format is suitable for viewing the difference report. If you want to copy this report to another program, use of the *.txt format is recommended.
	Click to expand or collapse the Difference Viewer dialog. When dialog is expanded, the element properties table is displayed. By default the dialog is expanded.
 **Show only differences**	Click to display only properties with differences.
 **Compare property values**	Click to compare the text differences. For more information, see "Text comparison" on page 585.

The first column of the Property window contains the same properties as the Quick Properties tab in the Browser. The second column title is the left-hand project name (with path) for local projects, and teamwork project name and version number for teamwork projects. The third column title is the right-hand project name (with path) for local projects, and teamwork project name and version number for teamwork projects. Modified properties are marked with the same color as in the model element tree.

Related concepts

Projects Comparison

Related references

Find dialog

Diagrams Comparison

To compare diagrams

1. In the **Difference Viewer** dialog, select the diagram you want to compare and click the **Compare Diagrams** button.
2. The **Diagrams Difference Viewer** dialog opens, which displays both diagrams with changes made in them.

Copyright © 1998-2015 No Magic, Inc.

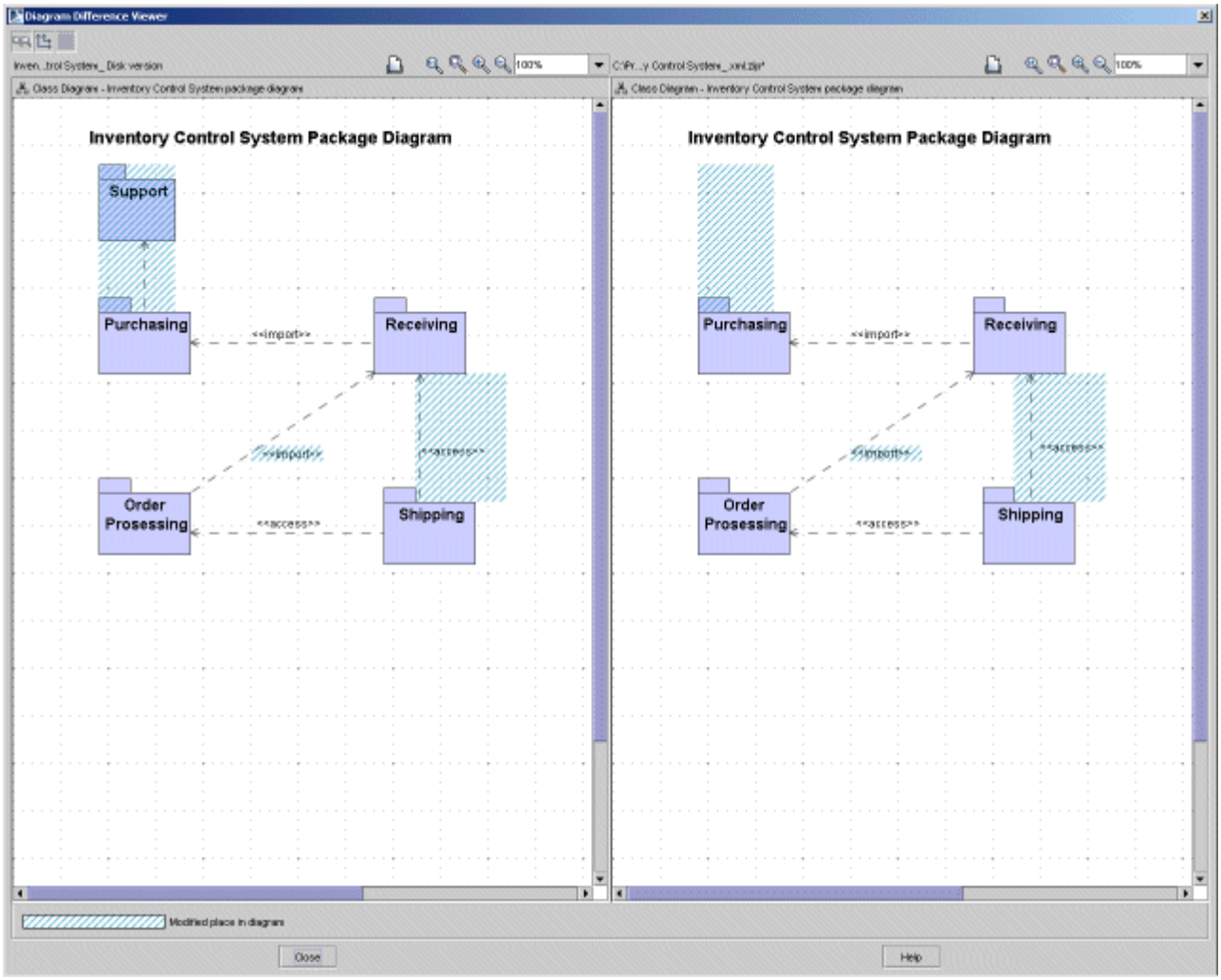


Figure 355 -- Diagram Difference Viewer dialog

Diagrams Difference Viewer displays two diagrams:

- Current (or first opened) project diagram is displayed at the right-hand side.
- Diagram that is compared with is displayed at the left-hand side.

Symbol changes that are reflected in the diagram:

- Modified symbol properties
- New symbol creation
- Symbol deletion
- Symbol bound changes (resize, bound changes because of element properties changes)

All changes are highlighted in light blue dashes.

Buttons available in the **Diagrams Difference Viewer** dialog:

Button	Function
Synchronize Zooming	If pressed, zooming affects both diagrams: zooming one diagram causes zooming of the other diagram.

Copyright © 1998-2015 No Magic, Inc.

Button	Function
 **Synchronize Scrolling**	If pressed, scrolling one diagram causes scrolling of the other diagram.
 **Mark Changes**	If pressed, places where diagram has changed are marked on both diagrams.
 **Print Diagram**	Prints the corresponding diagram together with marked changes.
 **Zoom 1:1**	Zooms the corresponding diagram(s) to the original size.
 **Fit in Window**	Zooms the corresponding diagram(s) to the size that fits the window.
 **Zoom In**	Zooms the corresponding diagram(s) in.
 **Zoom Out**	Zooms the corresponding diagram(s) out.
	Select the percentage for zooming the corresponding diagram(s).

Related concepts

Projects Comparison

Text comparison

As of version 17.0.3, MagicDraw can display exact changes of textual values when comparing models. You can now compare long texts in element properties, for example, element documentation, comments, pre and post conditions of a use case and so forth. You can see the differences of compared texts in a single dialog, where inserted and deleted parts of the text are appropriately highlighted.

Copyright © 1998-2015 No Magic, Inc.

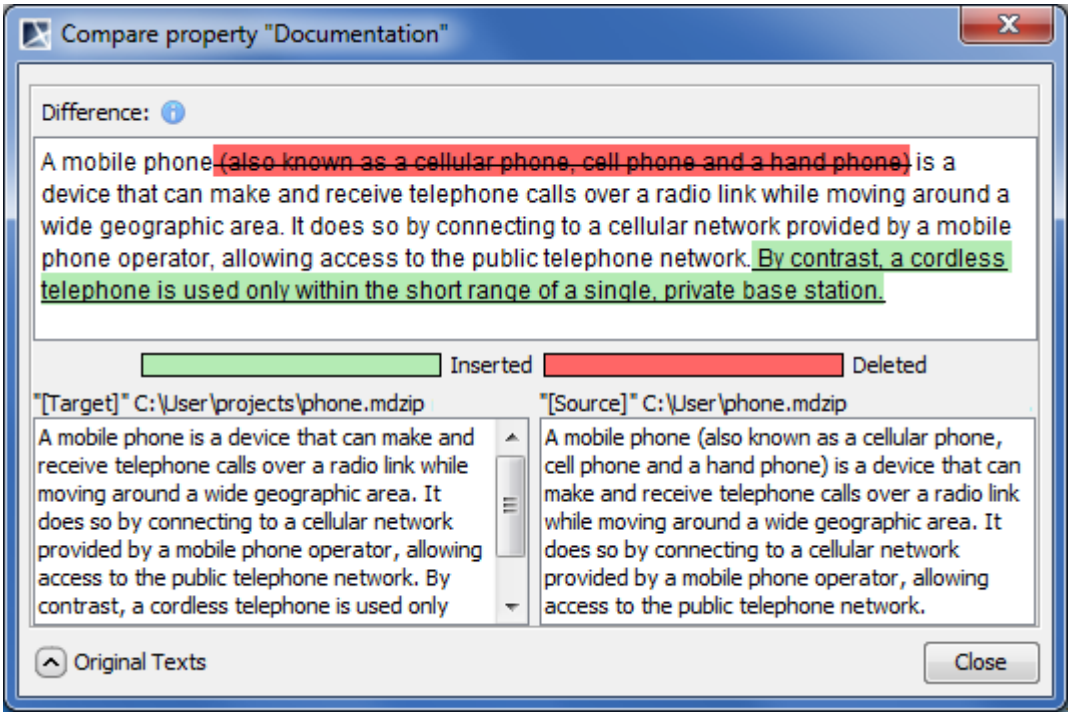


Figure 356 -- Compare property dialog

See an example, there the text differences are displayed for the Documentation property. The deleted text is highlighted with red color and the inserted text with green.

To open the Compare property dialog, do one of the following

- In the **Difference Viewer** dialog, click the **Compare property values** button. The **Compare property** dialog opens.

  NOTE

  The **Compare property values** button is enabled only when the property with the text difference is selected in the element properties table.

- In the **Difference Viewer** dialog, in the element properties table, double click the property that contains the text differences. The **Compare property** dialog opens.

The **Compare property** dialog contains the following elements:

Element name	Type	Description
Difference	**Informational tip**	Move the pointer over the button, to see the tooltip.
Inserted **Deleted**	**Explanatory text**	The text that is highlighted with green color is the inserted text and the red text is the deleted text.
Original Texts	**Button**	Click to expand or collapse the **Compare property** dialog. When expanded, the original texts from the target and source projects are displayed.

Related concepts

Projects Comparison

Copyright © 1998-2015 No Magic, Inc.

Metrics

NOTE

This feature is available in Architect and Enterprise editions only.

Metric is an item of information about some specific aspect of the model. The collection of various metrics can be used to evaluate the current state of your model. Calculated at regular intervals they can help tracking the evolution of the model development.

In this section you will learn *how to*:

- Create a metric suite
- Calculate metrics for a particular element according to selected metric suites
- Manage metrics in a metric table
- Use the obsolete metrics feature (available in MagicDraw 17.0.5 and earlier)

Concepts

For better understanding the further material, first of all get acquainted with the following concepts.

Metric suite

A collection of parameter and metric definitions used to calculate metrics that evaluate specific aspects of your model.

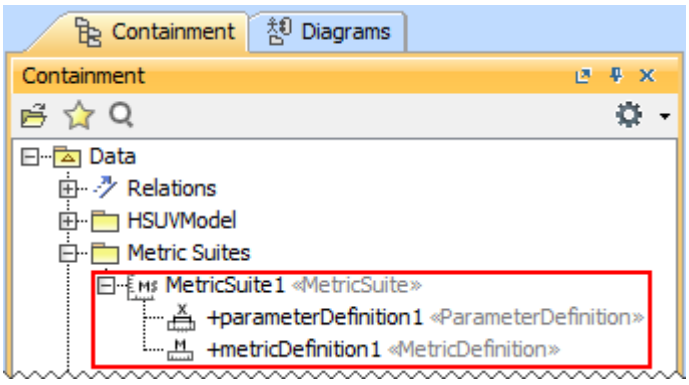
Parameter definition

A variable that is used for calculating metrics. It can be an element type (including both relationship and DSL element types), particular element, data type, or data type value.

Metric definition

A formula for calculating metrics according to the parameter values.

Actually, metric suite is a class with the *«MetricSuite»* stereotype applied. Both parameter and metric definitions are attributes of the metric suite.



One metric is the result of single calculation of a metric definition. The metrics of all the metric definitions of the metric suite together with relevant parameters are stored in a single instance of the metric suite (as slot values). To analyse and manage the instances of the metric suite with MagicDraw, use the metric table.

Copyright © 1998-2015 No Magic, Inc.

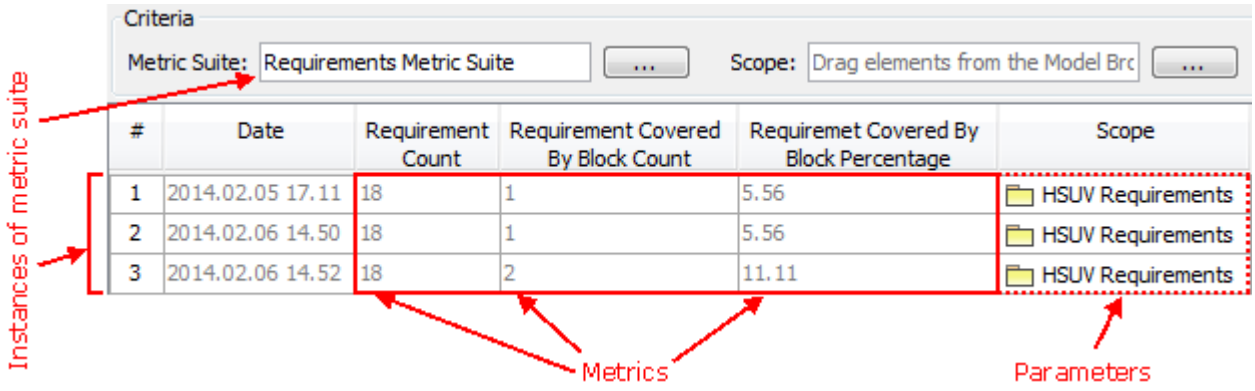


Figure 357 -- Instances of metric suite in metric table

Creating Metric Suites

There are several conventions that you must know about metric suites before creating your first one:

1. *Metric suite must have a target.* The target specifies model elements, for which metrics of the particular metric suite can be calculated. The target value can be any type of elements or even a particular element of your model.
2. *Metric suite must have one parameter definition declared as context.* An element, which is selected for calculating metrics, will be automatically set as a value of the context parameter definition.
3. *Metric suite must have a date tracker.* So that you could see a timestamp of each metric.

These conventions determine the main steps of creating a metric suite. They are as follows:

1. Adding a new metric suite to your model
2. Specifying the target for the metric suite
3. Defining parameters (including the context parameter definition)
4. Building metric definitions



It's recommended to have a *single* person per project, who is responsible for creating and managing all the metric suites of the project.

Adding a new metric suite to your model

There are several approaches of adding a new metric suite to your model:

- Creating more specific element for the *BaseMetricSuite*
- Starting from scratch

The *BaseMetricSuite* is a predefined metric suite stored in the *DSL Customization* standard/system profile.

You are free to choose either approach. Though, if you are a beginner, take the advice to choose the first one. It jump-starts the creation of a new metric suite: the more specific metric suite inherits the *BaseMetricSuite* attributes that are necessary for calculating metrics (for example, the context parameter definition and the metric definition for date tracking), and you don't have to specify them on your own.

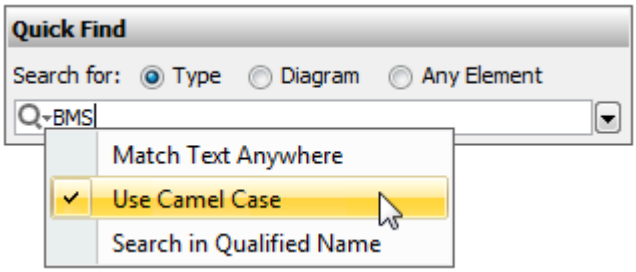
Copyright © 1998-2015 No Magic, Inc.

To create a new metric suite as more specific element of the *BaseMetricSuite*

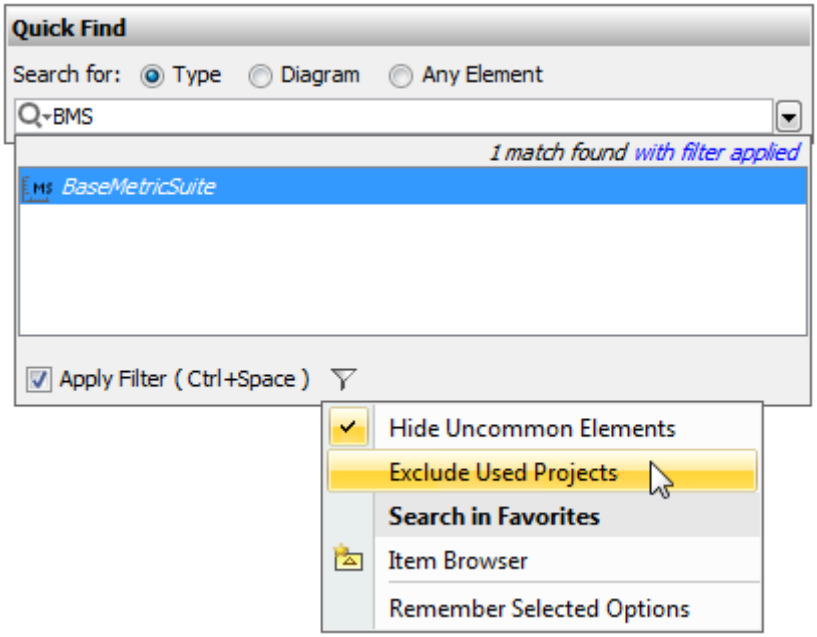
1. Create a package to store all the metric suites of your model.
2. In the package, create a profile diagram.
3. Press **Ctrl**+**Alt**+**F** and in the **Quick Find** dialog type “BMS”.

IMPORTANT!

- Be able to search by the upper case letters.



- Be sure the search includes auxiliary resources!



4. Click the *BaseMetricSuite* in the search results list. The *BaseMetricSuite* is selected in the Containment tree.
5. Drag the *BaseMetricSuite* to the diagram pane. A shape of it appears on the diagram.
6. On the diagram pallet, click MetricSuite , type diagram name, and then click a free space on the diagram pane. A new metric suite is created, and a shape of it appears on the diagram.
7. Draw a generalization relationship between the *BaseMetricSuite* and the newly created metric suite.

Copyright © 1998-2015 No Magic, Inc.

The newly created metric suite becomes more specific element of the *BaseMetricSuite*, which means that it inherits all the parameter and metric definitions of the *BaseMetricSuite.*

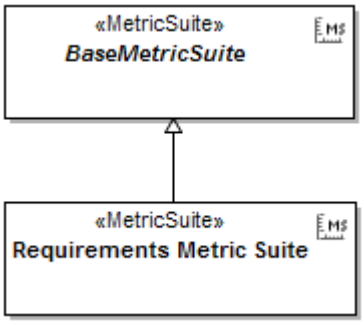


Figure 358 -- New metric suite as more specific element of BaseMetricSuite

To create a new metric suite from scratch

1. Create a package to store all the metric suites of your model.
2. In the package, create a profile diagram.
3. On the diagram pallet, click MetricSuite , type diagram name, and then click a free space on the diagram pane.

A new metric suite is created, and a shape of it appears on the diagram.



Figure 359 -- New metric suite from scratch

To specify description of the metric suite

1. Double-click a shape of the metric suite to open its Specification window.
2. Click **Documentation/Hyperlinks** on the left side of the dialog.
3. In the text box on the right side, type the description.
4. Click **Close**.

Copyright © 1998-2015 No Magic, Inc.

Specifying the target for the metric suite

Any metric suite must have a target. The target specifies model elements, for which metrics of the particular metric suite can be calculated. The target value can be any type of elements or even a particular element of your model.

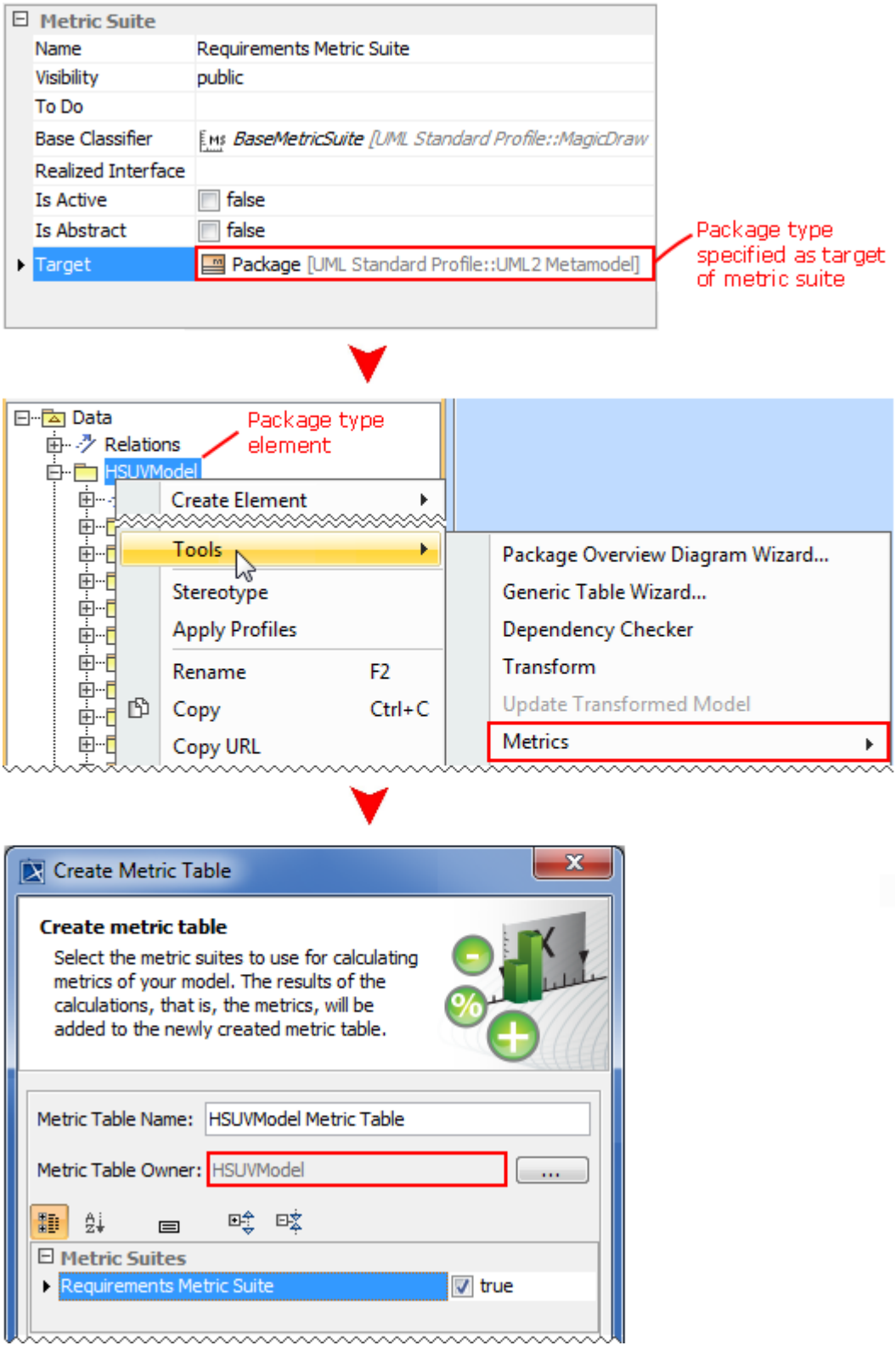


Figure 360 -- Applying metric suite, whose target is package type, on package "HSUVModel"

Copyright © 1998-2015 No Magic, Inc.

Actually, the target is a property of the metric suite.



Thought it can be any type of element or even an element of your model, we recommend selecting a particular package or at least the package type as the target for the metric suite.

To specify the target for the metric suite

1. Double-click the shape of the newly created metric suite to open its Specification window.
2. Click the cell of the **Target** property value and then click the Edit button. The element Selection dialog opens.
3. In the tree or list on the left side of the dialog, select an element and add it to the **Selected elements** list.



You can select more than one element.

For more information about the manipulations in the element Selection dialog, see "Elements Multiple Selection" on page 354.

The target is specified.

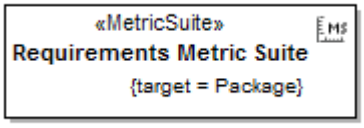


Figure 361 -- Metric suite with target specified



If you need to calculate metrics for the whole model, select the root package *Data* as the target of your metric suite.

Specifying parameter definitions

A parameter definition of the metric suite can be specified either:

- As the attribute with the *«ParameterDefinition»* stereotype.
- In the body of the metric definition.

If you need to track the parameter values, you should specify the parameter definition as the attribute of the metric suite. Then each value of the parameter becomes a slot of an instance created while running the metric suite.

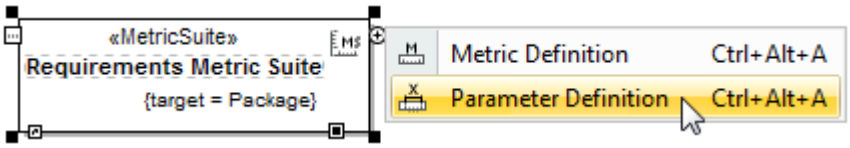
If your metric suite specifies the *BaseMetricSuite*, it has already got a few parameters (these are the inherited ones, such as *scope* and *coveringScope*). If it does not or the inherited parameters are not appropriate, you should define your own ones.

To define a parameter as an attribute of the metric suite

1. Select the shape of the metric suite.

Copyright © 1998-2015 No Magic, Inc.

2. Click ⊕ on the shape and then select **Parameter Definition**.



An attribute with the *«ParamterDefinition»* stereotype is created.

3. Type its name directly on the shape and then click a free space on the diagram.
4. Select the attribute on the shape and click T .
5. Select a type for the parameter.

IMPORTANT! This step is mandatory. A parameter must have a type!

The parameter is defined.

TIP! Instead of defining the parameters from scratch, you can copy ones from the *BaseMetricSuite* and then modify them according to your needs.

Declaring the context

One of the parameter definitions must be declared as the context of the metric suite. If the metric suite has the context declared, then any element, for which you select to calculate appropriate metrics, becomes the value of the context parameter definition of this metric suite and is used for calculating the metrics. Otherwise, calculating metrics for the selected element makes no sense (unless the context is declared in the body of the metric definition).

To declare the context for the metric suite

IMPORTANT! Make sure the type of the parameter you are going to declare as the context, corresponds the type of the target.

1. On the shape of the metric suite, double-click the parameter you need to define as the context of this metric suite. The Specification window of the parameter opens.
2. Click to select the check box in the cell of the **Set As Context** property value.

The property value becomes *true*, and the parameter is declared as the context of the metric suite.

TIP! If your metric suite specifies the *BaseMetricSuite*, it has already got the context parameter definition, that is, *scope*.

Building metric definitions

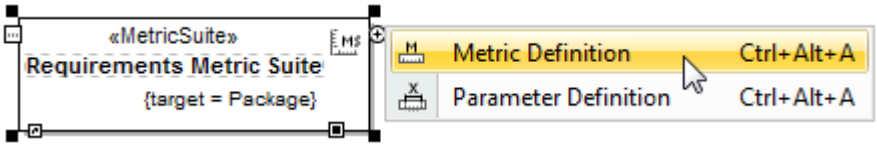
Metric definition is an attribute of the metric suite, with the *«MetricDefinition»* stereotype applied.

To build a metric definition

1. Select the shape of the metric suite.

Copyright © 1998-2015 No Magic, Inc.

2. Click ⊕ on the shape and then select **Metric Definition**.



An attribute with the *«MetricDefinition»* stereotype is created.

3. Type its name directly on the shape and then click a free space on the diagram.
4. Select the attribute on the shape and click T .
5. Select a type of result values that will be calculated according to the formula specified in the metric definition.

IMPORTANT! This step is mandatory. A metric definition must have a result type! For example, real or integer.

6. Press Enter to open the Specification window of the metric definition.
7. Click the cell of the **Default Value** property value. ... and ▶ buttons appear on the cell.
8. Click ▶ , point to **Value Specification**, and select **Opaque Expression**.
9. Click the cell of the **Default Value** property value again and then click ... . The **Default Value** dialog opens.
10. From the **Language** drop-down list, select a language.

NOTE To build a structured expression, select **StructuredExpression**.

11. In the **Body** box, specify the formula for the metric definition.

TIP! For more information about building structured expressions, see "Specifying Criteria for Querying Model" on page 551.

TIP! If your metric suite specifies the *BaseMetricSuite*, it has already got a metric definition for date tracking.

One metric definition can refer to another metric definition of either the same metric suite of a more general one (in case there is a generalization relationship between these metric suites).

How to...

How to build a metric definition for date and time tracking?

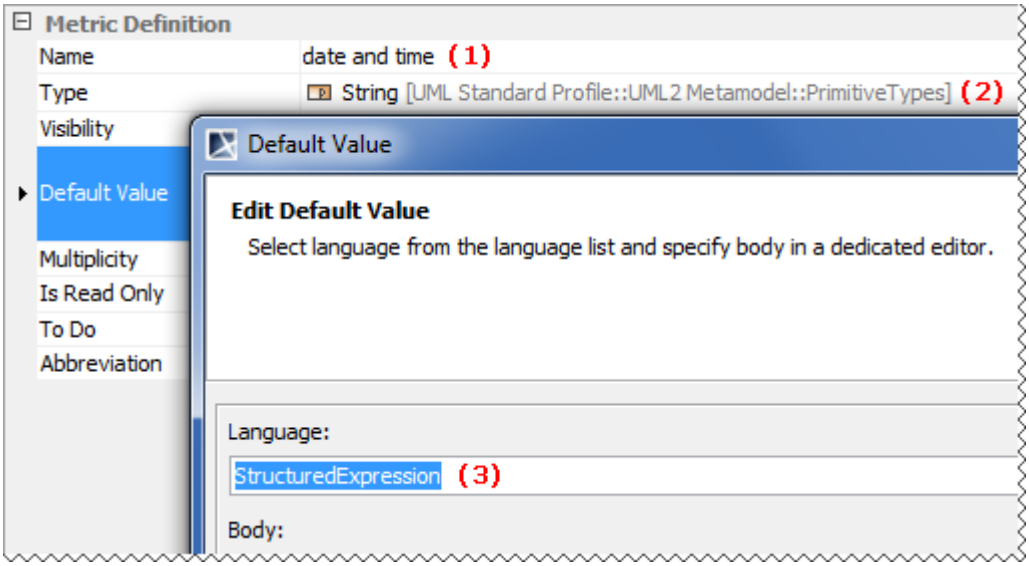
If your metric suite isn't a more specific element of the *BaseMetricSuite*, you must build a metric definition for the date and time tracking on your own.

For this, perform the following steps:

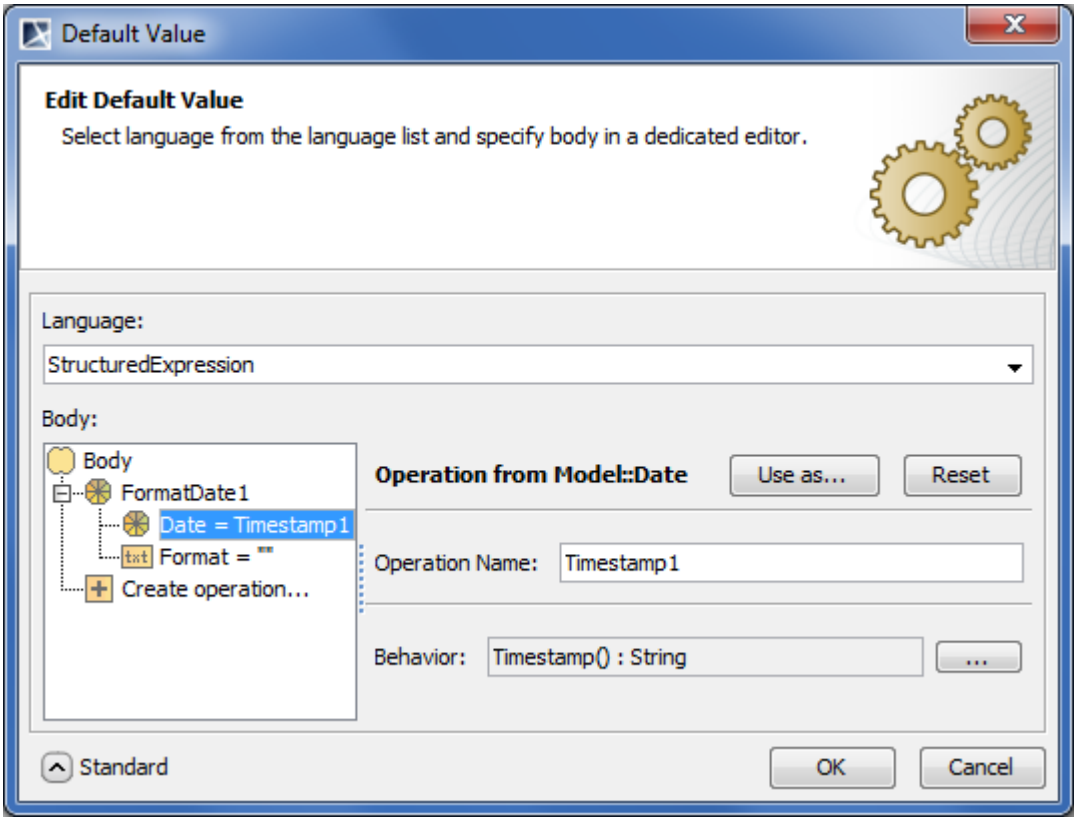
1. Add a new metric definition with the following characteristics to your metric suite:
   - *date and time* as name (1)
   - String as the result type (2)

Copyright © 1998-2015 No Magic, Inc.

- StructuredExpression as the language of the metric definition body (3)

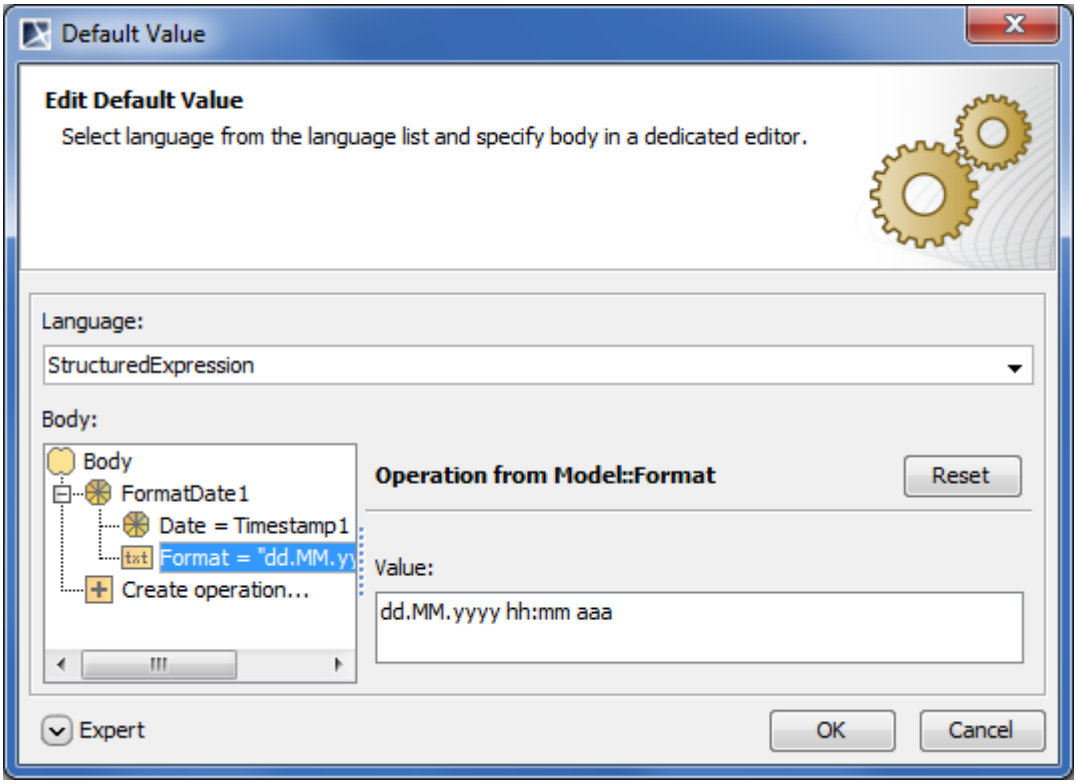


2. In the body of the metric definition, add a FormatDate built-in operation.
3. Use a Timestamp built-in operation as the Date parameter value.



Copyright © 1998-2015 No Magic, Inc.

4. Specify “dd.MM.yyyy hh:mm aaa” as the Format parameter value.



5. Confirm the changes and save your project.

Now you can track the date and time of each instance of your metric suite.

Date and time	
20.02.2014 03:35 PM	139
21.02.2014 04:55 PM	139

Figure 362 -- Date and time values calculated according to formula defined in “date and time” metric definition

How to access parameter values in a metric definition?

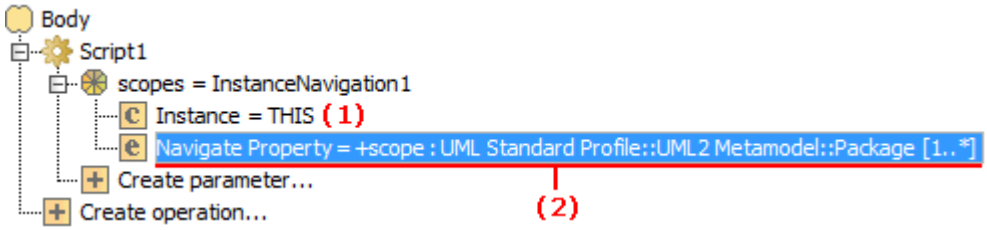
If you need to use values of some parameter for specifying calculations in the metric definition, you can easily access them with help of the functionality provided by the StructuredExpression language.

Let’s say, you need to use value of the *scope* parameter inherited from the *BaseMetricSuite* to calculate your metrics. For this, perform the following steps:

1. Add a new metric definition to your metric suite.
2. Select the StructuredExpression language for specifying the calculations.
3. In the structured expression, create a script operation with single parameter named *scopes*.
4. Select an InstanceNavigation built-in operation as the *scopes* parameter value. Specify parameter values of that operation:
   - Select Contextual Variable as the Instance parameter value (1).

Copyright © 1998-2015 No Magic, Inc.

- Select the *scope* parameter of the *BaseMetricSuite* as the Navigate Property parameter value (2).



5. In the script operation, refer to the *scopes* parameter simply by its name as the following Groovy script sample illustrates:

```
import static com.nomagic.magicdraw.modelmetrics.ScriptHelper.*;
import static com.nomagic.requirements.util.SysMLProfile.*;

getElementsRecursively(scopes)
 findAll {el -> isRequirement(el)}
 size()
```

An alternative way for accessing parameters is calling the *getValue()* method from MagicDraw Open API on the *valueContext* globally defined value as the following Groovy script sample illustrates:

```
def scopes = valueContext.getValue("scope")
```

How to build a metric definition that takes the result of another metric definition?

Let's say, we need to calculate the percentage of requirements that are covered by blocks. For this, we must have the following values:

- Overall requirements count
- Requirements covered by blocks count

Let's say, these values are the results of two already existing metric definitions: *requirementsCount* and *requirementsCoveredByBlocksCount* appropriately.

Now we need another metric definition, which takes the results of the above mentioned metric definitions and calculates the percentage. To access these values and make sure they are already calculated, we must call the *getOneValue()* method from MagicDraw Open API on the *valueContext* globally defined value for both metric definitions as the following Groovy script sample illustrates:

```
int all = valueContext.getOneValue("requirementsCount");
int covered = valueContext.getOneValue("requirementsCoveredByBlocksCount");
com.nomagic.magicdraw.modelmetrics.ScriptHelper.calcPercentage(all, covered)
```

Calculating Metrics

This section describes:

- "Creating a metric table" on page 597
- "Calculating metrics in the metric table" on page 600

Creating a metric table

While other types of tables can be created in several ways, *there is only one proper way to create a metric table*. It is described in the following procedure.

Copyright © 1998-2015 No Magic, Inc.

To create a metric table

1. In the Containment tree, right-click the element, for which you need to calculate metrics of particular metric suite.



The element or at least the type of it must be a target value of that metric suite.
For more information, see "Specifying the target for the metric suite" on page 591.

2. From the shortcut menu of the element, select **Tools** > **Metrics** > **New Metric Table**. The **Create Metric Table** dialog opens.

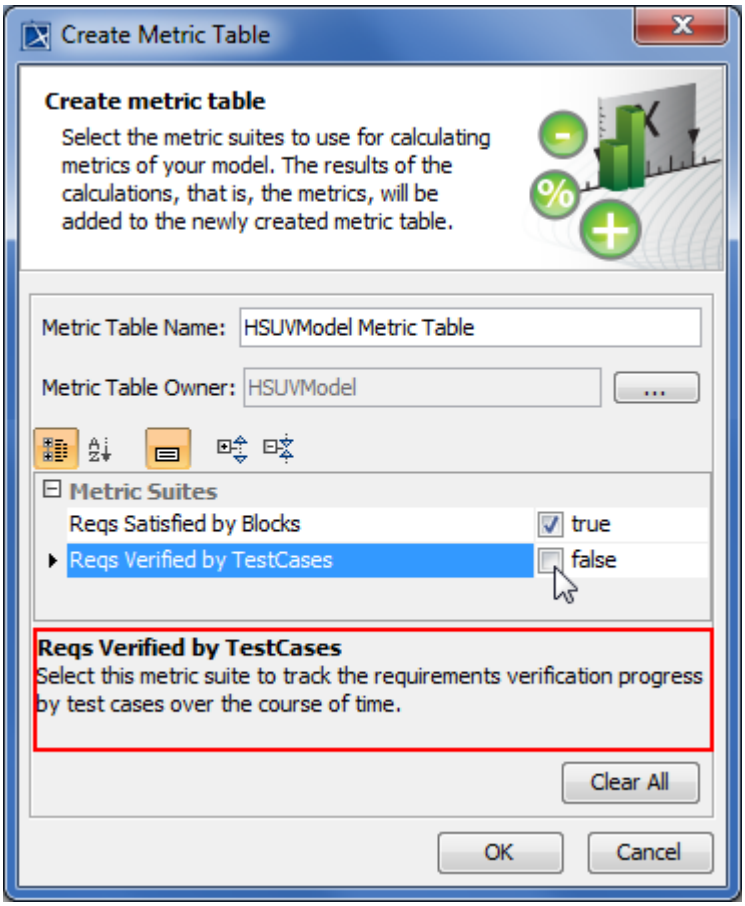
Copyright © 1998-2015 No Magic, Inc.

3. In the list of available metric suites, deselect unneeded metric suites by clicking nearby check boxes. Leave selected only the check box that is nearby the metric suite you need.



Carefully read descriptions of each metric suite. This helps to make a decision, which one of them best suites your needs.
Make sure you can see the description area (highlighted in the following figure) by clicking the Show Description button on the toolbar.

For more information, see the procedure "To specify description of the metric suite" on page 590.





If you cannot see the particular metric suite in this list, check the target value of that metric suite. It might be that the selected element is not a target of this metric suite.
For more information, see "Specifying the target for the metric suite" on page 591.

4. Click **OK**.

The table of the particular metric suite is created for the selected element. It is already filled with data, that is, metrics calculated at the table creation time. These metrics are actually stored in

Copyright © 1998-2015 No Magic, Inc.

the package owned by the element for which they have been calculated. The name of the package is <*element*> *Metrics*, for example, *HSUVModel Metrics*.

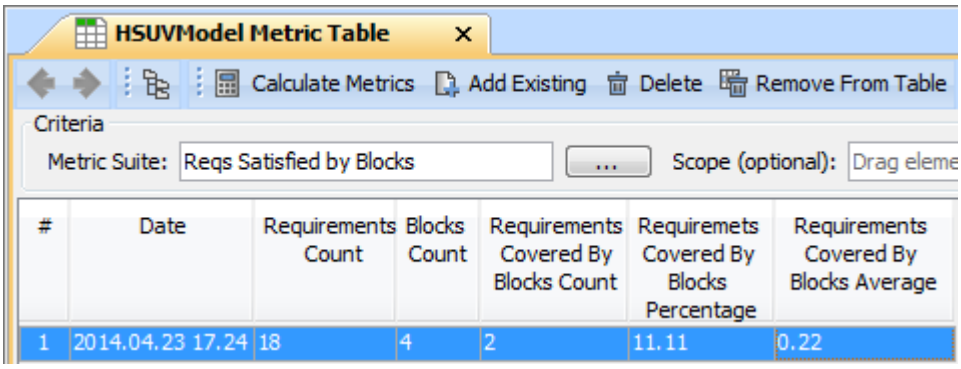


Figure 363 -- New metric table

NOTE

If one or more metrics are not calculated, make sure the metric suite is specified correctly. Here are some suggestions:

- Check the metric definition – the formula could be specified incorrectly.
- Check the parameter definition – the metric definition probably cannot access parameter values.

This way of creating metric tables saves much time in comparison with common ways of table (and even diagram!) creation. By using one of the common ways for a metric table creation you start with an empty table, which requires further efforts to fill it, that is, specify the metric suite, create an empty row, define parameters, and only then calculate the metrics.

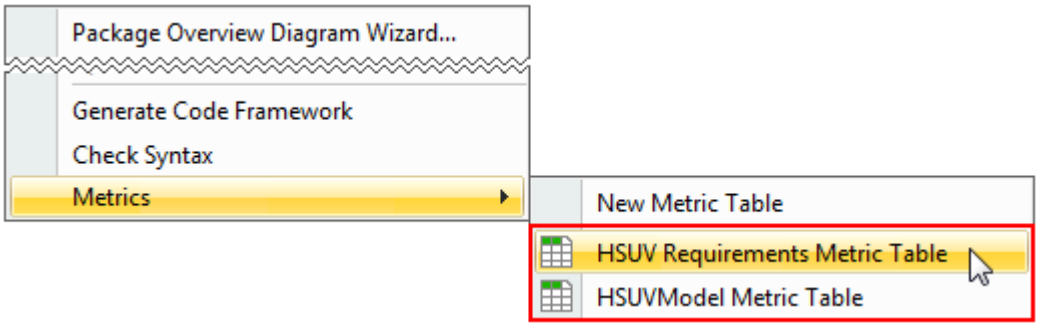
Calculating metrics in the metric table

Once you have created the metric table, which represents the metrics calculated at the table creation time, you can make further calculations and thus track the evolution of your model development.

If the table you need to append with new metrics is closed, you can easily open it.

To open the metric table

1. In the Containment tree, right-click the element that's metric table you need to append.
2. On the shortcut menu of the element, click **Tools** > **Metrics** and then select the metric table you need to append.



The selected metric table opens. Now you are ready calculate new metrics.

Copyright © 1998-2015 No Magic, Inc.

To calculate new metrics

1. In the metric table, select the row, below that you need to add a row with newly calculated metrics.
2. Do either:
   - On the Edit toolbar, click Calculate Metrics and then select **Calculate New Metric** from the open menu.
   - Right-click the selected row and then click **Calculate New Metric** on the shortcut menu.

The new metrics are calculated and represented in the table below the selected row.

To recalculate metrics

1. In the metric table, select the row, which metrics you need to recalculate.
2. Do either:
   - On the Edit toolbar, click Calculate Metrics and then select **Recalculate** from the open menu.
   - Right-click the selected row and then click **Recalculate** on the shortcut menu.

The new metrics are calculated and represented in the table instead of the selected row.

To calculate new metrics with different parameter values

1. In the metric table, select the row, below that you need to add a row for representation of newly calculated metrics.
2. Do either:
   - On the Edit toolbar, click Calculate Metrics and then select **Add New Metric with Different Parameters** from the open menu.
   - Right-click the selected row and then click **Add New Metric with Different Parameters** on the shortcut menu.

A new row with copied parameters is added to the metric table.

NOTE If the new row is empty, it might be that you haven't selected any row. You may continue the procedure.

3. Change parameter values.
4. Select the newly created row and do either:
   - On the Edit toolbar, click Calculate Metrics and then select **Recalculate** from the open menu.
   - Right-click the row and then select **Recalculate** from the shortcut menu.

The new metrics are calculated according to specified parameter values.

Metric Table Environment

In this section, you will find the brief information on each button and command that can be useful while working with metric tables. The section includes

- "Table toolbars" on page 602
- "Criteria area" on page 603

Copyright © 1998-2015 No Magic, Inc.

Table toolbars

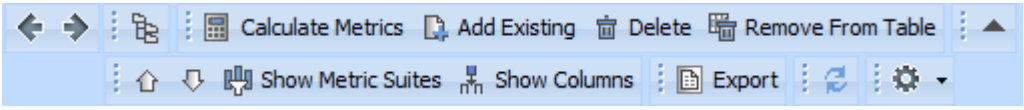


Figure 364 -- Metric table toolbars

NOTE

If the instance table toolbar buttons are not available in a server project, try to lock the table for edit (make sure you have the right to edit model of this project).

The following table describes the metric table toolbar buttons.

Button	Shortcut keys	Description
Navigation toolbar:		
	Alt+B	Click to select in the Containment tree the metrics of the selected row.
Edit toolbar:		
Calculate Metrics	N/A	Click to open the **Calculate Metrics** menu and then select the needed command. For more information, see "Calculating metrics in the metric table" on page 600.
Add Existing	Ctrl+Insert	Click to add metrics from the model.
Delete	Ctrl+D	Click to remove selected metrics both from the table and the model.
Remove From Table	Delete	Click to remove selected metrics only from the table.
Layout toolbar:		
Up	Ctrl+Up Arrow	Click to shift selected metrics (either grouped or non-grouped) up a row. The metrics are automatically renumbered after moving.
Down	Ctrl+Down Arrow	Click to shift selected metrics (either grouped or non-grouped) down a row. The metrics are automatically renumbered after moving.
Show Metric Suites	N/A	Click to select the metric suites that's columns you need to show on the metric table.
Show Columns	N/A	Click to select the columns for showing on the metric table. Keep in mind that after changing the metric suite, new columns are available only in the **Select Columns** dialog.
Publish toolbar:		
Export	N/A	Click to export the contents of the metric table to an **.html*, **.csv,* or **.xlsx* file.
View toolbar:		
	N/A	Click to update the contents of the metric table after specifying the smart package as the scope criteria.
Options toolbar:		

Copyright © 1998-2015 No Magic, Inc.

Button	Shortcut keys	Description
	N/A	Click to open the Options menu and then select: • **Show full paths** – to display the full paths of elements (where owner information can be displayed) in the metric table. • **Save Filter Criteria** – to keep the filter criteria unchanged after reloading the project or even after restarting MagicDraw. • **Show Detailed Column Name** – to show names of metric suites in column headers. For example, in the **Reqs Satisfied by Blocks.Blocks Count** column name, **Reqs Satisfied by Blocks** is the metric suite. This option works, when the metric table represents metrics of two or more metric suites. • **Show Column Type** – to show types of metric and parameter values in column headers. For example, **Blocks Count : Integer**. • **Show Abbreviation in Column Type** – to show abbreviations of metric or parameter names in column headers, if they are specified. • **Strip Multiline Text** – to strip the text that covers more than five rows. Three dots will be added at the end of the stripped text.
Suppress/Expand Criteria Area toolbar:		
	N/A	Click to hide the **Criteria** area.
	N/A	Click to show the **Criteria** area.
Other buttons:		
	Alt+Left Arrow	Click to navigate to the previously opened diagram, table, dependency matrix, or relation map.
	Alt+Right Arrow	Click to navigate to the previously opened diagram, table, dependency matrix, or relation map.

Criteria area

The **Criteria** area appears below the metric table toolbars. If it is not displayed, click the ▼ button on the Suppress/Expand Criteria Area toolbar.

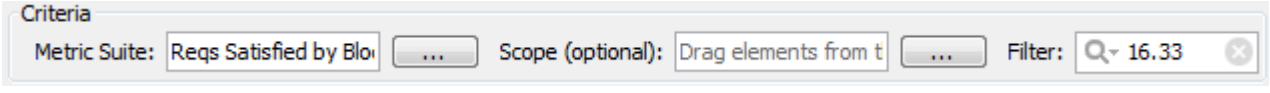


Figure 365 -- Metric table Criteria area

If the metric table is created following the procedure described in "Creating a metric table" on page 597, the **Metric Suite** box in the **Criteria** area has one or more default values. Otherwise, you have to specify the metric suite criteria in order to start working with the metric table. The other criteria are not necessary.

Copyright © 1998-2015 No Magic, Inc.

Using Obsolete Metrics

The metrics feature that has been available in MagicDraw 17.0.5 and earlier, is deprecated, thought you can still use it, if necessary. For this, you must enable the Obsolete Metrics Plugin.



For more information, refer to MagicDraw ObsoleteMetrics Plugin UserGuide.pdf.

Analyzing Package Dependencies

An element depends on the used project when at least one of its metaproperties reference elements from at least one share of that used project. In such a case the element has a dependency from the used project.



You can define options for the dependency checking. For this, open the **Project Options** dialog, select **General Project Options** in the option group list, and then find the **Dependency Checker** options subgroup.

To analyze package dependencies of the whole project or between the selected package/model and shared packages

1. Choose one of the following:

To analyse package dependencies of the whole project	On the **Analyze** menu, click **Dependency Checker**.
To analyze dependencies between the selected package/model and shared packages	Right-click the package or model in the Model Browser or on the diagram pane, point to **Tools** and then click **Dependency Checker**.

2. The **Dependency Checker** dialog appears. Click **OK.**
3. Dependencies between the selected package/model and shared packages will be analyzed and shown in the opened **Package Dependency** panel.



For more information about the Package **Dependency** panel, see "Package Dependencies panel" on page 606.

Copyright © 1998-2015 No Magic, Inc.

To analyze package dependencies while exporting or sharing packages

1. Choose one of the following:

To analyze package dependencies while exporting	From the package/model shortcut menu, choose **Project Usages** > **Export Packages to New Project**. The **Export Packages to New Project** dialog appears. Select the packages you want to export. For more information, see "Exporting packages to new projects" on page 149.
To analyze package dependencies while sharing	From the package/model shortcut menu, choose **Project Usages** > **Share Packages**. The **Shared Packages** dialog appears. Select the packages you want to share. For more information, see "Sharing project data" on page 150.

The question dialog appears asking for your confirmation to start dependency checking between the exported package/model and the rest of the project (including shared packages that belong to the project and used projects).

2. If you want to discover *cyclic dependencies*, select the **Check for cyclic dependencies among used projects** check box and click **Yes** to start dependency analysis. If the **Check for cyclic dependencies among used projects** check box is selected, only dependencies which have **Error** and **Warning** severity levels are displayed.



If there is a chain of dependencies such that:

A → M(1), M(1) → M(2), M(2) → M(3), ..., M(X) → M(A), where:

- A is an element from the used project M(A)
- M(1..X) are other used projects
- A → M(x) is element A dependency on used project M(x)
- M(y) → M(x) is a dependency of at least one element in used project M(y) on used project M(x),

then this chain is called a cyclic dependency and every atomic dependency in this chain is considered as part of cyclic dependency.

3. Dependencies between the selected package/model and shared packages will be analyzed and shown in the opened **Package Dependency** panel.



For more information about the Package **Dependency** panel, see "Package Dependencies panel" on page 606.

Copyright © 1998-2015 No Magic, Inc.

Package Dependencies panel

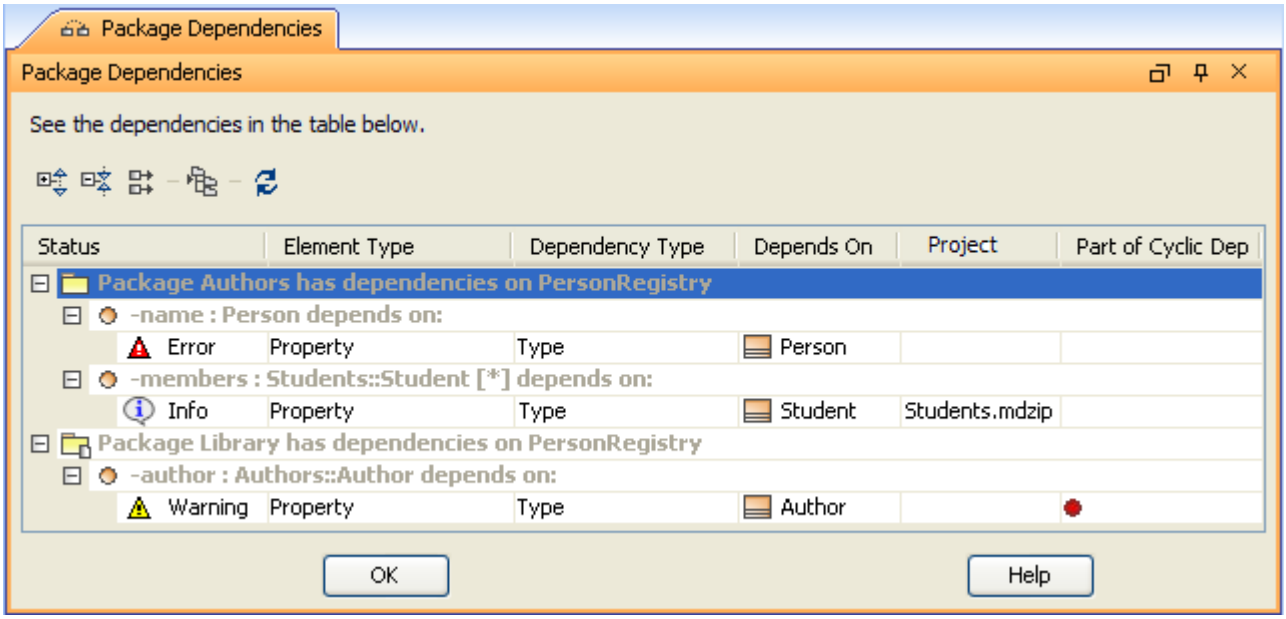


Figure 366 -- Independently opened Package Dependencies panel

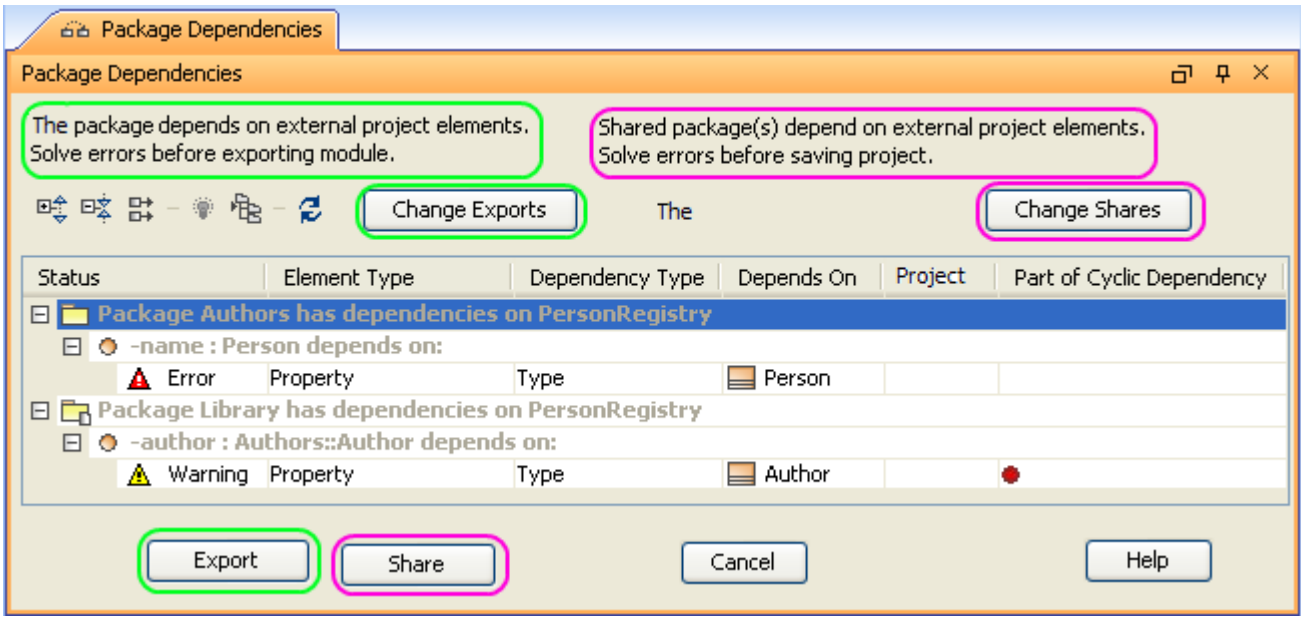


Figure 367 -- Package Dependencies panel, opened when exporting (highlighted in green GUI elements) or sharing package (highlighted in pink GUI elements)

The **Package Dependencies** panel has a table which shows the list of dependencies and buttons for managing data displayed in this table.

Button	Description
Expand All Tree Branches	Expands all nodes in the package dependencies tree.
Collapse All Tree Branches	Collapses all nodes in the package dependencies tree.
Show/hide the Full Path Names	Displays the element full path next to the element name.

Copyright © 1998-2015 No Magic, Inc.

Button	Description
Solve	The button is enabled, when a dependency whose status is **Error** is selected in the table and a solution for the problem can be found. Clicking the button opens the dialog for choosing the solution for a specific dependency problem. This button is only visible when exporting or sharing a package.
Select in Containment Tree	Shows the selected element in the Browser. The button is enabled, when a dependency is selected in the table.
Refresh	Performs dependency analysis and refreshes the dependency table with the new analysis results.
Change Shares	Opens the **Shared Packages** dialog for reselecting the packages to be shared. This button is visible only when sharing a package.
Change Exports	Opens the **Export Package to New Project** dialog for reselecting the packages to be exported. This button is visible only when exporting a package.
OK	Closes the **Package Dependencies** panel. This button is available when the dependency checker is opened independently by selecting **Analyze** > **Dependency Checker** from the main menu or **Tools** > **Dependency Checker** from the package/model shortcut menu.
Share	Closes the **Package Dependencies** panel and makes the package shared. The button is available, when the **Check Dependencies on Package Export/Sharing** environment option is set to **Allow dependencies** (to open the **Environment Options** dialog, select **Options** > **Environment**).This button is visible only when sharing a package.
Export	Closes the **Package Dependencies** panel and opens the **Save as**/ **Commit Settings** dialog for saving/committing the package as separate project. The button is available, when the **Check Dependencies on Package Export/Sharing** environment option is set to **Allow dependencies** (to open the **Environment Options** dialog, select **Options** > **Environment**). This button is visible only when exporting a package.
Cancel	Cancels package sharing or exporting. This button is visible only when exporting or sharing a package.
Help	Opens MagicDraw Help.

Copyright © 1998-2015 No Magic, Inc.

Column	Description
Status	Shows severity of the element dependency problem. The status can be **Error**, **Warning**, or **Info**. Dependencies that have **Error** status: • used project dependencies on the project Dependencies that have **Warning** status: • cyclic used project dependencies on other used projects Dependencies that have **Info** status: • project element dependencies on elements from shared packages (shared packages can belong both to the project and come from an external project)
Element Type	Displays element type.
Dependency Type	Displays dependency type.
Depends On	Displays the model element, on which the package/model element is dependent.
Project	Displays the name of the used project file that owns the model element, on which the package/model element depends.
Part of Cycling Dependency	indicates that the element is a part of the cycling dependency.

Analyzing Dependencies Among Elements

A package can be exported to an independent module[8] only if it does not depend on external elements (except other modules). Cyclical dependencies between several modules are not allowed.

There are three types of dependencies:

- Dependency by relationship
- Dependency by reference
- Diagram dependencies

Package dependencies by relationship

The module depends on external elements

If a module element has a relationship with an external element and this relationship is contained in the module package, an error message appears when exporting the module.

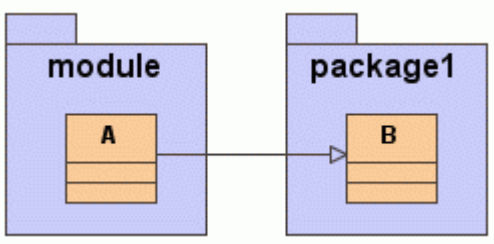


Figure 368 -- Example of a module dependency on an external element

8. Starting from version 18.1, referred as “used project” in MagicDraw UI. This section (including subsections) mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

Such dependencies on external elements are displayed in the Browser tree:



Figure 369 -- Package has a dependency on an external element

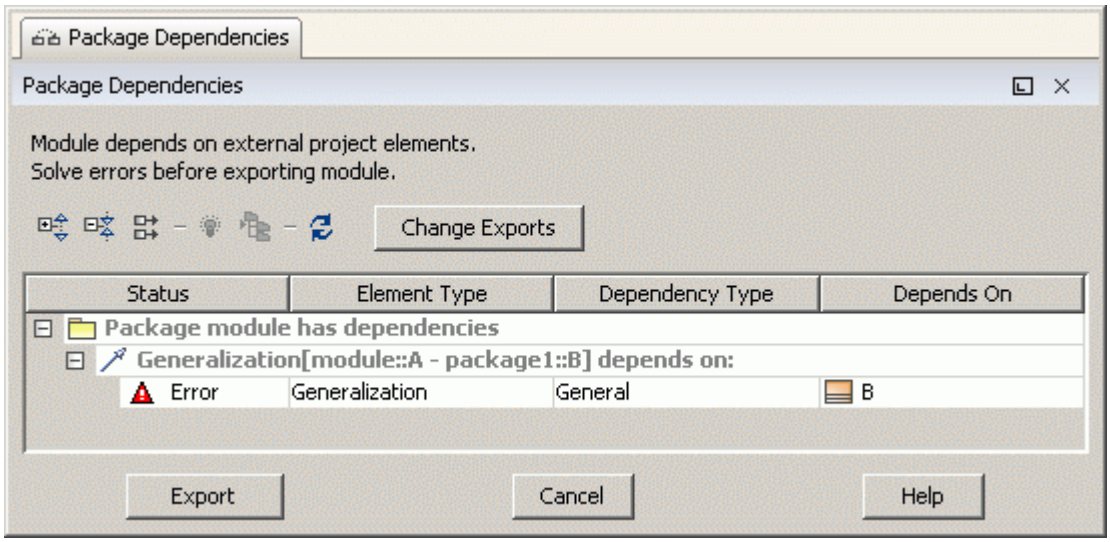


Figure 370 -- Error in the Package Dependencies dialog

In this situation, MagicDraw can suggest moving the relationship into the parent package of this external element. For example, **package1** is a parent of class **B**, so the relationship can be moved from the **module** into **package1**:

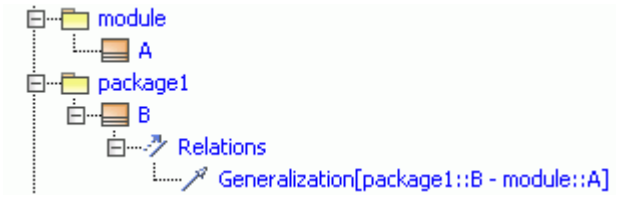


Figure 371 -- Resolved package dependency on an external element

Some movements can be achieved by clicking **Solve** in the **Package Dependencies** dialog. For a detailed description of this dialog, see "The module package can now be exported into an independent module." on page 611.

You can also drag-and-drop the relationship from one package to another in the Browser tree.

Copyright © 1998-2015 No Magic, Inc.

The module depends on an external element, but can be exported (with warning)

Though the module element has a relationship with an external element, this relationship is contained in an external package:

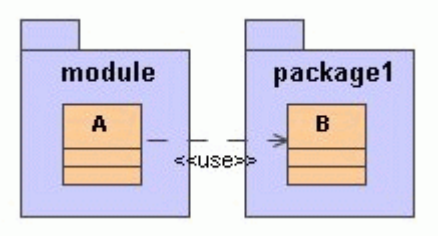


Figure 372 -- Example of a "legal" module dependency on an external element

In this case, the dependency on an external element is displayed in the Browser tree:

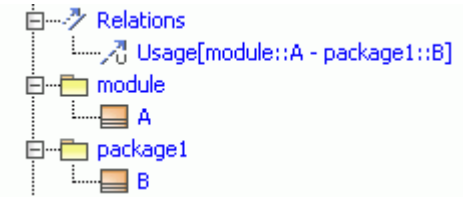


Figure 373 -- "Legal" module dependency on an external element in the data model browser

The package can be exported as a module because the relationship is contained in an external package.

The module does not depend on an external element

If the module element has a relationship with an external element is irrelevant in the context of UML (for instance, the external model uses the module, but not vice versa) and this relationship is contained in an external model, the package can be exported into an independent module:

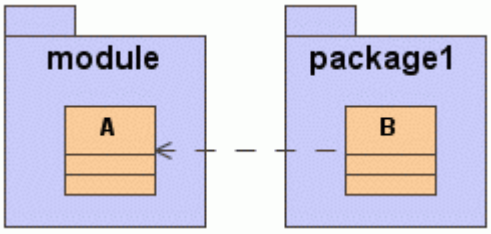


Figure 374 -- Example of a relationship when the module does not depend on an external element

Dependencies by reference

The module depends on external elements when the model elements from the module packages have references to external elements.

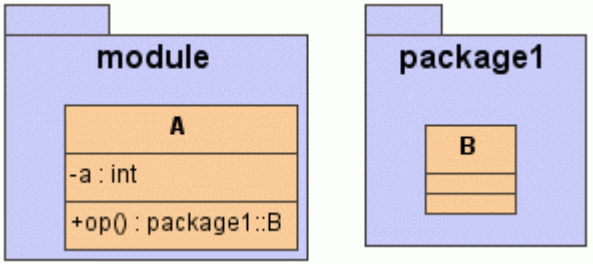


Figure 375 -- Example of a dependency by reference

Copyright © 1998-2015 No Magic, Inc.

In this case, the **module** package cannot be exported to an independent module.

Diagram dependencies

The diagram depends on all model elements displayed within it.

If the diagram is contained in a **module** package and depends on external elements, this package cannot be exported to a module.

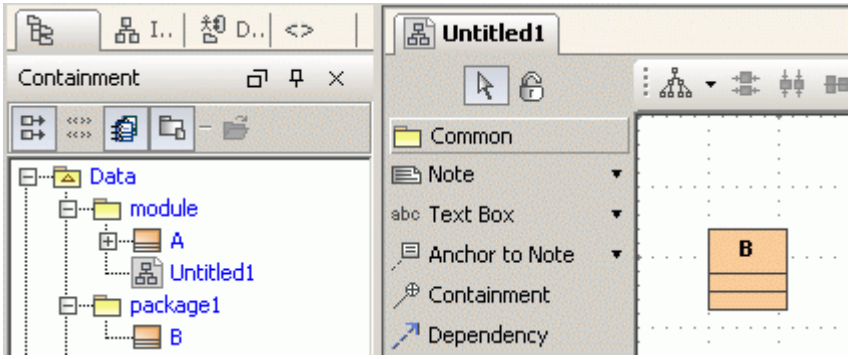


Figure 376 -- Example of the relationship when a diagram depends on an external element

For more information about the package dependencies on external elements, see "The module depends on external elements" on page 608.

In this case, if the diagram is not important to the module, it can be moved from the **module** package into any external package by dragging and dropping it within the Browser tree:

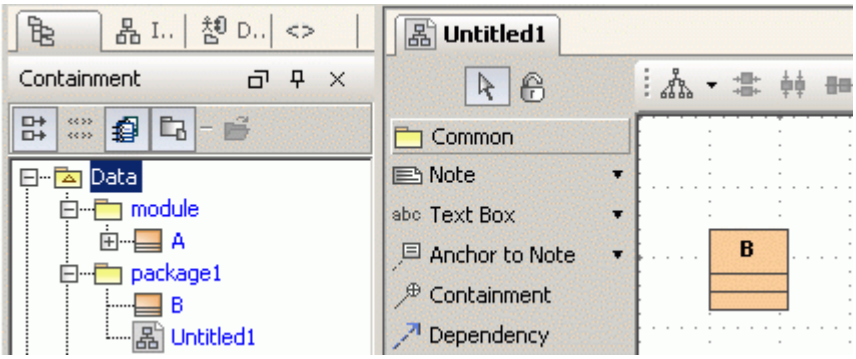


Figure 377 -- Diagram is moved from the module package to the package1 package.

The **module** package can now be exported into an independent module.

MagicDraw Teamwork Server is an ideal solution for a collaborative work on the same project. For more information about using MagicDraw in the collaborative environment, see "Working with Server Projects" on page 1039.

Unresolved dependencies

When a model part is exported to a separate module[9], if there are dependencies from the module back to the project, you are asked to resolve them (dependencies in the opposite direction - elements in project depending on elements in module - are OK).

9. Starting from version 18.1, referred as “used project” in MagicDraw UI. This section mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

The same situation occurs when you edit the module inside the project (when the module is mounted read-write on the project) and introduce dependencies from the module back to the project. In this case, you will be asked to resolve these dependencies on module save.

However, it might be inconvenient to resolve these dependencies at that moment (perhaps you have finished work for today and you will resolve dependencies tomorrow, and now you just want to save the project and leave; perhaps the particular dependency resolution is not a trivial task, which will take some time).

MagicDraw allows you to continue without resolving these dependencies. The elements, which were referenced, but are missing in the module will be shown as recovered elements (see "Maintaining Decomposed Model Integrity" on page 164).

This is one more improvement - in previous versions MagicDraw was strict in checking dependencies and did not allow dangling references. Now more flexibility is allowed.

This behavior is controlled by the **Check Dependencies on Package Export/Sharing** environment option (from the **Options** menu, select **Environment**, **General** section). There are three choices:

- **Do not allow dependencies** setting restores previous, strict checking.
- **Allow dependencies** is the default setting, described above.
- **Do not check** setting is an even more lax setting; it does not prompt the user to resolve dependencies at all. If you are not careful, this can lead to the proliferation of missing proxy elements, hence proceed with care.

Validation



Validation



This functionality is available in Architect and Enterprise editions only.

Introduction

MagicDraw has the functionality to check the created models. It consists of:

- A set of validation rules. Each validation rule captures some imperative conditions, which must be checked against the model. Validation rules are specified as invariant constraints in the model.
- One or more validation suites (modeled as packages). A validation suite is a simple concept of grouping the validation rules into meaningful groups, so that the collection of rules can be applied.

To run the validation, select some suites and validation scope - either the entire model or some part of it. When the validation is run, each rule from the suite is evaluated for each suitable element in the validation scope. Each element that violates the rule (constraint evaluates to false) is reported in the results table.

Since rules and suites are model elements, they can be manipulated using the standard MagicDraw modeling means - they can be copied, moved, and edited in the model; they can be refactored into modules[10], to

10. Starting from version 18.1, referred as “used project” in MagicDraw UI. This section (including subsections) mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

facilitate reuse in other projects, placed in the Teamwork Server for exchange, etc. And of course this approach allows editing predefined rules and defining new, custom rules for models and profiles.

Constraint Types

Each validation rule, modeled as a constraint has a target classifier property. This property determines on what type of element this rule applies. Thus the usual level - metalevel separation appears. Constraints that are defined on some particular classifiers are evaluated on the instances of these particular classifiers when validating. Inheritance is taken into account - instances of the subclasses of the class are also validated.

Thus there are 3 types of constraints that MagicDraw can evaluate:

- **Classifier level constraints.** Constraints that are placed on the classes, datatypes and other classifiers of the model are evaluated on all the instances of these classifiers - i.e. those InstanceSpecifications that have the particular classifier set as their type.
- **Constraints on metaclasses.** When a constraint is placed on a metaclass (one of the classes in the *UML Standard Profile::UML2 Metamodel*), this constraint is evaluated on all the model elements of that kind. E.g. if the constraint is placed on Actor metaclass, then this constraint applies to all the actor elements in the model. The following is an example of rule (specified in OCL2.0), which mandates that all actor names in the model must be capitalized:

  context Actor inv capitalize:

  let startswith:String = name.substring(1,1) in

  startswith.toUpper() = startswith

  These constraints are useful for specifying generic rules, which must apply on all the model elements of particular kind.
- **Constraints on stereotypes.** When a constraint is placed on some stereotypes of the profile, that constraint applies to all the model elements that have these stereotypes applied to them. These constraints are useful when creating domain specific profiles. When adapting UML to some specific modeling domain, a profile is usually created with extensions for that domain - stereotypes, tags etc. The constraints on these stereotypes allow enforcing the rules of that domain.

It is advisable not to mix the constraints from different metalevels into one suite (constraints on classifiers versus constraints on stereotypes and metaclasses).

Predefined Validation Suites

There are several validation suites (collections of validation rules) predefined in the profiles that come with MagicDraw. Since validation rules and validation suites are concepts, stored in the model, the availability of list of validation suites for validating depends on what profiles the model includes.

UML Standard Profile brings two predefined suites with it. These two suites are present in all models:

- UML completeness constraints;
- UML correctness constraints.

Completeness suite has a collection of rules, which check if a model is complete, that there are no gaps, and the essential information fields in the elements have been filled in (e.g. checks that all the properties have type specified etc.).

Correctness suite has a collection of rules, which check common mistakes while modeling in UML2 (NOTE: this collection is not exhaustive).

Copyright © 1998-2015 No Magic, Inc.

Additionally, there are validation suites for each of these modeling domains - XML schemas, DDL, Java, C++ plus DoDAF and SysML, if any. These validation suites are defined in the corresponding profiles of these modeling domains, hence they are included automatically when you start modeling in that domain.

EXAMPLE

If you create a new XML schema diagram, XML schema profile will be automatically included in your model and this profile brings in XML schema validation suite with it. So, from that moment, XML schema validation suite is available in the project.

Validating

To run the validation, you have to select a group of rules to be validated (validation suite) and indicate which part of the model to check (validation scope - either the entire model or some part of it).

To validate UML model for correctness

1. Open the **Validation** dialog. Do one of the following:
   - From the **Analyze** menu, select **Validation** and then click **Validate**.
   - On the **Validation** toolbar, click .

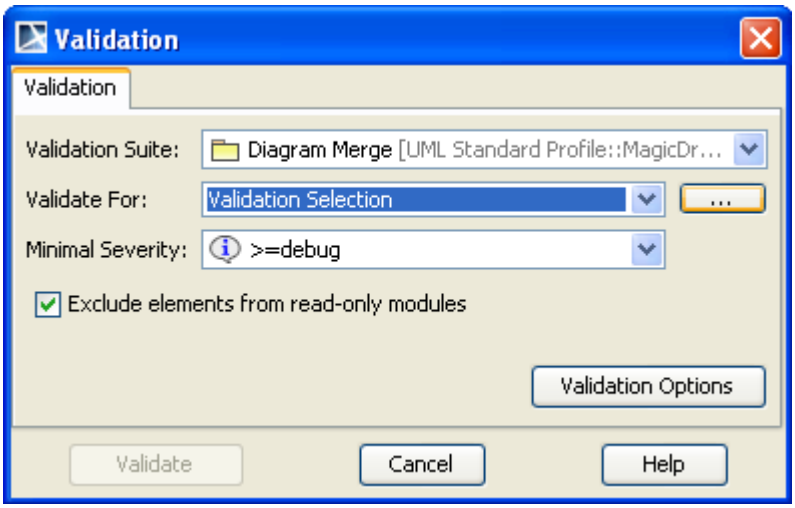


Figure 378 -- Validation dialog

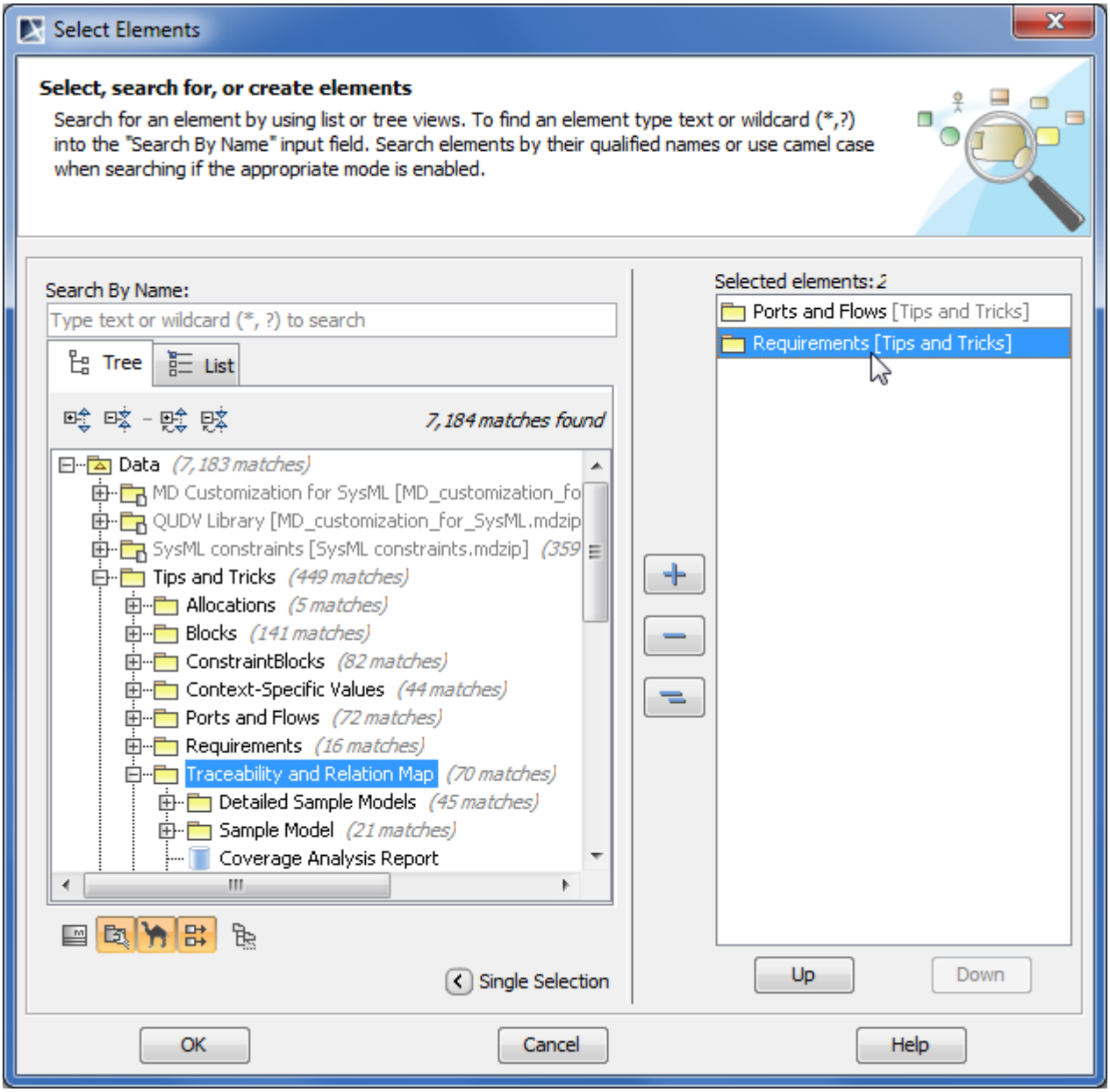
2. In the **Validation Suite** box, select the UML correctness constraints. All available validation suites are listed here.

The list of available validation suites depends on the opened project - the validation suites and validation rules are stored in the model as normal model elements. By default, a project has two suites - UML completeness constraints and UML correctness constraints - defined in the Standard profile.

If a project uses other profiles/modules - such as Java/XML schema/DDL profile, these profiles bring in their own predefined suites. You can also create your own validation rules and group them into a suite and this suite will be available in this box.

3. In the **Validate For** box, choose the validation target:
   - To run validation on the entire model, choose the **Whole Project**.
   - To run validation on the special packages and/or elements, choose the **Validation Selection,** then click the **...** button and select the scope for validation. The **Select Elements** dialog appears.

Copyright © 1998-2015 No Magic, Inc.



4. *Select Elements dialog* In the **Select Elements** dialog, select packages and/or elements for validation. Click **OK**.
5. Select the **Minimal Severity** level. Debug is the lowest possible severity level, all validation rules will be run.



Validation is always recursive, hence if you select a package for validation, you do not need to select its inner elements (no need of the **Add Recursively** button). In the case depicted above, all model elements in the *Classifiers Demo* and *Meta Demo* packages will be validated. Adding *Data* package is equivalent to validating the entire model.

6. If you want to run validation on the read-only modules and the elements that exist in those modules, clear the **Exclude elements from read-only modules** check box (by default it is selected).
7. Click **Validate**. Validation results are displayed in the **Validation Results** panel.

Copyright © 1998-2015 No Magic, Inc.

Validation Results Panel

Validation results are displayed in the **Validation Results** panel.

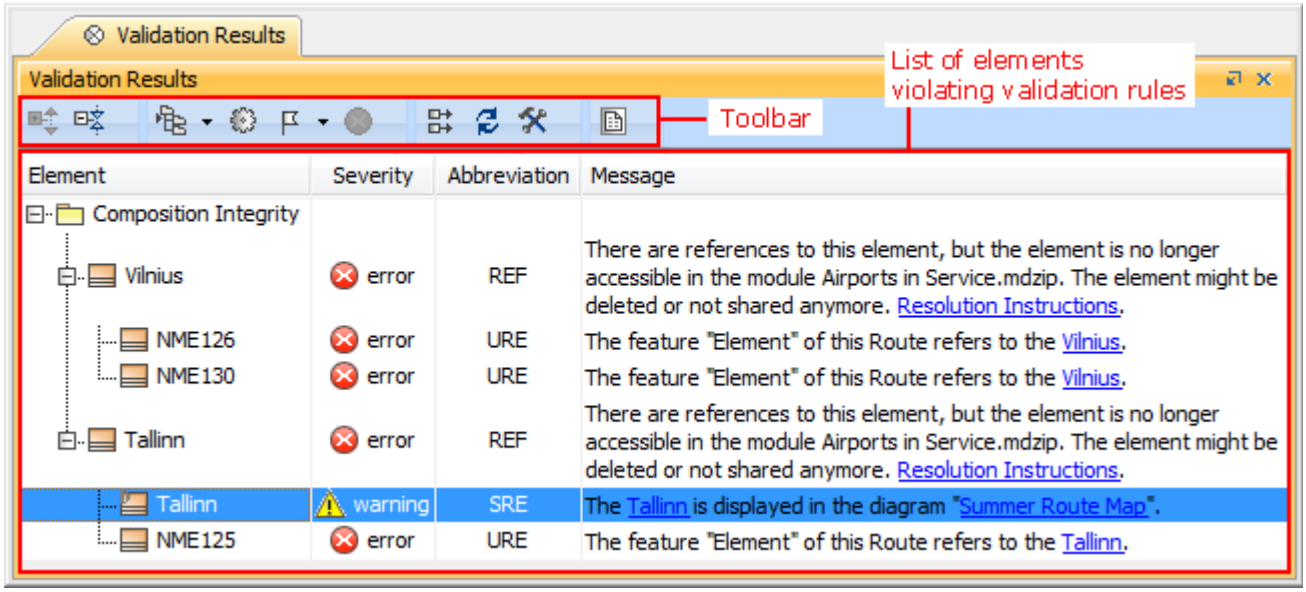


Figure 379 -- Validations Results panel

The **Validation Results** panel opens automatically after the validation has ended.

The panel includes

- Toolbar
- List of elements violating particular validation rules. The elements in the list are grouped by validation suite.

The following table describes the toolbar buttons.

Button	Button name	Description
	Expand All	Click to expand recursively all the elements in the list.
	Collapse All	Click to collapse recursively all the elements in the list.
	Navigate to validated object (ALT+B)	Click the button to select the element in the Model Browser. Click the arrow near the button to open the list of navigation commands that are available for a selected element.
	Solve	Click to open the list of available commands to solve a selected element violation.
	Ignore	Click to make a selected element violation ignored. **NEW!** In server projects, you can select whether to ignore the violation for you as a project user or for all users who works on that server project. The button is available, when an unignored element is selected in the list.
	Not Ignore	Click to make a selected element violation unignored. The button is available, when an ignored element in selected in the list.

Copyright © 1998-2015 No Magic, Inc.

Button	Button name	Description
	Toggle either to show the full path of the elements or not	Click to turn showing the full path of the elements on or off.
	Run validation with current settings	Click to run validation with current settings.
	Run validation with new settings	Click to run validation with new settings. The **Validation** dialog opens.
	Export	Click to export the list of validation results either to an .html file or to a .csv file.

The following table describes the columns of the validation results list.

Column name	Description
Element	Elements, which violate the constraint rule.
Severity	Rule severity violations.
Abbreviation	Simple short strings showing the abbreviation of the violated constraint. Mostly used for sorting/grouping.
Message	Texts describing the violations.
Is Ignored	Indicator whether the violation is ignored or not.

Sorting and filtering data in the Validation Results panel

You can sort or unsort the list by any column.

To sort the list

- Click the header of a column by which you want to sort the list data. A small arrow appears on the column header. This arrow shows, how data are sorted: ascending or descending.

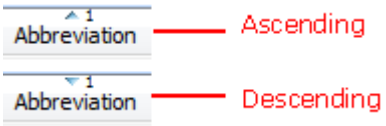


Figure 380 -- Data sorting by abbreviation

You can also filter the list by any value of every column.

To filter the list

1. Point to the header of a column whose value you want to select for filtering the list data.
2. Click the button at the right of the header and select a value from the drop-down list.

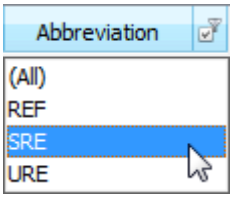


Figure 381 -- Data filtering by abbreviation

Copyright © 1998-2015 No Magic, Inc.

Case study: Model Validation

We will validate SysML model - SysML.mdzip for correctness and completeness. The model is located in MagicDraw installation directory/samples. This model requires that SysML plugin would be installed. This can be done from **Help** menu **Resource / Plugin Manager**.

1. Open model. From the **Analyze** menu, select **Validation** command, and then **Validation**. The **Validation** dialog opens.
2. Select **SysML Validation Constraints suite** and click **Validate**. The **Validations Results** view opens.
3. Narrow the validation scope. Click the **Run validation with a new**

   settings button, the **Validation** dialog opens. Change the scope to validation selection and select **SI Value Types** package, click **Validate**. The **Validations Results** view is refreshed with new options.
4. Double click the violating element *kg* in the results view or click **Select in**

   Containment Tree button. The element is selected in the model.

5. Fix the problem and click the **Refresh** button to rerun the validation suite with the same options and refresh validation results. Refreshed results do not include element kg.
6. Select other violating element N in the results view and click **Open all**

   Diagrams Containing the Selected Elements button to open all the diagrams, containing the selected model elements.

When the validation result view is opened and diagrams are shown on the screen, elements and links of the diagram, which have at least one result in the validation result view, are highlighted:

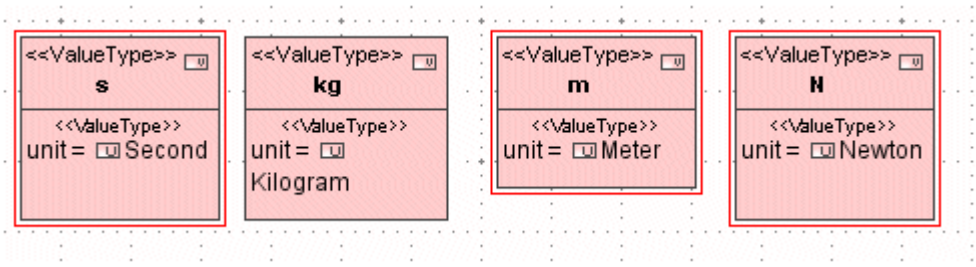


Figure 382 -- "N" element violating validation rule is highlighted

In general, highlighting depends on the validation rule severity:

- Error - red
- Warning - yellow
- Debug - gray
- Info - black

Copyright © 1998-2015 No Magic, Inc.

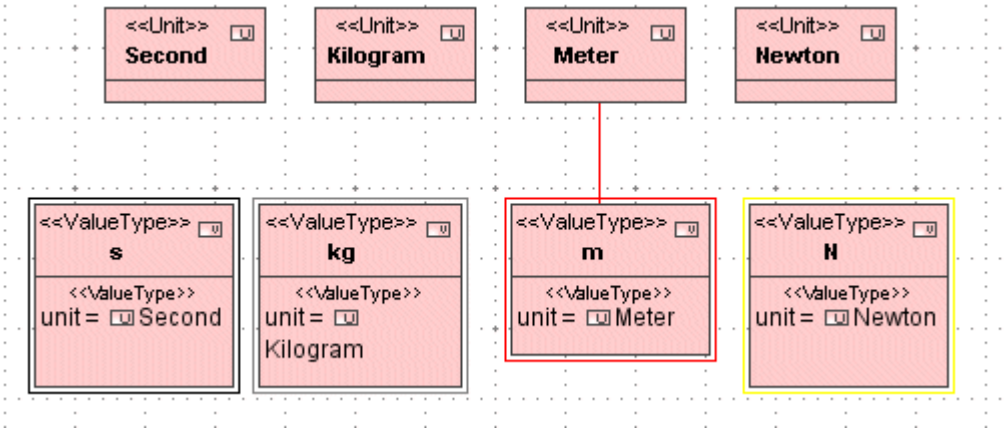


Figure 383 -- Elements and links violating constraints are highlighted.

If the element violates multiple rules, the color of the most severe rule is used.

Highlighting is shown until the rule is violated or the Validations Results view is not closed.

NEW! The results of the failing validation rules are marked in tables.

- In the generic tables, the entire row is marked.
- In the instance tables, either cell, or entire row is marked. A cell is marked in case of a faulty slot value. The entire row is marked in case of a faulty instance.

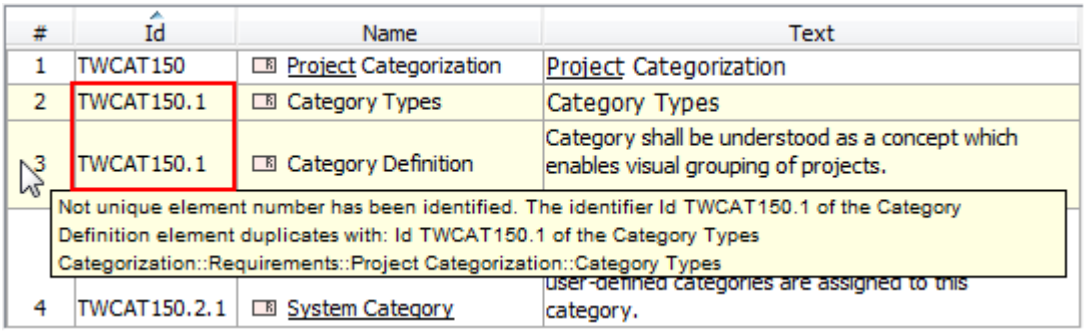


Figure 384 -- Rows marked because requirement ID's are non-unique

Other features available for the Validations Results view:

- By clicking the **Select Rule in Containment Tree** button, you may select rule in the model.
- You may move the selected results of the validation to the search result window by clicking the **Move to Search Results** button.
- The **Open all Diagrams Containing the Selected Elements** button opens all the diagrams, containing the selected model element.
- The Show / hide the Full Path names button shows element location in the model path.

Validation Rules

The validation rules are modeled as UML2 constraints. This approach allows treating the validation rules as simple model elements. They can be handled using usual modeling mechanisms. They can be copied, moved around in the model, refactored into a separate module, stored in the teamwork server for easy information exchange, etc.

Copyright © 1998-2015 No Magic, Inc.

Since constraints can have different semantic meanings in UML2, a special type of the constraint – invariant constraint is used for modeling validation rules.

To distinguish these constraints from the other types of constraints, *«invariant»* stereotype should be applied to them.

Additionally, validation rules require other pieces of information – severity level (for sorting/filtering), abbreviation string (a short string, for easy recognition) and error message (complete description of error explanation). This information is displayed in the validation result view.

For storing this information, a special stereotype *«validationRule»*, derived from the *«invariant»,* is used. If you want to run this constraint as validation rule, use the former stereotype. If you have created the constraint just for documentation purposes and do not intend to run it, the latter constraint is sufficient.

Validation rules can be placed anywhere in the model (where UML2 constraint can be placed), however, usually they are stored in the classifier, which is constrained – classes, datatypes, etc. (for classifier level constraints), stereotypes (for meta-classifier level constraints). This convention breaks down for the constraints, placed on metaclasses (since these classes are stored in read only profile). In this case, place constraints wherever you like (e.g. group them into a package).

To create a validation rule for a metaclass

1. Select a package where you want to store the rule.
2. Right-click this package, select **Create Element** > **Constraint**.

To create a validation rule for a classifier or stereotype

1. Open the Specification window of that classifier or stereotype.
2. Click Constraints on the left side of the dialog.
3. Click the **Create** button on the right side of the dialog and then select **Constraint**.

Case study:

Let’s say we have 2 stereotypes - *«product»* and *«part»*. We want to place a validation rule, that products must have at least one part in them.

Copyright © 1998-2015 No Magic, Inc.

1. Open the product stereotype specification, select *Constraints* section, and click **Create**.

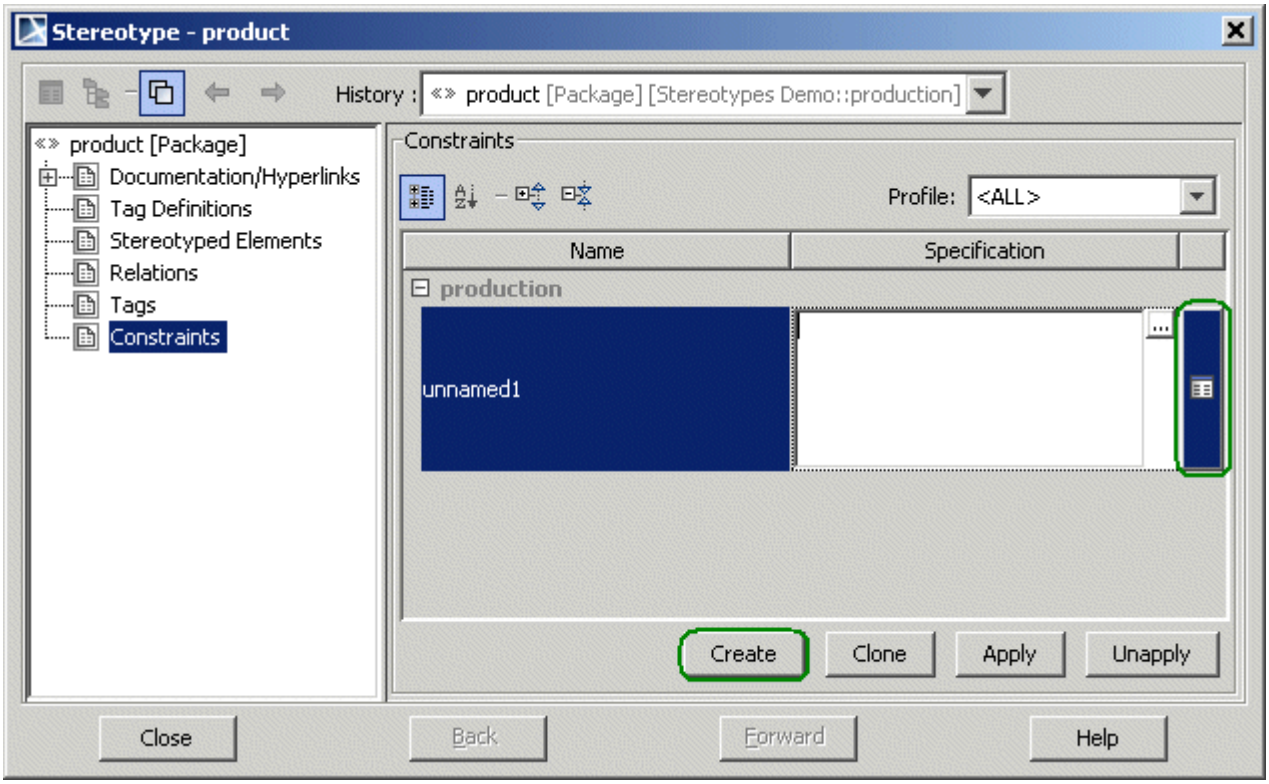


Figure 385 -- Creating constraints

Copyright © 1998-2015 No Magic, Inc.

2. The name of the constraint and the expression can be specified right away, but since we want to specify more information, we need to open the specification of a newly created constraint (press the button on the right of the constraint).

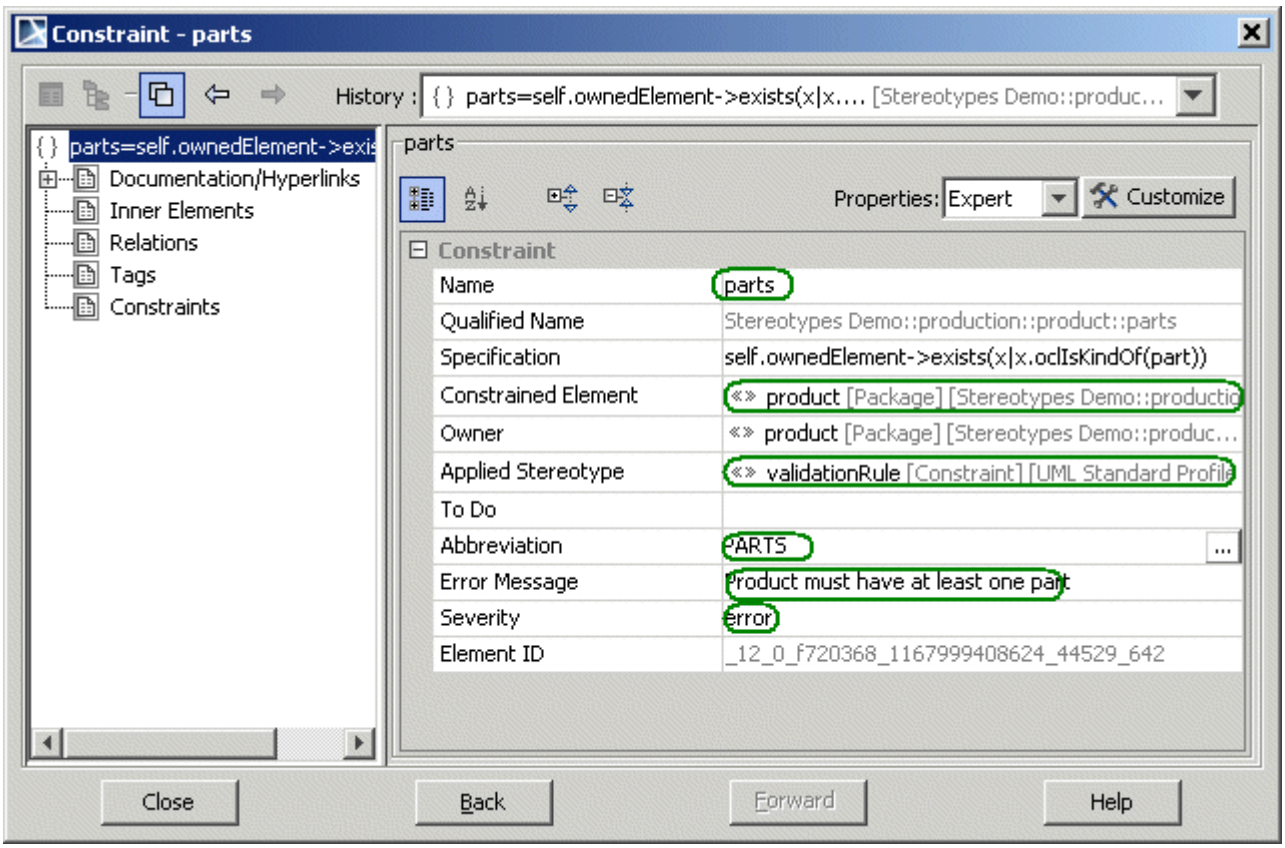


Figure 386 -- Specifying details of the constraint

3. In the specification panel of the constraint, specify the constraint name. Then ensure that the *Constrained Element* field points to the necessary classifier (*product* stereotype in our case). If we have created the constraint as described here, this field will be filled automatically. If we have created the constraint through the right-click, *Create Element*, *Constraint* route (e.g. constraint for metaclasses), we will need to specify the constrained element manually. For constraints on metaclasses select the appropriate element from the *UML Standard Profile::UML2 Metamodel*. In UML2 the constrained element field is multivalued, but only single value is supported for validation rules.
4. Now apply the «*validationRule*» stereotype on this constraint. Additional fields will open on the pane (*Abbreviation*, *Error Message* and *Severity*). If these fields do not open automatically, click **Customize** and then **Reset to Defaults** in the open customization dialog (you can also access these fields in the Tags section of the specification).
5. Fill in the values for those fields.

Usage of the *Severity* levels (approximate guidelines):

- **DEBUG**. This severity level should be assigned only to those validation rules, which fit the description of INFO, but are too numerous and annoying to constantly bother the user.
- **INFO**. Situations, which might be interesting to the user.
- **WARNING**. Used for less severe situations than ERROR, which are not errors per se, but have a high probability of causing errors. A good example would be – In Java model user redefines equals() method of the class, but does not redefine hashCode(). This is a dangerous coding situation.
- **ERROR**. Normal error message. For ordinary, run-of-the-mill errors.

Copyright © 1998-2015 No Magic, Inc.

- **FATAL**. Used for the errors, which lead to model corruption or are not valid from the UML metamodel structure viewpoint. There should be few or no validation rules of this level since MagicDraw automatically precludes such situations. This level is mostly reserved for future use.

Abbreviation is a simple (and preferably short) string, for quickly distinguishing the validation rules among other rules and sorting. Acronyms and short forms, used in the domain of this validation rule can be used here (e.g. NPE for hypothetical NullPointerException check).

Error Message is a longer string, fully describing the invalid situation.

Now that we have all the peripheral information about the validation rule, let's specify the actual validation rule expression. Validating expression is stored as the **Specification** property value. UML2 expression has two properties – **Language** and **Body**.

MagicDraw supports the following languages for expressions, that can be evaluated:

- **OCL2.0** is used for validation rules, specified in OCL language (version 2.0 of the spec - 06-05-01 specification document from OMG).
- **Binary** is used for more advanced expressions, which are not easily expressed in OCL.These expressions are written in Java, compiled, specified in the MagicDraw classpath. Then these expressions can be specified as validation rule expressions.
- Scripts, such as **JavaScript**, **Jython**, **JRuby**, **Groovy**, and **BeanShell**.



For the script writing instructions, refer to "Creating executable opaque behaviors" on page 952.
Since constraints cannot have parameters, you may skip the information about managing parameters.

- **StructuredExpression**.



For more information, refer to "Specifying Criteria for Querying Model" on page 551.

Other languages, such as OCL1.5 or English, are not evaluatable. They can be used for documentation purposes.

The result of a validation rule must be of the boolean type.

Copyright © 1998-2015 No Magic, Inc.

OCL constraints

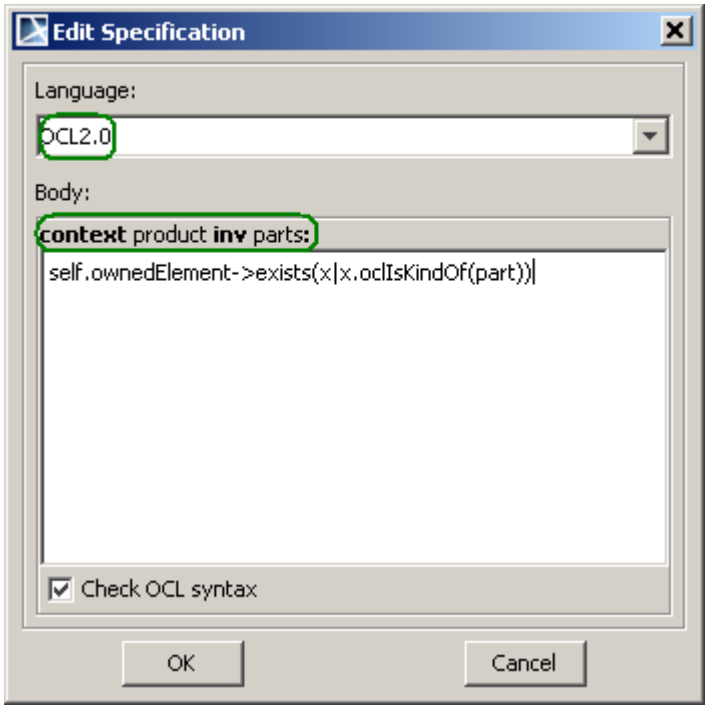


Figure 387 -- Validation rule in OCL

Continuing our example, in the constraint **Specification** dialog, click the button near the *Specification* field and open the **Edit Specification** dialog. Select **OCL2.0** language. Observe that MagicDraw has automatically generated the header of the expression from the constraint information, and we only need to specify the body of expression. The expression header is generated according to the following rules:

context <constrained element> <constraint type> <constraint name if any>:

Constraint type is one of the types, defined in the OCL2.0 spec:

- **inv** - when the expression is placed in the constraint with *«invariant»* stereotype applied.
- **def** – when the expression is placed in the constraint with *«definition»* stereotype applied.
- **init**, **derive** when the expression is placed in the default value of the property.
- **pre**, **post**, **body** when the expression is placed in the appropriate fields of operation.

Since our constraint is stereotyped with «*validationRule*» stereotype (derived from *invariant* stereotype), inv is shown in the header. Only invariant stereotype is used for the validation rules and are executed (plus derive expressions, when referenced from invariants – see "Advanced Topics" on page 628), other types of constraints can be used for documentation purposes.

MagicDraw checks the syntax of expression as you type. However this syntax check is not enough to catch all the errors. When the validation rule is run, additional checks are performed (semantic checks – such as checks for the existence of appropriate properties, type checks, multiplicity checks, etc.) to ensure that the expression can be evaluated correctly (internally, MagicDraw generates Java code from the expressions and then compiles it for execution).

Binary constraints

For more information about Binary Constraints, see MagicDraw OpenAPI UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

Validation Suites

You can create your own validation suites or use one of the predefined ones as: *UML completeness constraints* and *UML correctness constraints*.

The validation suite defines the set of validation rules, which will be applied when validating. The purpose of the validation suites is to group constraints without duplicating them.

To create a new validation suite

We need to check the Oracle model for correctness, but not all constraints in *Generic DDL constraints* suite are suitable for our Oracle model. We will create a new suite with a narrow constraint collection.

1. From the **Analyze** menu, choose **Validation** command, and then **Validation Options**. The **Validation Options** dialog opens.
2. Click the **Create New Validation Suite** button and name it *Oracle specific constraints*.
3. Define the validation rules in the **Validation Rules** pane.

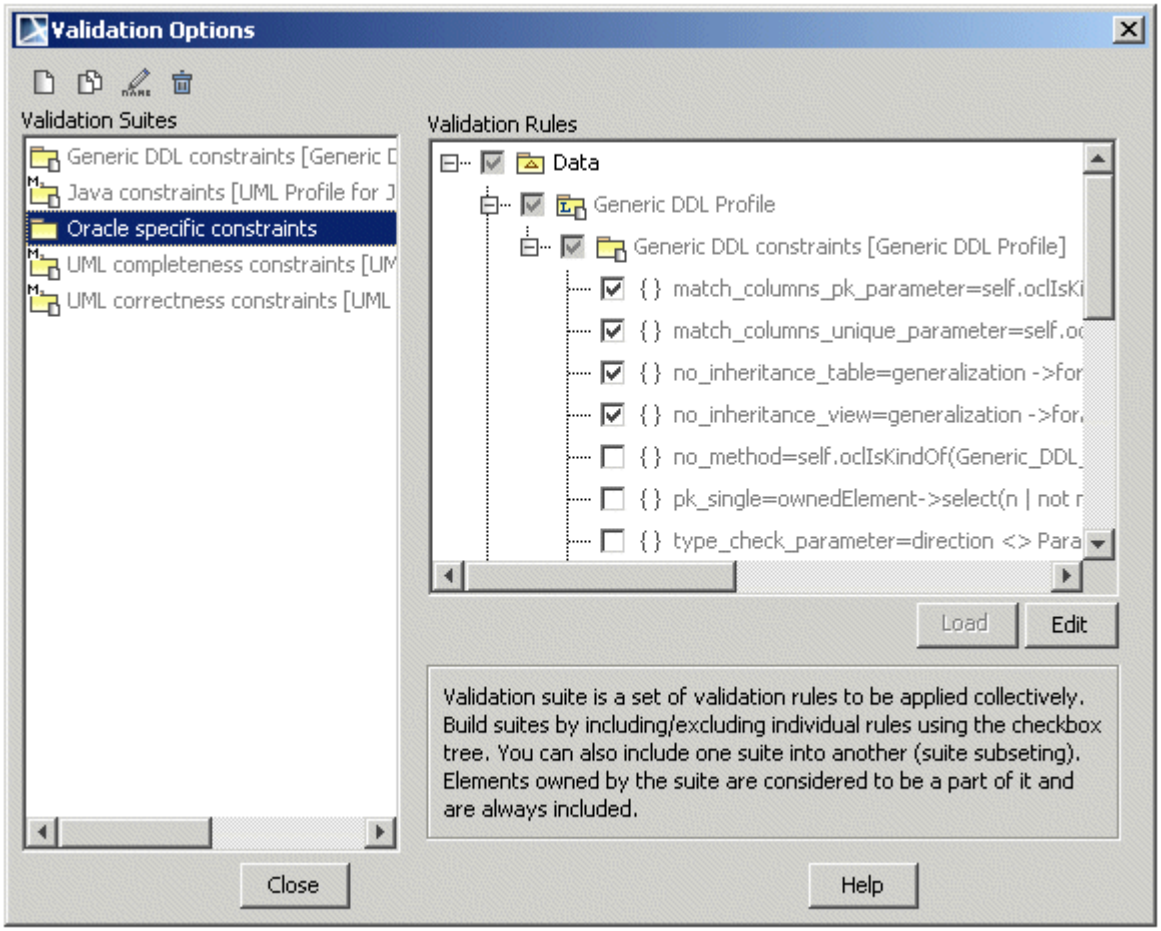


Figure 388 -- Parts of Generic DDL constraints suite are included into Oracle specific constraints suite.

The validation suite is stored in a model as a package, to which «*ValidationSuite*» stereotype is applied. The **Validation Suites** pane lists all such packages of the entire model as suites. So, the alternative method to create the validation suite is to apply «*ValidationSuite*» stereotype for a package.

Copyright © 1998-2015 No Magic, Inc.

When the user includes / excludes the constraint, the appropriate element import link is created/deleted in the model.

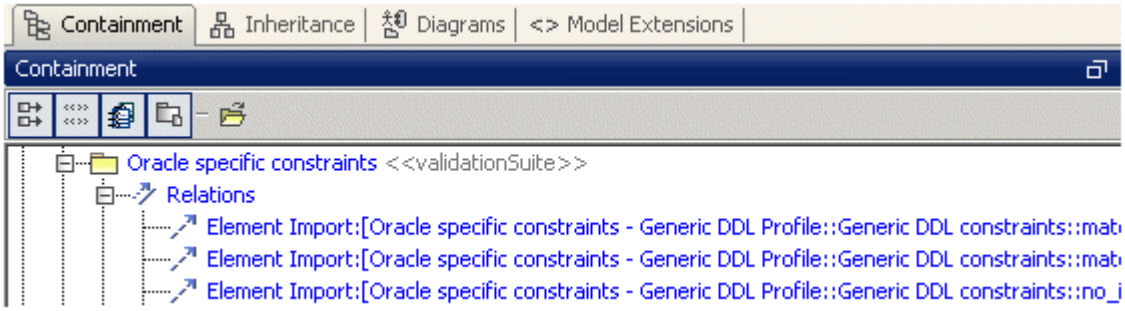


Figure 389 -- Element import relation showing in model that Oracle specific constraints suite includes other constraints.

Also, there can be constraints that are stored directly in the suite package - they are also considered as contained in this suite, and because they are physical in package those constraints cannot be excluded from the suite through **Validation Options** dialog. Typically, validation rules should be stored in a constrained element, but in cases when the constrained element is read-only, for example it is stored in a read-only profile, adding constraints to it requires profile editing and a separate constraints grouping is easier.

Constraint Tree

The constraint tree is shown in the right **Validation Options** pane. This tree shows all the constraints with «*invariant*» or «*validationRule*» stereotype applied, presented in the model, together with the appropriate grouping elements. Each item has a check box, indicating inclusion or exclusion of the constraint in the selected validation suite.

The constraint tree contains packages and other model elements. If it contains constraints, they are arranged according to their containment in a model. Additionally, this tree contains other validation suites. The user can include / exclude rules and these rules must suit the selected validation suite by selecting / unselecting these check boxes in the tree.

To group two or more suites into one

We have created an abstract system model, and modeled its implementation with Java specific classes. To check this model completeness, correctness, and conformity to Java language by using three suites. We will combine all these suites to one in order to simplify the model checking.

1. From the **Analyze** menu, choose **Validation** command, and then **Validation Options**. The **Validation Options** dialog appears.
2. Create a new suite, name it *General,* and select it.

Copyright © 1998-2015 No Magic, Inc.

3. Include the existing validation suites: *Java constraints, UML completeness constraints* and *UML correctness constraints* in the suite by selecting the check box in front of the packages in the **Validation Rules** panel.

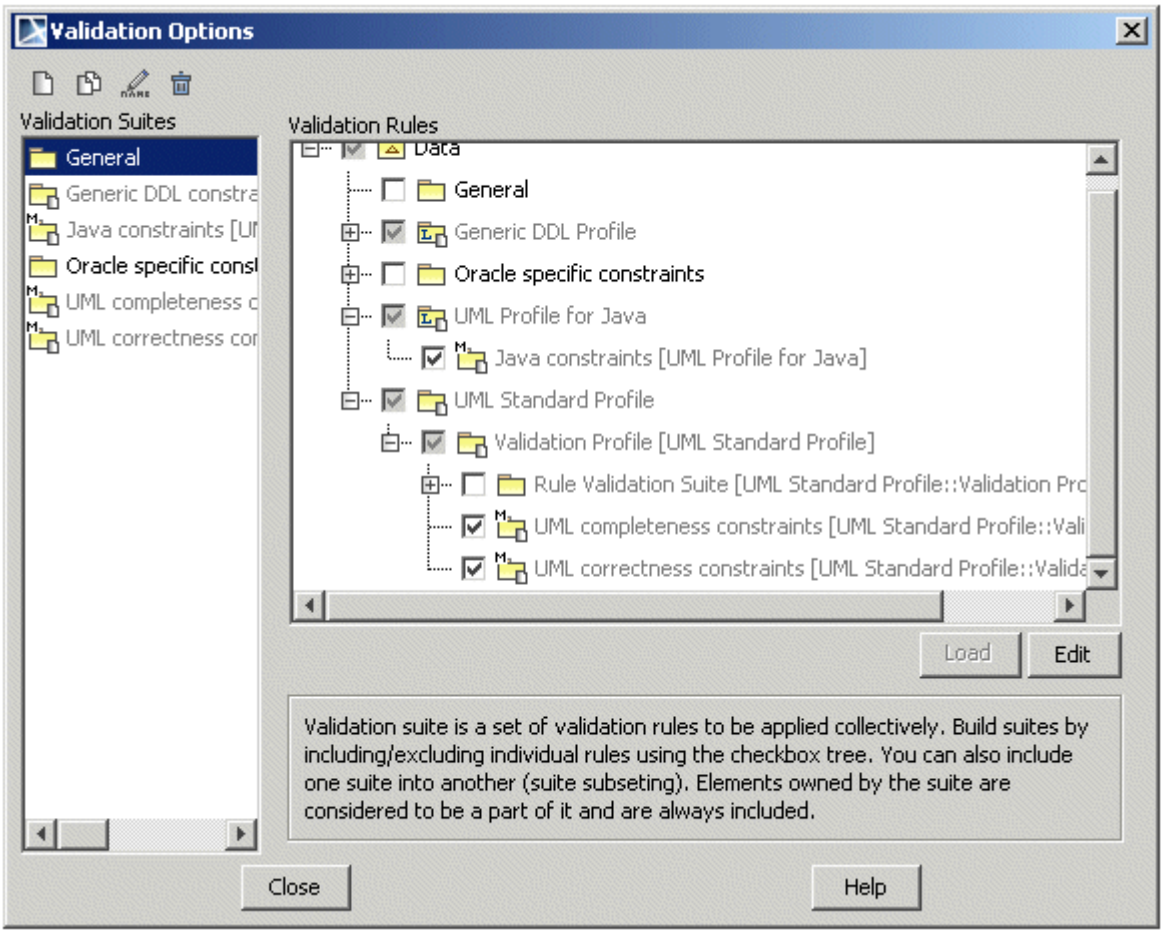


Figure 390 -- UML correctness, UML completeness, and Java validation suites included in the General validation suite

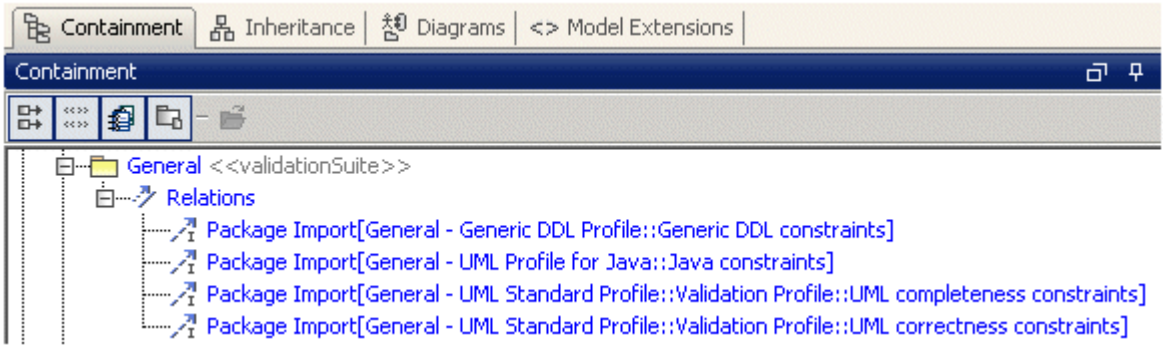


Figure 391 -- Package import is signifying in the model that one General suite includes the other as a subset.

To share constraints

Let's say we have created a validation suite with constraints and need to share it for other group members for their models validation. Validation rules/suites sharing is available through standard MagicDraw module mechanism. Package with constraints might be exported as module and used by any other project.

Copyright © 1998-2015 No Magic, Inc.

Another way of sharing constrains is copying them between projects. Since validation rules are simple model elements, any mechanism can be used on them.

1. From the model select *Java constraints* package to which «*ValidationSuite*» stereotype is applied.
2. From the package context menu select **Modules** and select **Export module.**
3. Save the exported package as *Java constraints.mdzip*.

Only constraints that are stored physically in the *Java constraints* package are exported together with the package.

Now the exported package can be used by other users and projects.

1. From the **File** menu, select **Use Module.** The **Use Module** dialog appears.
2. Select path to *Java constraints.mdzip* and select it to use.
3. Specify module import options.
4. Module is added into a project and the constraints can be used for validation.

The validation suite can be defined in the module, which is mounted as *read-only* in the project. **Read-only** and **Autoload** module will not be loaded into the project, but will be visible through validations dialog if «*ValidationSuite*» stereotype was applied to the exported package.

In this way the model and the profile will be smaller. And the validation suite is still visible through validation dialog. Java constraint validation suite module is mounted onto the project as Read-only and Autoload. In this way constraints are not added into project by default, until the validation suite is used.

.

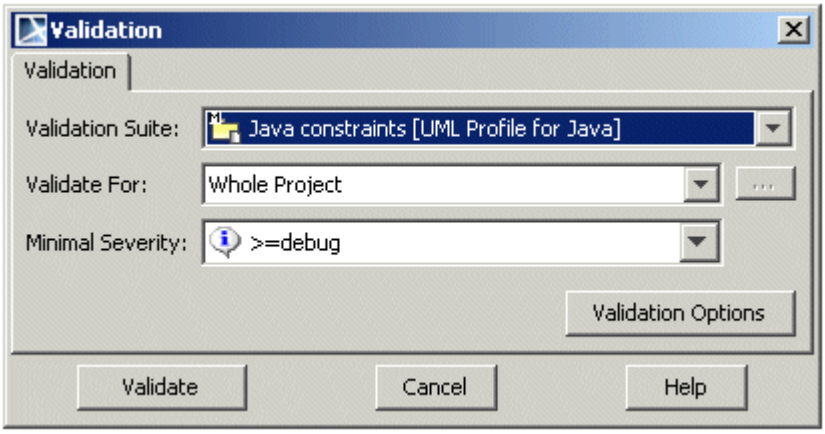


Figure 392 -- Unloaded module with Java constrains validation suite is avaialable through Validation dialog.

Advanced Topics

Global validation rules

Some of the validation rules, specified in OCL, do not refer to the current element (*self*). Such rules are often encountered when using allInstances() method to refer to all instances of a particular classifier. Such validation rules are called global validation rules in MagicDraw terminology. Evaluating these rules for each model element is pointless, hence they are evaluated only once per entire validation run. When reporting violations, such rules have a string <model> in the column of violating model elements. This means that it is not the concrete element that violates the rule, but the entire model itself.

Here is an example of such a rule (always fails):

Copyright © 1998-2015 No Magic, Inc.

```
context anything inv:
false
```

Such a rule is not very useful indeed. The following is another example:

```
context SomeSingletonClass inv:
SomeSingletonClass::allInstances()->size() <= 1
```

This rule checks that there is at most one instance of the *SomeSingletonClass* in the model. The following is a more complex example:

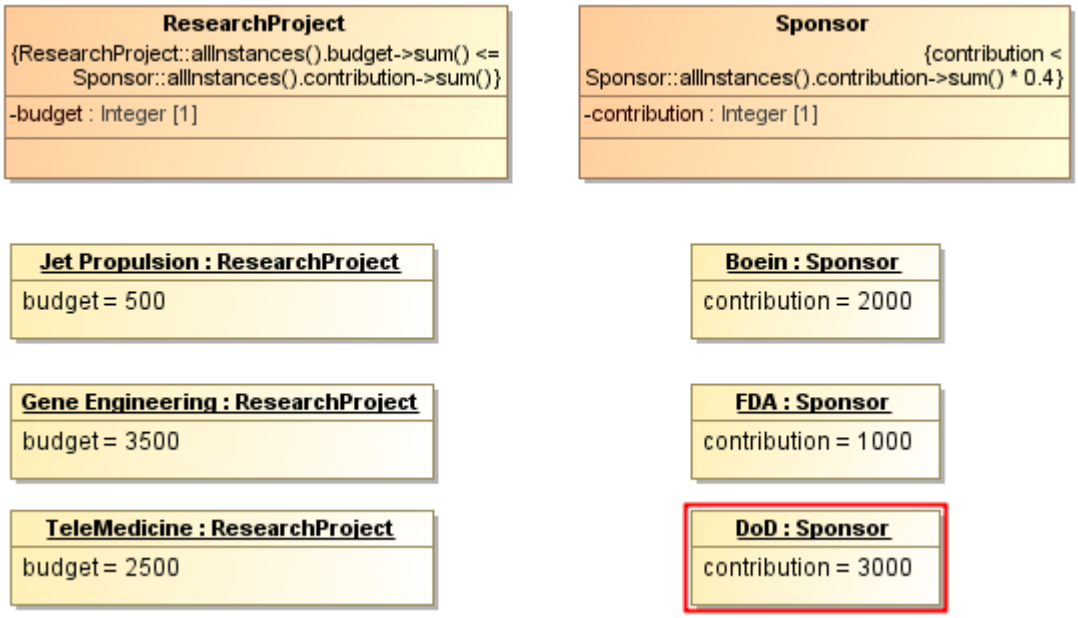


Figure 393 -- Example of global validation rule

Here, *ReseachProject* class has a following validation rule (budget must be balanced – sum of expenses of all projects must be less than sum of all sponsor contributions):

```
context ResearchProject inv balanced_budget:
ResearchProject::allInstances().budget->sum() <=
Sponsor::allInstances().contribution->sum()
```

Sponsor class has a following rule (anticorruption rule - each sponsor cannot contribute more than 40% of the funds):

```
context Sponsor inv anticorruption_law:
contribution < Sponsor::allInstances().contribution->sum() * 0.4
```


Copyright © 1998-2015 No Magic, Inc.

Now look at the results of applying these rules:

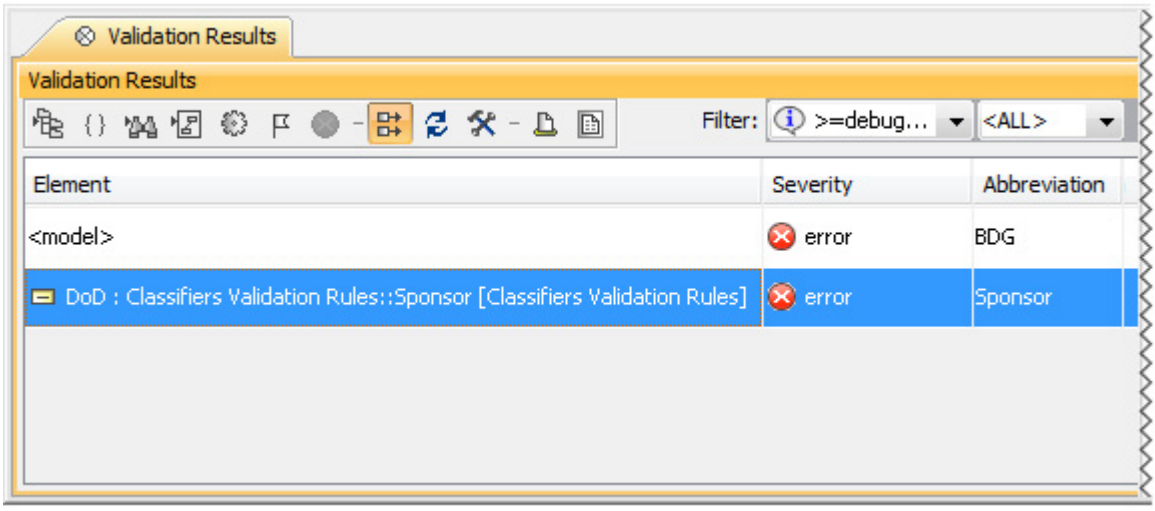


Figure 394 -- results of running the global validation rule

We see that budget balancing rule is a global rule – it is not the concrete instance of *ResearchProject* that violates the rule, but the entirety of instances in the model. Hence the string <model> in the column of that offending elements.

However, note that anticorruption rule is not a global rule - it refers to *contribution* field, which is really a shorthand for *self.contribution*, hence this rule refers to self variable and therefore, is not global and is evaluated for each instance of the *Sponsor* class separately.

MagicDraw has no means to determine if the binary validation rules are global, hence all binary rules are treated as local.

Expressions in error messages

When specifying error messages, more than a simple error string can be entered. Error messages can have template areas, which hold expressions, that will be evaluated and expanded when displaying validation results. Refer to the *ResearchProject*/*Sponsor* example above. The validation result shows:

Budget not balanced – overbudget by 500$

Where does the number 500 come from? It is not directly specified in the error message string (since it is different for different models) but a calculated value of the expression, embedded in an error string. This error string in this case is:

```
Budget not balanced - overbudget by {
ResearchProject::allInstances().budget->sum() -
Sponsor::allInstances().contribution->sum()}$
```

Expressions are embedded in the error messages by using curly brackets - {}. Everything between them is treated as an expression and evaluated for each validation result. Expressions are treated as OCL2.0 expression by default, however you can also use binary expressions. In this case use {bin: <binary expression>} syntax.

Modeling other types OCL2.0 constraints/expressions

Only **inv** constraints can be evaluated in MagicDraw. However, there are more constraint types defined in the OCL2.0 specification. There are also **def**, **init**, **derive**, **pre**, **post**, **body** constraints. These constraints are not evaluated, but can be modeled for documentation purposes. Here is how to model them:

Copyright © 1998-2015 No Magic, Inc.

- **def** – create a usual constraint, but apply *«definition»* stereotype, instead of *«invariant»* or *«validationRule»*.
- **init** – place an opaque expression in the defaultValue field of the property.

.

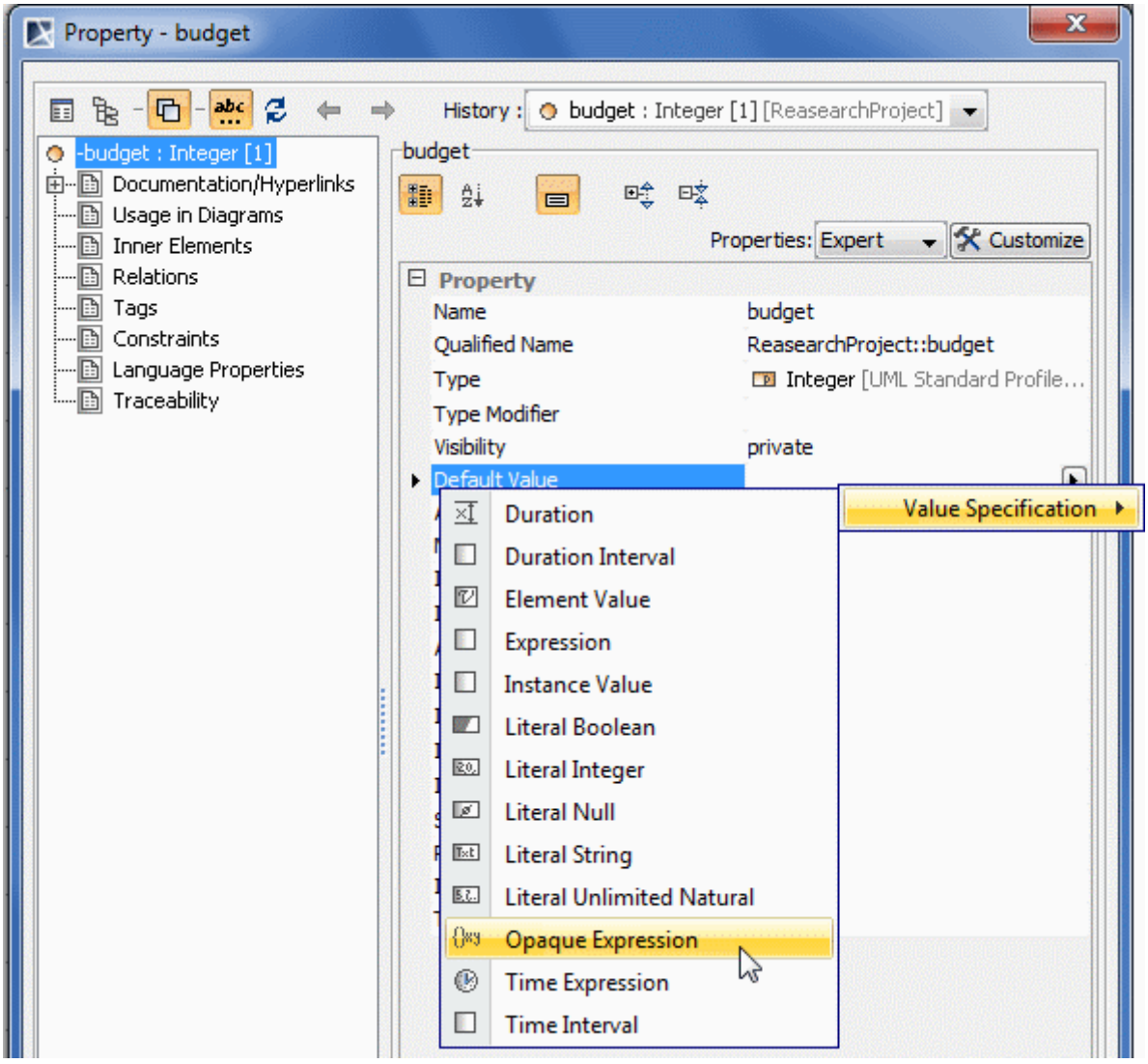


Figure 395 -- Setting opaque expression as default property value

To place an opaque expression in the default value of the property, click the button with a small black arrow pointed to the right or right click on the default value field in the specification of the property and from the shortcut menu select **Value Specification** > **Opaque Expression**.

- **derive** when the expression is placed in the default value of the property (the same as for **init** expressions) but the property is marked as derived in the specification.
- **pre**, **post**, **body** when constraints are placed in the appropriate fields of operation (precondition, postcondition and body condition respectively).

Note that **derive** expressions can be evaluated indirectly, when the validation rule (**inv** constraint) is referencing the property and the validation rule is evaluated.

Unsupported OCL2.0 features

Not all OCL2.0 features are supported in the current release of MagicDraw.

Copyright © 1998-2015 No Magic, Inc.

In particular these features are not supported:

- Distinction between null values and undefined values.
- Tuples.
- All the operations, defined in the UML2 superstructure specification on the metaclasses are not present and are not callable.
- Defining and calling operations on classifiers.

There may be some other features that are not working properly. These issues can be reported to https://support.nomagic.com. MagicDraw uses external library – Dresden OCL Toolkit for constraint evaluation.

Adding/customizing severity levels

If the default severity level choice, provided by MagicDraw is not enough for you, new severity levels can be added. This can be done by editing the *SeverityKind* enumeration in the *UML Standard Profile::Validation Profile* package. Each enumeration literal in this enumeration corresponds to available severity levels. Severity levels in this enumeration should be sorted in ascending order.

If you need to specify a new icon for your custom severity level:

1. Create stereotype, derived from the *imaged* stereotype, with the *EnumerationLiteral* as base class.
2. Set the necessary icon on this stereotype.
3. Apply this stereotype on your custom severity level enumeration literal.
4. Additionally, specify the *highlightColor* tagged value on the literal. This field (of the *String* type) determines how the offending elements will be highlighted in the diagrams. The string format is the same as for the specifying colors in HTML pages (as described in http://www.w3.org/TR/html4/types.html#h-6.5). Simple string constraints (such as *highlightColor="red"*) or numeric values (such as *highlightColor="#FF0000"*) can be used here.

Performance Issues

When validation rules, written in OCL are evaluated, MagicDraw generates Java source for them and invokes Java compiler to compile them into an executable form. Hence, on the first run of validation there is usually a delay of 20-30 seconds (depending on the computer performance) while Java compiler is loading. Subsequent runs will be faster that the first one.

Also, this process consumes some amount of RAM. If the validation process is run heavily on medium-large projects, increasing the default Java VM size is advisable. By default, VM size is set to 400MB in MagicDraw; increasing this to 600 (or 800 if the computer has sufficient RAM) might improve the performance.

Active Validation



Active Validation

Active Validation instantly checks the accuracy, completeness, and correctness of a model, displays invalid elements in the model, and suggests solutions.

The following modeling cases are validated with the Active Validation:

- Numbering validation

Copyright © 1998-2015 No Magic, Inc.

- Parameters synchronization. For more information, see "Parameters Synchronization" on page 430.
- Relations Ownership
- Shape Ownership
- UML Correctness
- others.

Invalid elements in the model are represented in the following ways:

- If an invalid or incomplete model is created, an invalid elements indicator will appear in the bottom right corner of MagicDraw.
- Invalid elements are marked in the Model Browser and on the diagram pane.

From the invalid element or symbol shortcut menu, you can analyze incorrect elements and solve problems. You can also use the **Active Validation Results** window.

Model is validated with predefined validation suites. You can modify these suites and create your own through the **Validation Options** dialog.

Active Validation is an extendable mechanism. Custom validation suites and constraints can be created using binary or OCL constraint.

Detecting Invalid Elements in Model

When Active Validation detects invalid elements in the model, they will be revealed in the following ways:

- in the Model Browser, an invalid element is marked with a small symbol which depends on the failure severity.
- on the diagram, invalid symbol is highlighted.
- on the status bar, the failure indicator is displayed.

Marking invalid elements in the Model Browser

An invalid model element is marked in the Model Browser with a small symbol. The owner of this element is marked with a small grey symbol.

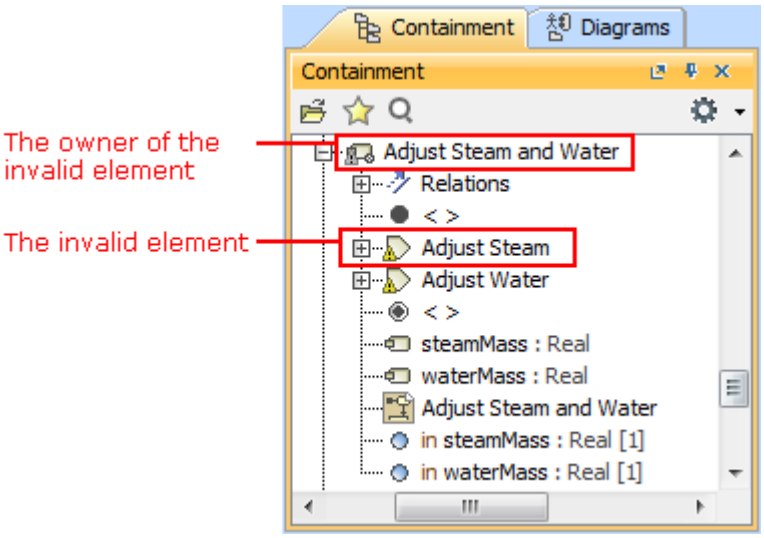


Figure 396 -- Invalid elements marking in Model Browser

Copyright © 1998-2015 No Magic, Inc.

Highlighting invalid elements on the diagram

The shape is colored according to the severity of the error on the diagram.

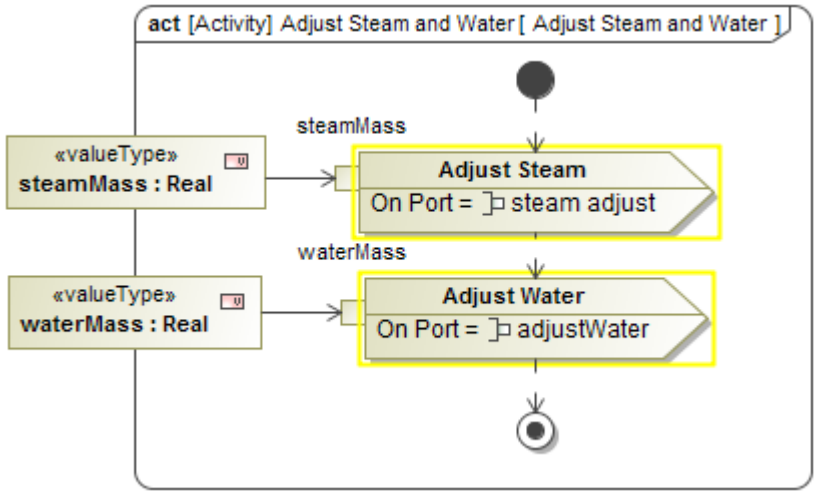


Figure 397 -- Invalid symbols marking on diagram

Failure indicator

If an invalid elements occur, the failure indicator ⚠ 1 W is displayed at the bottom-right corner of MagicDraw. Click the indicator and the **Active Validation Results** window opens.

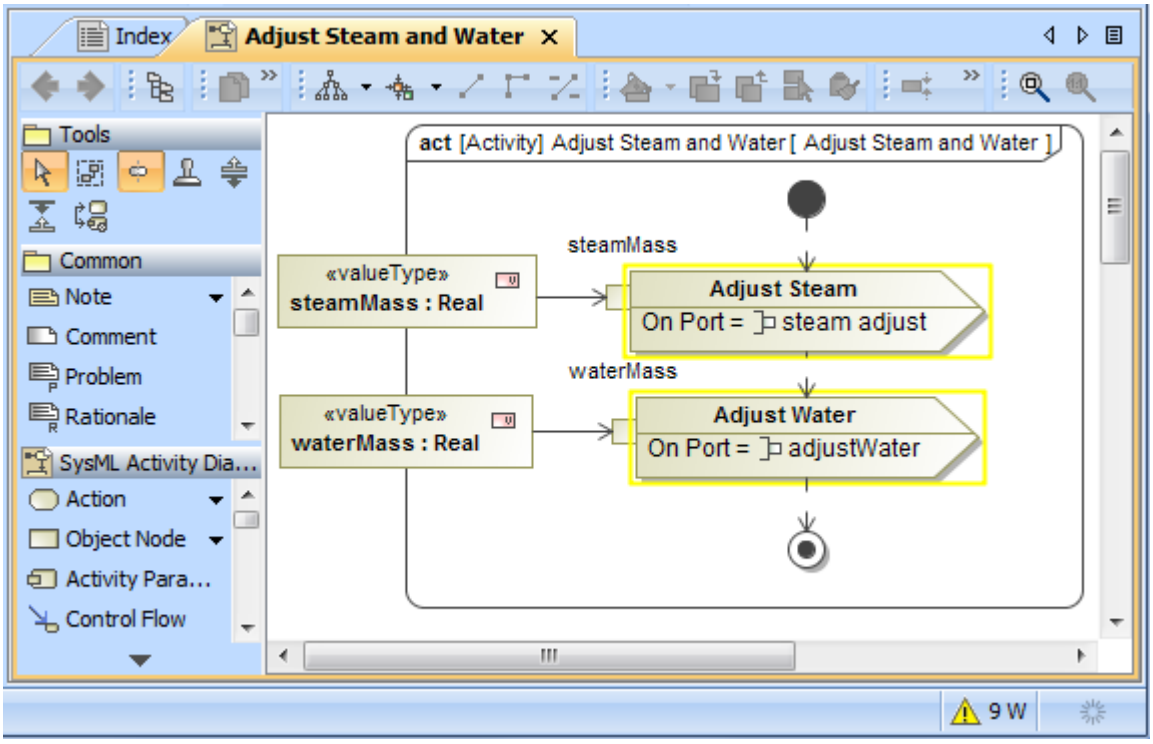


Figure 398 -- General validation notification

Copyright © 1998-2015 No Magic, Inc.

See the parts of the failure indicator represented in the following figure.

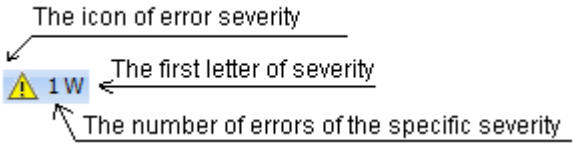


Figure 399 -- Failure indicator

The following table lists the parts of the failure indicator.

Example of the failure indication	Explanation	Possible values
⚠	Symbol showing the level of severity.	⚠ - indicating warning ⊗ - indicating error or fatal error. ⓘ - indicating debug error or info.
1	Number of errors of that specific severity	1, 2, 3 ...
w	The first letter of the error severity	F - fatal error E - error W - warning D - debug I - info

Solving Invalid Model

To handle the invalid model

Do one of the following:

- In the Model Browser, right-click the invalid element, on the shortcut menu, point to **Validation**, to the validation group, and then choose one of the suggested solutions.
- On the diagram pane, select the invalid shape, on the smart manipulator toolbar, click the failure indicator button, and then choose one of the suggested solutions.
- At the bottom-right corner of MagicDraw, click the failure indicator. The **Active Validation Results** window opens.

Copyright © 1998-2015 No Magic, Inc.

In the **Active Validation Results** window you can navigate to invalid elements or symbols, correct the invalid model, or ignore the validation.

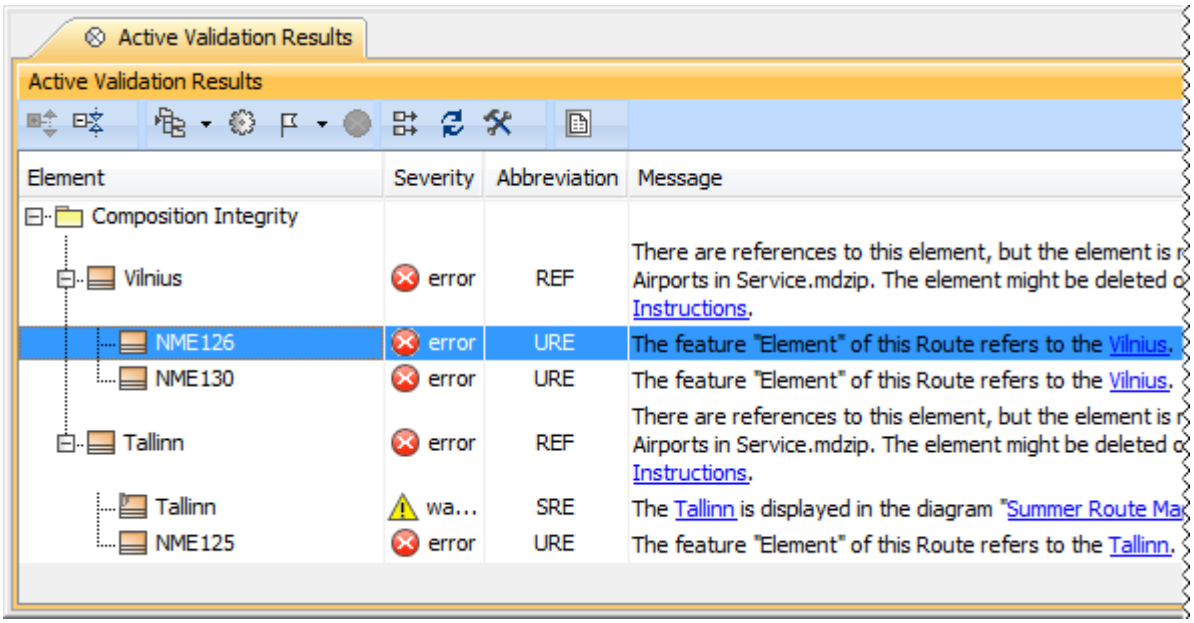


Figure 400 -- Active Validation Results window

Changing Active Validation Options

To change the active validation options:

1. From the **Options** main menu, select **Project**.
2. In the **Project Options** dialog, click to open the **General project options** group.
3. Under the **Active Validation** category, specify the active validation options.

Active Validation Suites

Validating Parameters Synchronization

More information about Parameters Synchronization can be found here "Parameters Synchronization" on page 430.

When synchronization is not possible or corrupted, the active validation will notify by highlighting the elements in the Model Browser and symbols on the diagram.

Shape Ownership

The term “symbol” means a visual representation of some model elements in the diagram. Symbols are further subdivided into shapes and paths (lines in the model, for displaying various relationships).

When drawing UML diagrams, the element ownership is not easily visible. One diagram can contain symbols for elements from several different packages. Element rearrangements in the model may lead to situations where the element ownership in the model does not match the symbol ownership as displayed in the diagram. Such situations are not easy to detect from diagram view.

MagicDraw version 15.0 has a built-in validation code to detect this mismatch. This feature is enabled by default and run unnoticed without requiring any additional input from the user.

Copyright © 1998-2015 No Magic, Inc.

When the symbol ownership on the diagram pane does not match the actual element ownership in the model, the symbol is highlighted with red. So, you will easily see it on the diagram and will be able to correct the problem (problem correction hints are also suggested).



Figure 401 -- Class symbol highlighted with red border

When the mismatch is resolved, the highlighted symbol will return to normal.

The symbol ownership validation feature uses the same mechanism for problem highlighting as the generic validation feature available in MagicDraw Enterprise edition. If you run some validation suites against the model, the element can be highlighted due to any of the validations failures:

- either validation rule(s) from that suite,
- or this automatic symbol ownership checking rule(s).

For more information about validation, see "Validation" on page 612.

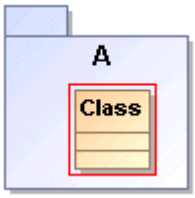
Symbol ownership validation covers two cases:

- Class diagram and its derivatives (use case, composite structure diagrams etc.) can display elements from many model locations and show their ownership. If shape owner is incorrect (not the same as the element owner in the model), this shape will be highlighted.
- Dynamic diagrams (state machine, activity and interaction diagrams) have a restriction that only elements from one definite state machine, activity or interaction can occur in each concrete diagram. Validation rule checks for these diagrams that only elements from correct state machine, activity, or interaction appear in the diagram. All foreign elements are highlighted as erroneous.

These cases are summarized in the table below

	Validation of ownership on the diagram pane	**Validation of diagram owner, on which element is drawn**
Validation rule	Check if the symbol owner on the diagram correctly reflects the element owner in the model.	Check if the element and the diagram on which the symbol is drawn belong to the same owner. This is valid for dynamic diagrams. The checking is performed according to the context of the diagram.
What is checked	Shapes are checked, excluding paths.	Validation works for shapes, including paths.

Copyright © 1998-2015 No Magic, Inc.

	Validation of ownership on the diagram pane	Validation of diagram owner, on which element is drawn
Cases	Check is performed: • if the element symbol is draw on the same package/model/profile to which it actually belongs; • if the element symbol is drawn on the same component/node to which it actually belongs; • if the element symbol is drawn on the same state to which it actually belongs; and other similar cases.	Check is performed: • if the communication diagram and elements from the communication diagram are in the same interaction; • if the sequence diagram and its elements are in the same interaction; • if the state diagram and its elements are in the same state machine; • if the protocol state machine diagram and its elements are in the same protocol state machine. • if the activity diagram and its elements are in the same activity.
Examples	For example, on the diagram pane, an element shape is nested in package A, but actually the element is in package B.	For example, a diagram belongs to activity A, but elements of this diagram belong to activity B.
		

TIP!

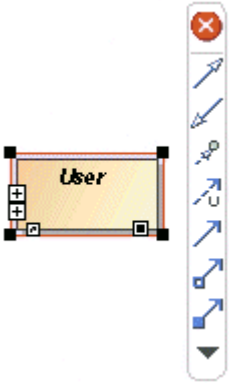
To locate the actual owner of an element:

- Right-click the element on the diagram pane. On the shortcut menu click **Select in Containment Tree**. The element will be selected in the Containment tree. The element's parent in the tree is its actual owner.
- In the element's Specification window, find the **Owner** property.

Solving the detected symbol ownership problems

In this section you will find out the reasons why an element is highlighted in red and how to solve symbol ownership problems quickly.

Select the highlighted element on the diagram pane. The smart manipulator toolbar opens. Notice the red button on the top of smart manipulator toolbar.



Copyright © 1998-2015 No Magic, Inc.

To see the reason why a shape is highlighted, move the mouse pointer over the red button in the smart manipulator toolbar. You will see a tool tip that explains why the element is highlighted. For example: “Shape ownership in the diagram does not correspond to the element ownership in the model”.

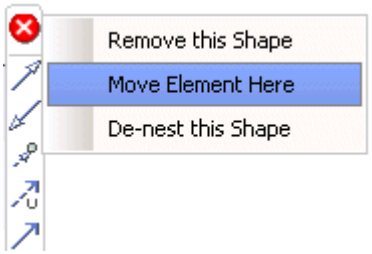
You may either solve the problem yourself or choose one of the suggested solutions. To select an available solution, click the red button in the smart manipulator toolbar. A menu with the following commands will open (note that some items might not be included depending on the situation in the model):

- **Remove this Shape**. Symbol is deleted from the diagram pane.
- **Move Element Here**. Element is moved to the new owner in the model, so that the element ownership in the model corresponds to the present shape ownership in the diagram.



When a symbol is deleted, the element is not deleted from the project.

- **De-nest this Shape**. Shape (not element!) will be extracted from the current owner and placed directly on the diagram. This solution is applicable only for some cases (e.g. De-nest, this Shape command is not available for shape, placed in the incorrect state machine, activity, interaction diagrams).



Related Topics

- "Customizing Environment Options" on page 96.
- "Smart Manipulation" on page 233.

UML model correctness

Active validation instantly check the most important correctness rules of UML model. The following validation rules are checked: Ports compatibility, Pin types compatibility, Slot and Tags multiplicity correctness and others. Automated solutions are suggested for solving the model errors.

Customizing Active Validation

Any constraint (binary or OCL), additionally optimized, can be used to validate models in real-time. MagicDraw provides predefined suites for validation of: Correct ownership in the model and on the diagram, Parameters and arguments synchronization, missing referenced elements (Orphaned Proxies) in the modules[11], and others.

A model is validated automatically without any additional input. Additional constraints can be added or properties can be customized from **Analyze** (main menu) > **Validation**.

11.Starting from version 18.1, referred as “used project” in MagicDraw UI. This section (including subsections) mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

Validate element that has no representation in diagram

Element or its symbol is highlighted in diagram pane and in Browser if it is owner of element that cannot be represented in diagram and has validation error.

See the examples of the spelling error in the table below:

Sample description	Representation in Diagram	Representation in Containment Tree	Notes
Use Case documentation (comment owned by Use Case and annotating Use Case) has a spelling error	Use Case symbol is highlighted with red dashed border (see Figure 402 on page 640).	The owner of the hidden element with error is marked with white x in red quadrant in the Containment tree as itself would have error (see Figure on page 640).	To the symbol smart manipulator toolbar there is added additional button (grey circle with cross inside) that presents the errors solving solutions.
Use Case extension point name has a spelling error	Use Case symbol is highlighted with red dashed border (see Figure 403 on page 641).	The owner of the corrupted and not represented in diagram element is marked with white X in grey quadrant in the Containment tree (see Figure 404 on page 641).	

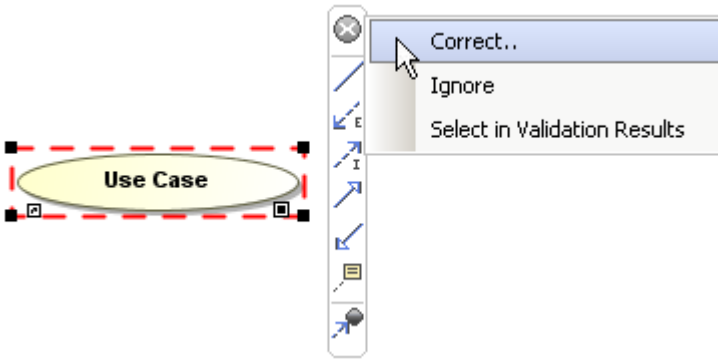
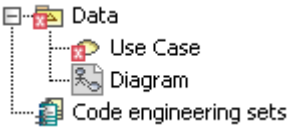


Figure 402 -- The Use Case is highlighted with red dashed border, because it has documentation with spelling error



The Use Case is highlighted with white X in red quadrant in the Containment tree, because it has documentation has a spelling error

Copyright © 1998-2015 No Magic, Inc.

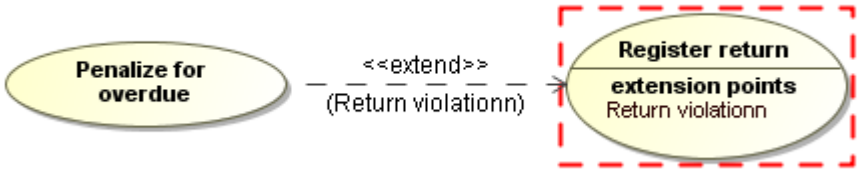


Figure 403 -- The "Register return" Use Case is highlighted with red dashed border, because the "Return violationn" Extension Point has a spelling error

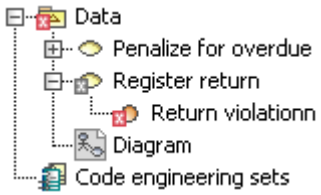


Figure 404 -- The “Register return” Use Case is highlighted with white X in grey quadrant in the Containment tree, because the “Return violationn” Extension Point has a spelling error

Model Visualizer



This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

MagicDraw contains tools that help to create elements from existing data and analyze the relationships between elements created in the UML model. It is also possible to analyze the inheritance and dependency relationships between classes.

All model visualizing and analyzing tools are presented in MagicDraw as Wizards with several steps that should be followed in order to accomplish the desired operation.

All Wizards have several common buttons:

- **Add** – add selected model elements from the **All** list to the **Selected** list.
- **Add All** – add all elements to the **Selected** list that are located in the same hierarchy level as the selected element.
- **Add Recursively** – add all elements in the selected packages and all elements from nested packages to the **Selected** list.
- **Remove** – remove the selected element from the **Selected** list.
- **Remove All** – remove all selected elements.
- **Back** – return to the previous dialog.
- **Next** – proceed to the next step.
- **Finish** – finish the configuration. All other options will be set by default. The **Wizard** exits and results are displayed.
- **Cancel** – cancel the wizard.
- **Reset To Defaults** – if changes were made to the element properties, values will be set to default.

Copyright © 1998-2015 No Magic, Inc.

Class Diagram Wizard

NOTE

This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

The **Class Diagram Wizard** helps you create a new class diagram when all classes and their relationships are already created and specified. You can select which classes, packages, and relationships will be included in a new class diagram and the details of the class representation to be configured (attributes, operations, accessibility). The **Class Diagram Wizard** frees you from creating the class diagram manually. The **Class Diagram Wizard** guides you through several steps and collects information along the way. It will automatically create a new class diagram and all the necessary elements.

To start the **Class Diagram Wizard**

1. From the **Analyze** menu, select **Model Visualizer**. The **Model Visualizer** dialog opens.
2. From the wizards list, select the **Class Diagram Wizard**.
3. Click **Start**. The **Class Diagram Wizard** opens.

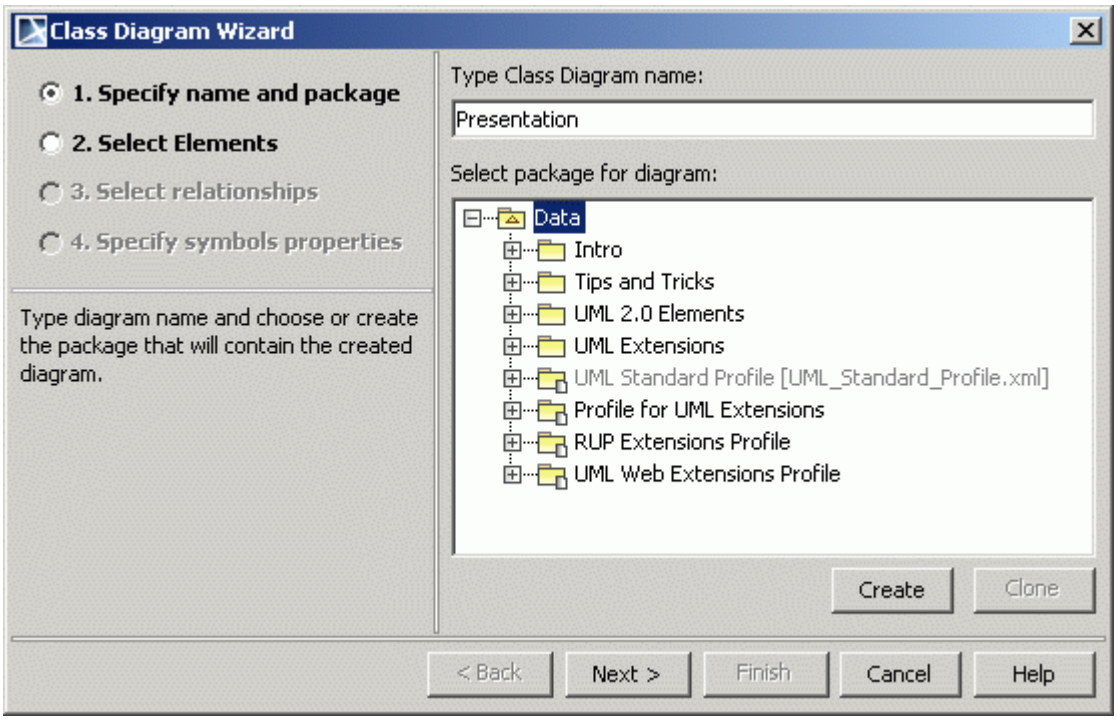


Figure 405 -- Class Diagram Wizard. Diagram name

Type the name of the new diagram in the **Type Class Diagram name** text box or leave the default name.

Select the package that will contain the created diagram. The hierarchy of UML model packages is displayed in the **Select package for diagram** list window. Select the package from the data tree that will be the parent for the newly created diagram or create a new package by clicking the **New** button.

When you select the **Specify name and package** option, the following functions are available:

- **Create** – create a new package.

Copyright © 1998-2015 No Magic, Inc.

- **Clone** – copy an existing package to a new one.

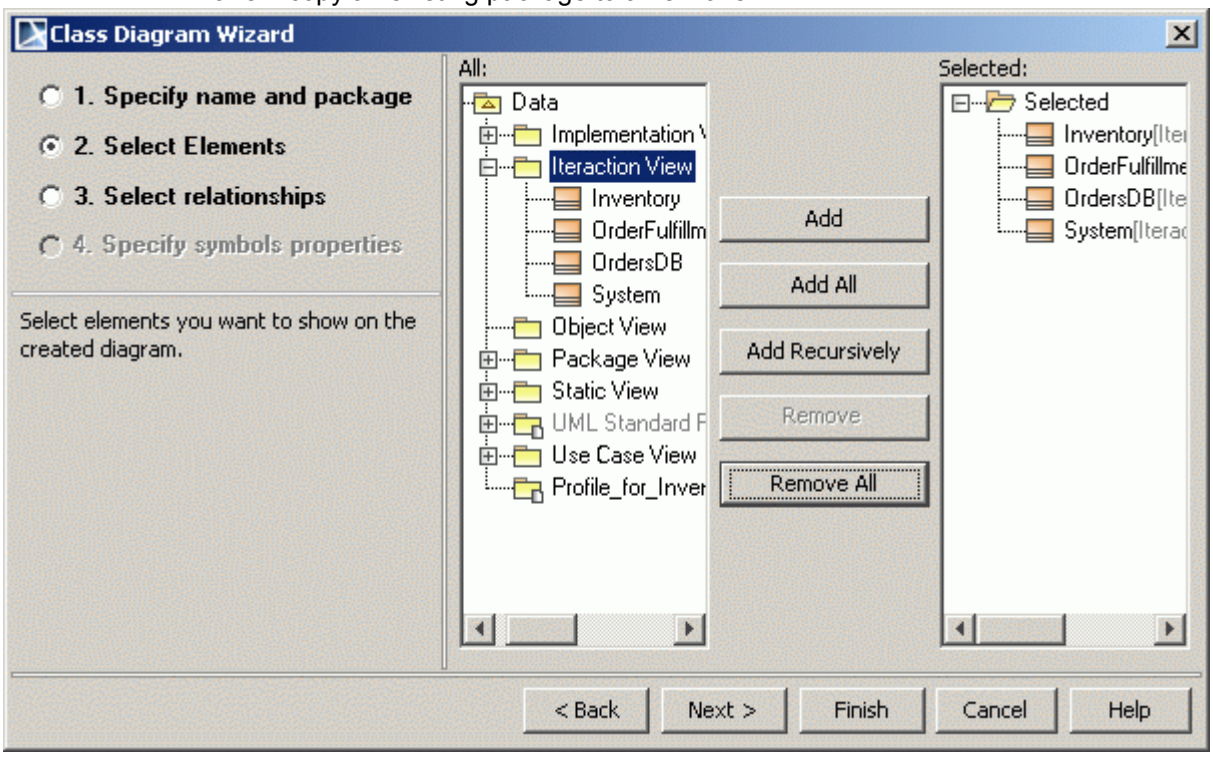


Figure 406 -- Class Diagram Wizard. Select Elements

- **All** list – contains all model elements.
- **Selected** list – contains the elements that are added to the class diagram.

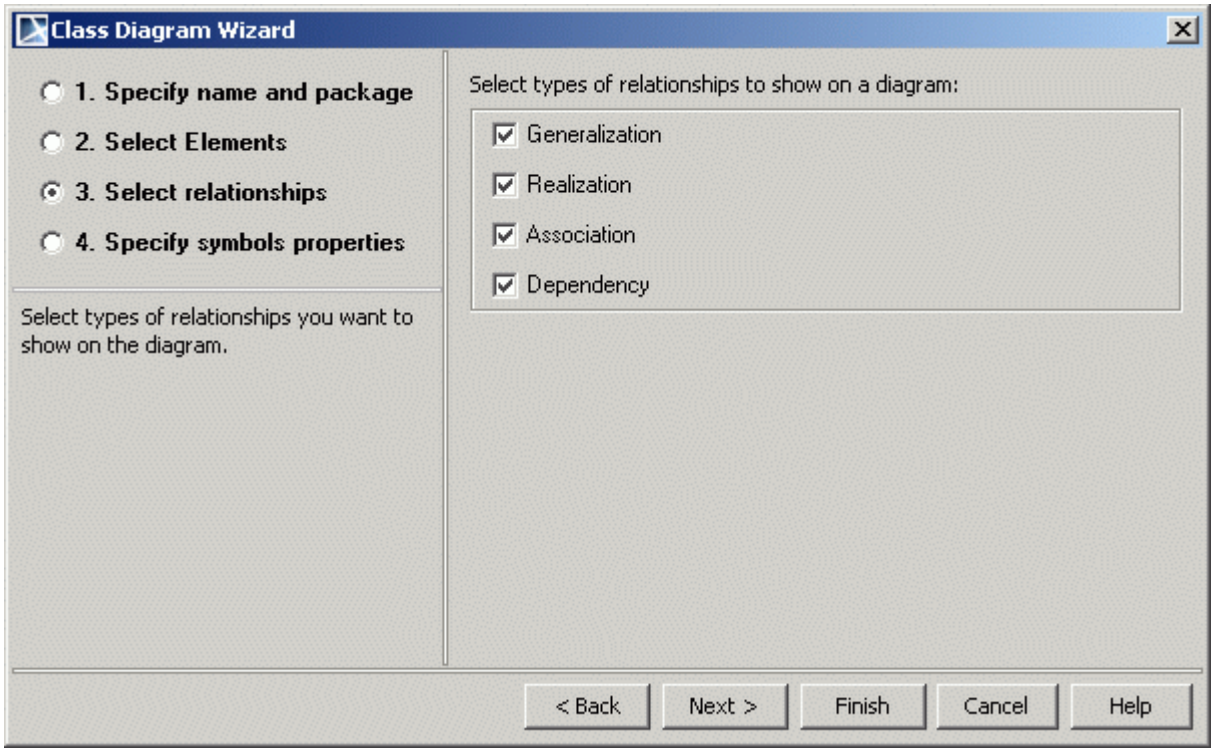


Figure 407 -- Class Diagram Wizard. Select Relationships

Select relationships to include in the class diagram

- **Generalization** – relationship between a general element and a more specific element (inheritance, extension).

Copyright © 1998-2015 No Magic, Inc.

- **Realization** – relationship between model elements where one of the elements implements the interface defined by the other model element.
- **Association** – semantic relationship between classes.
- **Dependency** – usage relationship between UML model elements.

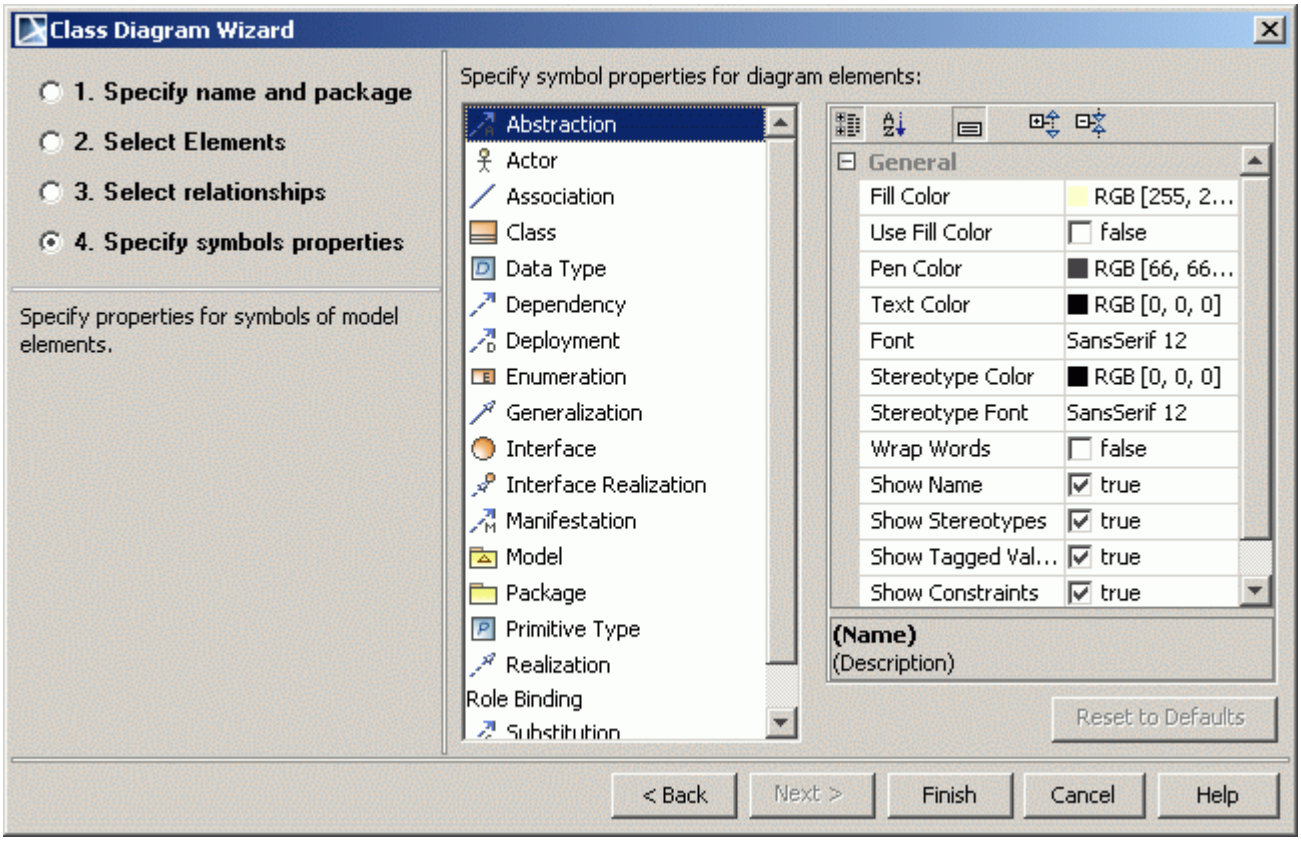


Figure 408 -- Class Diagram Wizard. Specify Symbols Properties

Select options for representing elements in the class diagram.

NOTE

If the **Suppress Attributes** and the **Suppress Operations** check boxes are selected, the class is displayed only as a rectangle with the class name in it.

Package Dependency Diagram Wizard

NOTE

This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

The **Package Dependency Diagram Wizard** creates the Package diagram containing packages (created within a project) and shows the relationships between them. The diagram may reflect all packages in the project, or just selected ones. The **Package Dependency Diagram Wizard** collects all information needed for both analyzing dependencies and generating a new diagram.

For information about the Package diagram, see "Package Diagram" on page 713.

If a package owns one or more elements used by or dependent on other package elements, the **Package Dependency Diagram Wizard** analyzes dependencies and creates «virtual» relations between the dependent packages (see the following example).

Copyright © 1998-2015 No Magic, Inc.

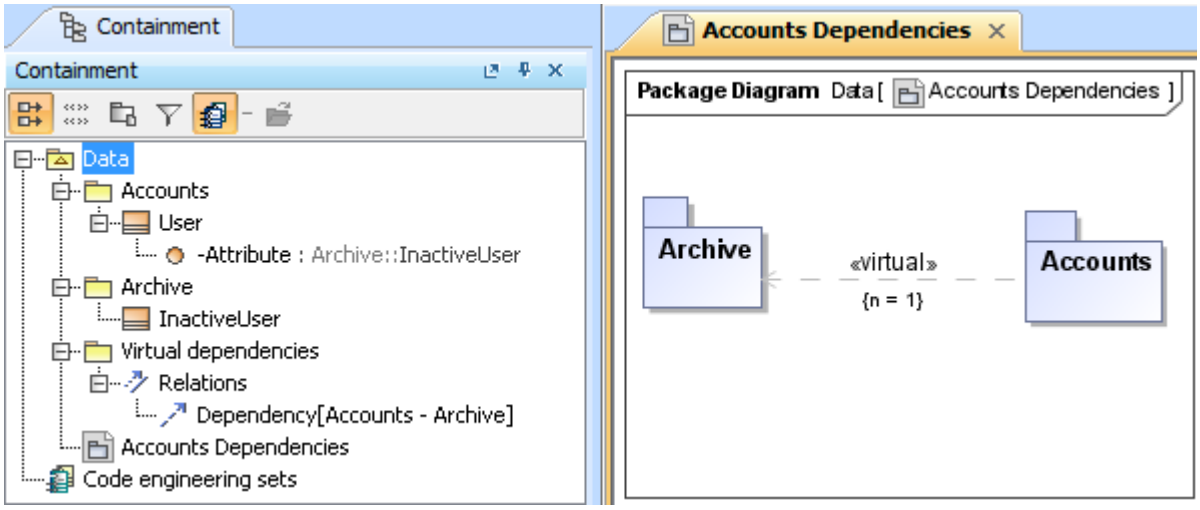


Figure 409 -- Example of virtual dependency

The **Virtual dependencies** package containing dependency links is created after finishing the wizard. If there are more than one dependency between elements in the packages selected to analyze, then the number of dependencies are assigned to the tagged value number **{**n=...**}** on the virtual dependency.

To start the **Package Dependency Diagram Wizard**

1. Do one of the following:
   - On the **Diagrams** menu, point **Diagram Wizards** and then click **Package Dependency Diagram Wizard**. The **Package Dependency Diagram Wizard** opens.
   - From the **Analyze** menu, select **Model Visualizer**. The **Model Visualizer** dialog opens.
2. From the wizards list, select the **Package Dependency Diagram Wizard**.
3. Click **Start**. The **Package Dependency Diagram Wizard** opens.

The Package Dependency Diagram Wizard contains of three steps:

1. Name and package specification.
2. Package selection.
3. Symbol properties specification [Optional].

Step #1: Name and package specification

In this step, you can specify the diagram name and select or create the diagram owner.

In the **Type Package Diagram name** text box, type the name for the diagram.

In the **Select owner for diagram** box, select the package wherein the created diagram will be stored. You can also create a new owner by clicking the **Create Owner** button. If you want to clone the package, select that package and click the **Clone** button.

Copyright © 1998-2015 No Magic, Inc.

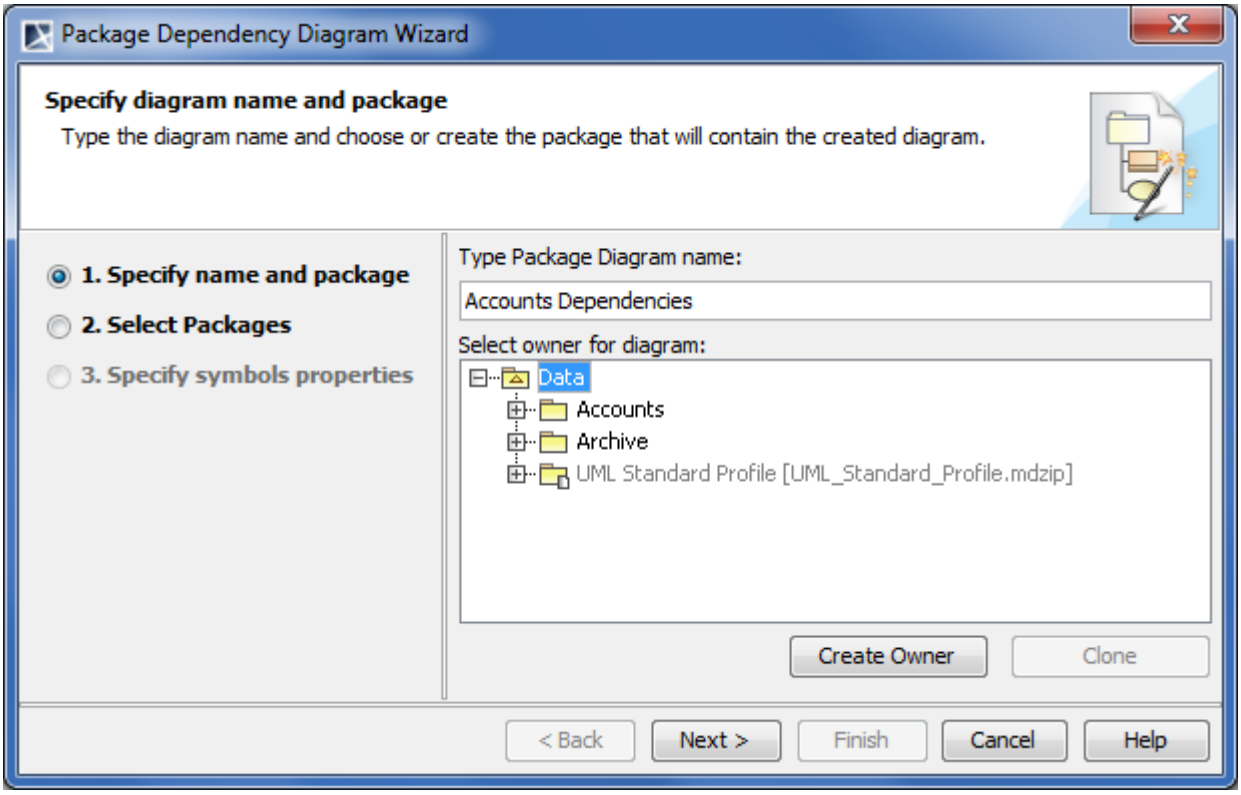


Figure 410 -- Package Dependency Diagram Wizard. Specifying diagram name and owning package

Step # 2: Package selection

In this step, select the packages that dependencies will be analyzed in the Package diagram.

You can select the packages in one of the following ways:

- In the Model Browser, select the package and the click the **Add** button. In this way you can select one package at the time.
- Click the **Add All** button. All the model packages are selected.
- If you want to add a package with all its owning packages, select one or more passaged in the browser and click the **Add Recursively** button.

Selected packages are added to the **Selected** box.

Copyright © 1998-2015 No Magic, Inc.

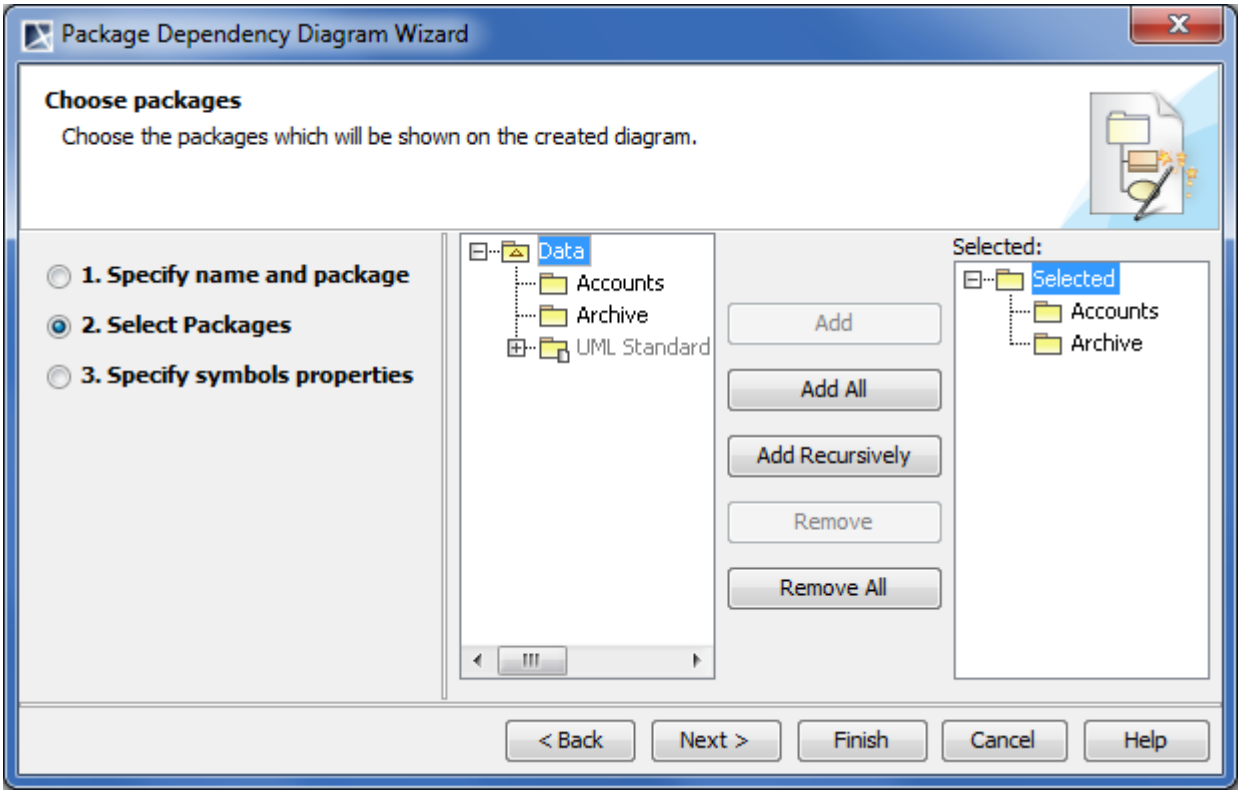


Figure 411 -- Package Dependency Diagram Wizard. Selecting packages

Step #3: Symbol properties specification [Optional]

In this step, you can specify the diagram symbol properties. All available diagram elements are displayed in the **Specify symbol properties for diagram elements** list. Select each element and specify its symbol properties.

Copyright © 1998-2015 No Magic, Inc.

For more information about specifying symbol properties, see "Formatting Symbols" on page 317.

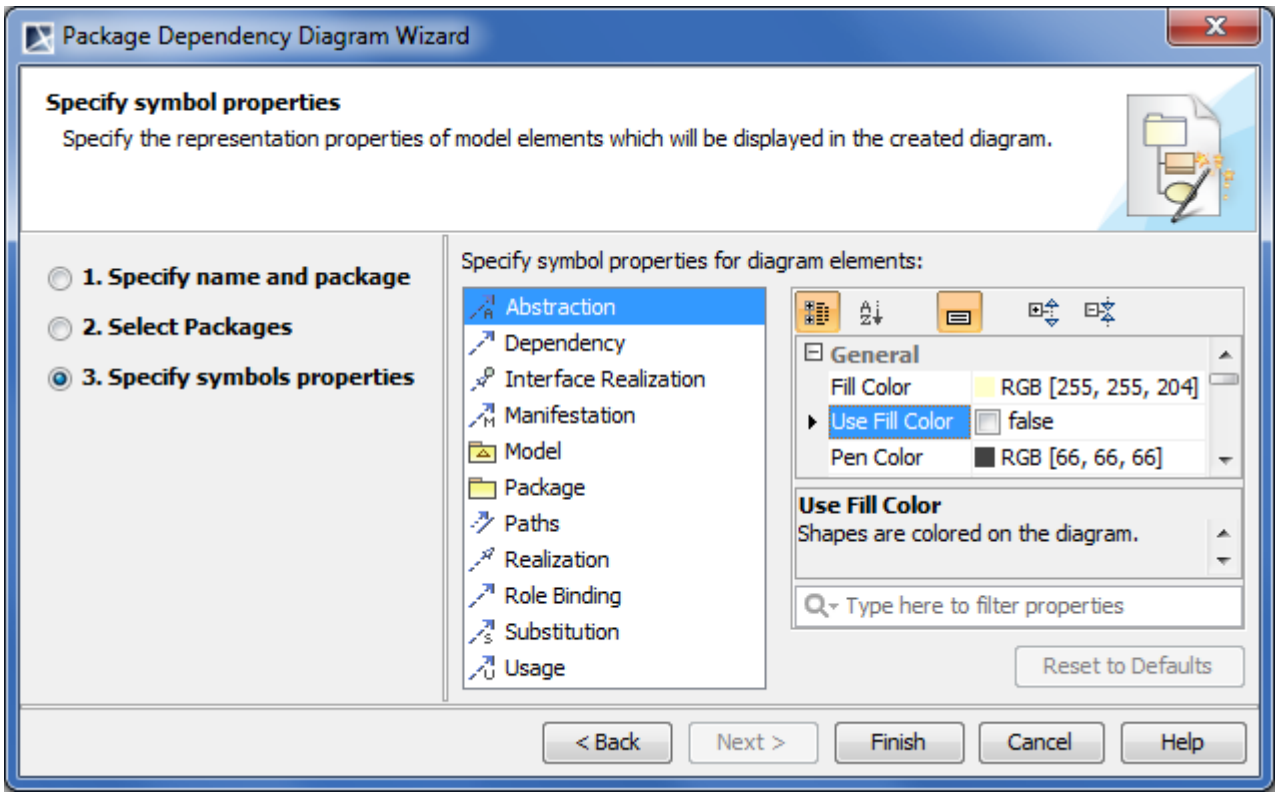


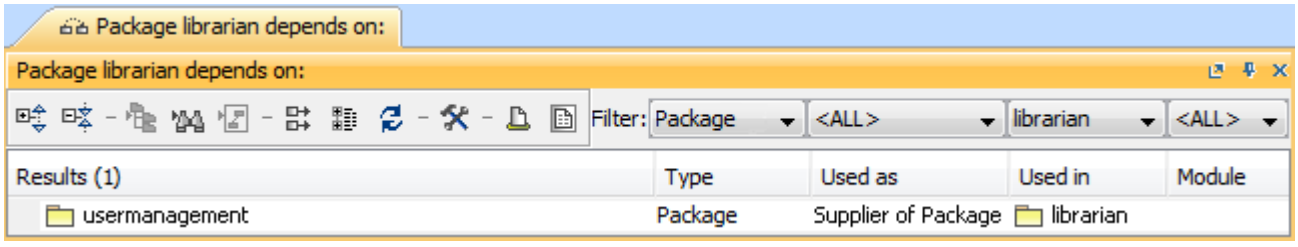
Figure 412 -- Package Dependency Diagram Wizard. Specifying symbols properties

Click **Finish**, when you are done. The Package diagram and the Virtual dependencies package are created. In the Step #2 selected packages and their dependencies are displayed on the created Package diagram.

Displaying dependency details

In the Package diagram, the virtual dependencies (that is, dependency links) are created. All the virtual dependencies are stored in the **Virtual dependencies** package which you can find in the model browser.

You can use the **Show Dependency Details** command to see what are dependencies of a particular package (that is, on what packages depends the particular package). Dependency results are displayed in the dependency details panel (see the following example).



To display the dependency details

1. In the Model Browser or on the diagram pane, right-click the dependency which details you want to display.
2. On the shortcut menu of the selected dependency, click **Related Elements** > **Show Dependency Details**. The dependency details panel opens where in the dependency details are displayed.

Copyright © 1998-2015 No Magic, Inc.

Package Overview Diagram Wizard

NOTE

This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

The **Package Overview Diagram Wizard** allows the creation of a diagram for every package from the selected scope (reversed packages). The created diagram displays the content of the packages – inner packages with inner elements connected with available relations.

To start the **Package Overview Diagram Wizard**

- Select the **Package Overview Diagram Wizard** from the **Diagrams** menu, **Diagram Wizards** submenu.
- Open the package shortcut menu, select **Tools** and then **Package Overview Diagram Wizard**.
- From the **Analyze** menu, select the **Model Visualizer** command. The **Model Visualizer** dialog opens. In the wizards list, select the **Package Overview Diagram Wizard**. Click the **Start** button.

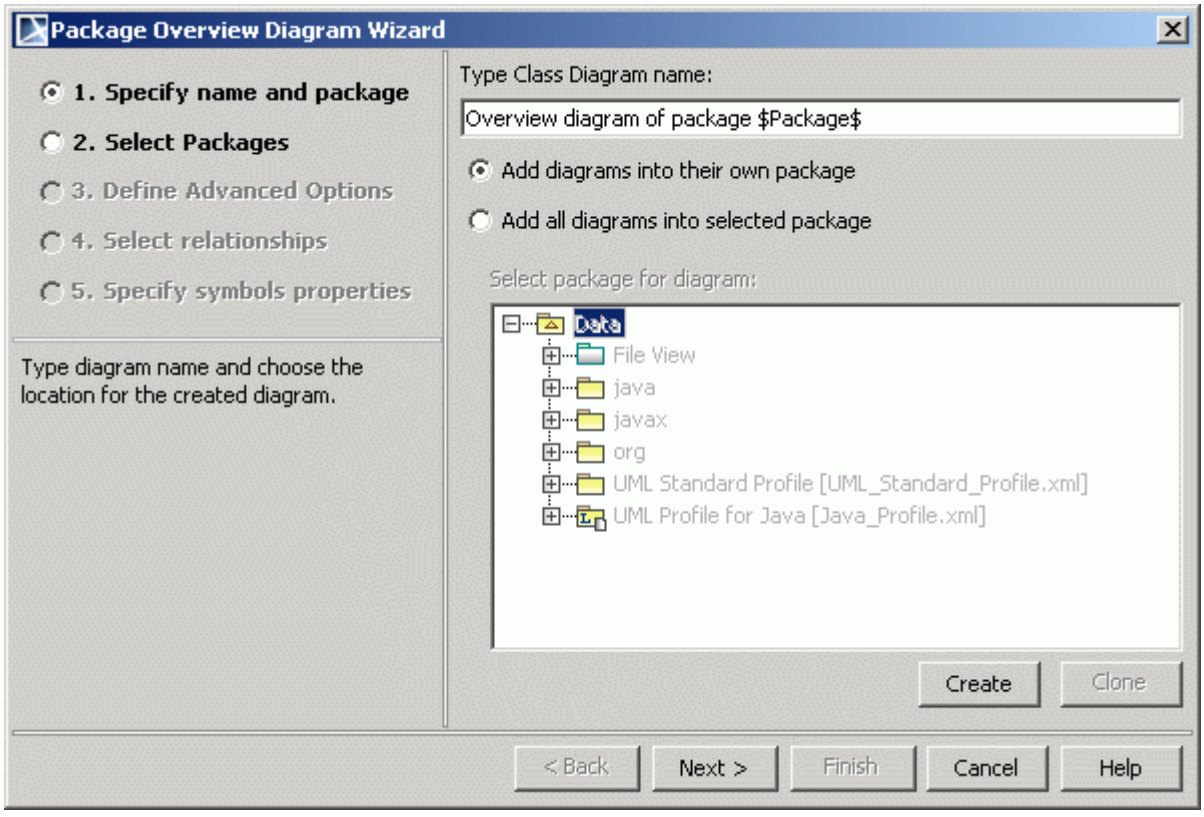


Figure 413 -- Package Dependency Diagram Wizard. Diagram name

Type a name for the newly created diagram in the **Type Class Diagram name** text box.

The **Add diagrams into their own package** option button - adds diagrams in the same package they are created.

The **Add all diagrams into selected package** - while creating new package overview diagrams, adds diagrams in the selected package. Select the package that will contain the created diagram. The hierarchy of UML model packages is displayed in the **Select package for diagram** list window. Select the package that will

Copyright © 1998-2015 No Magic, Inc.

be the parent for the newly created diagram from the Data tree or create a new package by clicking the **New** button.

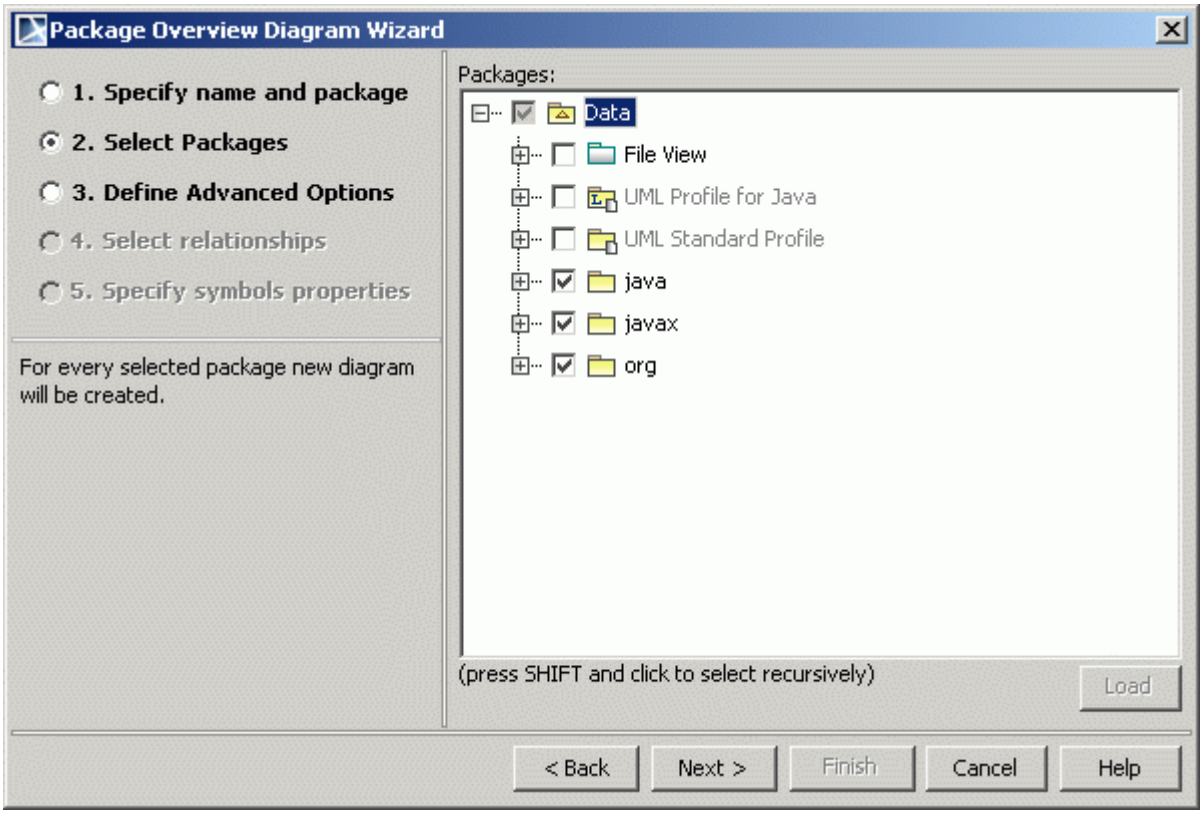


Figure 414 -- Package Overview Diagram Wizard. Select Package.

Select the packages, which will be represented in the new diagram. For every selection, a package diagram will be created.

Copyright © 1998-2015 No Magic, Inc.

If the selected package is read-only and the package for the diagrams is not specified, a warning will be displayed when the **Next** button is pressed.

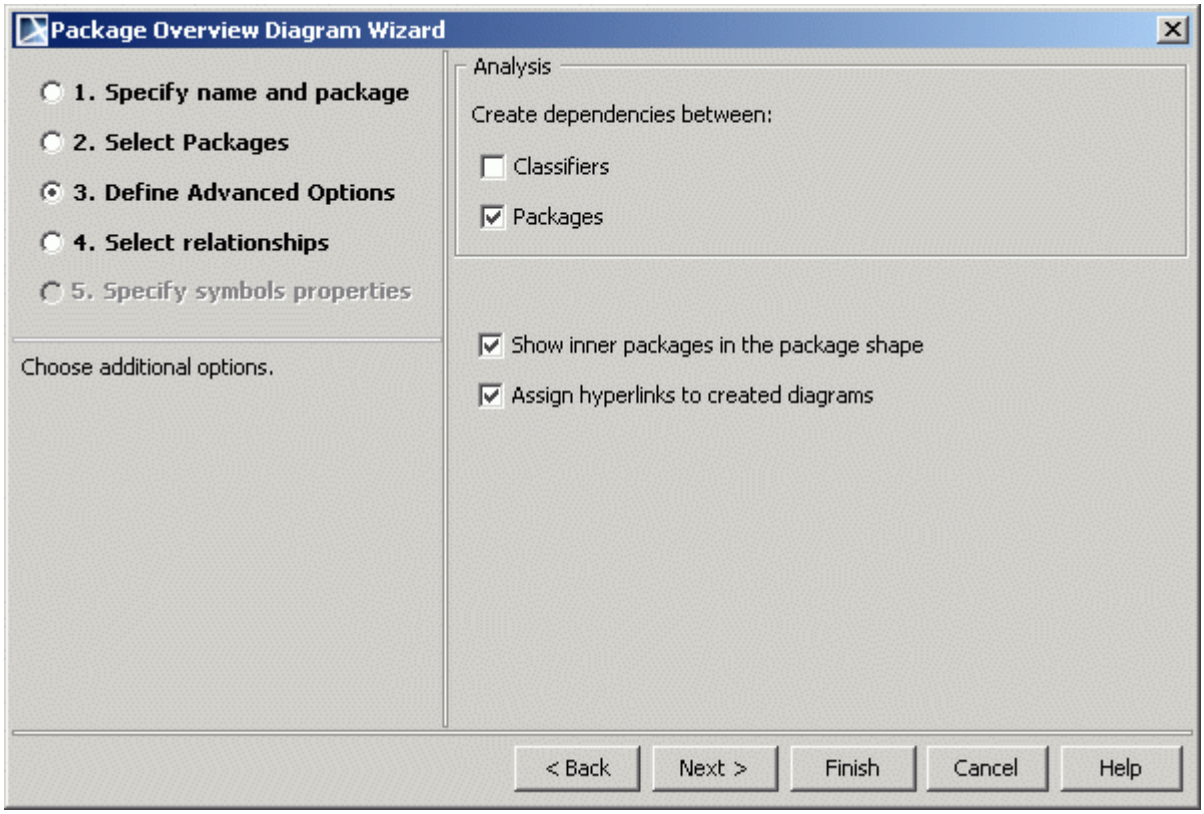


Figure 415 -- Package Overview Diagram Wizard. Define Advanced Options

Set the advanced properties for elements to be represented in the diagrams.

If you want to see the classifiers structure in the created class diagram, then select the create dependencies between **Classifiers** check box in the Analysis options group.

If create dependencies between **Packages** check box is selected, then only the package content class diagram will be created. Analyzes are performed of all inner elements, recursively by all criteria.

The **Show inner packages in the package shape** check box - displays one level of inner packages in every package shape, connected with dependencies.

Copyright © 1998-2015 No Magic, Inc.

The **Assign hyperlinks to created diagrams** check box - adds an active hyperlink to every package, referenced to the inner diagram of this package.

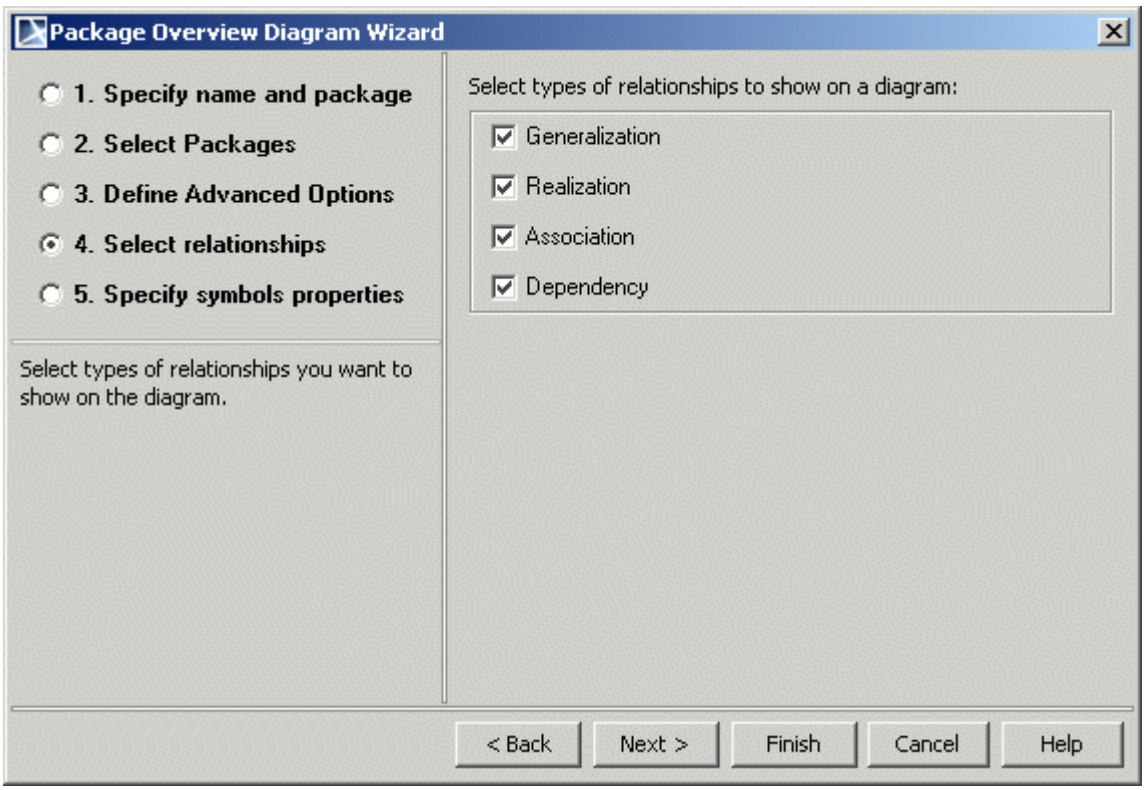


Figure 416 -- Package Overview Diagram Wizard. Select Relationships

Select the relationships you wish to include in the class diagram:

- **Generalization** – relationship between a general element and a more specific element (inheritance, extension).
- **Realization** – relationship between model elements where one of the elements implements the interface defined by the other model element.
- **Association** – semantic relationship between classes.

Copyright © 1998-2015 No Magic, Inc.

- **Dependency** – usage relationship between UML model elements.

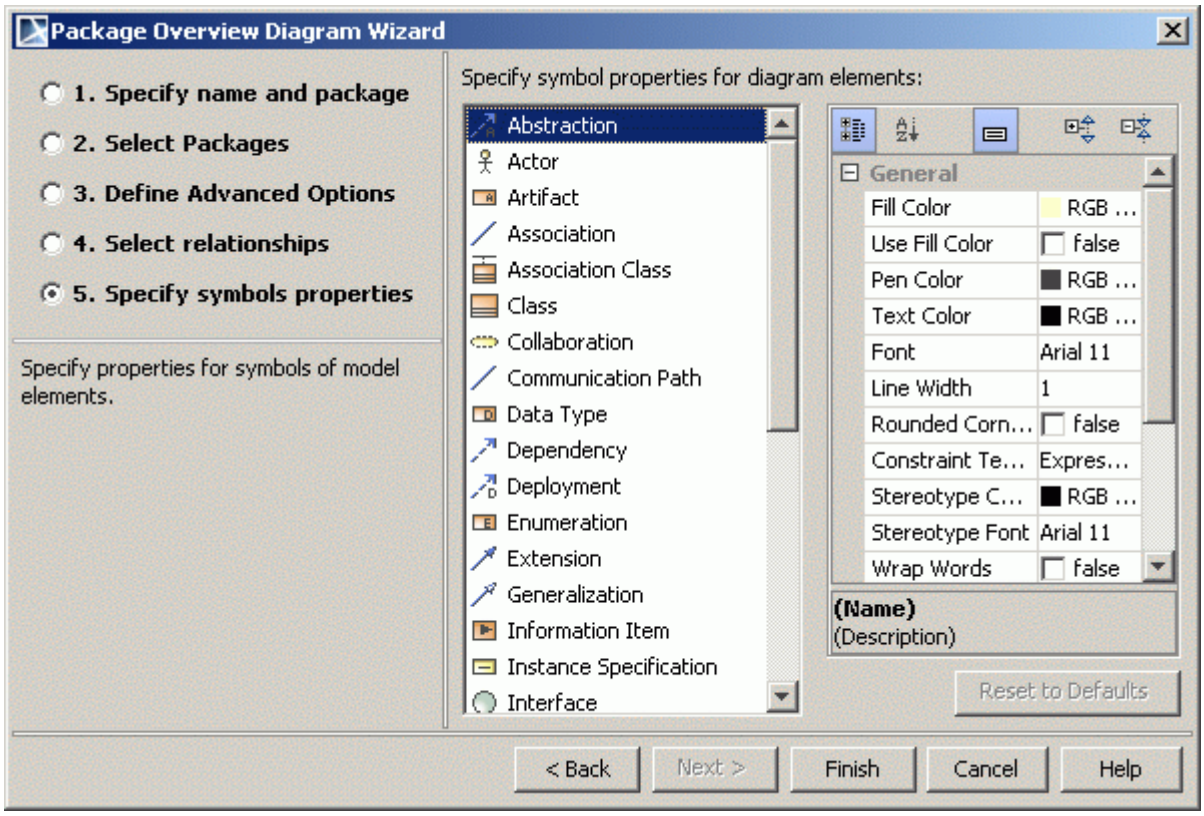


Figure 417 -- Package Overview Diagram Wizard. Specify symbols properties

Select options for representing elements in the diagram.

Hierarchy Diagram Wizard

NOTE

This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

The **Hierarchy Diagram Wizard** collects the largest hierarchies and allows every of them to be displayed as separate diagrams or all in one diagram.

To start the **Hierarchy Diagram Wizard**

- From the **Diagrams** main menu, select the **Diagram Wizards** command and then **Hierarchy Diagram Wizard**.
- From the **Analyze** menu, select the **Model Visualizer** command. The **Model Visualizer** dialog opens. From the wizards list, select the **Hierarchy Diagram Wizard**.

Copyright © 1998-2015 No Magic, Inc.

- From the model element shortcut menu, select **Tools** and then **Hierarchy Diagram Wizard**.

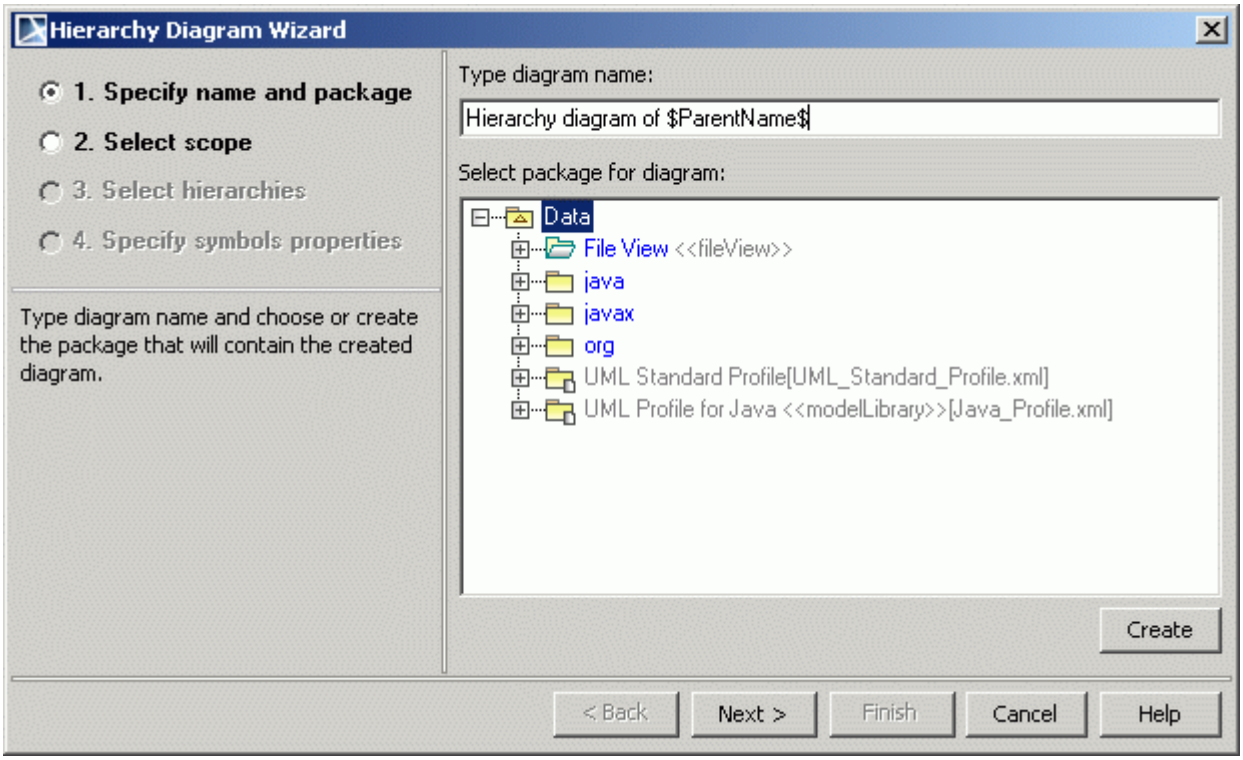


Figure 418 -- Hierarchy Diagram Wizard. Specify Name and Package

Type a name for the new diagram in the **Type diagram name** field.

Select the package that will contain the created diagram. The hierarchy of UML model packages is displayed in the **Select package for diagram** list window. Select the package that will be the parent for the newly created diagram from the Data tree or create a new package by clicking the **Create** button.

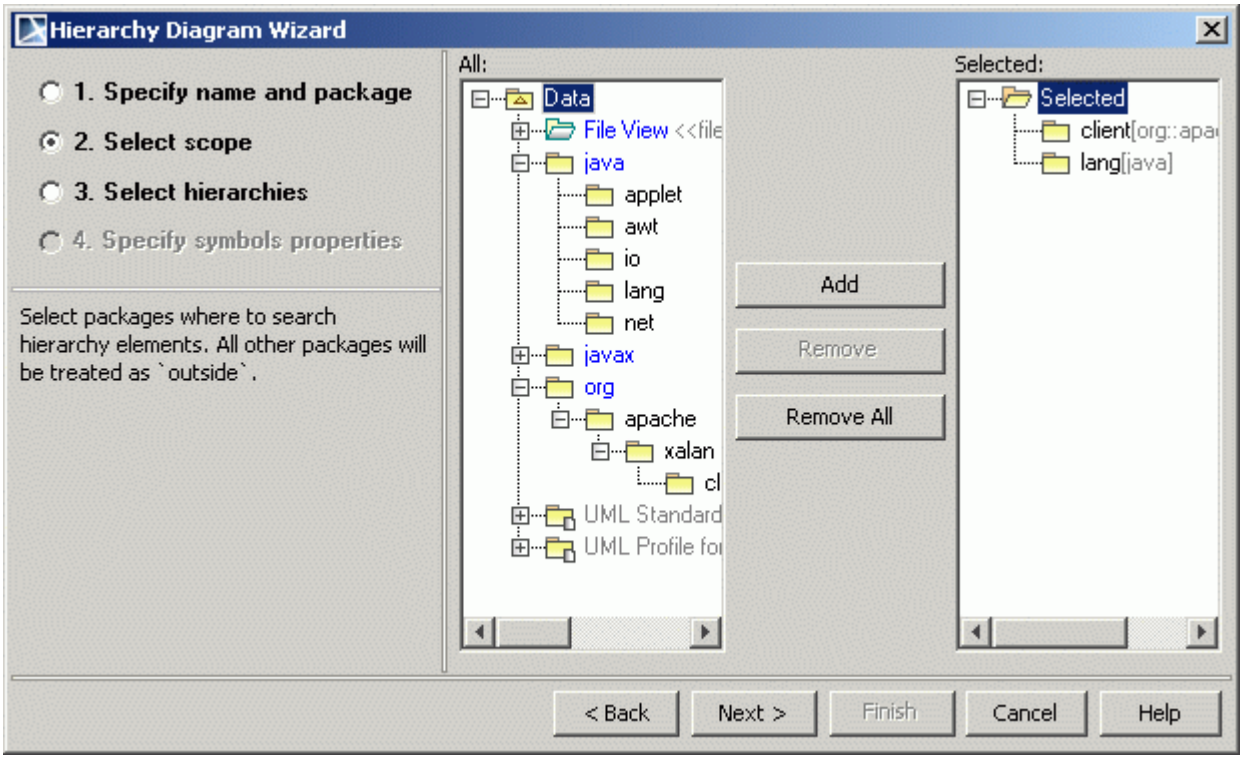


Figure 419 -- Hierarchy Diagram Wizard. Select Scope

Copyright © 1998-2015 No Magic, Inc.

Select packages from the **All** list to the **Selected** list.

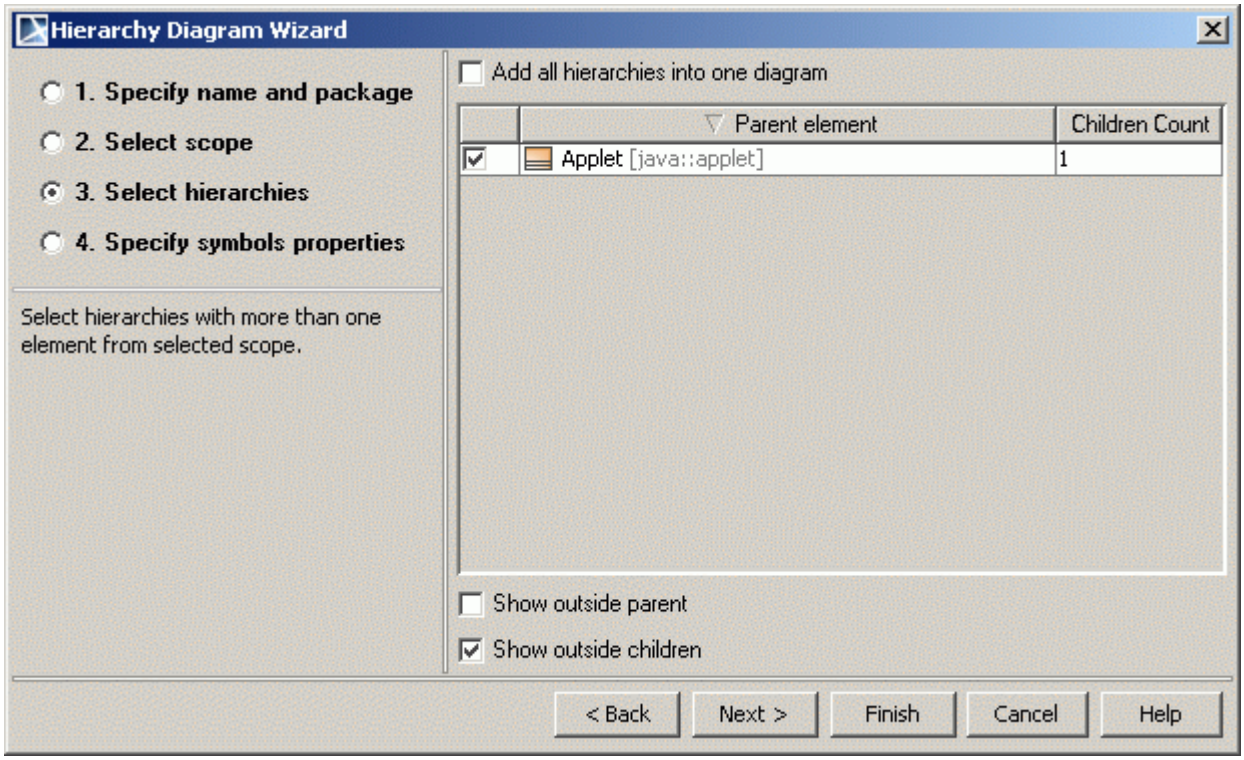


Figure 420 -- Hierarchy Diagram Wizard. Select Hierarchies.

The **Add all hierarchies into one diagram** check box creates only one diagram for all selected hierarchies. This option is enabled only if the selected hierarchies can be added into one diagram (the same diagram type).

All available hierarchies are listed in the **Parent Element** column. In the **Children Count** column, the number of model elements is presented.

The **Show outside parent** check box shows hierarchies, when derived packages are in the scope, but specializations is from outside the scope.

Copyright © 1998-2015 No Magic, Inc.

The **Show outside children** check box counts outside derived elements from the displayed hierarchies. Otherwise the hierarchy will not be fully displayed and the diagram may be not valid.

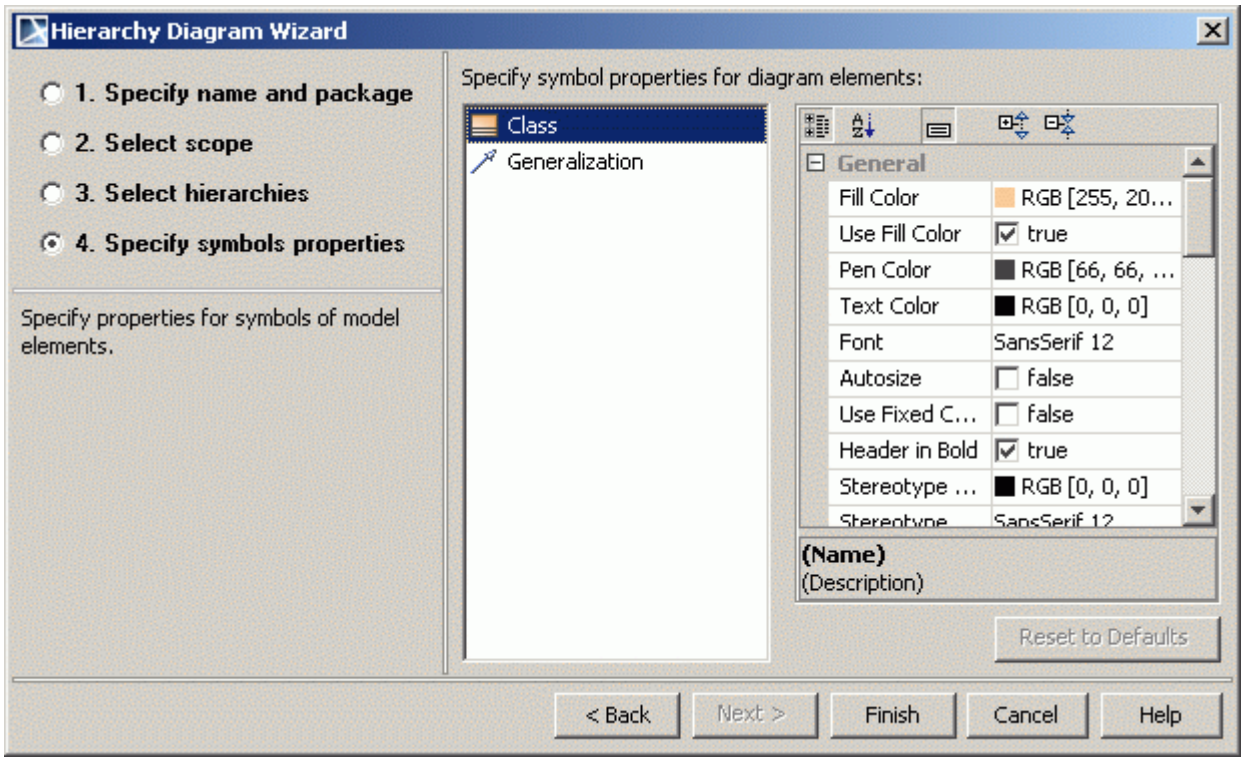


Figure 421 -- Hierarchy Diagram Wizard. Specify Symbols Properties

Select options for representing elements in the diagram.

Realization Diagram Wizard

NOTE

This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

The **Realization Diagram Wizard** shows a table of the largest element groups that realize some interface.

To start the **Realization Diagram Wizard**

- From the **Diagrams** main menu, select the **Diagram Wizards** command and then **Realization Diagram Wizard**.
- From the **Analyze** menu, select the **Model Visualizer** command. The **Model Visualizer** dialog opens. From the wizards list, select the **Realization Diagram Wizard**.

Copyright © 1998-2015 No Magic, Inc.

- From the model element shortcut menu, select **Tools** and then **Realization Diagram Wizard**.

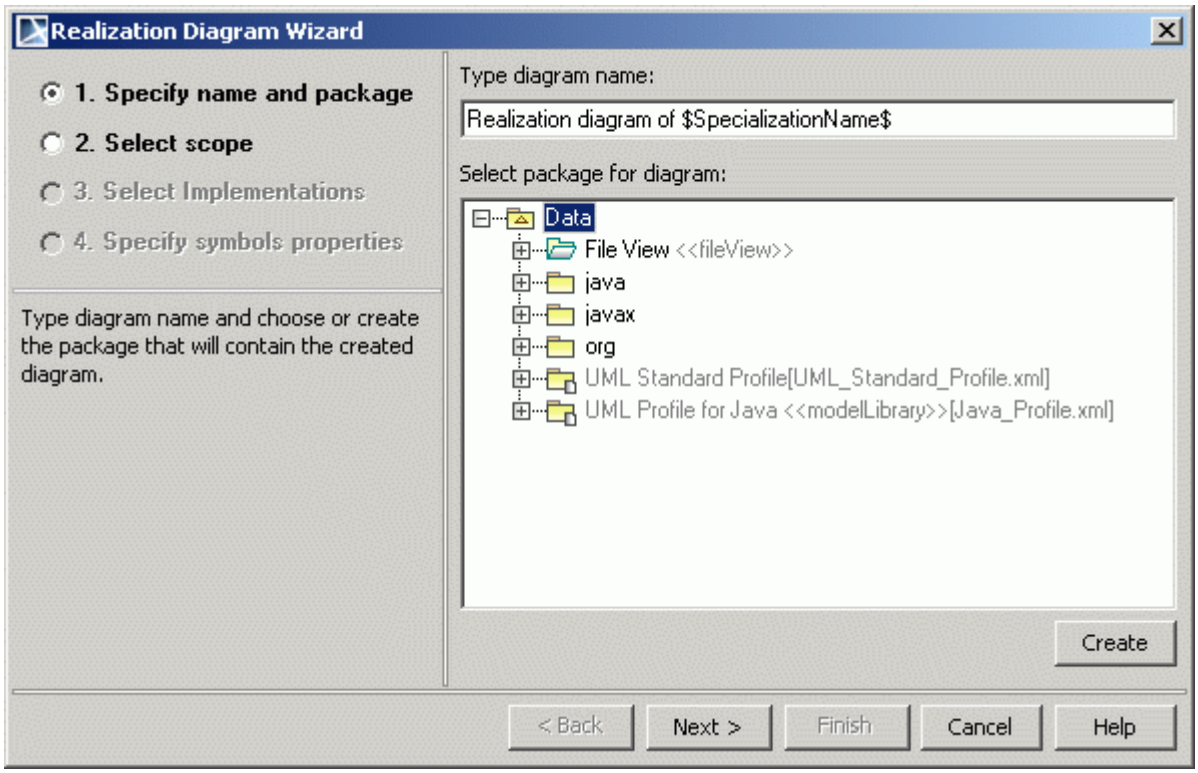


Figure 422 -- Realization Diagram Wizard. Specify Name and Package

Type a name for the new diagram in the **Type diagram name** field.

Select the package that will contain the created diagram. The hierarchy of UML model packages is displayed in the **Select package for diagram** list window. Select the package that will be the parent for the newly created diagram from the Data tree or create a new package by clicking the **Create** button.

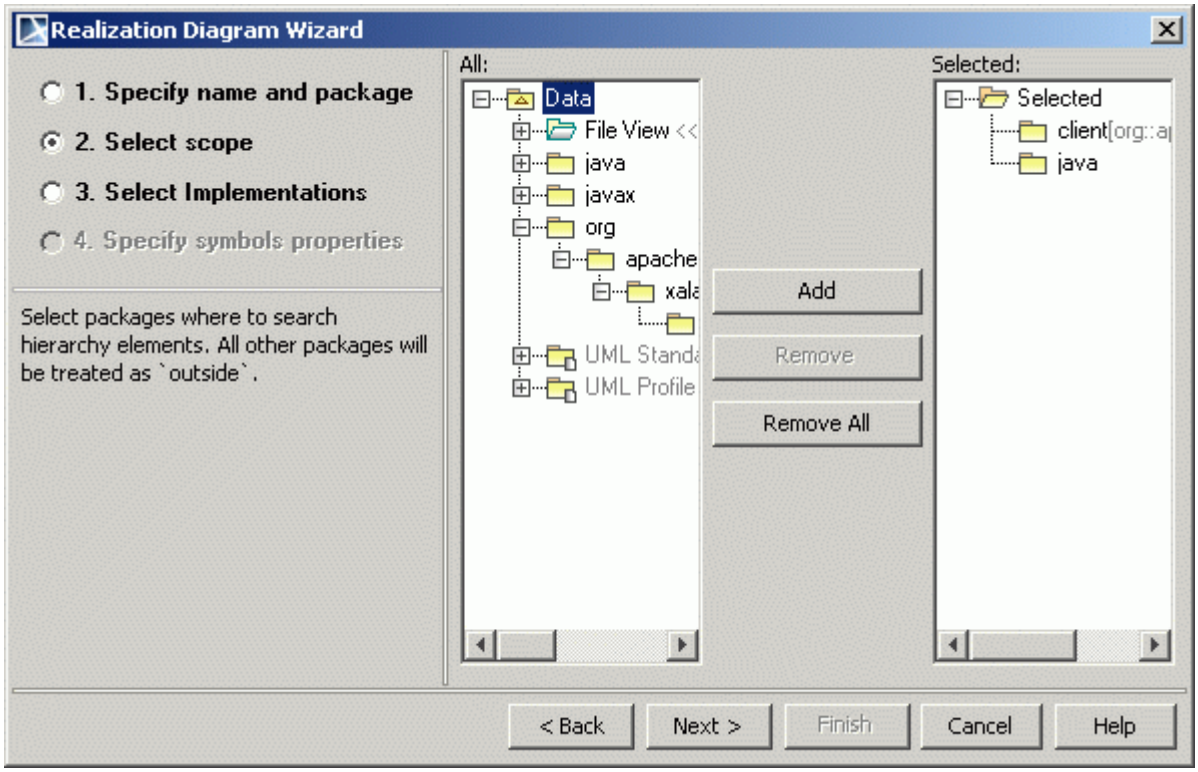


Figure 423 -- Realization Diagram Wizard. Select Scope

Copyright © 1998-2015 No Magic, Inc.

Select packages from the **All** list and add them to the **Selected** list to search for hierarchy elements.

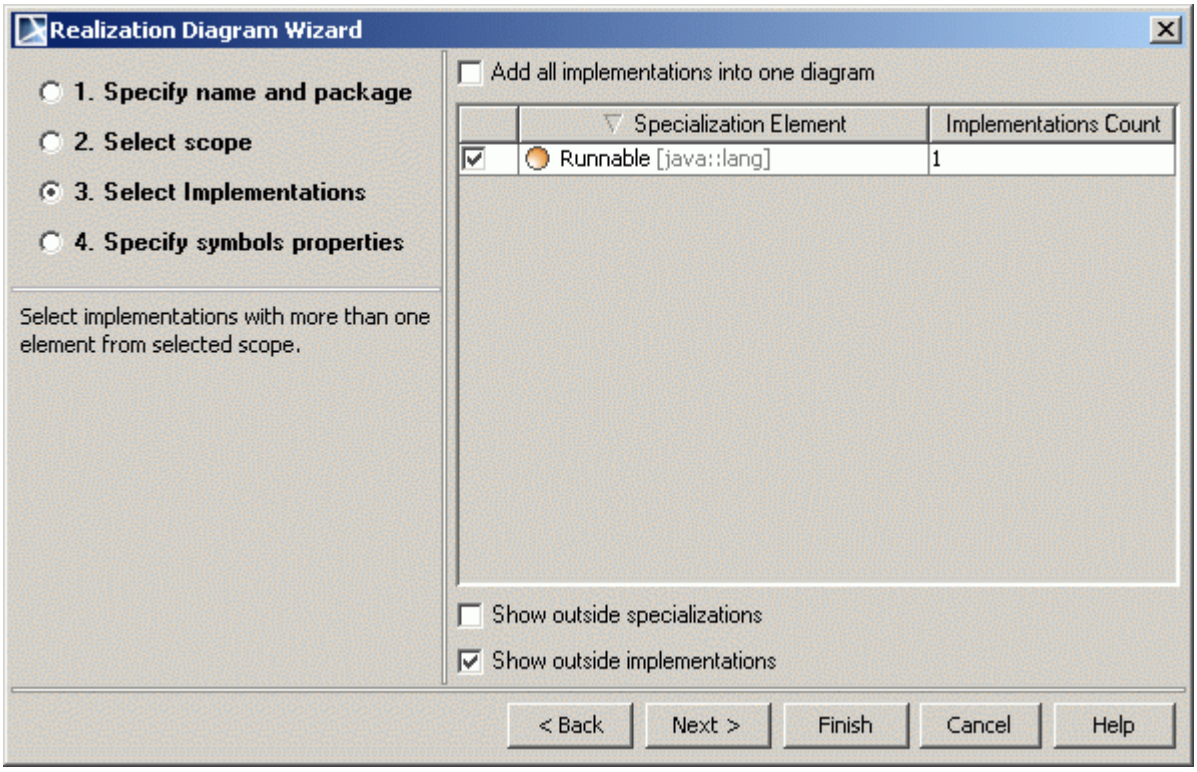


Figure 424 -- Realization Diagram Wizard. Select Implementations

The **Add all implementations into one diagram** check box creates only one diagram for all selected realizations. This option is enabled only if the selected realizations can be added into one diagram (the same diagram type).

All available implementations are listed in the **Specialization Element** column. In the **Implementations Count** column, the number of model elements is presented.

The **Show outside specializations** check box show realizations, when derived interfaces are in the scope, but specializations is from outside the scope.

Copyright © 1998-2015 No Magic, Inc.

The **Show outside implementations** check box count outside derived elements from displayed realizations. Otherwise realization will not be fully displayed and the diagram cannot be valid.

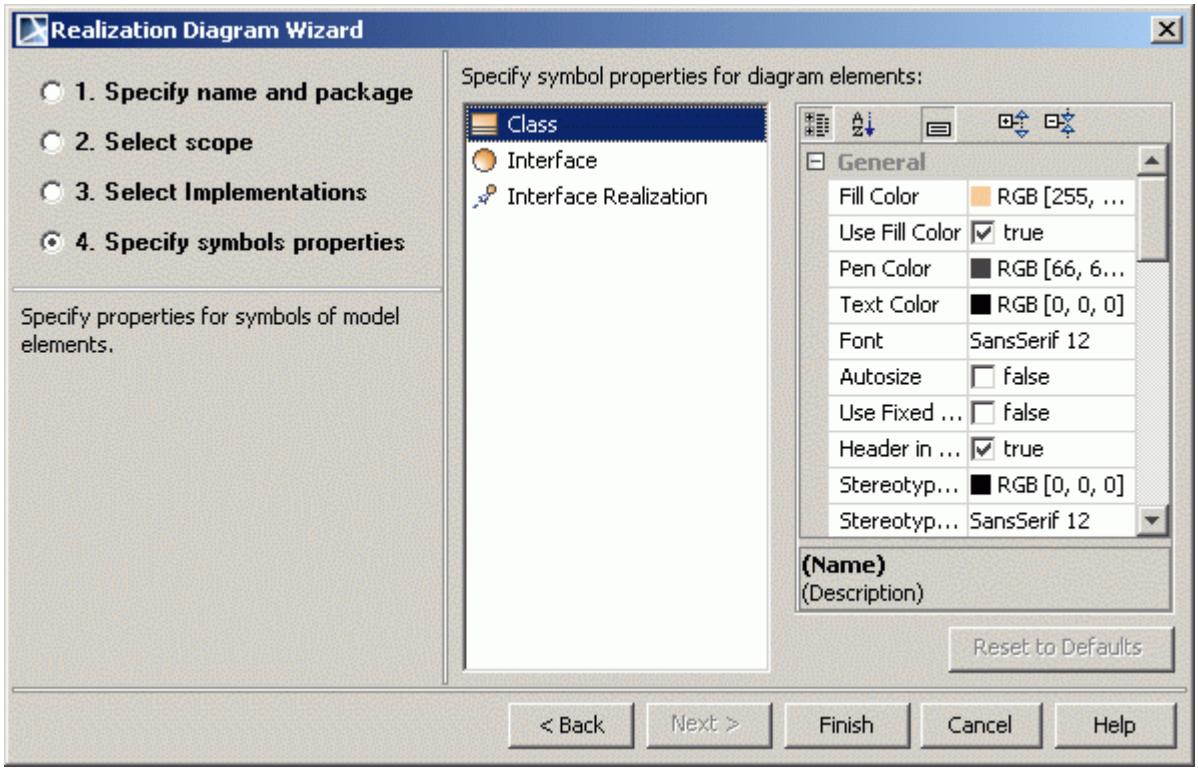


Figure 425 -- Realization Diagram Wizard. Specify Symbols Properties

Select options for representing elements in the diagram.

Activity Decomposition Hierarchy Wizard



This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

Activity Decomposition Hierarchy Wizard allows converting activity into class and SysML Block Definition Diagram. This gives the capability to represent, analyze, and document activity hierarchies in the structure diagrams.

Diagram generation rules:

- Behaviors will be connected with contained object node types by compositions. The name of the object node that corresponds to the composition will be used as the end name of the association on the end towards the object node type.
- Pins are not included in the calculation.
- CallBehaviorActions that are not directly in the Activity, but are in the Structured Activity Nodes contained by the Activity, for example, are also included in the calculation.
- Activity will be connected by composition association with other behaviors that are called by CallBehaviorActions. The part end name must be the same as the name of a CallBehaviorAction in the composing activity. If the action has no name, then the end name is as same as that of the invoked activity.
- Hierarchical layout - Top to Bottom is used to arrange the generated diagram.
- If CallBehaviorAction calls the same activity, the composition to self will be displayed on the generated diagram.

Copyright © 1998-2015 No Magic, Inc.

Recursive structure analysis will be stopped after reaching the same behavior, which has already been analyzed. This requirement prevents an endless cycle.

- In such a case, the composition will be created in a previously analyzed activity in the diagram. The new behavior symbol will not be created.
- There will be as many compositions from one activity into another as different CallBehaviorActions call this activity.

To start the **Activity Decomposition Hierarchy Wizard**

- From the **Diagrams** main menu, select the **Diagram Wizards** command and then **Activity Decomposition Hierarchy Wizard**.
- From the **Analyze** menu, select the **Model Visualizer** command. The **Model Visualizer** dialog opens. From the wizards list, select the **Activity Decomposition Hierarchy Wizard**.

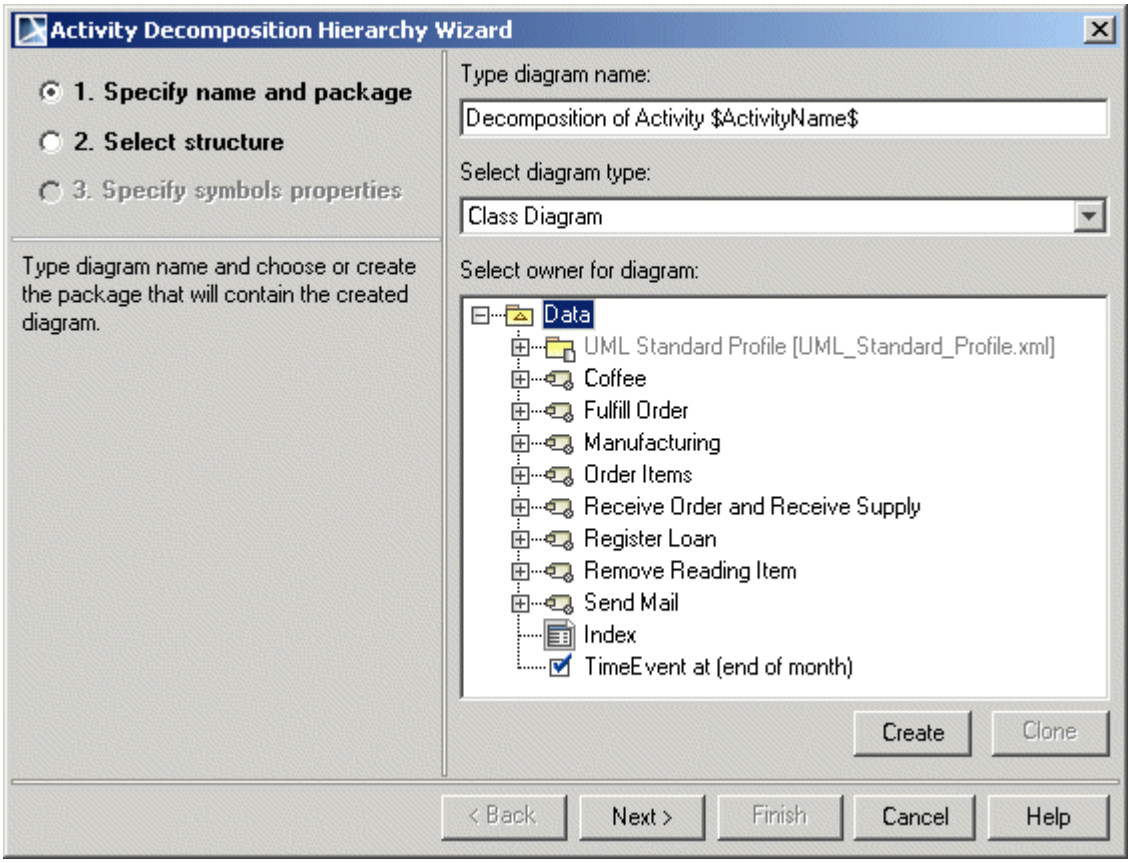


Figure 426 -- Activity Decomposition Hierarchy Wizard. Specify name and package step

In the **Specify name and package** step, type the diagram name, select the diagram type the activity will be converted and select or create a package that will contain the desired activity diagram.



If you are using the SysML plugin, SysML Block Definition and Class diagrams are available as diagram types. For other domains this list depends on a plug-in of those domains.

Copyright © 1998-2015 No Magic, Inc.

.

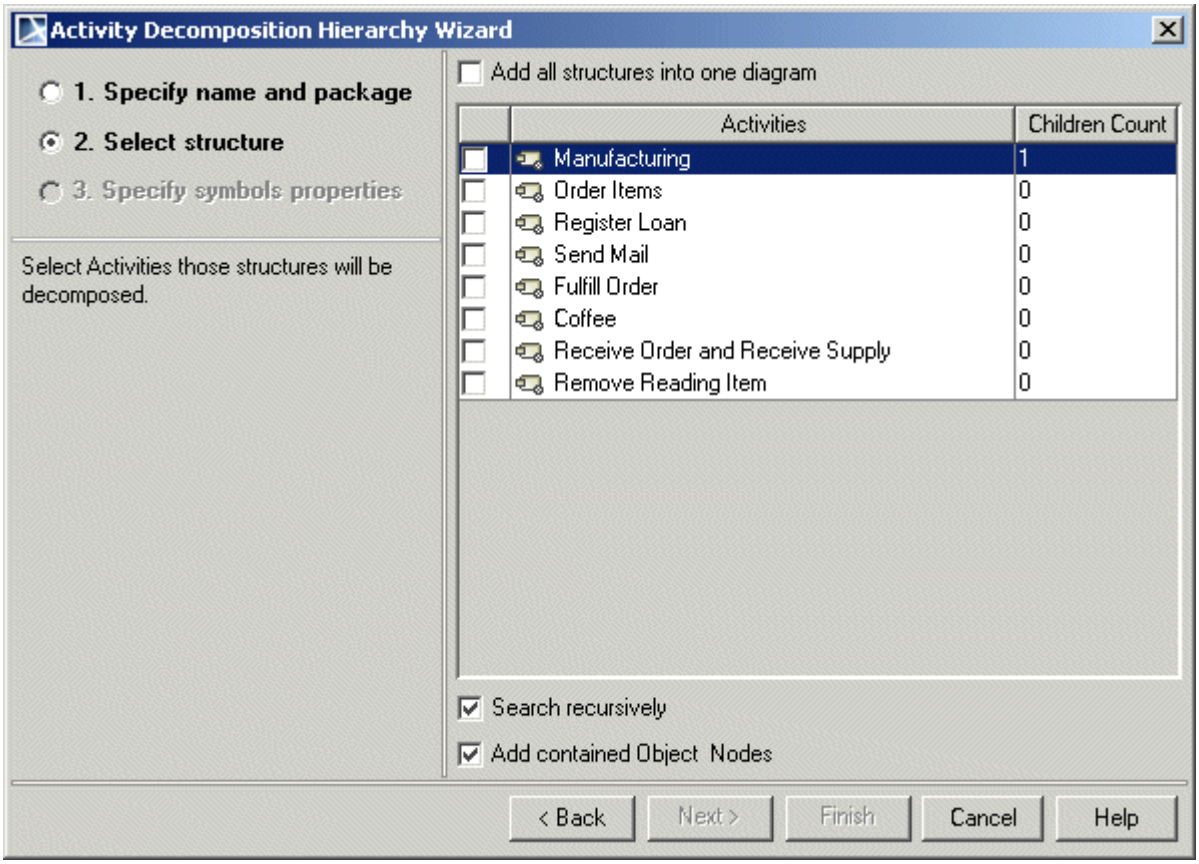


Figure 427 -- Activity Decomposition Hierarchy Wizard. Select structure step

In the **Select structure** step, select Activities structures that will be decomposed.

To add all activity structures into one diagram, select **Add all structures into one diagram** check box.

Select check boxes of the desired activity structures. The **Children Count** column shows the number of included behaviors (also owned object nodes if the **Add contained Object Nodes** check box is selected). The number also depends on the option **Search recursively**.

- The **Add contained Object Nodes** check box is selected by default. If selected, types of object nodes are displayed and connected to the composition with activities containing object nodes.
- The **Search recursively check box** is selected by default:
  - If not selected, the search will be conducted in only one level of the selected activity.
  - If selected, the search will be conducted in the selected activity and those activities that are invoked by CallBehaviorActions that are in the selected activity. This search is recursive.

Copyright © 1998-2015 No Magic, Inc.

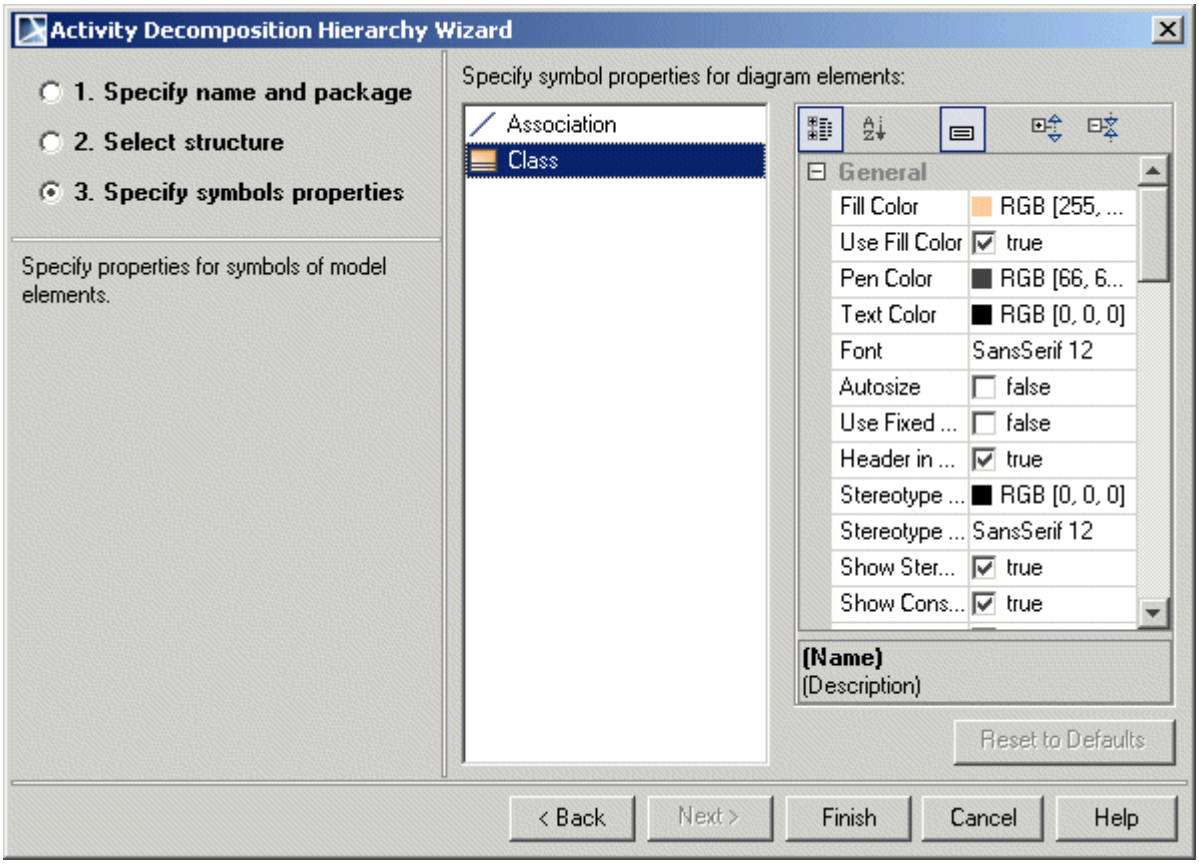


Figure 428 -- Activity Decomposition Hierarchy Wizard. Specify symbols properties step

Specify properties for symbols for model elements.

Content Diagram Wizard

NOTE

This feature is available in Standard, Professional, Architect, and Enterprise editions.

The **Content Diagram Wizard** generates content of diagrams that are used in the project.

To start the **Content Diagram Wizard**

Do one of the following:

- On the main menu select **Diagrams** > **Diagram Wizards** > **Content Diagram Wizard**.

Copyright © 1998-2015 No Magic, Inc.

- On the main menu select **Analyze** > **Model Visualizer**. The **Model Visualizer** dialog opens. Click **Content Diagram Wizard** in the wizards list and then click the **Start** button.

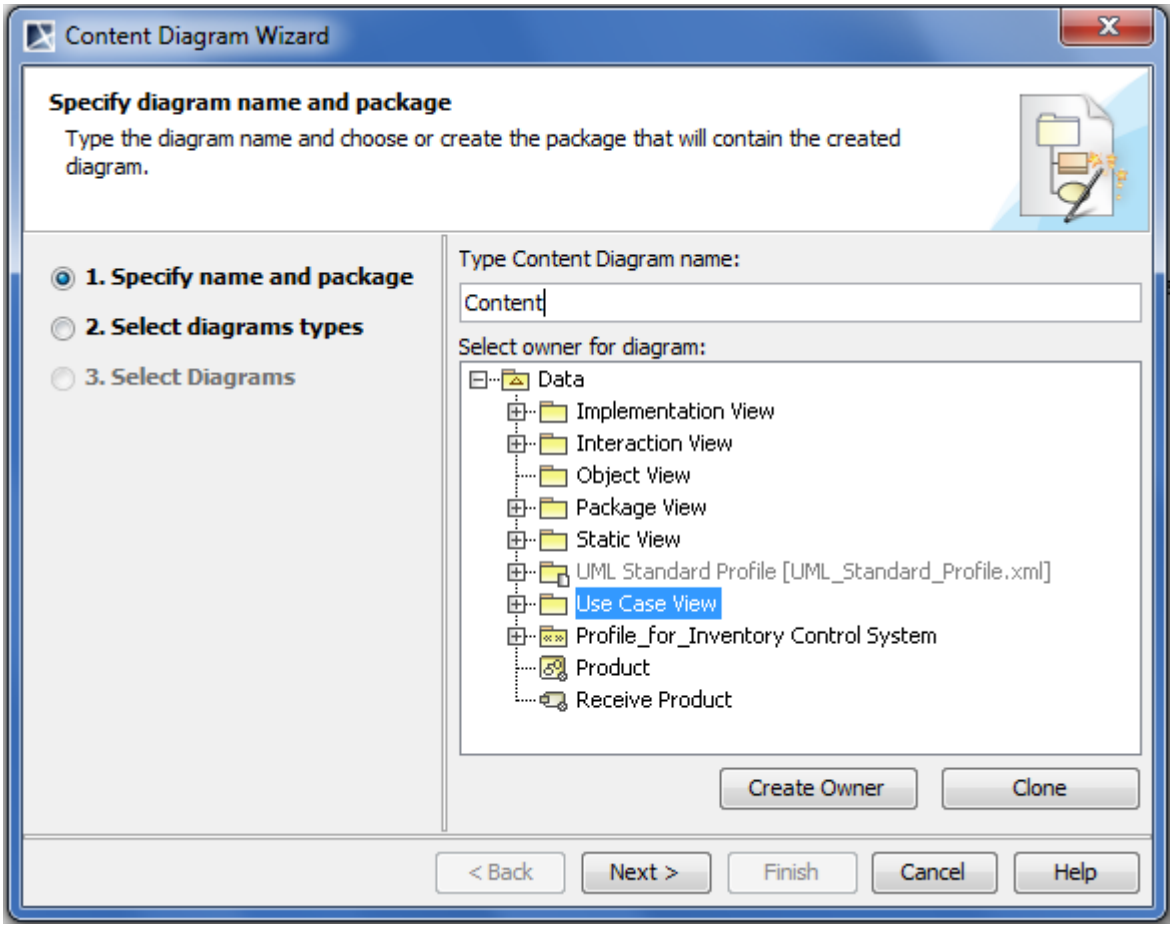


Figure 429 -- Content Diagram Wizard. Specifying name and package

Type a name for the new diagram in the **Type Content Diagram name** box.

Select the package that will contain the created diagram. The hierarchy of UML model packages is displayed in the **Select owner for diagram** tree. You can select the existing package or create a new one by clicking the **Create Owner** button, or copy the selected package with all its content by clicking the **Clone** button.

Copyright © 1998-2015 No Magic, Inc.

Click **Next** or choose the **Select diagram types** option for further steps.

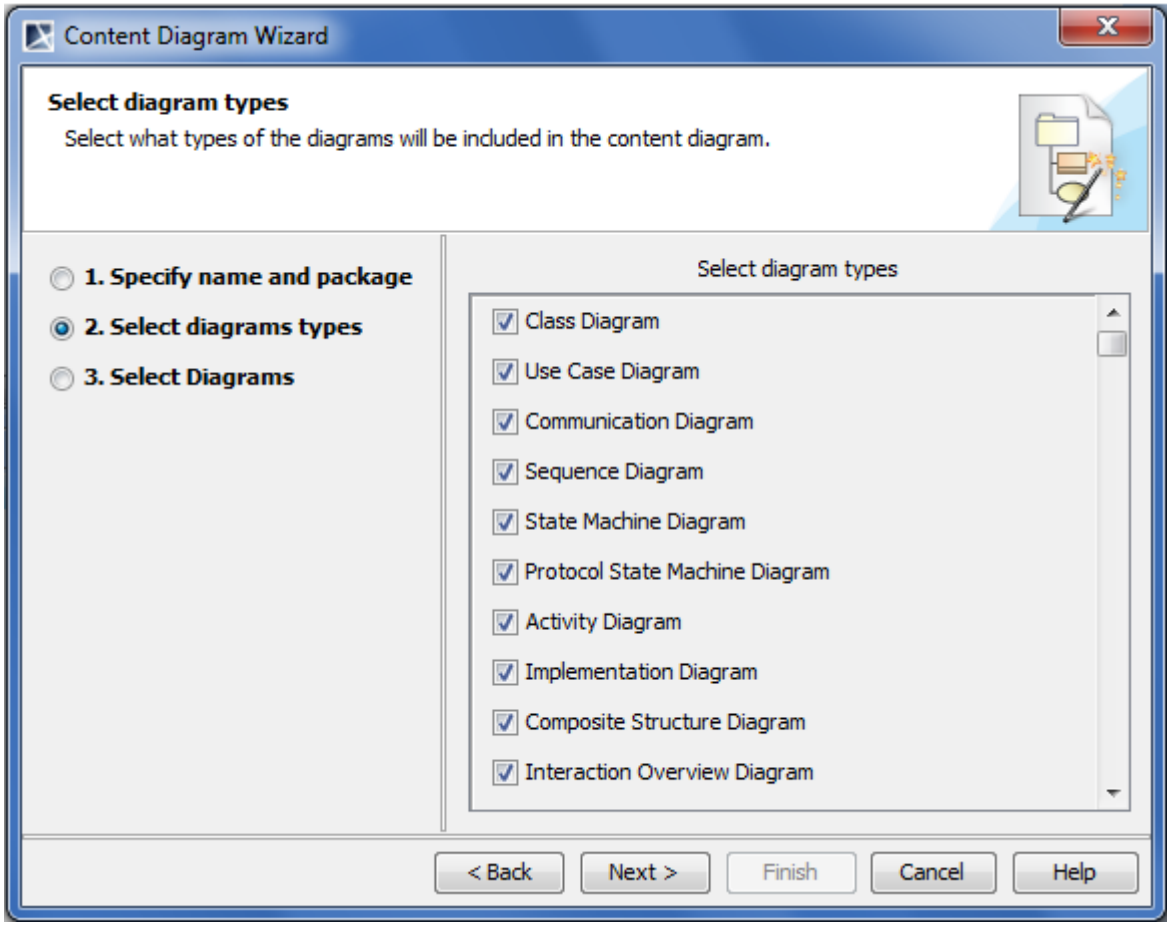


Figure 430 -- Content Diagram Wizard. Selecting diagram types

Copyright © 1998-2015 No Magic, Inc.

Select the types of diagrams to be included in the content diagram. Then click **Next** or choose the **Select Diagrams** option.

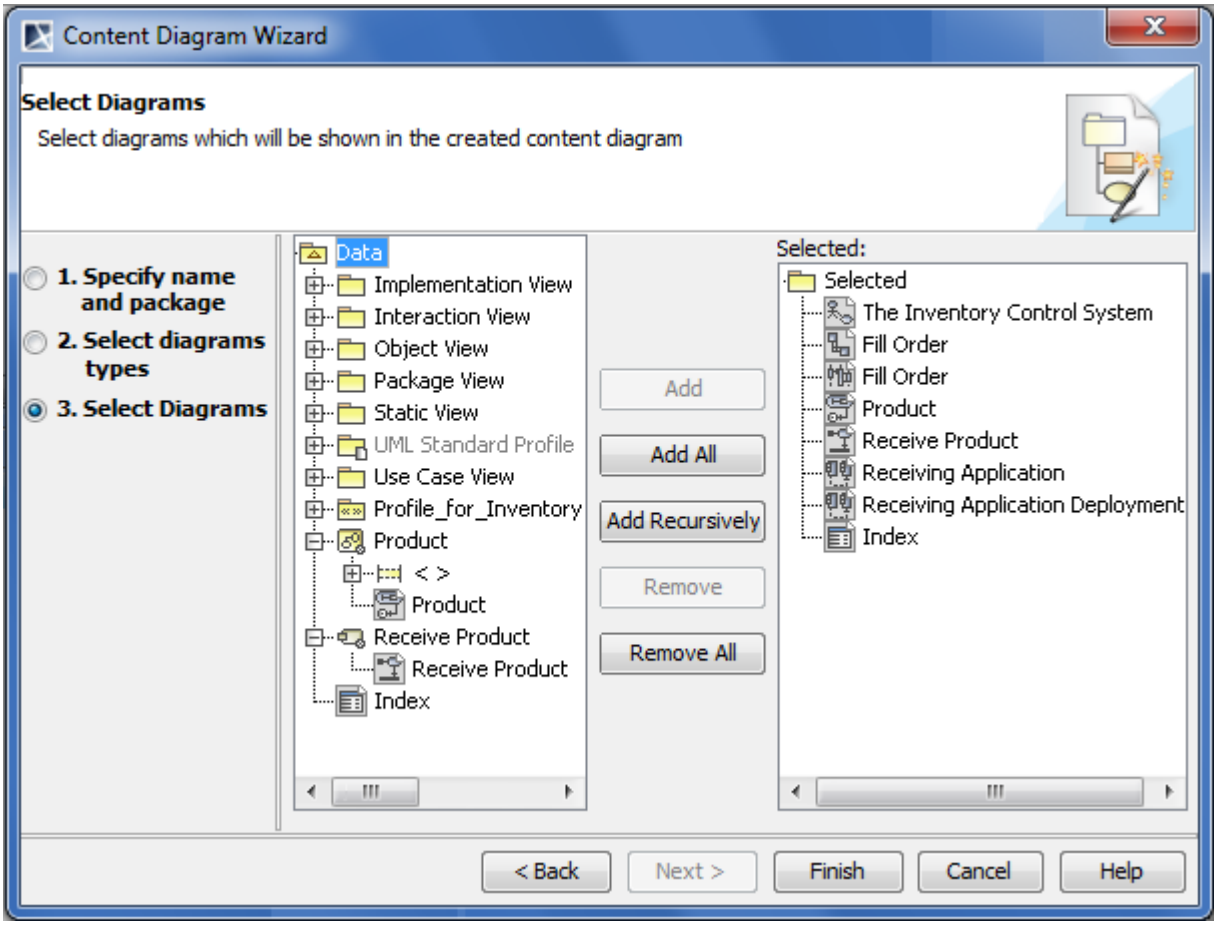


Figure 431 -- Content Diagram Wizard. Selecting diagrams

Select diagrams to be included in the content diagram and click **Finish**. The content diagram is now created.

Sequence Diagram from Java Source Wizard

NOTE

This feature is available only in the Enterprise edition.

The **Sequence Diagram from Java Source Wizard** allows for visualizing Java method implementation within UML Sequence diagrams. Though UML Sequence diagrams cannot show Java code with 100% accuracy yet, MagicDraw provides a mechanism for generating a diagram that reflects the essence of Java method content.

If you want to create a sequence diagram from the Java source, first of all you need to reverse the Java source code to a model. For the reverse procedure please refer to Section “Reverse” in MagicDraw Code Engineering UserGuide.pdf.

You can also to create a model corresponding to your Java code structure manually.

IMPORTANT!

Be sure, the model you use to create a sequence diagram fully corresponds the Java source code you want to represent in the diagram.

One sequence diagram can represent one method in a Java code. Classes are represented as lifelines, and method calls are represented as messages in the diagram. The sequence diagram can also be used to visualize dependencies for all classes used in this method.

Copyright © 1998-2015 No Magic, Inc.

To open the **Sequence Diagram from Java Source Wizard**

- Via the **Model Visualizer** dialog:
    1. From the main menu, select **Analyze** > **Model Visualizer**. The **Model Visualizer** dialog opens.
    2. In the dialog, select **Sequence Diagram from Java Source Wizard**.
    3. Click **Start**. The **Sequence Diagram from Java Source Wizard** opens.
- Via the **Diagram Wizards** submenu:
    1. From the main menu, select **Diagrams** > **Diagram Wizards** > **Sequence Diagram from Java Source Wizard**.
- In a manually created model:
    1. Create a class with an operation.
    2. Create a new Java code engineering set and drag the class to this set.
    3. Select the operation in any of the following places: the Containment tree, the Code engineering sets tree, or a message on a diagram pane with that operation assigned. How to assign an operation to a message, see Section "Assigning operations to messages" on page 920.
    4. Right-click the selected operation and, from the shortcut menu, select **Reverse Implementation**.

Creating sequence diagram from Java source

The **Sequence Diagram from Java Source Wizard** consists of these four steps:

- The diagram name and package specification
- The selection of an operation
- The selection of classes for the diagram
- The symbol properties specification

Copyright © 1998-2015 No Magic, Inc.

STEP #1: The diagram name and package specification

This step allows for specifying diagram name and selecting or creating a new owning package for the diagram.

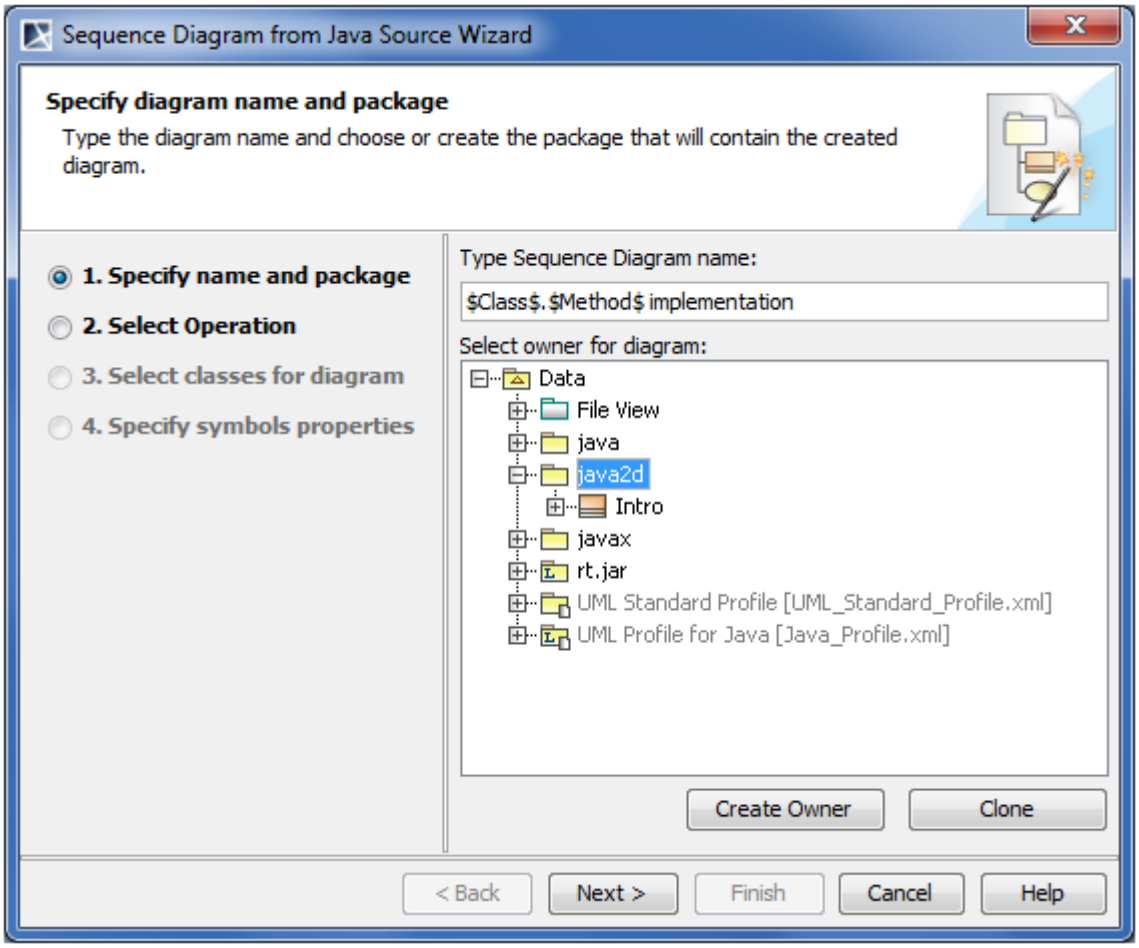


Figure 432 -- Sequence Diagram from Java Source Wizard. Specifying name and package

In the **Type Sequence Diagram name** box, type a name for the new sequence diagram.

Select the package that will contain the created diagram. The hierarchy of UML model packages is displayed in the **Select owner for diagram** dialog. Select the package that will be the owner for the newly created diagram, or create a new package by clicking **Create Owner** or **Clone**. For more information about an element creation refer to Section "Creating New Elements" on page 356.

Copyright © 1998-2015 No Magic, Inc.

STEP #2: The selection of an operation

This step is designed to select the operation that will be displayed in the sequence diagram and the Java source file for constructing the diagram.

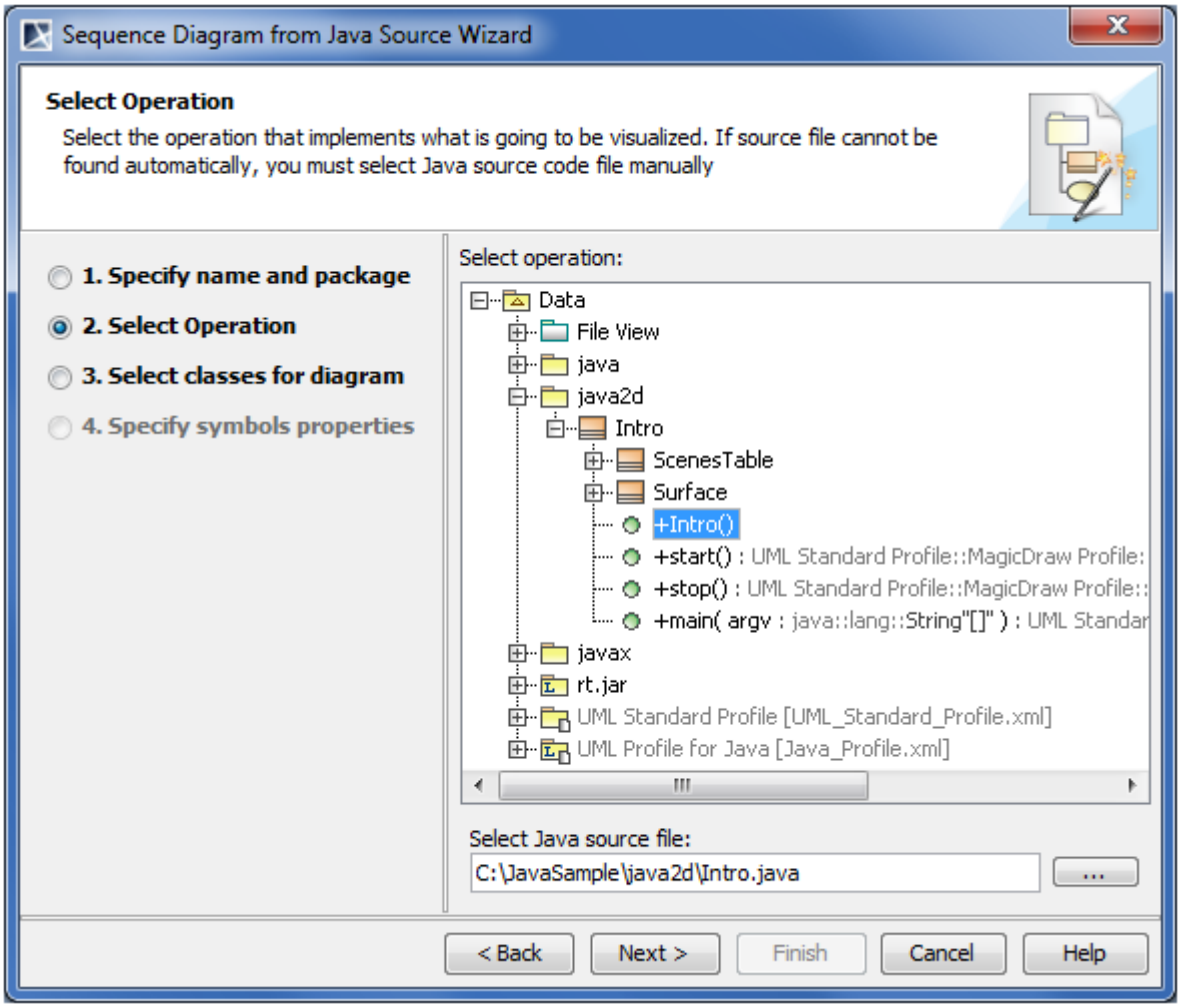


Figure 433 -- Sequence Diagram from Java Source Wizard. Selecting operation

When you start the wizard from either the operation shortcut menu or the Sequence diagram, both the operation and the Java source file are selected by default. If the Java source file cannot be found automatically, specify it in the **Select Java source file** box.

Copyright © 1998-2015 No Magic, Inc.

STEP #3: The selection of classes for the diagram

This step allows for selecting classes to represent them in a diagram.

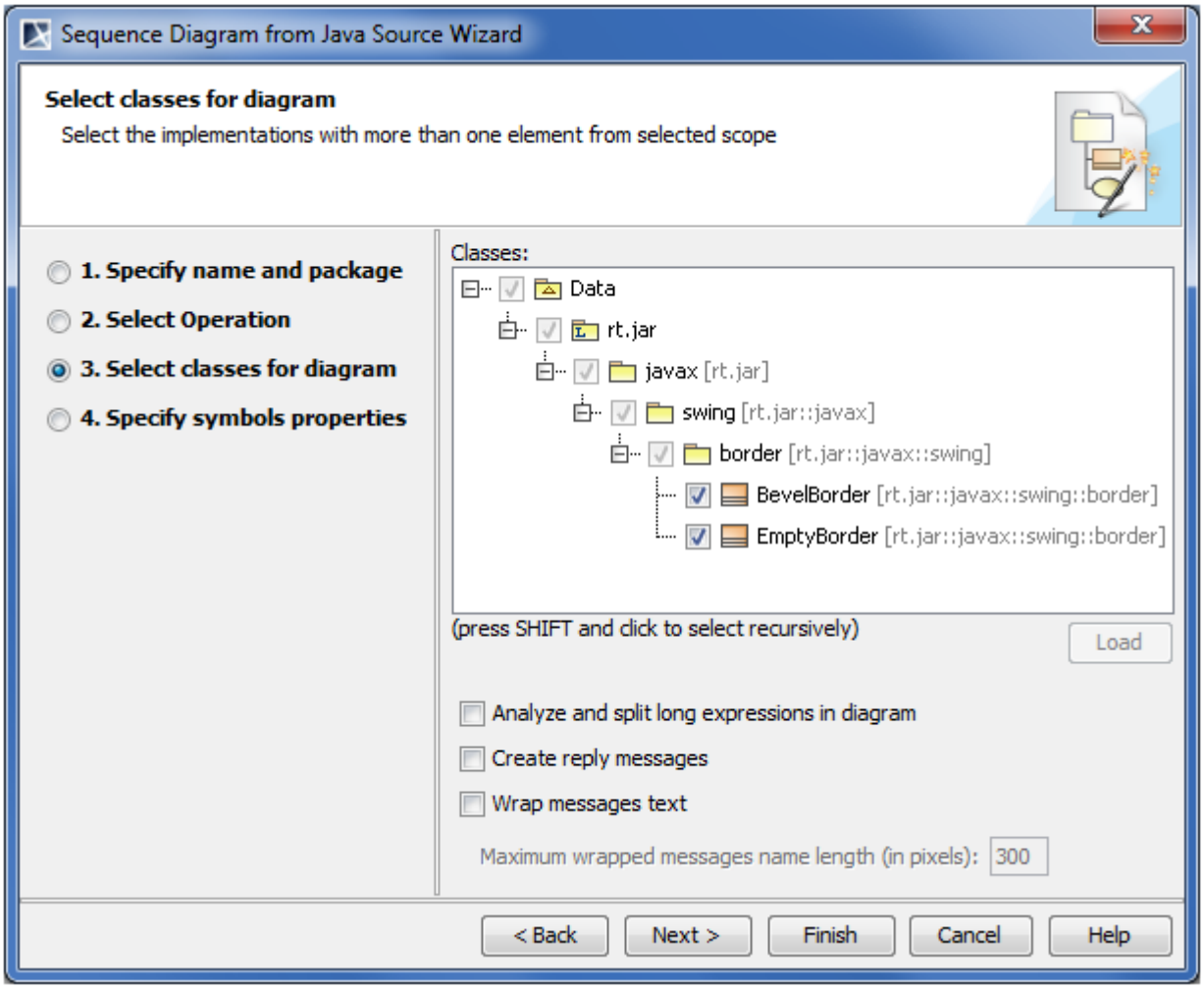


Figure 434 -- Sequence Diagram from Java Source Wizard. Selecting classes for diagram

Check boxes used in this step are described in the following table.

Check box	Description
Analyze and split long expressions in diagram	Select to display every call as a separate call message with a temporary variable initialization, if the expression containing calls cannot be displayed as a call message. In the final expression message, these calls are replaced with appropriate temporary variable names.
Create reply message	Select to display the return message for every call message.
Wrap message text	Select to wrap message text in the diagram. In the **Maximum wrapped messages name length (in pixels)** box, specify the maximum message text length in pixels.

Copyright © 1998-2015 No Magic, Inc.

STEP #4: The symbol properties specification

You can select properties for the elements that will be represented in your sequence diagram.

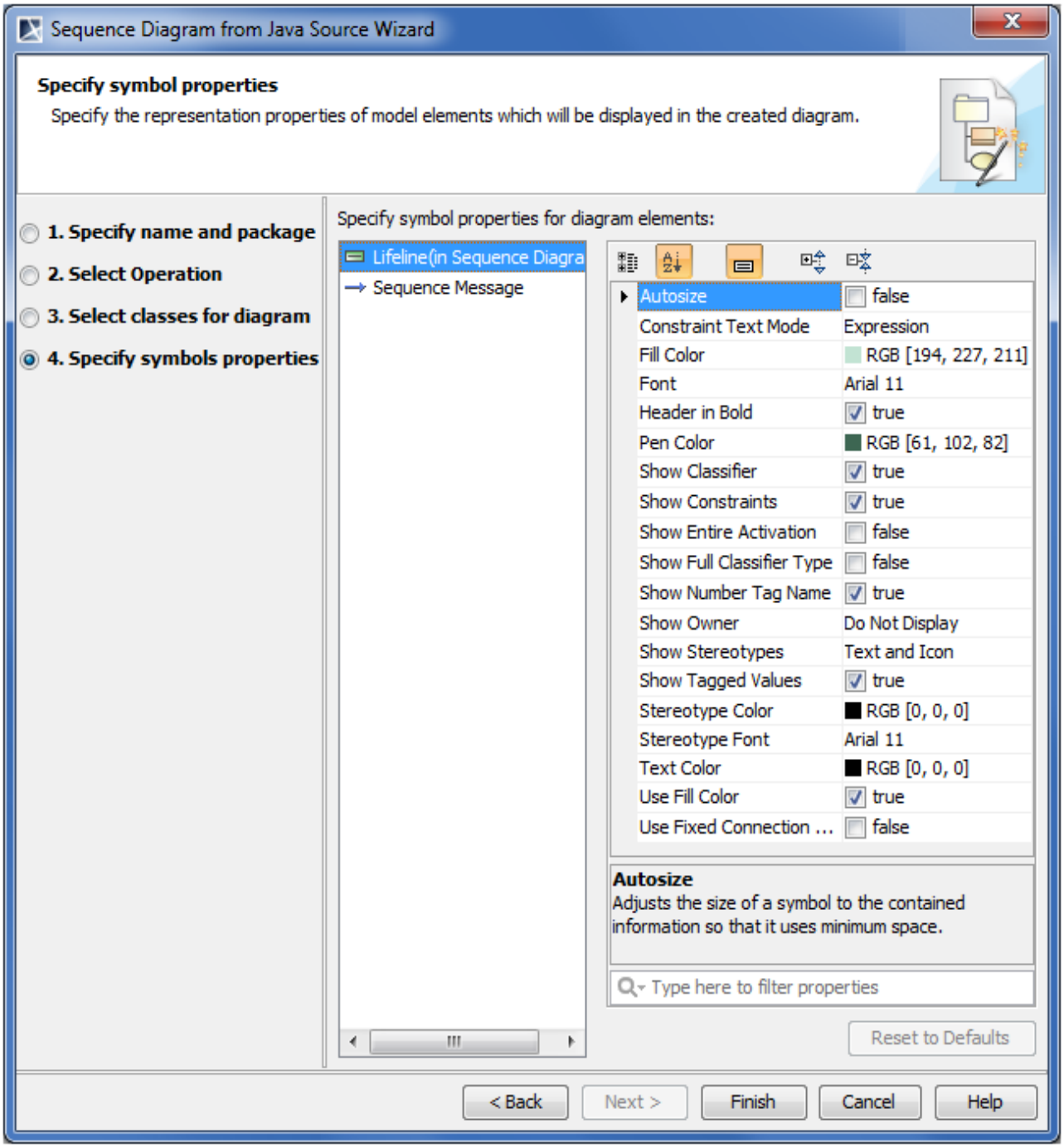


Figure 435 -- Sequence Diagram from Java Source Wizard. Specifying symbol properties

Extending sequence diagrams

Creating a sequence diagram from a Java code allows for analyzing dependencies between the methods represented on the diagram and the classes they are referencing to and / or the other methods they call for. This means that you can create a sequence diagram for any method selected in the already created sequence diagram.

Copyright © 1998-2015 No Magic, Inc.

To create a new sequence diagram for a selected method

1. Select the method (call message with an operation assigned), whose details you want to see.
2. Right-click the message and, from the shortcut menu, select **Reverse Implementation**. The **Sequence Diagram from Java Source Wizard** opens to create a sequence diagram for the selected method.



If you have started the wizard via the operation assigned to the message, be aware that in this case the first step of the wizard will be skipped, and the wizard will start from the step #2.

All selected method dependencies will be represented in a newly created diagram.

Copyright © 1998-2015 No Magic, Inc.

9 UML DIAGRAMS

In software development, the diagram is the equivalent of a blueprint. To meet the various needs of many parties, we often need several different “blueprints” of the same system. Furthermore, every system is described by many different aspects. For example:

- Functional (static structure and dynamic interactions)
- Nonfunctional (timing requirements, reliability, and deployment)
- Organizational (work organization and mapping to code modules)

MagicDraw supports the following diagrams that are defined in UML 2:

	Class Diagram
	Use Case Diagram
	Sequence Diagram
	Activity Diagram
	State Machine Diagram
	Component Diagram
	Object Diagram
	Package Diagram
	Deployment Diagram
	Communication Diagram
	Protocol State Machine Diagram
	Composite Structure Diagram
	Interaction Overview Diagram
	Profile Diagram

Copyright © 1998-2015 No Magic, Inc.

Architectural Views

UML defines 13 diagrams that describe 4+1 architectural views:

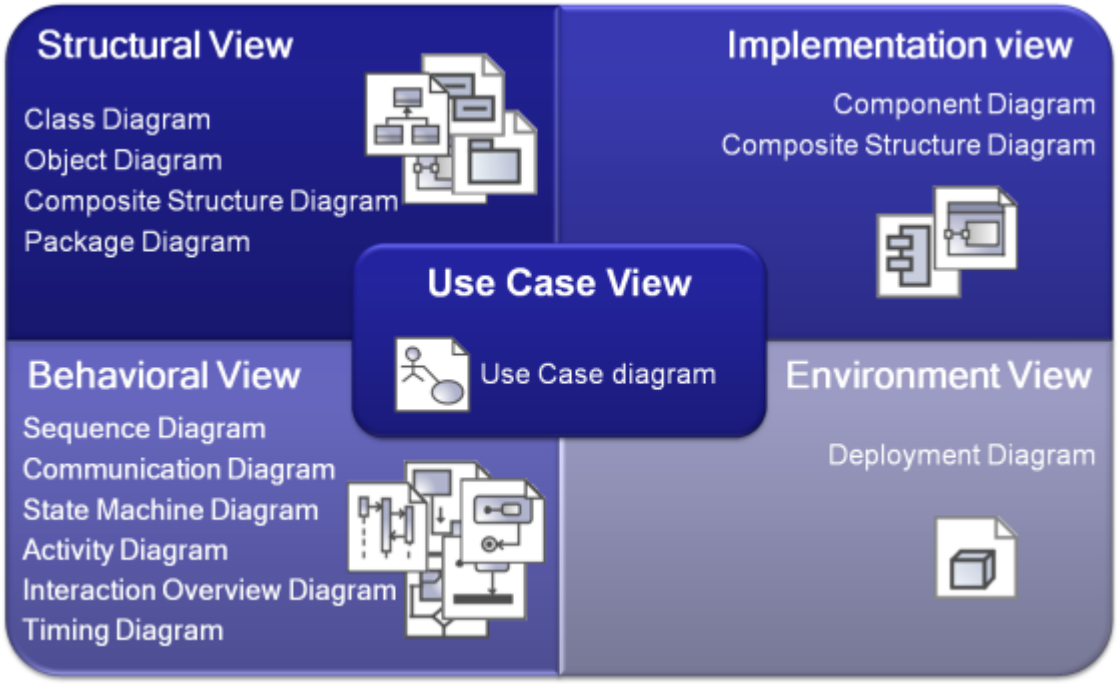


Figure 436 -- Architectural views

Several kinds of diagrams provide a visual notation for the concepts in each view.

Use Case View

The use case view represents the functionality and behavior of a system or subsystem as it is perceived by external users. This view is targeted mainly at customers, designers, developers, and testers.

The use case view usually is presented as a number of use cases and actors in Use Case diagrams. Occasionally it is used in Activity and Sequence diagrams.

The use case view is central because the contents drive the development of the other views. It is also used for project planning. Every single use case unit is deemed as a manageable unit during the project execution.

Structural View

The structural view represents structural elements for implementing a solution for defined requirements. It identifies all of the business entities and how these entities are related to each other. Usually entities are represented as classifiers and their instances in class and object diagrams in multiple abstraction levels. System decomposition to different layers can be displayed using Package diagrams. A Composite structure diagram can be used to represent the classifier inner structure. The system structural view artifacts are created by software architects and represent the system implementation design solutions.

Behavioral View

The dynamic behavior of the system is displayed on the Interaction (sequence and collaboration), State, Activity, Interaction overview, and Timing diagrams. It focuses mainly on the interactions that occur between objects inside a system, activities and work performed by the various parts of a system, and state changes

Copyright © 1998-2015 No Magic, Inc.

within a particular object or collaboration. Rather than defining the participants of the system, it defines how particular use cases are executed, which provides value for the external user. The dynamic view is concerned about what is happening inside the system and how those actions impact other participants.

Implementation View

The implementation view describes the implementation artifacts of logical subsystems defined in the structural view. It can include the intermediate artifacts used in a system construction (code files, libraries, data files, etc.) This view defines dependencies between the implementation components and their connections by the required and provided interfaces. Components and their relationships are displayed on the Component diagram. Inner parts of the component can be represented with the Composite structure diagrams. The implementation view helps analyze system parts and their dependencies in a higher component level.

Environment View

The environment view represents the physical arrangement of a system, such as computers and devices (nodes) and how they are connected to each other. In contrast to the component view, the deployment view is concerned with the physical structure of the system and the location of the software modules (components) manifested by artifacts within the system.

The environment view is displayed on the deployment diagram.

Class Diagram

A class diagram is a graphic representation of the static structural model. It shows classes and interfaces, along with their internal structure and relationships. The classes represent types of objects that are handled in a system. A class diagram does not show temporal information, it describes only the classification. The instances of those types (objects) are instantiated only on the runtime and are represented by an object and the interaction diagrams.

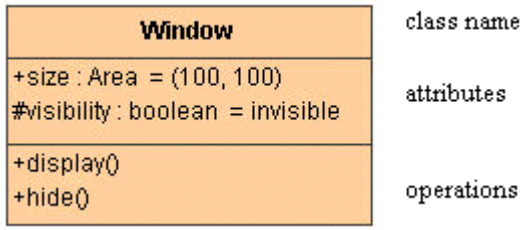
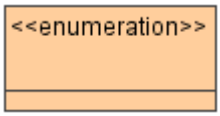
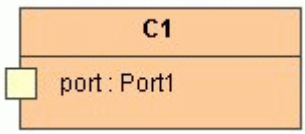
The classes can be related to each other in a number of ways: associated (connected to each other), dependent (one class depends/uses another class), specialized (one class is a subtype of another class), or packaged (grouped together as a unit – package). A class diagram does not express anything specific about the relationships of a given object, but it does abstractly describe the potential relationships of one object with other objects.

A system typically has a number of class diagrams – not all classes are inserted into a single class diagram. A class can have multiple levels of meaning and participate in several class diagrams.

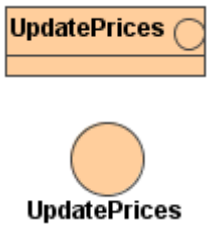
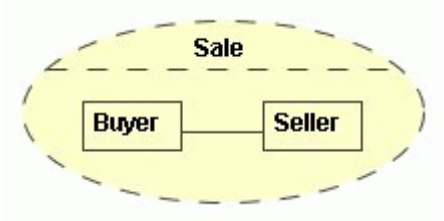
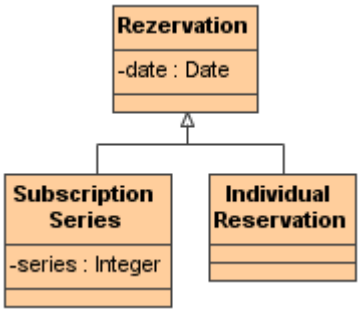
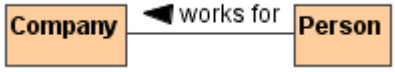
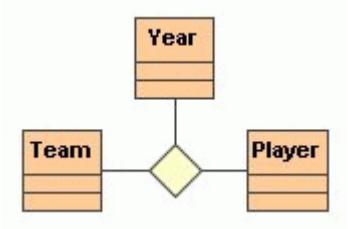
A class diagram is the logical map of an existing or future source code.

Copyright © 1998-2015 No Magic, Inc.

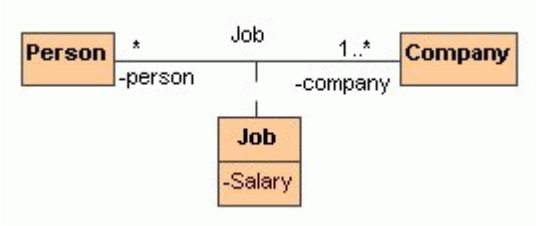
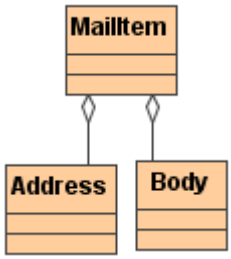
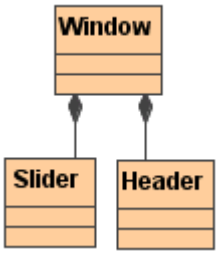
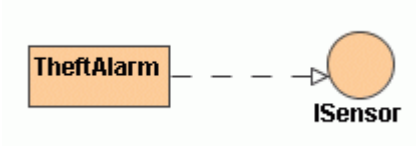
Class Diagram Elements

Model element	Button (hot key)	Notation
Class A descriptor for a set of objects with similar structures, behaviors, and relationships.	(C)	
Structured Class		
Class by Pattern	(SHIFT+P)	
Signal		
Data Type		
Primitive Type		
Enumeration A user-defined data type whose instances are a set of user-specified named enumeration literals. The literals have a relative order but no algebra is defined on them.	(K)	
Port A port is a property of a classifier that specifies a distinct interaction point between that classifier and its environment or between the (behavior of the) classifier and its internal parts.	(SHIFT+R)	

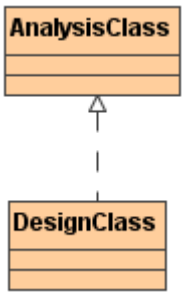
Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Interface The description of a visible behavior of a class, a component or a package. Attributes and operations inside the Interface can be suppressed.	 (I)	
Collaboration A collaboration is represented as a kind of classifier and it defines a set of cooperating entities to be played by instances (its roles) as well as a set of connectors that define communication paths between the participating instances.	 (Q)	
Generalization A relationship between a more general and a more specific element. **Note:** Choose a different Generalization direction from the toolbar to draw a line with an opposite arrow end.	 (G)	
Association A connection among classes, which also means a connection among objects of those classes.	 (S)	
Directed Association		
Non-navigable Association		
N-ary association An association among two or more classes (a single class can appear more than once).	 (O)	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Association Class The Association Class is a declaration of a semantic relationship between Classifiers. The Association Class, which has a set of features of its own, is both an Association and a Class.		
Non-navigable Association Class		
Aggregation An aggregation is an association that represents a whole-part relationship.	 (A)	
Directed Aggregation		
Composition A composition is a form of aggregation with a stronger ownership and coincident lifetime of part with the whole.	 (F)	
Directed Composition		
Interface Realization A relationship is usually used between an interface and an implementation class. **Note:** Choose a different Interface Realization direction from the toolbar to draw a line with an opposite arrow end.	 (R)	

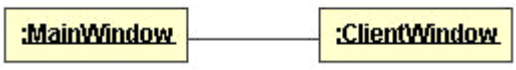
Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Realization A relationship between a specification and its implementation.	 (E)	
Substitution A substitution is a relationship between two classifiers.		
Usage A usage is a relationship in which one element requires another element (or set of elements) for its full implementation or operation. **Note:** Choose a different Usage direction from the toolbar to draw a line with an opposite arrow end.		
Template Binding A binding is a relationship between a template and a model element generated from the template.	 (B)	

Object Diagram

The Object diagram displays instances of classifiers and links (instances of associations) between them.

Object Diagram Elements

Model element	Button (hot key)	Notation
Instance An instance specification specifies existence of an entity in a modeled system and completely or partially describes the entity.	 (SHIFT+O)	
Link A connection between two or more objects.	 (SHIFT+L)	

Copyright © 1998-2015 No Magic, Inc.

Use Case Diagram

A use case is a description of the functionality (a specific usage of a system) that a system provides. The use case descriptions can exist in a textual form (a simple table), where the use case diagram provides additional information about the relationship between the use cases and the external users. The diagram also allows a definition of the system's boundary.

The Use cases are described only in terms of how they appear when viewed externally by the user (a system's behavior as the user perceives it), and do not describe how the functionality is provided inside the system. The Use cases are not object-oriented, but they are included in the UML to simplify the approach of the project's lifecycle -- from the specification to the implementation.

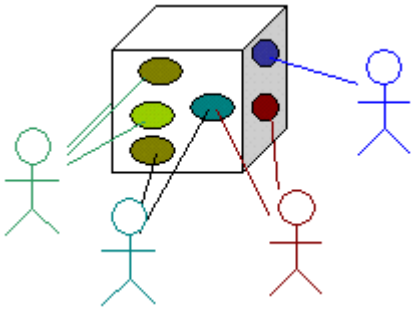
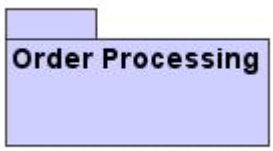
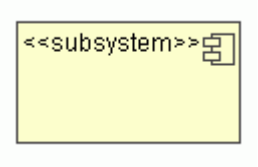
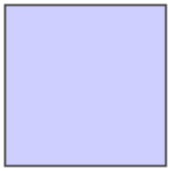
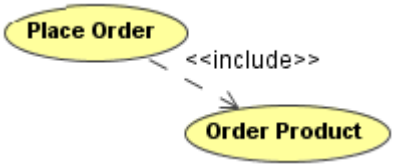
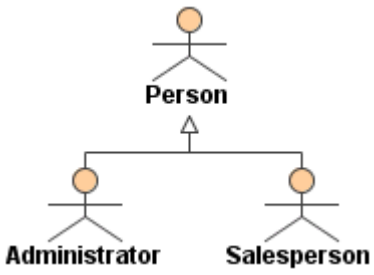


Figure 437 -- The schematic view of the use cases in the system.

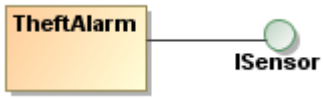
Use Case diagram elements

Model element	Button (hot key)	Notation
Actor Actors represent roles played by human users, external hardware, and other subjects. An actor does not necessarily represent a specific physical entity but merely a particular facet (that is, "role") of some entities that is relevant to the specification of its associated use cases.	 (A)	
Use Case A use case is a kind of behavior-related classifier that represents a declaration of an offered behavior. Each use case specifies a particular behavior, possibly including the variants that the subject can perform in collaboration with one or more actors. The subject of a use case could be a physical system or any other element that can initiate a behavior, such as a component, a subsystem, or a class.	 (U)	
Package A group of classes and other model elements. A package can contain other packages.	 (P)	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Subsystem A subsystem is treated as an abstract single unit. It groups model elements by representing the behavioral unit in a physical system.	(Y)	
System Boundary Another representation of a package. A system boundary element consists of use cases related by Exclude or Include (uses) relationships, which are visually located inside the system boundary rectangle.	(B)	
Include An include (uses) relationship from use case A to use case B indicates that an instance of the use case A will also contain the behavior as specified by B.	(C)	
Extend A relationship from an extending use case to an extended use case that specifies how and when the behavior defined in the extending use case can be inserted into the behavior defined in the extended use case. The extension takes place at one or more specific extension points defined in the extended use case. **Note:** Choose a different Extend direction from the toolbar to draw a line with an opposite arrow end.	(E)	
Association The participation of an actor in a use case, i.e. instances of the actor and instances of the use case communicate with each other. This is the only relationship between actors and use cases.	(S)	
Generalization A relationship between a more general and a more specific element. **Note:** Choose a different Generalization direction from the toolbar to draw a line with an opposite arrow end.	(G)	

Copyright © 1998-2015 No Magic, Inc.

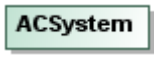
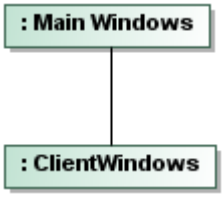
Model element	Button (hot key)	Notation
Interface Realization The classifier at the tail of the arrow implements the interface that is located at the arrow head or uses that interface. **Note:** Choose a different Interface Realization direction from the toolbar to draw a line with an opposite arrow end.	 (R)	

Communication Diagram

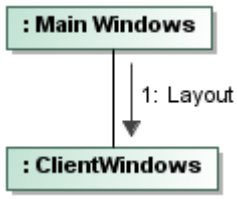
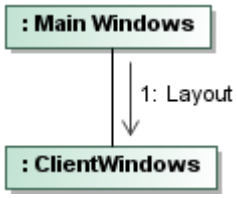
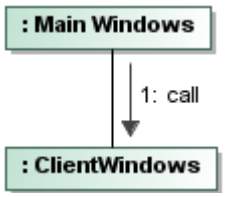
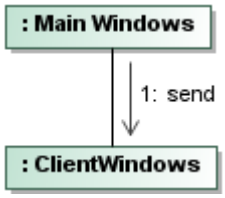
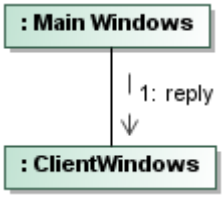
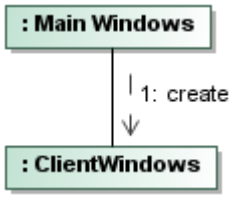
The Communication diagram illustrates the various static connections between objects, and models their interactions. It also presents a collaboration that contains a set of instances as well as their required relationships given in a particular context, and includes an interaction that defines a set of messages. These messages specify the interaction between the classifier roles within a collaboration that will serve to achieve the desired result.

A Communication diagram is given in two different forms: at the instance level or at the specification level.

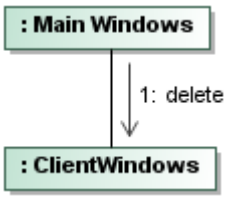
Communication Diagram elements

Model element	Button (hot key)	Notation
Lifeline A lifeline represents an individual participant in the Interaction. The Lifelines represent only one interacting entity.	 (O)	
Connector Specifies a link that enables communication between two or more lifelines. Each connector can be attached to two or more connectable elements, each representing a set of lifelines.	 (C)	
Connector to Self Self connector for self-calls. It begins and ends on the same lifeline.	 (S)	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Message to Right **Message to Left** A Message defines a particular communication between the Lifelines of an Interaction. It implies that one object uses the services of another object, or sends a message to that object. A communication can be formed by e.g., raising a signal, invoking an Operation, creating or destroying an Instance. Messages can be synchronous and asynchronous. Synchronous messages are used when the operation should be completed before the caller resumes the execution. Asynchronous messages are used when the sender is not waiting for the recipient's acceptance.	 	**Synchronous message**  **Asynchronous message** 
Call Message to Right **Call Message to Left** A call message represents the request to invoke a specific operation.	 	 Since the version 17.0.1 brackets "()" are not added to message names anymore.
Send Message to Right **Send Message to Left** A send message specifies the sending of a request to invoke a specific operation.	 	
Reply Message to Right **Reply Message to Left** A reply message returns the values to the caller of the previous call, completing the execution of the call.	 	
Create Message to Right **Create Message to Left** A create message specifies the creation of a specific operation.	 	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Delete a Message to Right **Delete Message to Left** A delete message represents the destruction of the instance described by the lifeline.		

NOTE When an operation or a signal is assigned to a message, the name of the operation or the signal appears instead of the message name.

Sequence Diagram

The Sequence diagram focuses on the Message interchange between a number of Lifelines.

A sequence diagram shows the interaction information with an emphasis on the time sequence. The diagram has two dimensions: the vertical axis that represents time and the horizontal axis that represents the participating objects. The time axis could be an actual reference point (by placing the time labels as text boxes). The horizontal ordering of the objects is not significant to the operation, and you can rearrange them as necessary.

In the less below you will find the description and notation of all the elements available in the Sequence diagram.

Related sections

"Message"

Copyright © 1998-2015 No Magic, Inc.

Sequence diagram elements

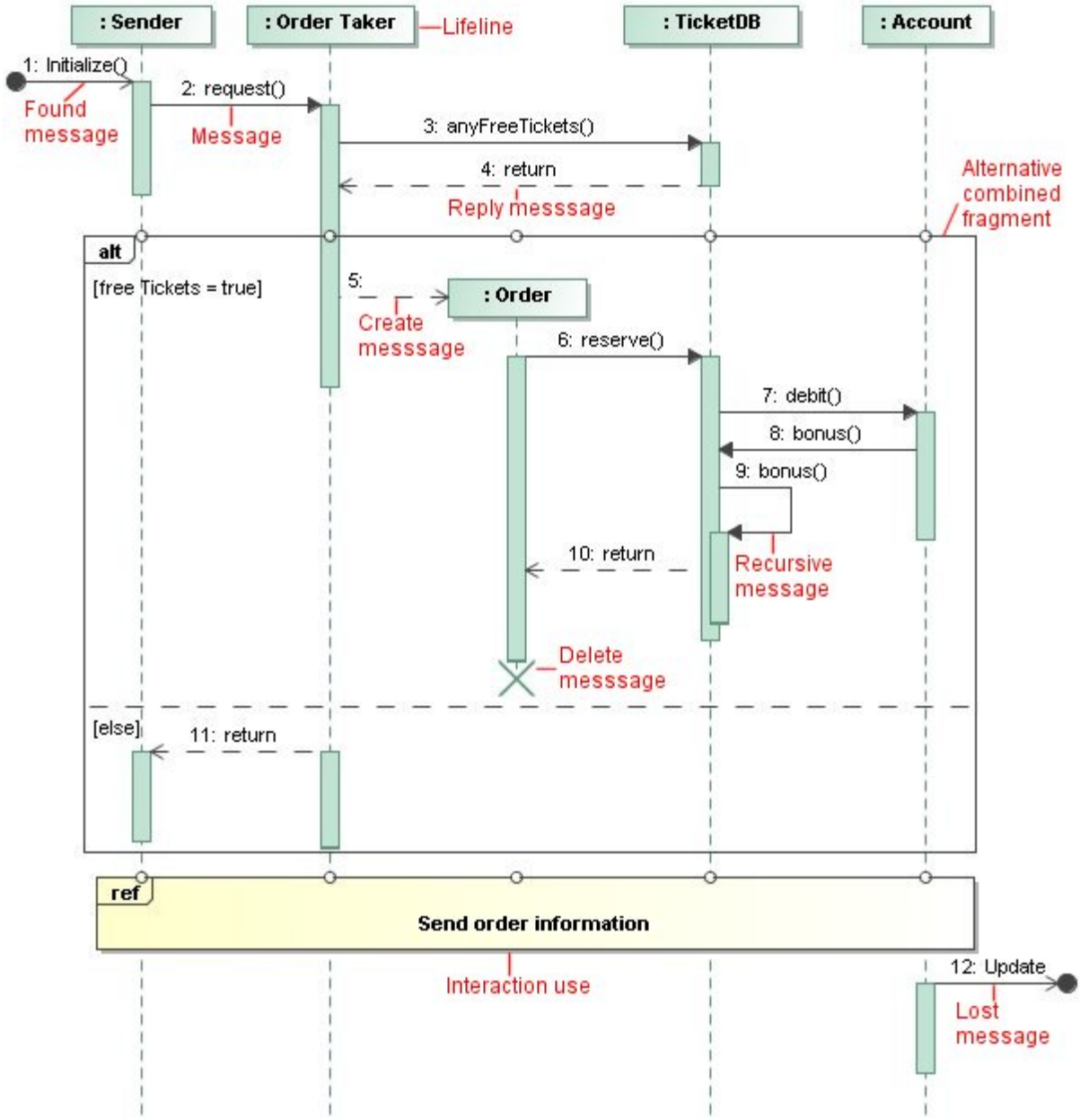
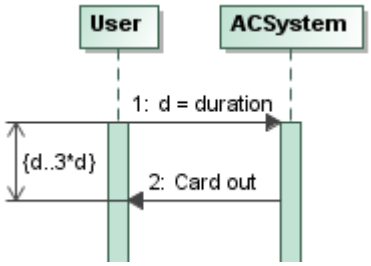
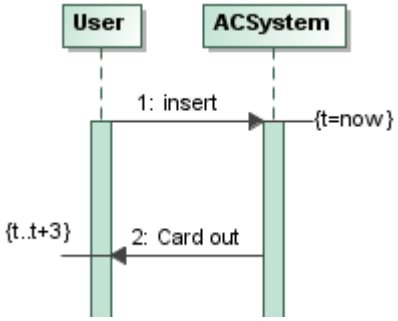
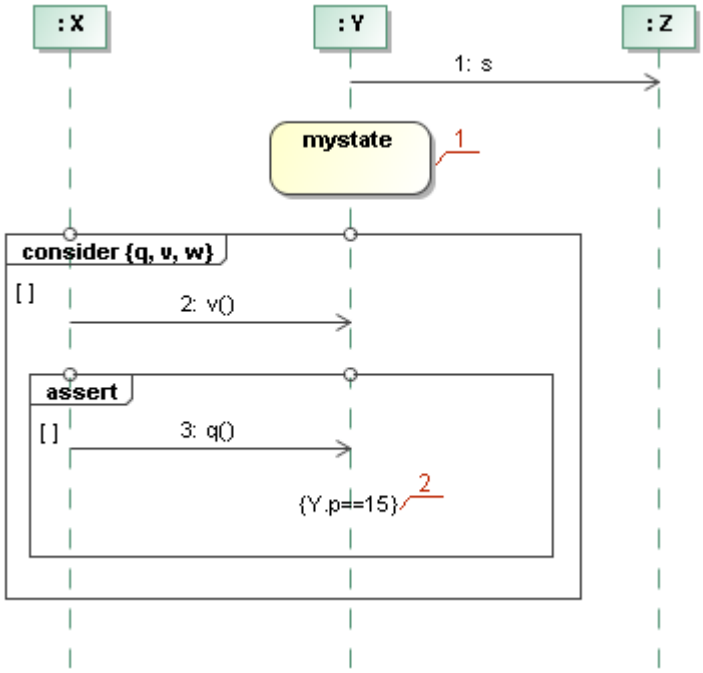


Figure 438 -- Elements in sequence diagram

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Lifeline Represents the existence of an object at a particular time. **Activation Bar** Focus of control. Shows the period during which an object is performing an action either directly or through a subordinated procedure.	 (O)	
Interaction Use A reference to interactions, communication diagram, sequence diagram, and time diagram can be created.	 (SHIFT+T)	See Figure 438 on page 684.
Duration Constraint A duration defines a value specification that specifies the temporal distance between two time instants.		
Time Constraint Specifies the combination of min and max timing interval values.		

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
State Invariant A StateInvariant is a runtime constraint on the participants of the interaction. It can be used to specify a variety of different kinds of constraints, such as values of attributes or variables, internal or external states, and so on.	 (SHIFT+V)	1. State Invariant with a assigned “mystate” State. 2. State Invariant with a defined constraint Y.p=15. 

Combined fragments

A combined fragment defines an expression of interaction fragments. A combined fragment is defined by an interaction operator and corresponding interaction operands. Through the use of combined fragments you will be able to describe a number of traces in a compact and concise manner.

Interaction operands can be of the following types and are described in the table below.

Interaction operand name and description	Button (hot key)
Alternatives The alternative combined fragment **alt** represents a choice of behavior. Alternative combined fragment has several operands. At most one of the operands has to be chosen. Using alternative combined fragment you can model if-then-else statement.	 (SHIFT+A)
Loop The loop combined fragment represents that the **loop** operand will be repeated a number of times. If the loop contains a separate interaction constraint with a specification, the loop will only continue if that specification evaluates to true during execution regardless of the minimum number of iterations specified in the loop.	 (SHIFT+L)
Option The option combined fragment **opt** represents a choice of behavior where either the (sole) operand happens or nothing happens. An option combined fragment is used to model "if-then" construct.	 (SHIFT+O)

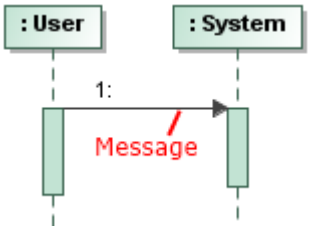
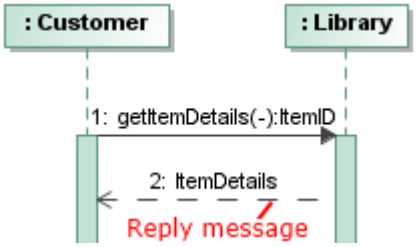
Copyright © 1998-2015 No Magic, Inc.

Interaction operand name and description	Button (hot key)
Parallel The interaction operator **par** designates that the combined fragment represents a parallel merge between the behaviors of the operands. A parallel merge defines a set of traces that describes all the ways that occurrence specifications of the operands can be interleaved without obstructing the order of the occurrence specifications within the operand.	 (SHIFT+P)
Break The interaction operator **brk** designates that the combined fragment represents a breaking scenario in the sense that the operand is a scenario that is performed instead of the remainder of the enclosing interaction fragment. A break operator with a guard is chosen when the guard is true and the rest of the enclosing interaction fragment is ignored. When the guard of the break operand is false, the break operand is ignored and the rest of the enclosing interaction fragment is chosen. A combined fragment with interaction operator break should cover all Lifelines of the enclosing interaction fragment.	 (SHIFT+B)
Negative The interaction operator **neg** designates that the combined fragment represents traces that are defined to be invalid. The set of traces that defined a combined fragment with interaction operator negative is equal to the set of traces given by its (sole) operand, only that this set is a set of invalid rather than valid traces. All interaction fragments that are different from Negative are considered positive meaning that they describe traces that are valid and should be possible.	 (SHIFT+G)
Critical Region The interaction operator **crt** designates that the combined fragment represents a critical region. A critical region means that the traces of the region cannot be interleaved by other occurrence specifications (on those Lifelines covered by the region). This means that the region is treated atomically by the enclosing fragment when determining the set of valid traces.	 (SHIFT+R)
Consider The interaction operator **con** designates which messages should be considered within this combined fragment. This is equivalent to defining every other message to be ignored.	 (SHIFT+C)
Ignore The interaction operator **ign** designates that there are some message types that are not shown within this combined fragment. These message types can be considered insignificant and are implicitly ignored if they appear in a corresponding execution. Alternatively, one can understand ignore to mean that the message types that are ignored can appear anywhere in the traces.	 (SHIFT+I)
Weak Sequencing The interaction operator **seq** designates that the combined fragment represents a weak sequencing between the behaviors of the operands. It is the same as parallel execution, except that event on the same lifeline from different subfragments are ordered in the same order as the subfragments within the enclosing weak sequencing fragment.	 (SHIFT+W)
Strict Sequencing The interaction operator **str** designates that the combined fragment represents a strict sequencing between the behaviors of the operands. The semantics of strict sequencing defines a strict ordering of the operands on the first level within the combined fragment with interaction operator strict.	 (SHIFT+S)

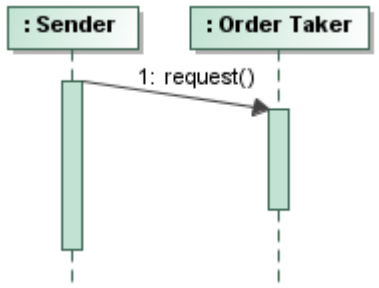
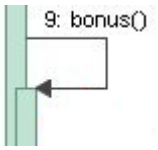
Copyright © 1998-2015 No Magic, Inc.

Interaction operand name and description	Button (hot key)
Assertion The interaction operator **asr** designates that the combined fragment represents an assertion. The sequences of the operand of the assertion are the only valid continuations. All other continuations result in an invalid trace.	(SHIFT+R)

Messages

Model element	Button (hot key)	Notation
Message A communication between objects that conveys information with the expectation that an action will ensue. This Message button is used to create abstract messages - that you can represent a sequence diagram in an abstract level. For more information about message, see "Message", on page 917.	(M)	 **Note**: Since the version 17.0.1, brackets “()” are not added to message names anymore.
Call Message A call message represents the request to invoke a specific operation.	(A)	
Send Message A send message specifies the sending of a request to invoke a specific operation.	(E)	
Reply Message The reply message returns the values to the caller of the previous call, completing the execution of the call.	(R)	
Create Message A create message specifies the creation of a specific operation.	(C)	The message is connected directly to an object (not lifeline).
Delete Message Destroy message represents the destruction of the instance described by the lifeline.	(T)	A large X mark is displayed on the object’s lifeline in the message’s destination.

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Diagonal Message Requires some time to arrive, during which another action occurs.	 (D)	
Message to Self	 (S)	
Recursive message A connected set of messages can be enclosed and marked as iteration.	 (U)	See Figure 438 on page 684.
Lost Messages A lost message is a message where the sending event occurrence is known, but there is no receiving event occurrence. We interpret this to be because the message never reached its destination.	 (L)	See Figure 438 on page 684.
Found Messages A found message is a message where the receiving event occurrence is known, but there is no (known) sending event occurrence. We interpret this to be because the origin of the message is outside the scope of the description. This can be noise for example or other activity that we do not want to describe in detail.	 (F)	See Figure 438 on page 684.

State Machine Diagram

The behavior of objects of a class can be described in terms of states and events, using a state machine connected to the class under a construction.

The state machine is a specification of the sequence of states through which an object or an interaction goes in response to events during its life, together with its responsive actions. The state machine can represent the sequence of states of a particular collaboration (for example, collection of objects) or even the whole system (which is also considered as a collaboration). The abstraction of all possible states defined in a state machine is

Copyright © 1998-2015 No Magic, Inc.

similar to the way class diagrams are abstracted: all possible object types (classes) of a particular system are described.

Objects that do not present a very pronounced reactive behavior can always be considered to stay in the same state. In such a case, their classes do not possess a state machine.

State diagrams (also called Statechart diagrams) represent the behavior of entities capable of dynamic behavior by specifying its response to the receipt of event instances. Typically, the state diagrams describe the behavior of classes, but the statecharts can also describe the behavior of other model entities such as use-cases, actors, subsystems, operations, or methods.

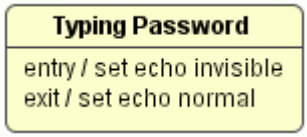
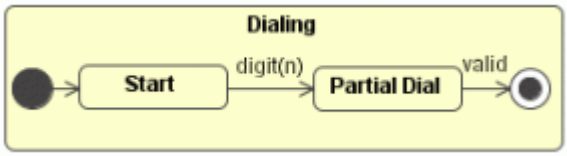
A state diagram is a graph that represents a state machine. States and various other types of vertices (pseudostates) in the state machine graph are rendered by the appropriate state and pseudostate symbols, while transitions are generally rendered by directed arcs that inter-connect them. The states can also contain subdiagrams by a physical containment or tiling. Note that every state machine has a top state, which contains all the other elements of the entire state machine. The graphical rendering of this top state is optional.

The states are represented by the state symbols, while the transitions are represented by arrows connecting the state symbols.

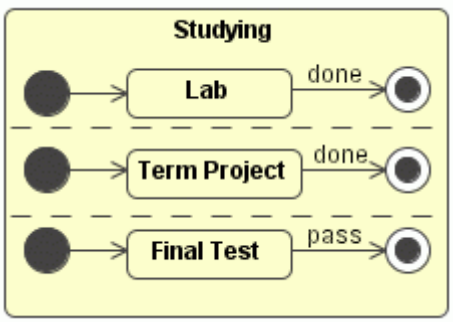
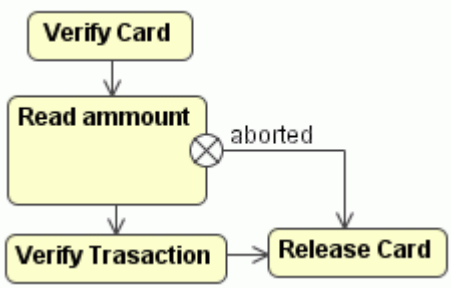
The state diagram concerns with an internal object changes (as opposed to the external object interaction in a collaboration). Do not attempt to draw them for all classes in the system, because they are used only for modeling a complex behavior. The state diagram shows all the possible states that objects or collaborations may have, and the events that cause the state to change. An event can be another object that sends a message to it announcing for example that a specified time has elapsed or that some conditions have been fulfilled. A change of a state is called a transition. A transition may also have a connected to it action that specifies what should be done in connection with the state transition.

State Machine Diagram elements

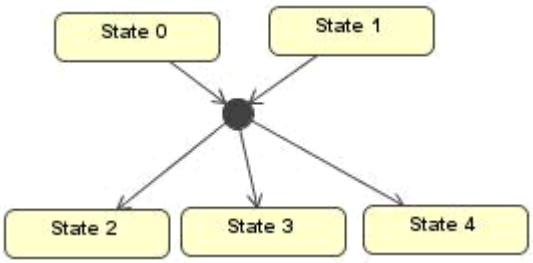
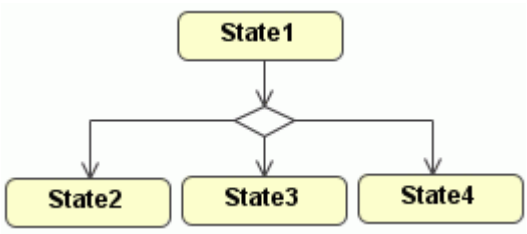
An initial pseudo state and a state connected to it are automatically created when a state machine diagram is created.

Model element	Button (hot key)	Notation
State A state models a situation during which some (usually implicit) invariant conditions holds. The invariant may represent a static situation such as an object waiting for some external events to occur. However, it can also model dynamic conditions such as the process of performing some behavior.	 (SHIFT+S)	
Composite State A composite state either contains one region or is decomposed into two or more orthogonal regions. Each region has a set of mutually exclusive disjoint subvertices and a set of transitions.	 (SHIFT+C)	

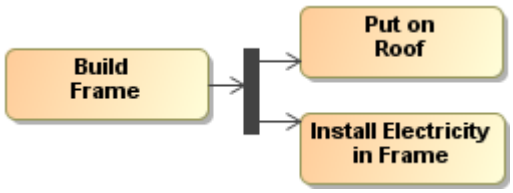
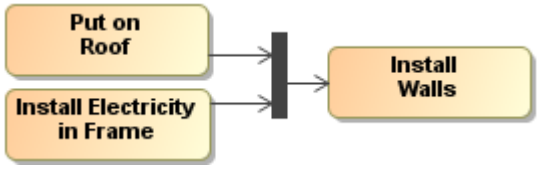
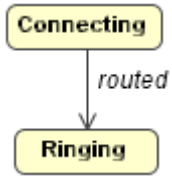
Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Orthogonal State An orthogonal state is a composite state with at least 2 regions.	(C)	
Submachine State The submachine state specifies the insertion of the specification of a submachine state machine. The submachine state is a decomposition mechanism that allows factoring of common behaviors and their reuse.	(A)	
Initial Pepresents a default vertex that is the source for a single transition to the *default* state of a composite state. There can be at most one initial vertex in a region.	(I)	
Final State A special kind of state signifying that the enclosing region is completed. If the enclosing region is directly contained in a state machine and all other regions in the state machine also are completed, then it means that the entire state machine is completed.	(F)	
Terminate Implies that the execution of this state machine by means of its context object is terminated.	(R)	
Entry Point The entry point connection points a reference as the target of a transition. This implies that the target of the transition is the entry point pseudostate as defined in the submachine of the submachine state.	(Y)	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Exit Point The exit point connection points a reference as the source of a transition. This implies that the source of the transition is the exit point pseudostate as defined in the submachine of the submachine state that has the exit point connection point defined.	 (U)	
Connection Point Reference The Connection point references of a submachine state can be used as the sources/targets of the transitions. They represent entries into or exits out of the submachine state machine referred by the submachine state.	 (Z)	
Deep History Represents the most recent active configuration of the composite state that directly contains the pseudostate (e.g., the state configuration that was active when the composite state was last exited).	 (P)	
Shallow History Represents the most recent active substate of its containing state (but not the substates of that substate). A composite state can have at most one shallow history vertex.	 (SHIFT+R)	
Junction The junction vertices are semantic-free vertices that are used to chain together multiple transitions. They are used to construct the compound transition paths between states.	 (J)	
Choice The choice points are used to split transition paths. In the dynamic choice point, a decision concerning which branch to take is only made after the transition from State1 is taken and the choice point is reached.	 (O)	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Fork Vertical/Horizontal Serves to split an incoming transition into two or more transitions terminating on orthogonal target vertices (i.e., vertices in different regions of a composite state).	 (G)  (D)	
Join Vertical/Horizontal Serves to merge several transitions emanating from source vertices in different orthogonal regions.	 (G)  (D)	
Transition A directed relationship between a source vertex and a target vertex. It may be part of a compound transition, which takes the state machine from one state configuration to another, representing the complete response of the state machine to an occurrence of an event of a particular type.	 (T)	
Transition to Self When an object returns to the same state after the specified event occurs.	 (E)	

Displaying Inner Elements on State Machine Diagrams

NEW! To display inner elements on a State Machine diagram

Do one of the following:

- Right-click a State Machine in the Containment tree, point to **Create Diagram**, and then click **State Machine Diagram**. Note that inner elements are displayed on the State Machine diagram upon the first diagram creation under the State Machine.
- Right-click the empty area on the State Machine diagram, point to **Related Elements**, and then click **Display Inner Elements**.



To display inner elements on the Composite State symbol, right-click the Composite State on the diagram, point to **Related Elements**, and then click **Display Inner Elements**.

Copyright © 1998-2015 No Magic, Inc.

Protocol State Machine Diagram

A protocol state machine is always defined in the context of a classifier. It specifies which operations of the classifier can be called, in which state, and under which condition, thus specifying the allowable call sequences on the classifier's operations.

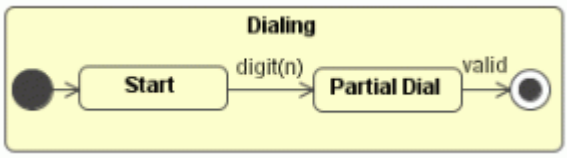
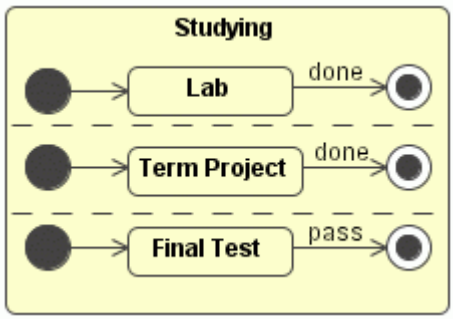
The protocol state machine presents the possible and permitted transitions on the instances of its context classifier, together with the operations that carry the transitions.

In this manner, an instance lifecycle can be created for a classifier, by specifying the order in which the operations can be activated and the states through which the instance progresses during its existence.

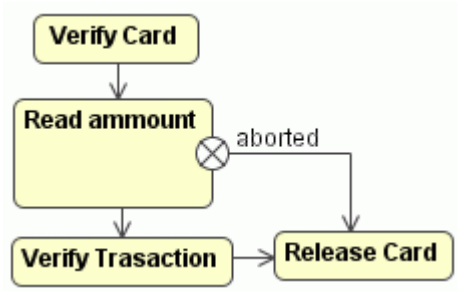
The Protocol State Machine Diagram is created for use with the Protocol State Machine and the Protocol Transitions.

Protocol State Machine Diagram elements

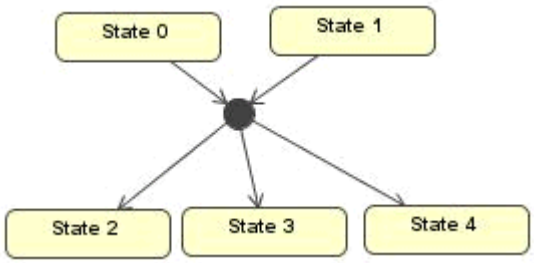
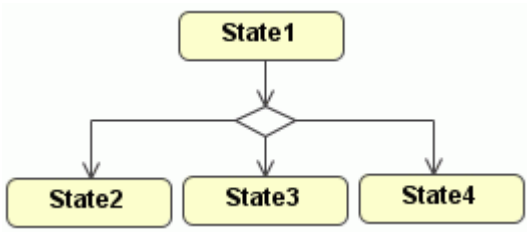
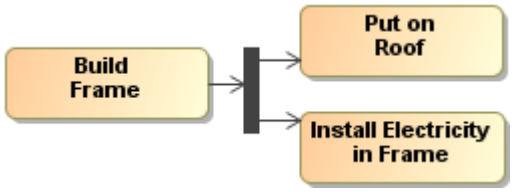
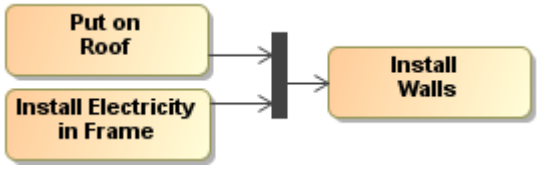
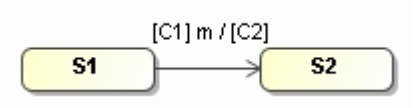
Since MagicDraw version 17.0.2, an initial pseudo state and a state connected to it are automatically created when the protocol state machine diagram is created.

Model element	Button (hot key)	Notation
State The states of the protocol state machines are exposed to the users of their context classifiers. A protocol state represents an exposed stable situation of its context classifier: When an instance of the classifier is not processing any operation, the user of this instance can always know its configuration state.	 (SHIFT+S)	
Composite State A composite state either contains one region or is decomposed into two or more orthogonal regions. Each region has a set of mutually exclusive disjoint subvertices and a set of transitions.	 (SHIFT+C)	
Orthogonal State An orthogonal state is a composite state with at least 2 regions.	 (C)	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Submachine State A submachine state specifies the insertion of the specification of a submachine state machine. The submachine state is a decomposition mechanism that allows factoring of common behaviors and their reuse.	 (A)	
Initial Pepresents a default vertex that is the source for a single transition to the *default* state of a composite state. There can be at most one initial vertex in a region.	 (I)	
Final State A special kind of state signifying that the enclosing region is completed. If the enclosing region is directly contained in a state machine and all other regions in the state machine also are completed, then it means that the entire state machine is completed.	 (F)	
Terminate Implies that the execution of this state machine by means of its context object is terminated.	 (R)	
Entry Point The entry point connection points a reference as the target of a transition. This implies that the target of the transition is the entry point pseudostate as defined in the submachine of the submachine state.	 (Y)	
Exit Point The exit point connection points a reference as the source of a transition. This implies that the source of the transition is the exit point pseudostate as defined in the submachine of the submachine state that has the exit point connection point defined.	 (U)	
Connection Point Reference The connection point references of a submachine state that can be used as the sources/targets of the transitions. They represent entries into or exits out of the submachine state machine referenced by the submachine state.	 (Z)	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Junction The junction vertices are semantic-free vertices that are used to chain together multiple transitions. They are used to construct the compound transition paths between states.	 (J)	
Choice The choice points are used to split transition paths. In the dynamic choice point, a decision on which branch to take is only made after the transition from State1 is taken and the choice point is reached.	 (O)	
Fork Vertical/Horizontal Serves to split an incoming transition into two or more transitions terminating on orthogonal target vertices (i.e., vertices in different regions of a composite state).	 (G)  (D)	
Join Vertical/Horizontal Serves to merge several transitions emanating from source vertices in different orthogonal regions.	 (G)  (D)	
Protocol Transition A protocol transition (transition as specialized in the ProtocolStateMachines package) specifies a legal transition for an operation. Transitions of the protocol state machines have the following information: a pre condition (guard), on trigger, and a post condition. Every protocol transition is associated to zero or one operation that belongs to the context classifier of the protocol state machine.	 (T)	
Protocol Transition to Self When an object returns to the same state after the specified event occurs.	 (E)	

Copyright © 1998-2015 No Magic, Inc.

Activity Diagram

An activity graph is a variation of a state machine. In the state machine, the states represent the performance of actions or subactivities, while the transitions are triggered by the completion of the actions or subactivities. It represents a state machine of a procedure itself. The entire activity diagram is attached (through the model) to a class, such as a use case, or to a package, or to the implementation of an operation. The purpose of this diagram is to focus on flows driven by the internal processing (as opposed to external events). You should use the activity diagrams in situations where all or most of the events represent the completion of internally-generated actions (that is, procedural flow of control). You should use the ordinary state diagrams in situations where asynchronous events occur. An activity diagram is a variant of a state diagram. Organized according to actions, the activity diagrams are mainly targeted towards the representation of the internal behavior of a method (the implementation of an operation) or a use case.

Though activity diagrams are often classified alongside the interaction diagrams, they actually focus on the work performed by a system instead of an object interaction. An activity diagram captures actions and displays their results.

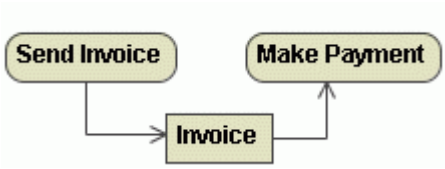
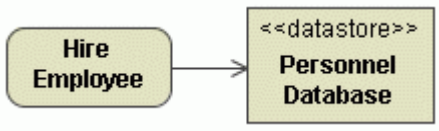
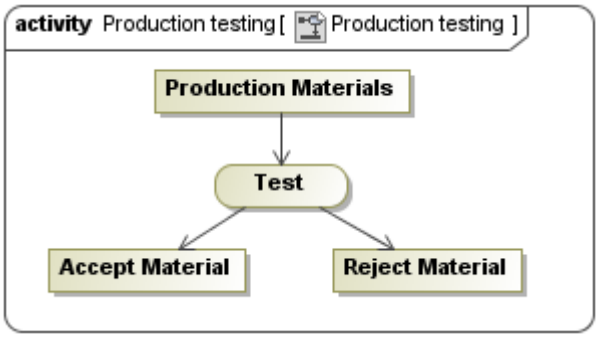
A state diagram may also represent this sequencing of steps. However, given the procedural nature of the implementation of the operations – in which most events simply correspond to the end of the preceding activity – it is not necessary to distinguish states, activities, and events systematically (i.e. state changes and external events have less importance inside the method). It is therefore beneficial to have a simplified representation for directly displaying activities.

The activity diagram provides a convenient way to describe complex algorithms, parallel operations, and business processes. Together with the collaboration and sequence diagrams, they are used to relate use cases.

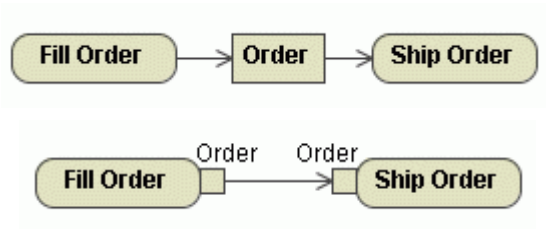
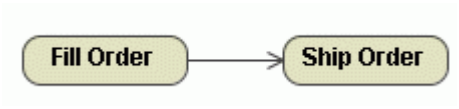
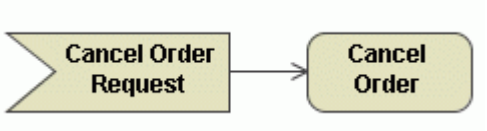
Activity Diagram elements

Model element	Button (hot key)	Notation
Action An action is a named element that is the fundamental unit of an executable functionality. The execution of an action represents some transformations or processing in the modeled system. When the action is to be executed or what its actual inputs are is determined by the concrete action and the behaviors in which it is used.	 (B)	
Call Operation Action An action that transmits an operation call request to the target object, where it may cause the invocation of the associated behavior. The argument values of the action are available to the execution of the invoked behavior.	 (O)	

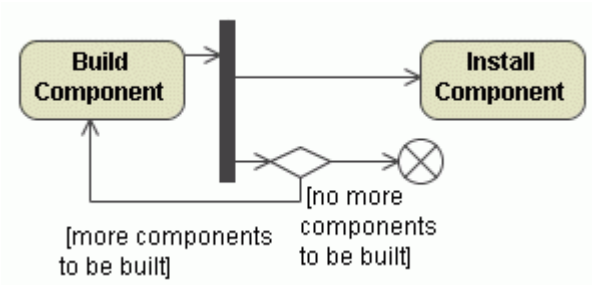
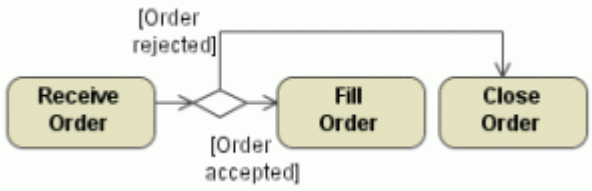
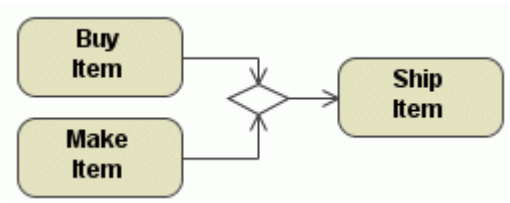
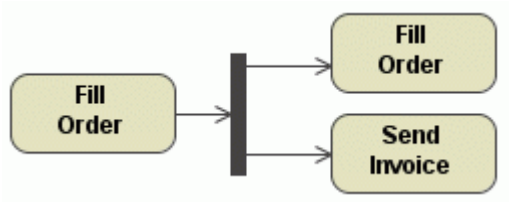
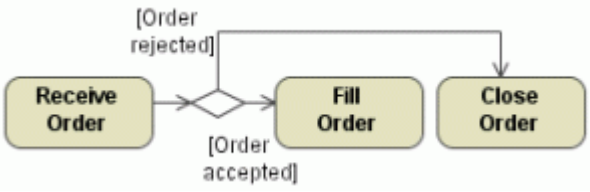
Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Opaque Action An opaque action is introduced for implementing specific actions or for use as a temporary placeholder before some other actions are chosen.		
Any Action This element is introduced in order to maintain any other desirable action element with an appropriate metaclass stereotype applied.		
Object Node The Activity nodes are introduced to provide a general class for the nodes connected by activity edges. The ActivityNode replaces the use of the StateVertex and its children for activity modeling in UML.	 (SHIFT+B)	
Data Store A data store node is a central buffer node for a non-transient information. A data store keeps all tokens that enter it, copies them when they are chosen to move downstream. Incoming tokens containing a particular object replace any tokens in the object node containing that object.	 (SHIFT+D)	
Activity Parparameter Node It is an object node for inputs and outputs to the activities. The Activity parameters are object nodes at the beginning and end of the flows, to accept inputs to an activity and provide outputs from it.		
Input Expansion Node An expansion node is an object node used to indicate a flow across the boundary of an expansion region. A flow into a region contains a collection that is broken into its individual elements inside the region, which is executed once per element.		

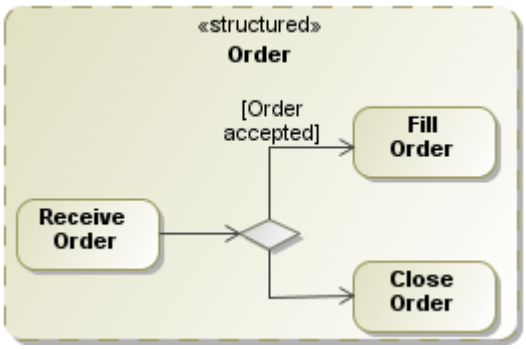
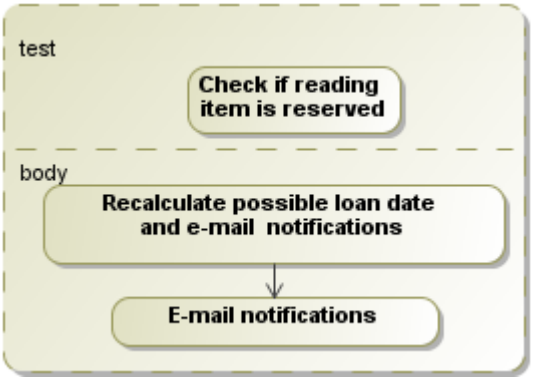
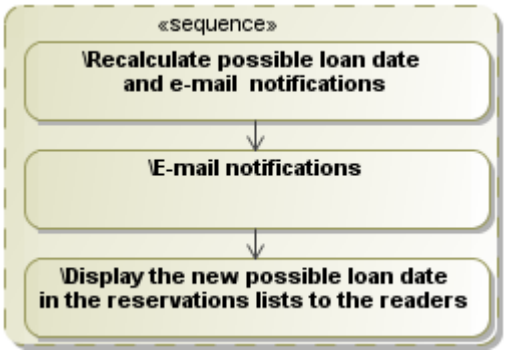
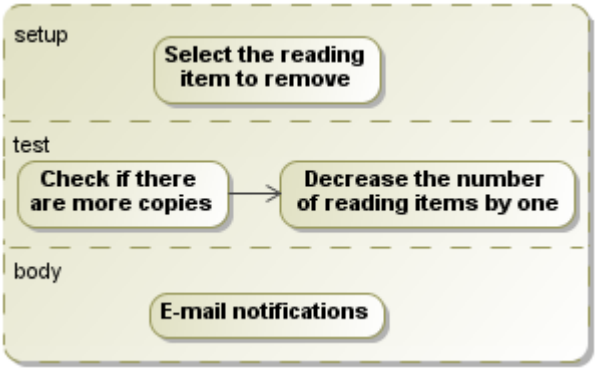
Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Output Expansion Node A flow out of a region combines individual elements into a collection for use outside the region.		
Object Flow Is an activity edge that can have objects or data passing along it. An object flow models the flow of values to or from the object nodes.	 (SHIFT+F)	
Control Flow Is an edge that starts an activity node after the previous one is finished. Objects and data cannot pass along the control flow edge.	 (F)	
Send Signal Action Is an action that creates a signal instance from its inputs, and transmits it to the target object, where it may trigger the state machine transition or the execution of an activity.	 (SHIFT+S)	
Accept Event Action Is an action that waits for the occurrence of an event that meets the specified conditions. The Accept event actions handle event occurrences detected by the object owning the behavior.	 (E)	
Initial Node An initial node is a starting point for executing an activity. It has no incoming edges.	 (T)	
Activity Final An activity final node is a final node that stops all flows in an activity.	 (D)	

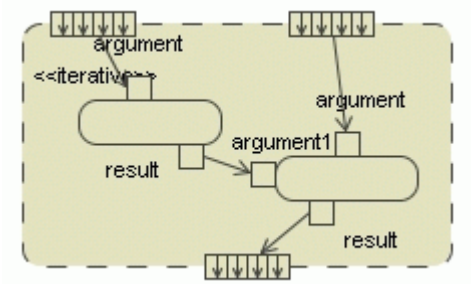
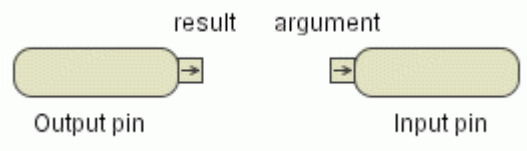
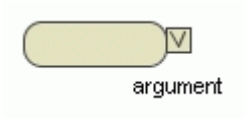
Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Flow Final The Final node that terminates a flow and destroys all tokens that arrive at it. It has no impact on other flows in the activity.	 (L)	
Decision Decision is a control node that chooses between outgoing flows. A decision node has one incoming edge and multiple outgoing activity edges.	 (G)	
Merge A merge node is a control node that brings together multiple alternate flows. It is not used to synchronize concurrent flows but it is used to accept one among several alternate flows.	 (G)	
Fork/Join Horizontal Helps to control parallel actions.	 (K)	
Fork/Join Vertical Helps to control parallel actions.	 (SHIFT+K)	
Exception Handler An exception handler is an element that specifies a body to execute in case the specified exception occurs during the execution of the protected node.	 (P)	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Structured Activity Node A structured activity node is an executable activity node that may have an expansion into the subordinate nodes. The structured activity node represents a structured portion of the activity that is not shared with any other structured node, except for nesting.		
Conditional Node A conditional node is a structured activity node that represents an exclusive choice among alternatives.		
Sequence Node A sequence node is a structured activity node that executes its actions in order.		
Loop Node A loop node is a structured activity node that represents a loop with the setup, test, and body sections.		

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Expansion Region An expansion region is a structured activity region that executes multiple times corresponding to the elements of an input collection.		
Time Event A time event specifies a point of time by an expression. The expression might be absolute or might be relative to some other points of time.		
Input Pin An input pin is a pin that holds input values to be consumed by an action. They are object nodes that receive values from other actions through object flows.	 (SHIFT+I)	
Output Pin A pin that holds output values produced by an action. Output pins are object nodes that deliver values to other actions through object flows.	 (SHIFT+O)	
Value Pin A value pin is an input pin that provides a value to an action that does not come from an incoming object flow edge.		
Swimlanes Swimlanes are used to organize responsibility for actions and subactivities according to the class, dividing an activity diagram.	 (SHIFT+V)  (SHIFT+W)	

Copyright © 1998-2015 No Magic, Inc.



- You can create a new activity diagram under the following elements: Action, Structured Activity Node, Expansion Region, Conditional Node, Loop Node, Sequence Node.
- For more information on dragging elements to the activity diagram, see "Dragging in Activity diagram" on page 252.

Creating Behavior Automatically in Activity Diagram

To turn on the automatic behavior creation in the activity diagram

- In the diagram pallet, click the  *Automatic Behavior Creation* button.

When the *Automatic Behavior Creation* mode is turned on:

- On a call behavior action creation a behavior is created. The behavior's type is the same as the owner's of the call behavior action.
- If a call behavior action and its behavior are without name, type a name on the call behavior action shape on the diagram and the typed name will be set as behavior's name automatically.



For more information about the name display mode on call behavior actions, see "To change the name display mode on the call behavior action" on page 829.

Copyright © 1998-2015 No Magic, Inc.

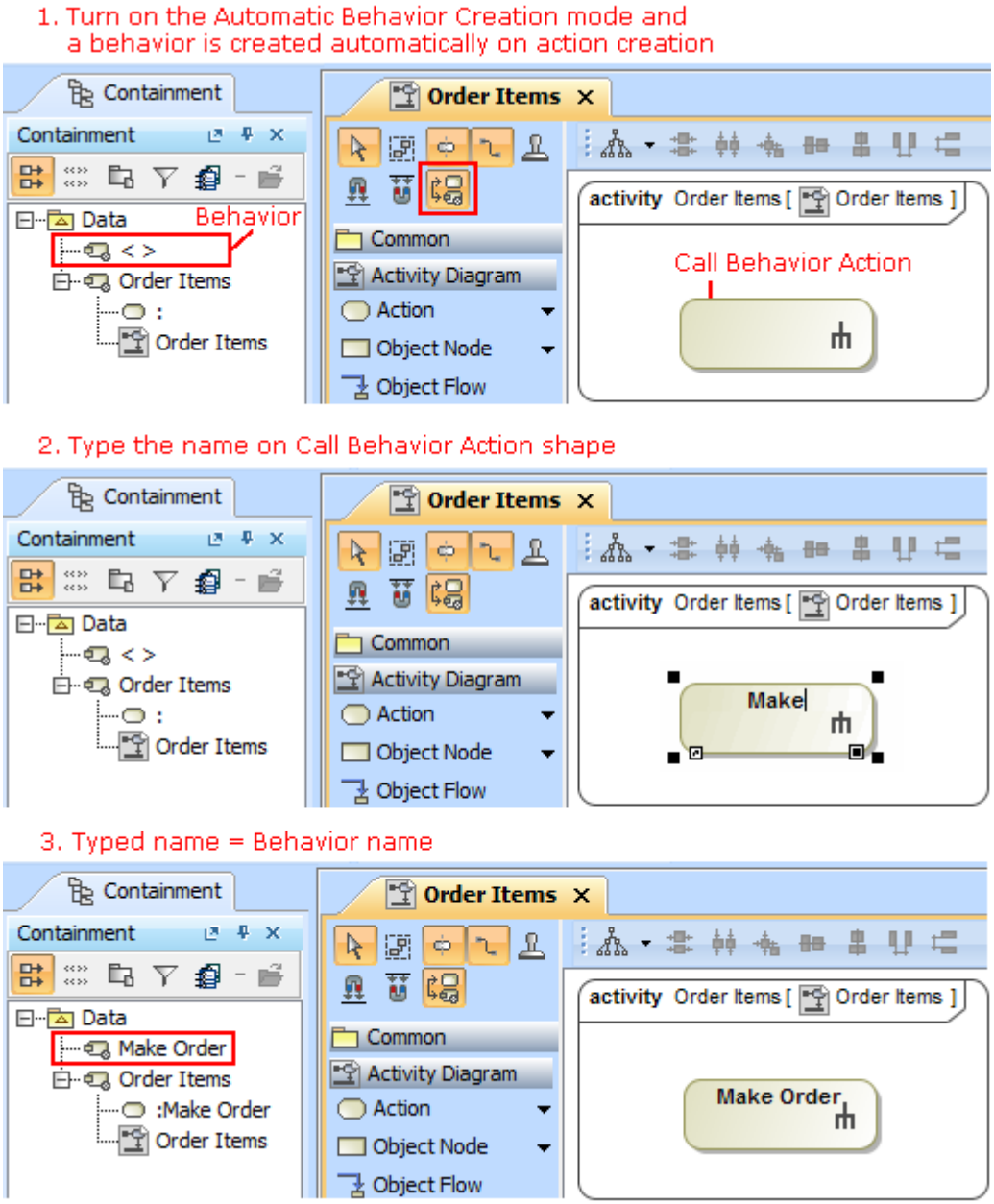


Figure 439 -- Example of creating call behavior action and typing name on it, when Automatic Behavior Creation mode is on

Related diagrams

Activity Diagram

Displaying Inner Elements on Activity Diagrams

NEW! To display inner elements on an Activity diagram

Do one of the following:

- Right-click an Activity in the Containment tree, point to **Create Diagram**, and then click **Activity Diagram**. Note that inner elements are displayed on the Activity diagram upon the first diagram creation under the Activity.

Copyright © 1998-2015 No Magic, Inc.

- Right-click the empty area on the Activity diagram, point to **Related Elements**, and then click **Display Inner Elements**.



To display inner elements on the Structured Activity Node symbol, right-click the Structured Activity Node on the diagram, point to **Related Elements**, and then click **Display Inner Elements**.

Smart Activity Diagram Layout

Dynamic centerlines

The centerlines are displayed only when a center of the shape that was moved or newly drawn is located near the center of another shape that already exists in the diagram. These lines help to draw diagram with aligned shapes easily.

When the center of the shape that was moved coincides with a center of any shape that is placed to its right or left, a horizontal centerline is displayed. When the center of the shape is close to any center of a shape that is located above or below it, a vertical centerline is displayed.

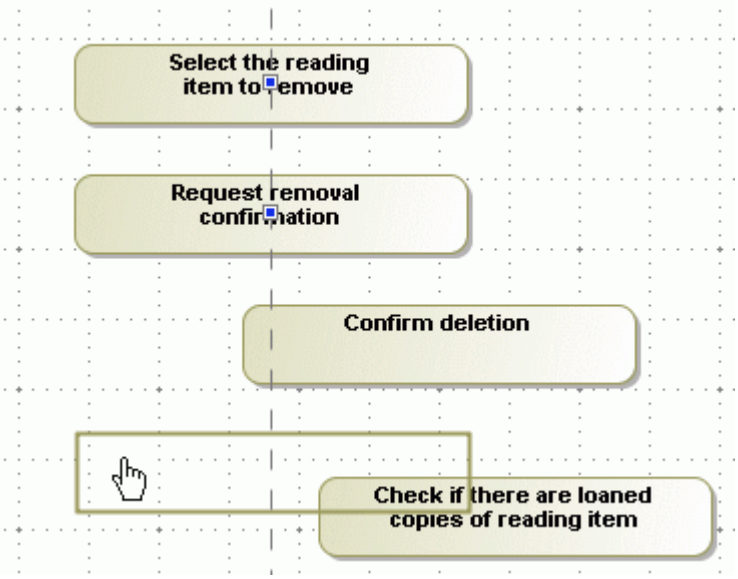


Figure 440 -- Dynamic vertical centerline

To switch off the dynamic centerlines

- Click the **Show Centerlines** button in the diagram toolbar or press **C**.
- From the **Options** main menu, select **Environment**. In the open dialog box, click the **Diagram** node and clear the **Show centerlines in flow diagrams** check box in the **Display** properties group.

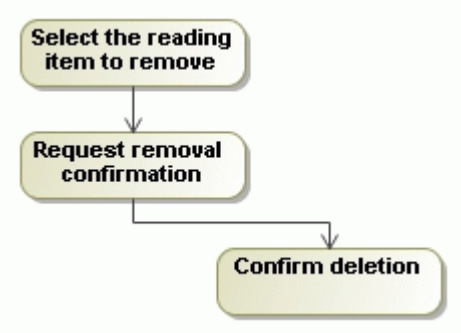
Diagram orientation

The diagram orientation is used to assign the correct rectilinear path breaks and draw paths between the activity diagram shapes. The paths can be drawn from side to side, or from the lower to the upper shape borders.

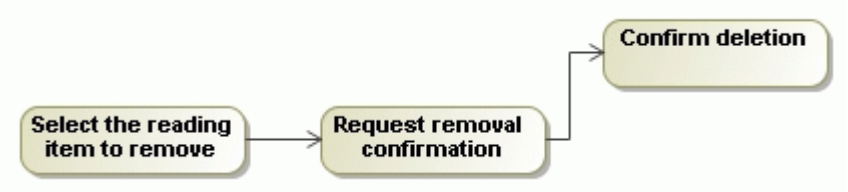
Copyright © 1998-2015 No Magic, Inc.

Example:

For a vertical diagram orientation - in this case if two shapes are not in the same centerline, the paths will be connected from the lower border of the first shape to the upper border of the next shape, adding break points:



For a horizontal diagram orientation - in this case the paths will be connected from the side border of the first shape to the next side border of the second shape, adding break points:



To change the diagram orientation

- From the **Options** main menu, select **Project**. In the open dialog box, expand **Diagram** group and in the right side properties list, change the value for the **Diagram Orientation** property.
- From the diagram pane shortcut menu, select **Diagram Properties** and change the value for the **Diagram Orientation** property.

Deleting from Activity diagrams

- Be advised that after deleting the last symbol of an element from the Activity diagram, the element will be automatically removed from the model. This does not affect Pins and Activity Parameter Nodes.
- If a call behavior action that has an assigned behavior is deleted from the Activity diagram, then this behavior is deleted from the model too.



This is valid if the assigned behavior is editable and does not have more usages (there is no more call behavior action that has reference to this behavior).

Component Diagram

A component diagram describes logical components that make up the system.

A component contains information about the logical class or classes that it implements, thus creating a mapping from a logical view to a component view. Dependencies between the components makes it easy to analyze how a change in one component affects the others. The components may also be shown with any of

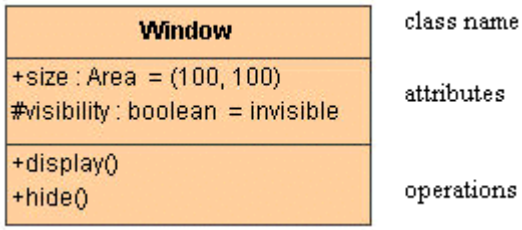
Copyright © 1998-2015 No Magic, Inc.

the interfaces that they expose. They, as with almost any other model elements, can be grouped into packages, much like classes or use cases.

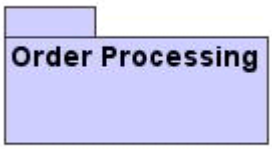
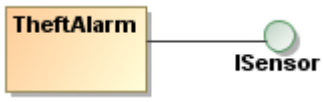
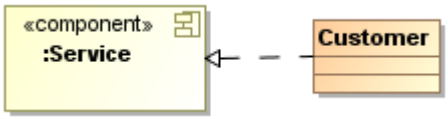
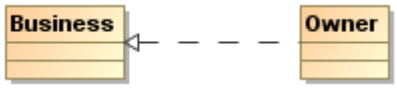
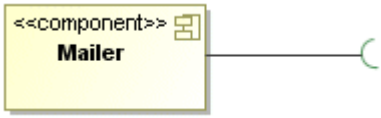
The component diagrams are used in the later phases of the software development, when there is a need to divide up classes among different components. When working with the CASE facilities, the components are used for file-class mapping during code generation, reverse engineering, and round-trip engineering operations.

Copyright © 1998-2015 No Magic, Inc.

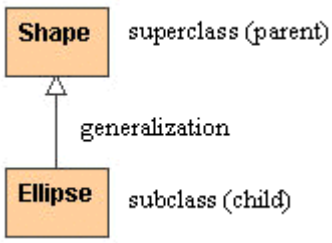
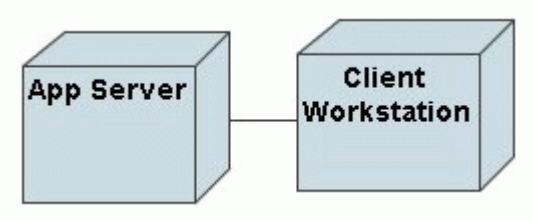
Component Diagram elements

Model element	Button (hot key)	Notation
Class A descriptor for a set of objects with similar structures, behaviors, and relationships.	 (C)	
Component The Component may be used to define the requirements for each physical software element.	 (K)	
Component Instance An instance of a component.	 (A)	
Port Ports represent interaction points between a classifier and its environment. A port has the ability to specify that any requests arriving at this port are handled.	 (SHIFT+R)	
Artifact An artifact represents a physical piece of information that is used or produced by a software development process. Examples of Artifacts include models, source files, scripts, and binary executable files. An Artifact may constitute the implementation of a deployable component.	 (B)	
Deployment Specification It indicates a set of properties that defines how a component should be deployed.		
Artifact Instance		
Deployment Specification Instance An instance of a deployment specification element.		

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Package A group of classes and other model elements.	 (P)	
Interface All functionalities implemented by a particular component.	 (F)	
Interface Realization The classifier at the tail of the arrow implements the interface that is located at the arrow head or uses that interface. **Note:** Choose a different Interface Realization direction from the toolbar to draw a line with an opposite arrow end.	 (R)	
Component Realization A component realization concept is specialized in the Components package to (optionally) define the Classifiers that realize the contract offered by a component in terms of its provided and required interfaces.		
Realization A realization signifies that the client set of elements are an implementation of the supplier set, which serves as the specification.	 (E)	
Usage A usage is a relationship in which one element requires another element (or set of elements) for its full implementation or operation. Usage is also used to create required interface. **Note**: Choose a different Usage direction from the toolbar to draw a line with an opposite arrow end.	 (U)	
Manifestation A manifestation is the concrete physical rendering of one or more model elements by an artifact.		

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Generalization	 (G)	
Link A relationship between a specification element and an implementation element.	 (L)	
Communication Path A communication path is an association between two DeploymentTargets, through which they are able to exchange signals and messages.		



Since MagicDraw version 17.0.1, Component and Deployment diagrams are created instead of the Implementation diagram.

From now on, Implementation diagrams created with earlier MagicDraw versions are realized as the following diagrams:

- Deployment diagram, if node shapes were used in the Implementation diagram.
- Component diagram, if node shapes were not used in the Implementation diagram.

Customized diagrams based on the Implementation diagram are based on the Component diagram now.

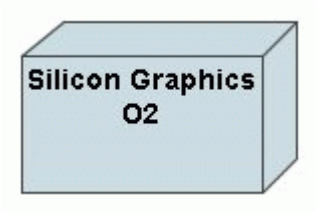
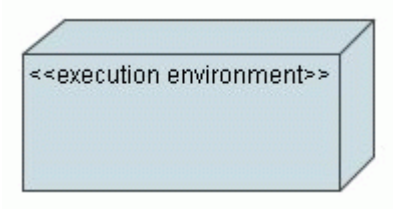
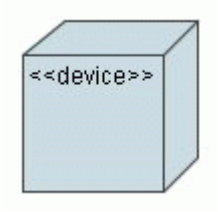
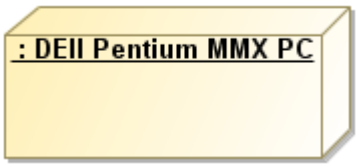
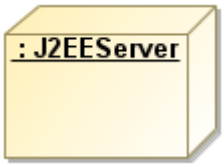
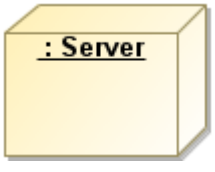
Deployment Diagram

Deployment diagrams show the physical layout of various hardware components (nodes) that compose a system as well as the distribution of executable programs (software components) on this hardware.

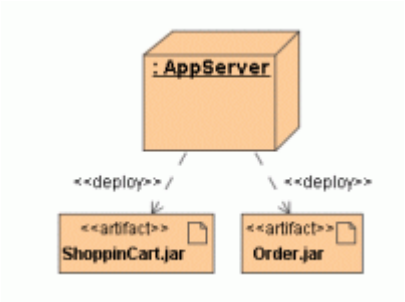
Deployment diagrams are crucial when dealing with distributed systems. You may show the actual computers and devices (nodes), along with the connections they have to each other, thus specifying a system topology. Inside the nodes, executable components and objects are located in a way that it shows where the software units are residing and on which nodes they are executed. You may also show dependencies between components.

Copyright © 1998-2015 No Magic, Inc.

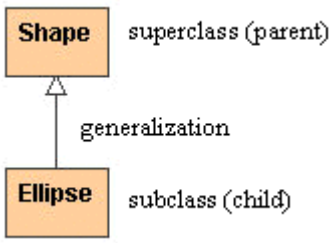
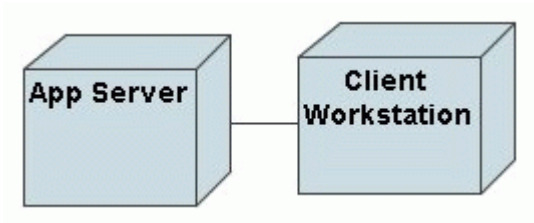
Deployment Diagram elements

Model element	Button (hot key)	Notation
Node A Node is a physical object that represents a processing resource, generally having at least a memory and often the processing capability as well.	 (O)	
Execution Environment The element that is used to indicate that a node is the execution environment.		
Device A physical computational resource with the processing capability upon which artifacts may be deployed for an execution.		
Node Instance An instance of a node.	 (T)	
Execution Environment Instance The element that is used to indicate that a node is an instance of the execution environment.		
Device Instance		

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Artifact An artifact represents a physical piece of information that is used or produced by a software development process. Examples of Artifacts include models, source files, scripts, and binary executable files. An Artifact may constitute the implementation of a deployable component.	 (B)	
Deployment Specification It indicates a set of properties that defines how a component should be deployed.		
Artifact Instance		
Deployment Specification Instance An instance of a deployment specification element.		
Package A group of classes and other model elements.	 (P)	
Deployment A deployment is the allocation of a deployment target to an artifact or artifact instance.		
Manifestation A manifestation is the concrete physical rendering of one or more model elements by an artifact.		

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Generalization	 (G)	
Link A relationship between a specification element and an implementation element.	 (L)	
Communication Path A communication path is an association between two DeploymentTargets, through which they are able to exchange signals and messages.		



Since MagicDraw version 17.0.1, Component and Deployment diagrams are created instead of the Implementation diagram.

From now on, Implementation diagrams created with earlier MagicDraw versions are realized as the following diagrams:

- Deployment diagram, if node shapes were used in the Implementation diagram.
- Component diagram, if node shapes were not used in the Implementation diagram.

Customized diagrams based on the Implementation diagram are based on the Component diagram now.

Package Diagram

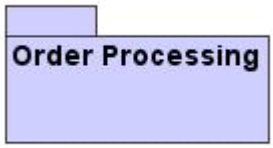
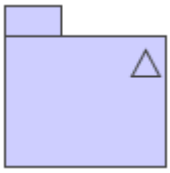
Package diagram shows packages and dependencies between the packages.

The classes can be grouped into packages. The packages can be nested within other packages. A package, as an entity, may have all the relationships that can be drawn for a class. Those relationships are derived from the classes or packages that are nested within two particular packages (i.e., the relationship between packages reflects a set of relationships between classes placed in those packages).

Package Diagram elements

Element	Button (hot key)	Notation

Copyright © 1998-2015 No Magic, Inc.

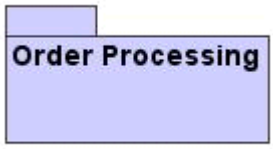
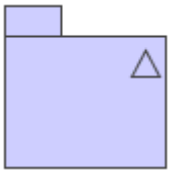
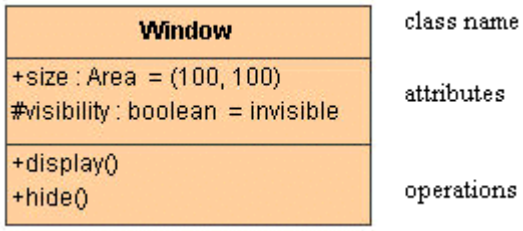
Package A group of classes and other model elements.	 (P)	
Model A model is an abstraction of a physical system from a particular point of view. A model contains a hierarchy of packages/ subsystems and other model elements that describe the system.	 (M)	
Profile A Profile is a kind of Package that extends a reference metamodel.		
Package Merge A package merge is a directed relationship between two packages that indicates that the contents of the two packages are to be combined.		
Package Import A package import is defined as a directed relationship that identifies a package whose members are to be imported by a namespace.		
Element Import An element import is defined as a directed relationship between an importing namespace and a packageable element.		

Profile Diagram

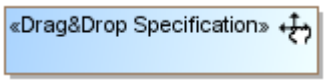
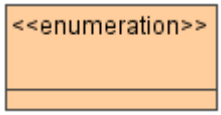
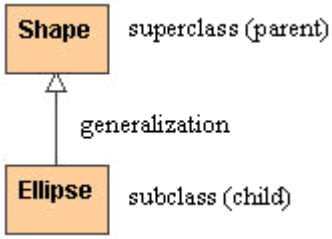
The Profiles package contains mechanisms that allow metaclasses from existing metamodels to be extended to adapt them for different purposes.

Copyright © 1998-2015 No Magic, Inc.

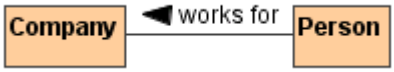
Profile Diagram elements

Model element	Button (hot key)	Notation
Stereotype A stereotype is an extension mechanism that defines a new and more specialized element of the model based on an existing element.	 (SHIFT+S)	
MetaClass A class whose instances are classes. Metaclasses are typically used to construct metamodels.		
Profile A Profile is a kind of Package that extends a reference metamodel.		
Package A group of classes and other model elements.	 (P)	
Model A model is an abstraction of a physical system from a particular point of view. A model contains a hierarchy of packages/ subsystems and other model elements that describe the system.	 (M)	
Class	 (C)	
Customization		

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Drag&Drop Specification For more information, see MagicDraw UMLProfiling&DSL UserGuide.pdf, the “Customizing drag-and-drop” section.		
Data Type		
Primitive Type		
Enumeration A user-defined data type whose instances are a set of user-specified named enumeration literals. The literals have a relative order but no algebra is defined on them.	 (K)	
Profile Application A profile application is used to show which profiles have been applied to a package.		
Extension An extension is used to indicate that the properties of a metaclass are extended through a stereotype, and gives the ability to flexibly add (and later remove) stereotypes to classes.		
Package Merge A package merge is a directed relationship between two packages that indicates that the contents of the two packages are to be combined.		
Package Import A package import is defined as a directed relationship that identifies a package whose members are to be imported by a namespace.		
Generalization	 (G)	

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Association A connection among classes, which also means a connection among objects of those classes.	 (S)	
Direct Association A directed relationship represents a relationship between a collection of source model elements and a collection of target model elements.		

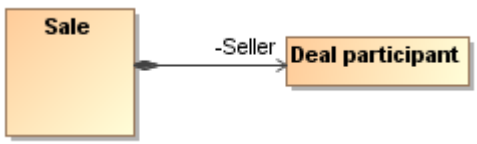


If the Class, Data Type, Primitive Type, Enumeration, Association, Direct Association, Generalization are not available on the diagram pallet, make sure you are working in the Expert mode.

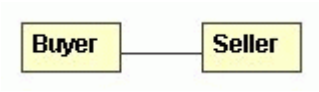
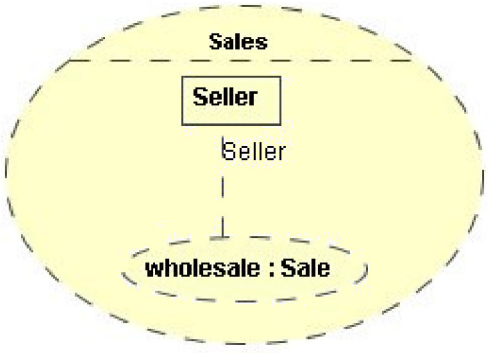
Composite Structure Diagram

The Composite Structure diagram allows a decomposition and modeling of the internal structure of the classifiers, such as classes, components, nodes, and other elements that are able to own other elements.

Composite Structure Diagram elements

Model element	Button (hot key)	Notation
Classifier Is extended with the capability to own collaboration uses. These collaboration uses link a collaboration with the classifier to give a description of the works of the classifier.	 (SHIFT+G)	
Part A part is a classifier (for example, a class, collaboration, component) property owned by a composition.	 (P)	 

Copyright © 1998-2015 No Magic, Inc.

Model element	Button (hot key)	Notation
Port A port may appear either on a contained part representing a port on that part, or on the boundary of the class diagram, representing a port on the represented classifier itself.	 (SHIFT+R)	
Collaboration A collaboration describes a structure of collaborating elements (roles), each performing a specialized function, which collectively accomplishes some desired functionalities. Its primary purpose is to explain how a system works and, therefore, it typically only incorporates those aspects of reality that are deemed relevant to the explanation.	 (Q)	
Collaboration Use A collaboration use represents one particular use of a collaboration to explain the relationships between the properties of a classifier. The collaboration use shows how the pattern described by a collaboration is applied in a given context, by binding specific entities from that context to the roles of the collaboration.	 (U)	
Connector Each connector may be attached to two or more connectable elements, each representing a set of instances. Each connector end is distinct in the sense that it plays a distinct role in the communication realized over the connector.	 (C)	
Role Binding Is a relationship from parts to the Collaboration Use (in diagram).	 (B)	

Copyright © 1998-2015 No Magic, Inc.

Working with Parts in Composite Structure Diagram

Displaying parts on shapes

You can display the existing Parts on the structured classifier shape.

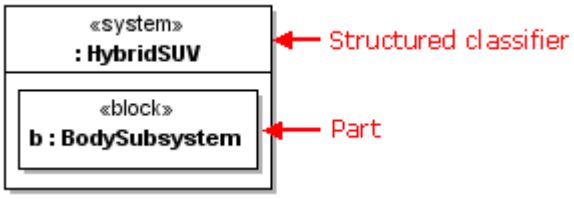
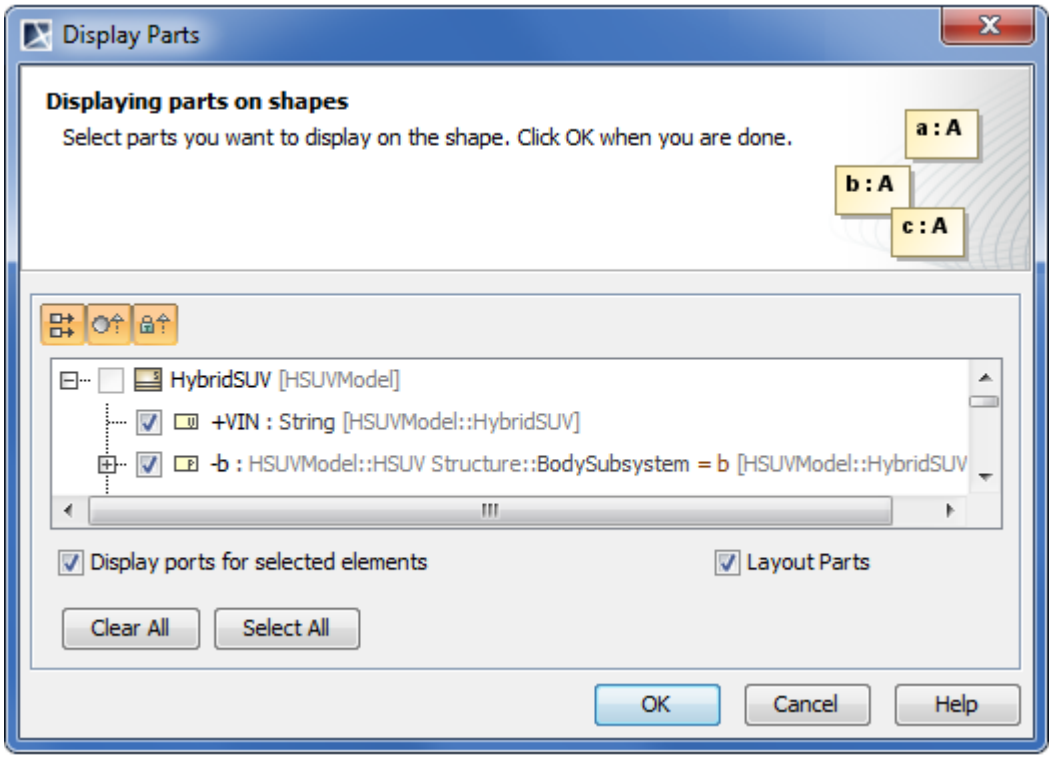


Figure 441 -- Example of part displayed on structured classifier

To display parts on a structured classifier shape

1. On the diagram pane, right-click a shape whereon you want to display parts. The shortcut menu opens.
2. Select **Related Elements** > **Display Parts**. The **Display Parts** dialog opens.



3. In the dialog, select parts you want to display on the shape.

TIP!

- You can select parts either individually or recursively. To select a part recursively, hold down SHIFT while selecting the part.
- Click to clear the **Display ports for selected elements** check box, if you do not want to display ports on parts.

NOTE

NEW! Make sure the **Layout Parts** check box is selected. Otherwise the part shapes will not be arranged in the automatic layout.

4. Click **OK** when you are done.

See the parts displayed on the structured classifier shape.

Copyright © 1998-2015 No Magic, Inc.

Nesting parts that are association ends

In this section you will find the description on how to nest a part to another part in the composite structure diagram. Until MagicDraw version 17.0.2, nesting of parts was not available if a part was a property of an association, that is, if the part represented the association end.

If a classifier has an association connection with another classifier, the association end in the classifier is referenced as a property. In a composite structure diagram, a part is representation of a property. So the part can represent the property that is association end too. If the association end is composite, parts are represented in a nested structure.

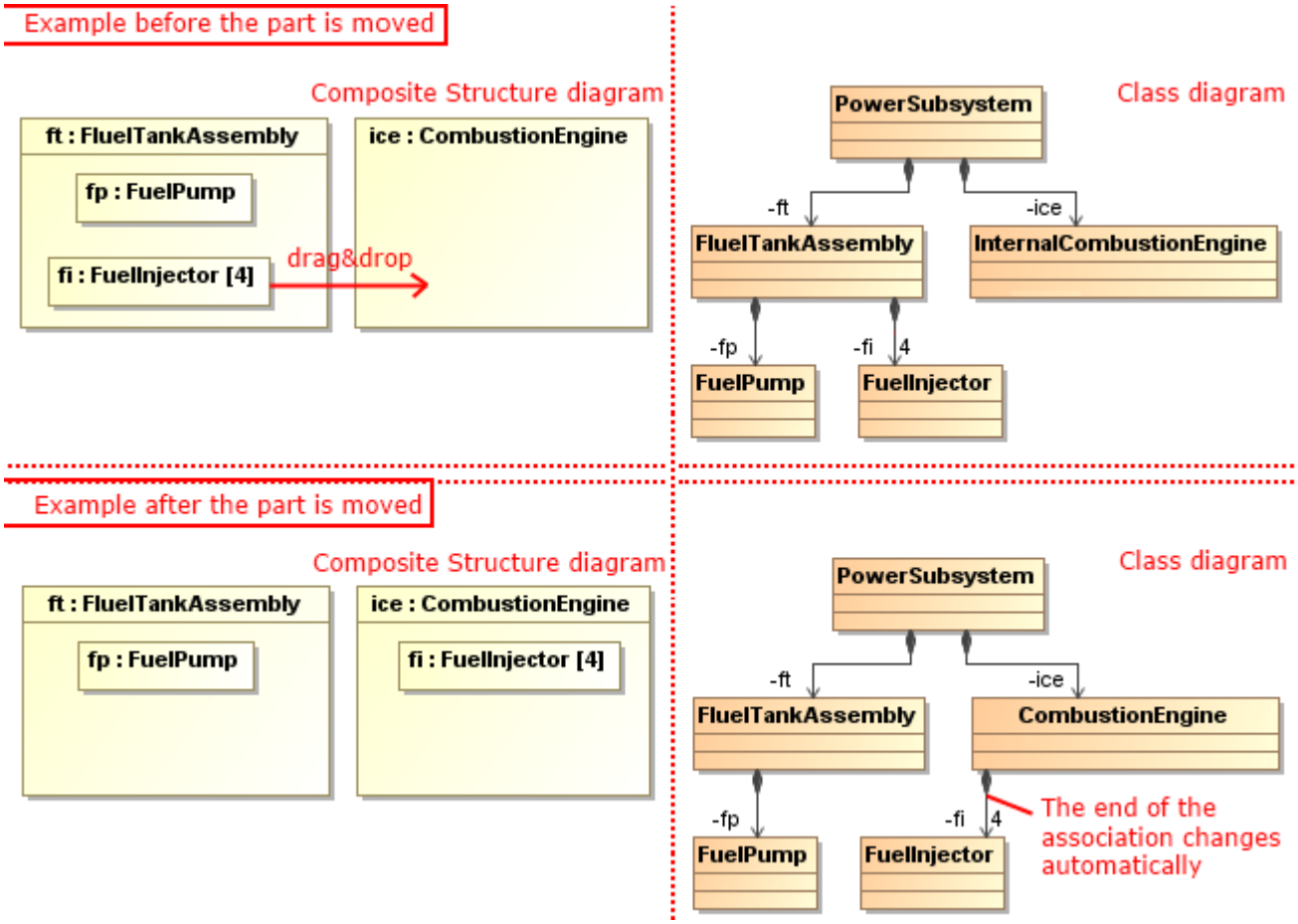


Figure 442 -- Example of part movement in Composite Structure diagram and results representation in Class diagram

See an example in the preceding figure. In the class diagram, the *FuelTankAssembly* class is connected with the *FuelInjector* class with the composite association. In the composite structure diagram, the same association is represented in the following way - the part with the *FuelInjector* type is nested into the part with *FuelTankAssembly* type. By dragging and dropping the part with *FuelInjector* type to the part with the *CombustionEngine* type, will change the association in the class diagram - the *FuelInjector* class will be connected with the *CombustionEngine* class.

Related diagrams

Composite Structure Diagram

Related procedures

Working with Parts in Composite Structure Diagram

Copyright © 1998-2015 No Magic, Inc.

Mapping ports on a part type change

On a part type change, ports of the original part type are automatically replaced with ports of a new type. Connectors are reconnected accordingly too.

Ports are mapped automatically if particular rules are met. Otherwise, the **Port Mapping** dialog appears.

Rules of automatic port mapping

Ports are mapped automatically:

- When the following properties of the port of the original part type are the same as the properties of the port of the new part type:
  - **port type and name**
  - **port type** (note that only one port of the new part type has the same type as port(s) of the original part type)
  - **port name** (valid if the type of the original port is **not** specified)
- Ports of the original part type and port of the new part type has the same interface.
- When the new part type is subtype of the original part type (inherited ports are mapped automatically with ports which they are inherited).
- When the ports of a new type redefines ports of original part type.

The Port Mapping dialog

The **Port Mapping** dialog appears on changing a part type when ports of the type cannot be mapped automatically.

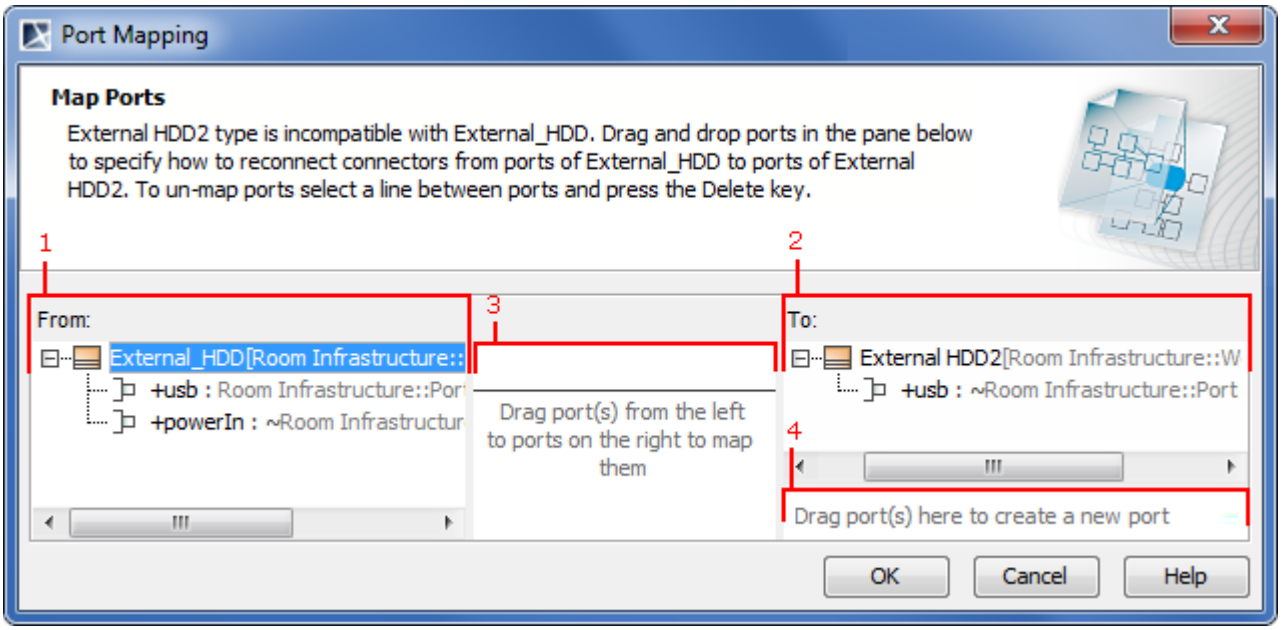
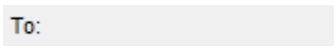
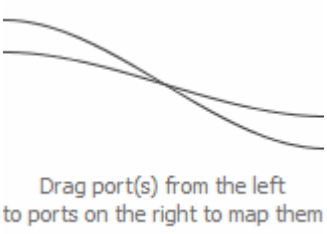
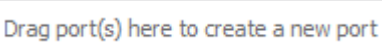


Figure 443 -- Port Mapping dialog

See the elements of the **Port Mapping** dialog described in the following table.

	Element	Function
1	From:	Lists the type of the part from which you want to make a replacement. Ports of the type are listed under the type.

Copyright © 1998-2015 No Magic, Inc.

	Element	Function
2		Lists the type of the part to which you want to make a replacement. Ports of the type are listed under the type.
3		Displays lines between the mapped ports.
4		Creates a new port in the **To** list. In the **From** list, click a port and then drag it to the "Drag port(s) here to create a new port" box. The port for the new type will be created and mapped.

To map ports in the **Port Mapping** dialog

1. Select a port in the **From** box.
2. Drag the selected port onto a port in the **To** box. The ports are mapped and the line between them is displayed.



You can also select multiple number of ports in the **From** box and drag onto a port in the **To** box. The selected ports will be mapped with the port.

To create a new port in the **To** box

1. Select a port in the **From** box.
2. Drag the selected port onto the following box: "*Drag port(s) here to create a new port*". The new port is created in the **To** box and ports are mapped.



You can also select multiple number of ports in the **From** box and drag onto the box: "*Drag port(s) here to create a new port*". The selected number of ports will be created.

To un-map ports in the **Port Mapping** dialog

1. Select the line between the mapped ports.
2. Press the **Delete** key. The line is deleted and ports unmapped.

Displaying Internal Structure on Structured Classifiers

You can represent the internal structure of the structured classifier on the diagram pane. For example, in a composite structure diagram on a part shape, or in a class diagram on a class shape you can simple the owned ports, parts, and connectors.

To display the internal structure of a selected classifier

1. On the diagram pane, select the classifier shape which internal structure you want to display.
2. Open its shortcut menu, click **Related Elements** > **Display Internal Structure**, select the name of the classifier, and then select a diagram containing the internal structure.



In the composite structure diagram, in the part's shortcut menu, the **Related Elements** > **Display Internal Structure** is available only if the part has its own internal structure.

Copyright © 1998-2015 No Magic, Inc.

3. The internal structure is displayed inside the selected classifier.

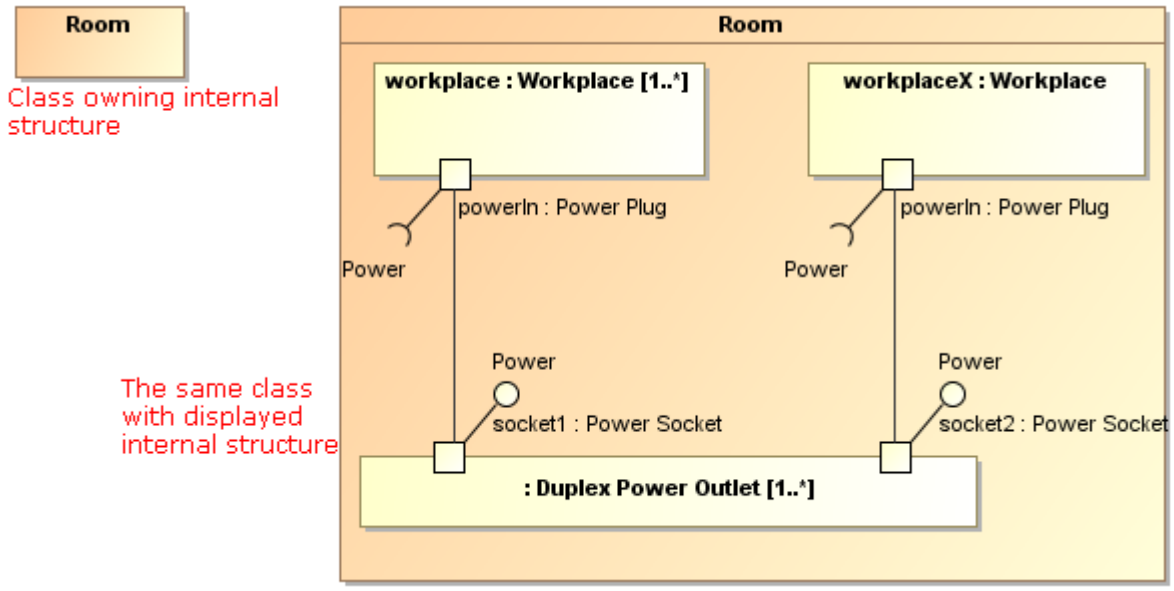


Figure 444 -- Example of class with displayed internal structure

Interaction Overview Diagram

The Interaction Overview diagram focuses on the overview of the flow of control between Interactions. It is based on the Activity diagram notation. Interactions in the Interaction Use diagram are represented using the Interaction Use element. See the sample of the Interaction Overview diagram in Figure 445 on page 724.

Copyright © 1998-2015 No Magic, Inc.

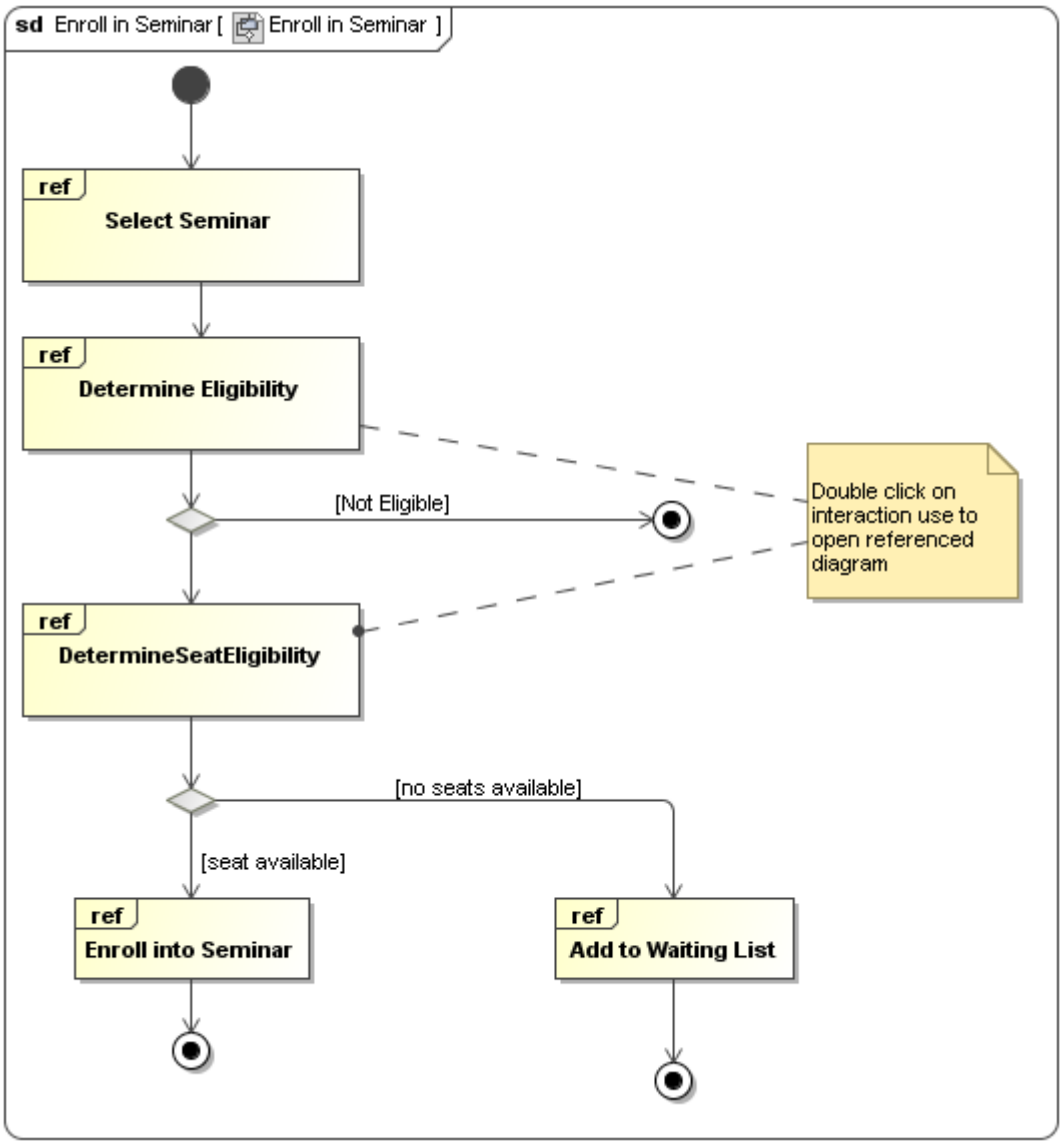


Figure 445 -- Sample of the Interaction Overview diagram

Element	Button (hot key)	Notation
Interaction Use Represents interactions.	(Shift-T)	ref Select Seminar

Other elements of the Interaction Overview diagram are the same as in the Activity diagram. For more information about the Activity diagram, see “Activity Diagram elements” on page 697.

Copyright © 1998-2015 No Magic, Inc.

10 EXTENSION DIAGRAMS

MagicDraw supports these extensions of UML diagrams:

- "Content Diagram" (Standard, Professional, Architect, and Enterprise editions), see page 751.
- "Robustness Diagram" (Standard, Professional, Architect, and Enterprise editions), see page 752.
- "User Interface Modeling Diagram" (Standard, Professional, Architect, and Enterprise editions), see page 727.

Moreover, MagicDraw provides various types of other diagrams:

- "Web Diagram" (Standard, Professional, Architect, and Enterprise editions), see page 753.
- "CORBA IDL Diagram" (Architect and Enterprise editions), see page 756.
- "WSDL Diagram" (Architect and Enterprise editions), see page 758.
- "Time Diagram" (Standard, Professional, Architect, and Enterprise editions), see page 759.
- "Struts Diagram" (Professional, Architect, and Enterprise editions), see page 760.
- "Networking Diagram" (Standard, Professional, Architect, and Enterprise editions), see page 761.
- "Free Form Diagram" (Standard, Professional, Architect, and Enterprise editions).
- "Relation Map Diagram" (Standard, Professional, Architect, and Enterprise editions), see page 764.
- "Dependency Matrix" (Standard, Professional, Architect, and Enterprise editions), see page 764.
- "Generic Table" (Standard, Professional, Architect, and Enterprise editions), see page 785.
- "Instance Table" (Standard, Professional, Architect, and Enterprise editions), see page 803.
- "Glossary", see page 819.
- "Metric Table" (Architect and Enterprise editions), see page 822.

Working with Diagrams



For the general information about working with diagrams see “Diagramming” on page 198.

If there is a need, you can enable or disable a plugin that allows to create the appropriate extension diagram in the **Plugins** options group of the **Environment Options** dialog.



For more information see "Customizing Environment Options" on page 96.

Patterns

Various types of classes can be created in every class diagram using a **Pattern Wizard**. It contains GOF, Java, Junit, CORBA IDL, XML Schema, and WSDL design patterns.

Copyright © 1998-2015 No Magic, Inc.

New patterns and modifications of the existing ones can also be created using Java code or JPython scripts. For the detailed description see MagicDraw OpenAPI UserGuide.pdf.



For a detailed description of the **Pattern Wizard**, see "Pattern Wizard" on page 459.

To open the Pattern Wizard

- In the class diagram, click the **Class by Pattern** button.
- From the class shortcut menu, select **Tools** and then select the **Apply Pattern** subcommand.
- Select a class and select **Apply Pattern** from the **Tools** main menu.

Common Elements

Toolbar Button	Button (Hot key)	
Selection	 (Escape)	
Sticky Button	 (Z)	
Text Box	 (X)	
Text Box (HTML Text)	 (SHIFT+X)	
Note	 (N)	
Note (HTML Text)	 (SHIFT+N)	
Comment		
Comment (HTML Text)		

Copyright © 1998-2015 No Magic, Inc.

Toolbar Button	Button (Hot key)	
Anchor	 (H)	
Constraint	 (SHIFT+H)	
Containment	 (SHIFT+C)	
Dependency	 (SHIFT+D)	
Separator	(W)	
Rectangular Shape	 (SHIFT+Q)	

User Interface Modeling Diagram



User Interface Modeling



Web UI sample project, which represents the UI Modeling Diagram usage for the web based interface modeling is available at <*MagicDraw installation directory*>*samples**diagrams**User* Interface Modeling.

User Interface (UI) Modeling diagram makes it possible to build prototypes of user interfaces, connect UI mock-ups with whole Architectural model, export them as images, and create browsable reports for presentations. In short, they help gather information faster and thus save time and money.

The merits of User Interface Modeling lies in its ability to:

- Create WYSIWYG User Interface prototypes rapidly.
- Integrate User Interface development with UML specifications.
- Get immediate feedback from prospective users on real situations and reuse it for next designs.
- Create browsable reports with the **Report Wizard** (for more information about report generation see MagicDraw ReportWizard UserGuide.pdf).

Copyright © 1998-2015 No Magic, Inc.

Why Prototyping Is So Important?

Various versions of user interfaces need to be tested in order to see how clients respond. This is especially true if you work on key dialogs like for example a sign up screen for a website or an e-commerce application. Working with User Interface prototypes instead of “real” user interfaces will thus enable you to work with small details such as color and the position of a button and substantially reduce designer and programmer overhead in the design phase.

You can build up mock-ups or prototypes, get quick feedback from prospective users, and take and reuse the feedback for future designs. All this is possible with User Interface prototyping. The feedback loop makes for quicker mature designs that work for everybody, which is what really matters.

You can build mock-ups or prototypes to meet following objectives:

- **Integrate GUI development with UML specifications** since this is a new field with increased design capability and since it is easier to see missing parts in UML or User Interface modeling.
- **Help business analysts gather requirements** because of the permanent feedback from prospective users, which makes it easier for them to get all the information needed.
- **Create browsable User Interface prototypes / GUI simulation / Story boarding** since several prototypes can be connected via hyperlinks with one another and then be presented together in a report and simulate the workflow of an application.
- **Resolve flow issues** as it is easier to think through a problem when a user-related interface can be changed quickly.
- **Get "buy-in" from stakeholders** as it can be shown more rapidly how a particular user interface will look like.
- **Run a usability test before full production** so that potential errors in usability like overfilled screens or a usage too complicated can be avoided.
- **Test a series of interaction widgets**. You can think, for example, that another text field or button would be good on a screen. Since modifying a prototype is so easy, it is not a problem to present these suggestions to others.



For more information about prototyping, please visit
http://today.java.net/pub/a/today/2005/08/23/prototyping.html
http://www.scottberkun.com/essays/12-the-art-of-ui-prototyping/

Working with User Interface Modeling Diagrams

To start working with a User Interface Modeling diagram

Do one of the following:

- Click the **User Interface Diagram**  button in the Diagram toolbar.
- Select **Create Diagram** > **Other Diagrams** > **User Interface Modeling Diagram** from the Package or Model shortcut menu in Browser.

After a project is loaded and a diagram is created, UI modeling elements can be added to the diagram pane.

To add new elements to the User Interface Modeling diagram

- Simply drag and drop them out of the Diagrams modeling elements toolbar.



For further information, see "User Interface Modeling" on page 729
and "Case Studies for User Interface Modeling" on page 743.

Copyright © 1998-2015 No Magic, Inc.

To load a sample with an already created User Interface model

1. Open either the **UI Modeling Samples.mdzip** or **UI Modeling in System Development.mdzip** projects which you will find in <MagicDraw installation folder>/samples/diagrams/User Interface Modeling.
2. After loading the sample, select the models from the **Index-Diagram** by clicking the hyperlinks.

User Interface Modeling

This section gives the basic information you need to know about user interface modeling in MagicDraw. It includes the following sections:

- "User Interface Modeling diagram elements" on page 729
- "Modifying a table" on page 735
- "Modifying a tree" on page 738
- "Nesting" on page 739
- "Reusability" on page 739
- "Specifying elements" on page 739
- "Icon usage" on page 741
- "Using symbol properties" on page 742
- "User interface prototyping" on page 743

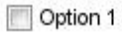
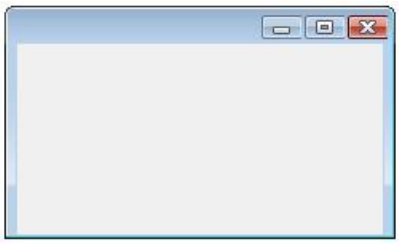
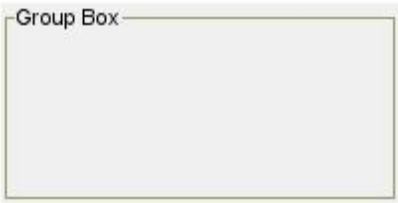
For User Interface Modeling, see "Case Studies for User Interface Modeling" on page 743. Three case studies will provide step-by-step instructions how to build user interfaces and create browsable reports.

User Interface Modeling diagram elements

The following table lists all available User Interface modeling elements and their properties.

Element	Description	Properties	Example
Button	A regular button with the possibility to add text, icon or both.	- **Icon**: select one of the available icons or choose your own. - **Inactive**: activates/ deactivates the button. - **Text**: displays text on the button on the diagram pane. - **Selected**: sets the button as selected or not.	
Toolbar button with Icon	Small sized button with predefined icon (close, copy, cut, delete, new, open, paste, print, redo, save, search, undo, zoom in, zoom out).	- same as for regular buttons.	

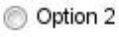
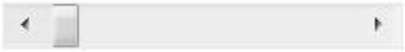
Copyright © 1998-2015 No Magic, Inc.

Element	Description	Properties	Example
Button with Text	Button with predefined text (Back, Cancel, Close, Finish, Next, OK).	- same as for regular buttons.	
Check Box	Check box with text.	- **Inactive**: activates/ deactivates the box. - **Text**: displays text next to the check box on the diagram pane. - **Selected**: marks/ unmarks the check box.	
Combo Box	Regular combo box in non-editable style with possibility to show a value.	- **Inactive**: activates/ deactivates the box. - **Text**: displays text in the combo box.	
Frame*	Main container component represented by a regular Internal Frame. Any other component can be nested inside.	- **Icon**: select one of the available icons or choose your own. - **Inactive**: activates/ deactivates the frame. - **Maximize**: defines if the maximize-icon should be visible or hidden. - **Minimize**: defines if the minimize-icon should be visible or hidden. - **Title**: displays the title of the frame.	
Group Box*	A panel with a titled border. Any other component can be nested inside.	- **Border Type**: defines the style of the border. - **Title**: displays the title of the border. - **Titled**: defines whether border text should be shown or not.	
Hyperlink	A blue colored and underlined item for showing some text or an icon or both.	- **Icon**: select one of the available icons or choose your own. - **Inactive**: activates/ deactivates the link. - **Text**: displays the text of the hyperlink.	

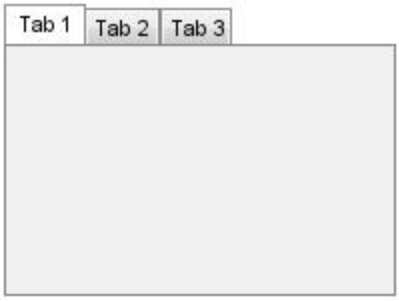
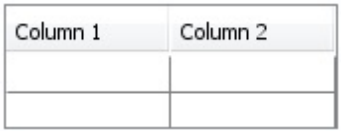
Copyright © 1998-2015 No Magic, Inc.

Element	Description	Properties	Example
Label	Item with ability to present a text, an icon or both.	- **Icon**: select one of available icons or choose your own. - **Inactive**: activates/ deactivates the label. - **Text**: displays the text in the label.	This is a Label
List	Bordered item which stores and shows numerous values. Values can be shown as selected.	- **Horizontal Scroll Bar**: defines the visibility of the horizontal bar. - **Inactive**: activates/ deactivates the list. - **Selected Value**: defines which value should be shown as selected. - **Values**: add/remove values to/from the list. - **Vertical Scroll Bar**: defines the visibility of the vertical bar.	Test Value 1 Test Value 2 Test Value 3
Menu Bar	A bar that can present several text items.	- **Values**: add/remove menus to/from the bar.	File Edit View Tools Help
Panel*	A bordered panel.		
Password Field	A bordered field with a text which is hidden through symbols.	- **Hidden**: decides whether the text should be shown as text or symbolized. - **Inactive**: activates/ deactivates the field. - **Text**: displays the text of the password field.	********

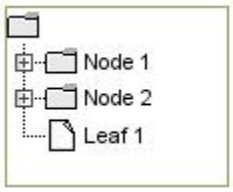
Copyright © 1998-2015 No Magic, Inc.

Element	Description	Properties	Example
Progress Bar	A bar which presents the state of progress.	- **Maximum Value**: sets the maximum value of the bar. - **Minimum Value**: sets the minimum value of the bar. - **Value**: sets the display text on the bar. - **Vertical**: switches between horizontal and vertical orientation.	
Radio Button	A radio button with possibility to present some text.	- **Inactive**: activates/ deactivates the button. - **Text**: displays text next to the radio button. - **Selected**: marks/ unmarks the button.	
Scroll Bar	Item which represents regular scroll bar.	- **Inactive**: activates/ deactivates the bar. - **Vertical**: switches between horizontal and vertical orientation.	
Scroll Pane*	A panel which can contain a vertical and/or a horizontal scroll bar.	- **Horizontal Scroll Bar**: defines the visibility of the horizontal bar. - **Vertical Scroll Bar**: defines the visibility of the vertical bar.	
Separator	A line with ability to split up a component.	- **Vertical**: switches between horizontal and vertical orientation.	

Copyright © 1998-2015 No Magic, Inc.

Element	Description	Properties	Example
Slider	An item for presenting a range of several values. The knob of this item can be moved to these values.	- **Inactive**: activates/ deactivates the slider. - **Invert**: switches maximum and minimum values. - **Knob Position**: defines the location of the knob. - **Maximum Value**: sets the maximum value of the slider. - **Minimum Value**: sets the minimum value of the slider. - **Spacing**: sets the space between two markings. - **Values**: defines at which position the slider should be a text. - **Vertical**: switches between horizontal and vertical orientation.	
Spinner	Regular spinner that can show a value.	- **Inactive**: activates/ deactivates the spinner. - **Text**: displays value on the spinner.	
Tabbed Pane*	Item which represents a panel with tabs.	- **Active Tab**: sets which tab should be shown as selected. - **Tab Position**: selects the placement of the tabs. - **Tabs**: add/remove tabs to/from the pane.	
Table	A table item with the ability to add and/or remove columns and/or rows. See "Modifying a table" on page 735 for more information.	- **Column Header**: defines whether the column header should be visible or not.	

Copyright © 1998-2015 No Magic, Inc.

Element	Description	Properties	Example
Text Area	A multi-line bordered item to present long text.	- **Horizontal Scroll Bar**: defines the visibility of the horizontal bar. - **Inactive**: activates/ deactivates the area. - **Text**: displays text in the text area. - **Vertical Scroll Bar**: defines the visibility of the vertical bar.	
Text Field	A single-line bordered item to present some text.	- **Inactive**: activates/ deactivates the field. - **Text**: displays text in the text field.	
Tool Bar*	A bar which can consist of any other component. Usually used to nest buttons or labels with icons.		
Tree	An item to present a tree structure. It can add numerous nodes and/or leaves to it. For information on how to do this, see "Modifying a tree" on page 738.	- **Expand**: defines whether the whole tree should be expanded or not. - **Horizontal Scroll Bar**: defines the visibility of the horizontal bar. - **Icon**: select one of available icons or choose your own. - **Text**: displays the text of the root node. - **Vertical Scroll Bar**: defines the visibility of the vertical bar.	

NOTE

The elements marked with an asterisk (*) are the **Container** elements and can nest other components.

Copyright © 1998-2015 No Magic, Inc.

Modifying a table

As the following figure shows, a newly created table has two columns and two rows already added to the table. Each row has cells, which number equals to the number of columns.

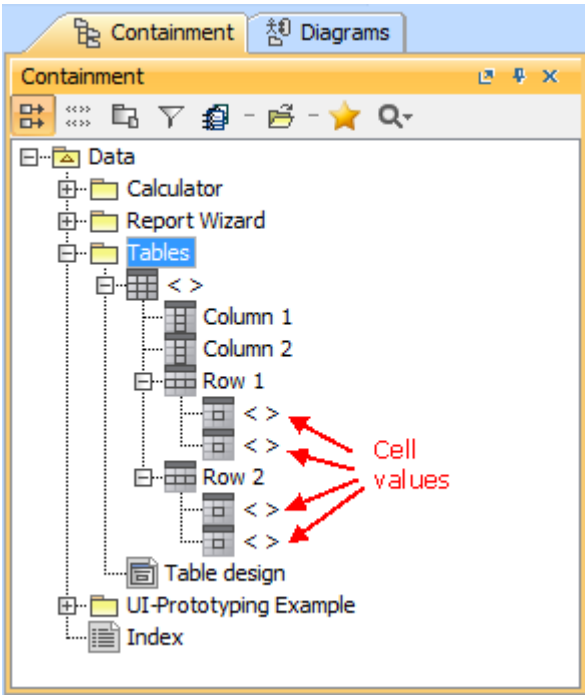


Figure 446 -- Structure of newly created table

To modify the contents of the table, you can do the following:

- Change the titles of the existing columns.
- Add a new column.
- Change the order of the columns.
- Add a new row.
- Change the order of the rows.
- Add cell values.

To add a new column to a table

1. Right-click the table in the Containment tree and from the shortcut menu, select **Create Element** > **Column**. A new column is added to the table.
2. Type the title for the new column.

NOTE

With the new column, a new cell is added to each row of the table.

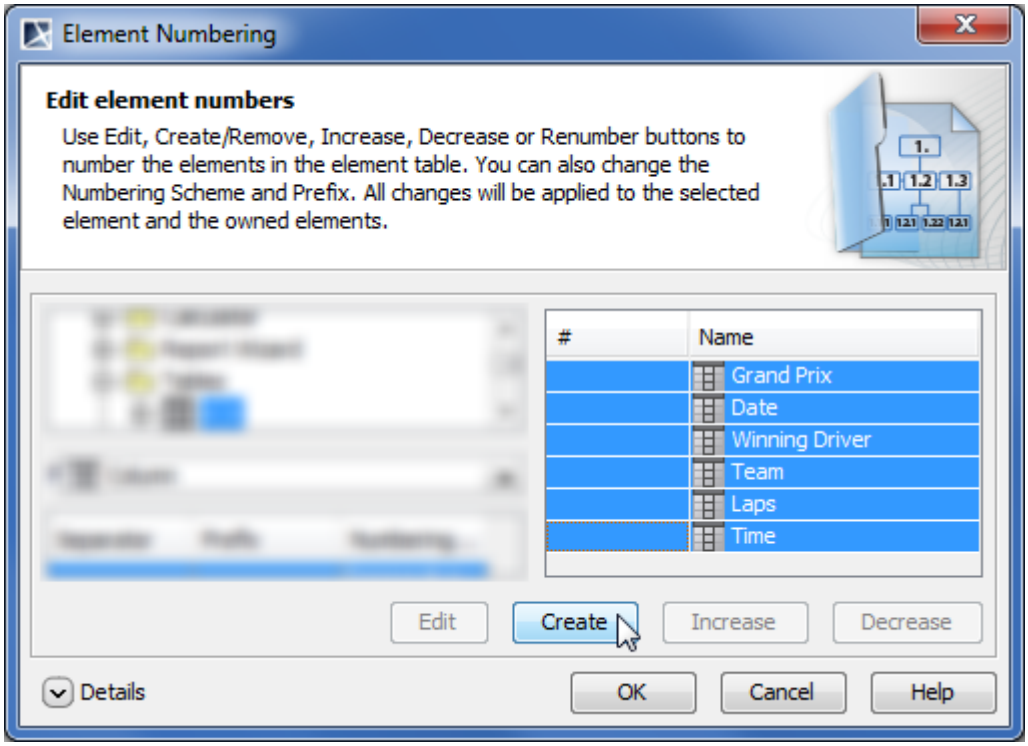
To change the order of the columns in a table

1. Right-click a column of the table in the Containment tree and from the shortcut menu, select **Element Numbering**. The **Element Numbering** dialog opens.
2. In the list on the right side of the dialog, select all columns and click **Create**.

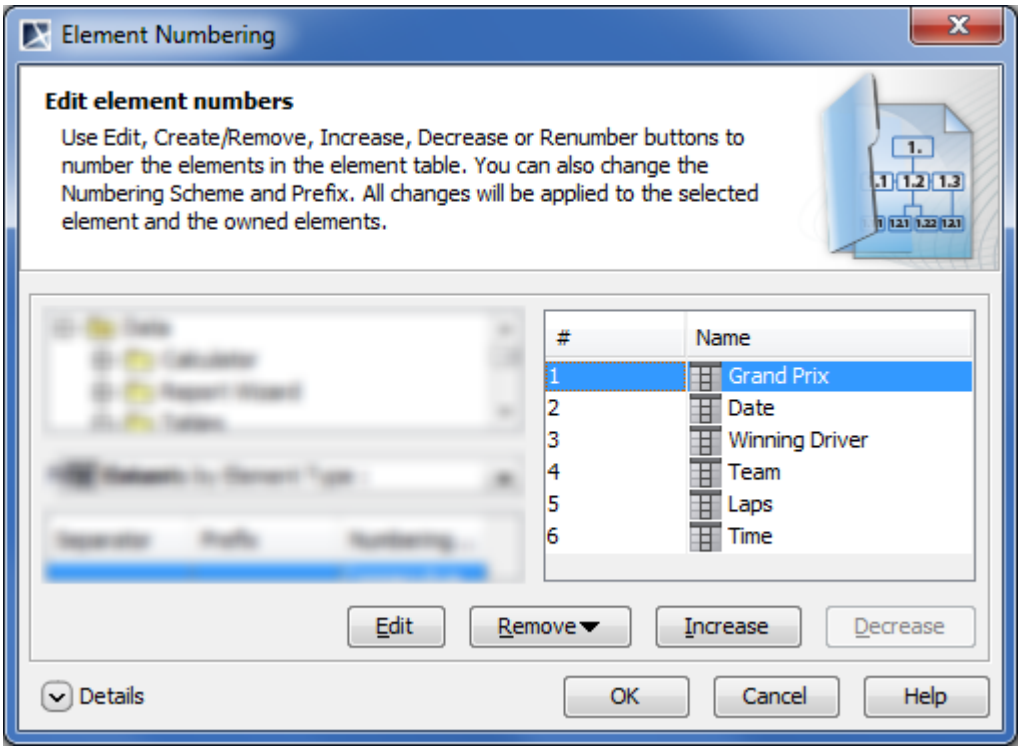
TIP!

To select all columns, click the first column in the list and then hold down the Shift key when selecting the last one.

Copyright © 1998-2015 No Magic, Inc.



All the columns become numbered.



3. Change the order of columns in the table by changing their positions in the list appropriately (see the following figure).
    - To move a column to the left, select the column and click the **Decrease** button.
    - To move a column to the right, select the column and click the **Increase** button.

TIP!

For more information about using the **Element Numbering** dialog, see "Element Numbering dialog" on page 399.

Copyright © 1998-2015 No Magic, Inc.

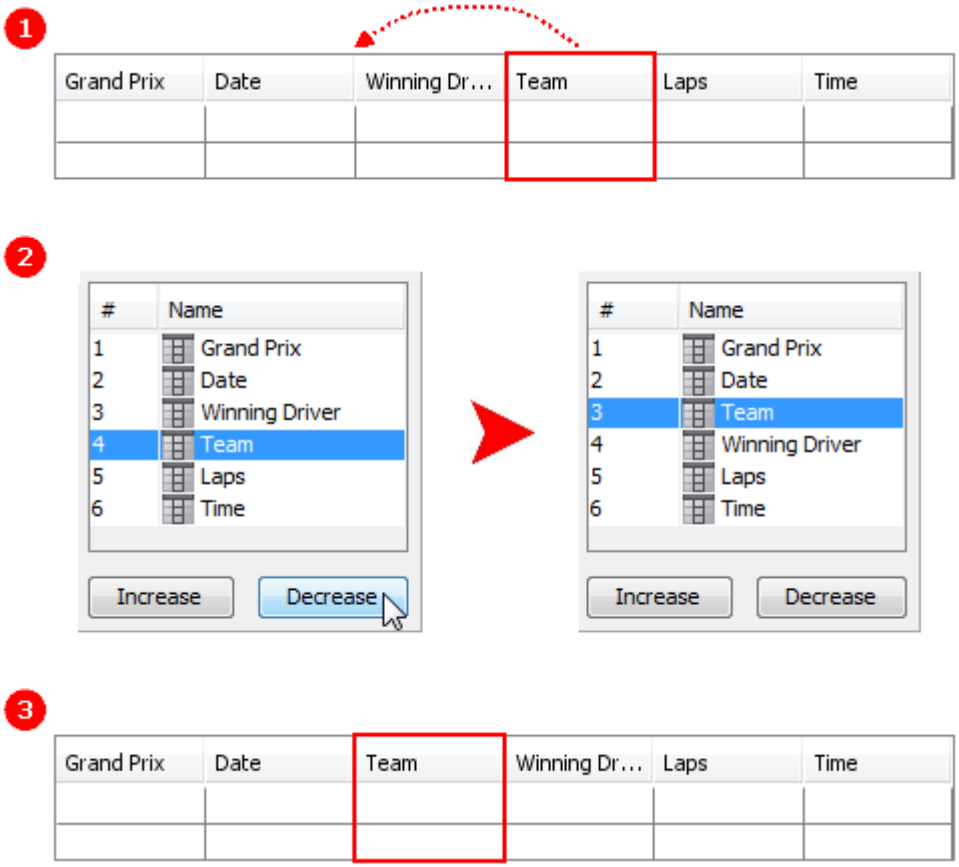


Figure 447 -- Moving column "Team" to left

To add a row to a table

1. Right-click the table in the Containment tree and from the shortcut menu, select **Create Element** > **Row**. A new row is added to the table.
2. Type the title for the new row.

To change the order of the rows in a table

1. Right-click a row of the table in the Containment tree and from the shortcut menu, select **Element Numbering**. The **Element Numbering** dialog opens.
2. In the list on the right side of the dialog, select all rows and click **Create**.



To select all rows, click the first row in the list and then hold down the Shift key when selecting the last one.

All the rows become numbered.

3. Change the order of rows in the table by changing their positions in the list appropriately.
   - To move a row up, select the row and click the **Decrease** button.
   - To move a row down, select the row and click the **Increase** button.



For more information about using the **Element Numbering** dialog, see "Element Numbering dialog" on page 399.

To change a cell value

1. Select the cell of the table row in the Containment tree.

Copyright © 1998-2015 No Magic, Inc.

2. Change the cell value in one of the following ways:
    - Press the F2 key. Type a new value.
    - Click the cell value again and wait while it becomes editable. Type a new value. Press Enter when you are done.
    - Right-click the cell to open the Specification window of the cell. Type a new value in the cell of the **Text** property value.

Modifying a tree

You can add to a tree as many nodes and leaves as you need and also change the order of the elements in the tree.

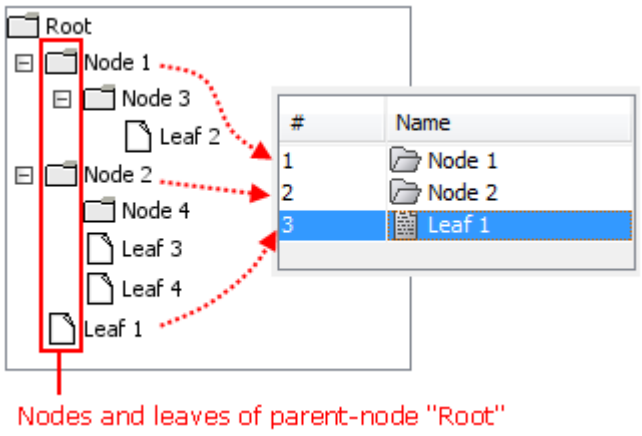
To add a node or a leaf to a tree or to another node

1. Right-click the tree or a node in the Containment tree and from the shortcut menu, select **Create Element** > **Node** or **Leaf**.
2. Type a name for the new node or leaf.

To change the order of the elements in a tree

NOTE

At a time, you can rearrange the order only of the same parent-node leaves and/or nodes.



1. Right-click a leaf or a node of the tree and from the shortcut menu select **Element Numbering**. The **Element Numbering** dialog opens.
2. In the list on the right side of the dialog, select all leaves and/or nodes, and then click **Create**.

TIP!

To select all leaves and/or nodes, click a leaf or a node that is the first one in the list and then hold down the Shift key when selecting a leaf or a node that is the last one in the list.

All the leaves and/or nodes become numbered.

3. Change the order of leaves and/or nodes in the tree by changing their positions in the list appropriately.
    - To move a leaf or a node/leaf up, select the row and click the **Decrease** button.
    - To move a leaf or a node/leaf down, select the row and click the **Increase** button.

TIP!

For more information about using the **Element Numbering** dialog, see "Element Numbering dialog" on page 399.

Copyright © 1998-2015 No Magic, Inc.

Nesting

You can move and arrange user interface modeling elements, since they nest each other, and thus create deep structured User Interface models. However, not all elements have the nesting ability: only **Container** elements can nest other elements (these elements are marked with an asterisk (*) in the section “User Interface Modeling diagram elements” on page 729).

As Figure 448 on page 739 shows, all frames nest in each others and, as a result, if you were to move the element with title *Frame 1*, all other elements would stay in position within this element.

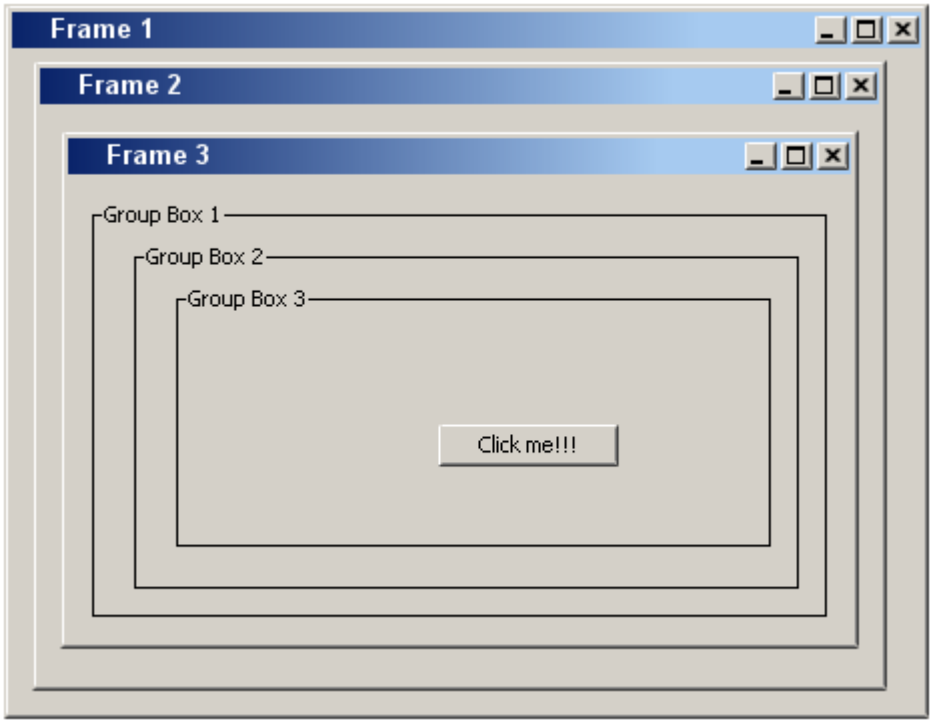


Figure 448 -- Example of a deep structure that was created using containers

Reusability

If you have created a complex model and need to use it again, you do not have to create a new one. All you need to do is select the elements, copy them and then paste them into the same diagram or in any other diagram. You can also reuse just a single element by copying and pasting it or by dragging it from the Browser onto the diagram pane.

Specifying elements

You can modify UI modeling elements by editing their properties in the following three ways:

- Via the **Specification** dialog box.
- Via the shortcut menu.
- On a diagram pane.

To edit properties via the **Specification** dialog box

1. Select an element on the diagram.
2. Right-click on it and select **Specification** on the shortcut menu (Figure 449 on page 740).
3. Edit one of the properties to modify the element (for the effects of editing properties, see section "User Interface Modeling diagram elements" on page 729).

Copyright © 1998-2015 No Magic, Inc.

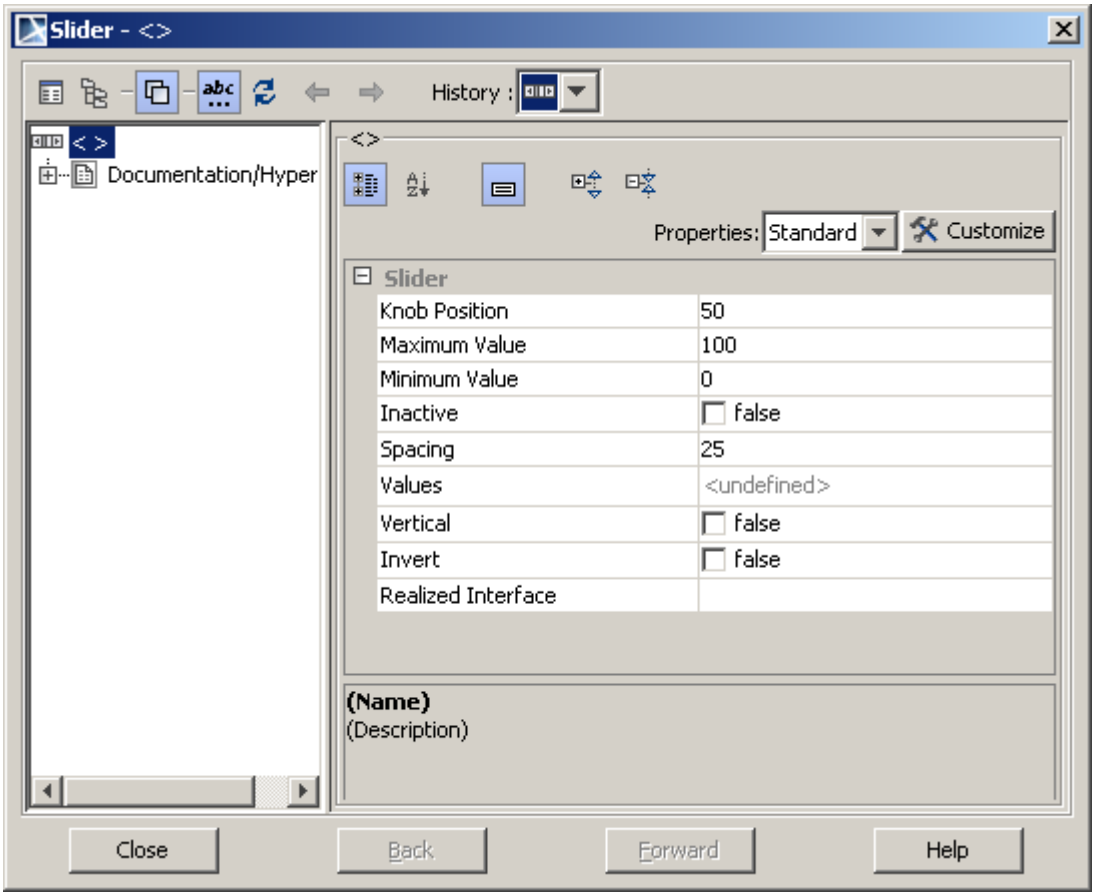


Figure 449 -- Specification dialog box of the Slider element

The so called boolean properties that have only two values can also be edited via the shortcut menu.

To edit properties via the shortcut menu

NOTE This is possible for the boolean properties only.

1. Select an element on the diagram.
2. Right-click on it. The shortcut menu will appear and the properties which can be edited will be displayed at its bottom (Figure 450 on page 741).
3. Select the property you want to change.

Copyright © 1998-2015 No Magic, Inc.

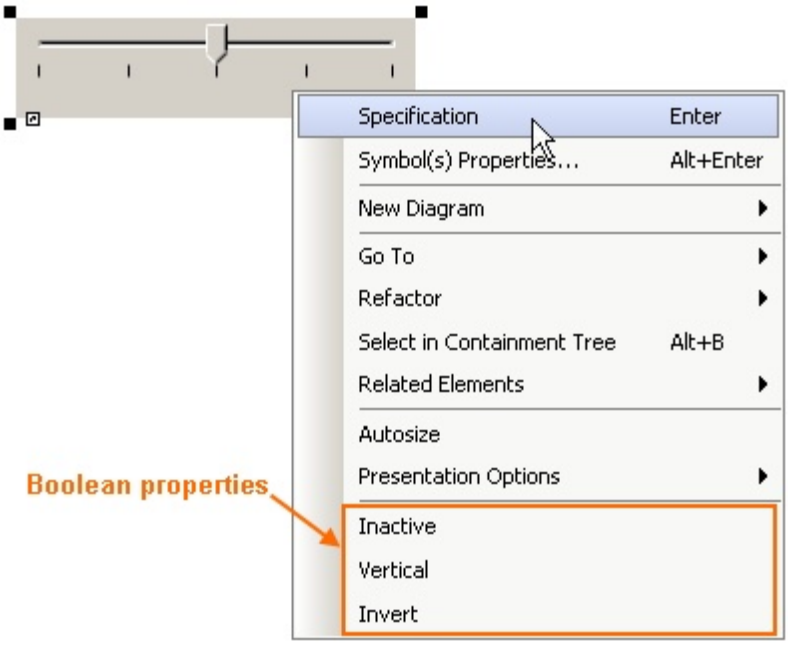


Figure 450 -- Shortcut menu of the Slider element with boolean properties

For the elements that own a text or a title, for example a button or a frame, it is also possible to edit the appropriate properties straight on a diagram.

To edit the element properties on a diagram

NOTE This is possible for the text or title properties only.

1. Select an element on the diagram.
2. Click the selected element.
3. Type a text on the element (see the following figure).
4. Press ENTER. The text you have typed will appear on the element and in the **Text/Title** property of the element as well.

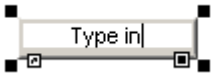


Figure 451 -- Editing title of element

Icon usage

Some elements are capable of owning an icon. User Interface Modeling provides a number of frequently-used icons such as cut, delete, undo/redo, etc. Of course, it is also possible to use any images from your computer as icons.

To add an icon

1. Select an element which can own an icon, for example a button.

   NOTE To find out which elements can have an icon, see "User Interface Modeling diagram elements" on page 729.

2. Right click on the element and select **Specification** in the shortcut menu.

Copyright © 1998-2015 No Magic, Inc.

3. In the **Icon** list, select an icon you want to appear on the element (Figure 452 on page 742).

NOTE

If you want to use any image from your computer as an icon, click **custom**.

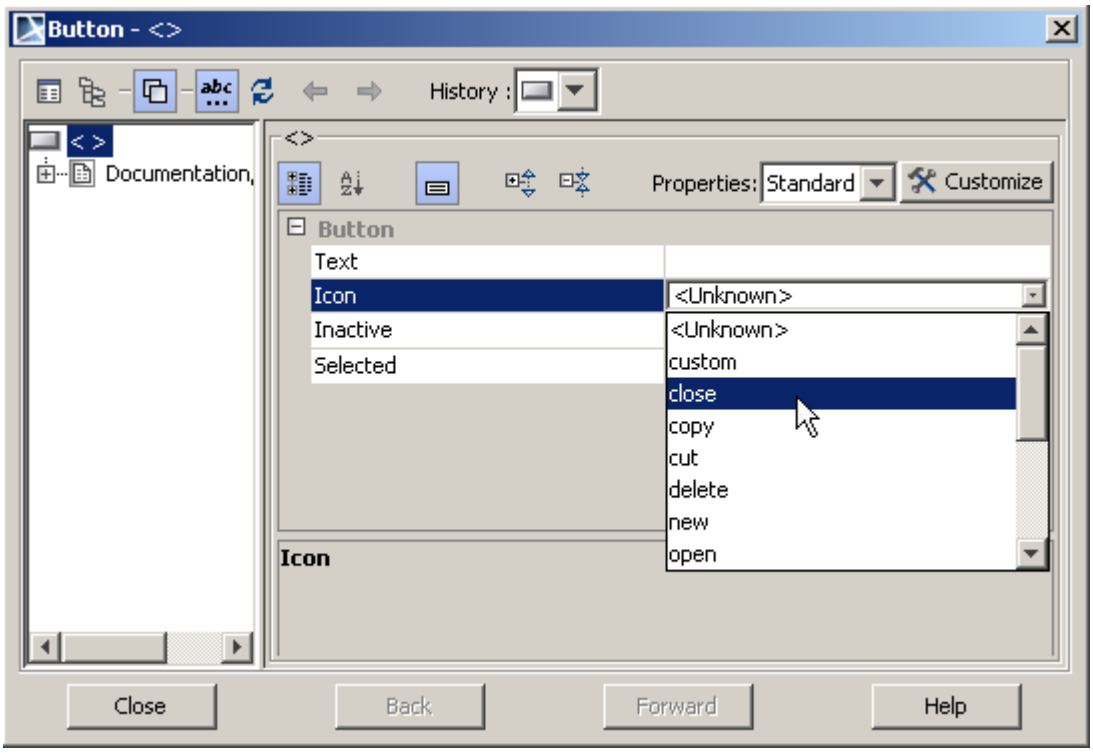


Figure 452 -- Setting an icon for a button

Using symbol properties

User Interface Modeling also allows you to modify user interface elements via symbol properties.

To edit an element via symbol properties:

1. Select an element on the diagram.
2. Right-click and select **Symbol Properties** on the shortcut menu.
3. Change one of the properties listed in the table below.

TIP!

If you want to set, for example, the same background color to more than one UI modeling element, you can select all these elements in the diagram and then change the required property for all of them in the same way.

The following table lists the supported symbol properties:

Symbol Property	Effect
Fill Color	Sets the background color of the component to the chosen one, property **Use Fill Color** must be marked.
Font	Sets the font if there is any text in the component.
Text Color	Sets the text color of the component to the chosen one.
Pen Color	Sets the color of the border for components that have one.

Copyright © 1998-2015 No Magic, Inc.

User interface prototyping

User Interface Modeling allows you to create browsable reports. All you need to do, when creating a breathable report, is add a hyperlink to the UI model element, is to and link it to the other one. Once a report of these models has been created, click on any element in the report. You will then be directed to the other diagram.

As Figure 454 on page 745 shows, when you click **OK** in the upper left-hand frame you will be directed to the next frame. And using the **OK** of this frame you can get to a couple of frames in which you can step forward and backward as you want since they are all linked to one another.

"Case Study 3 - User Interface Prototyping Example" on page 749 provides an example of how this feature works and explains how to add hyperlinks and create browsable reports.

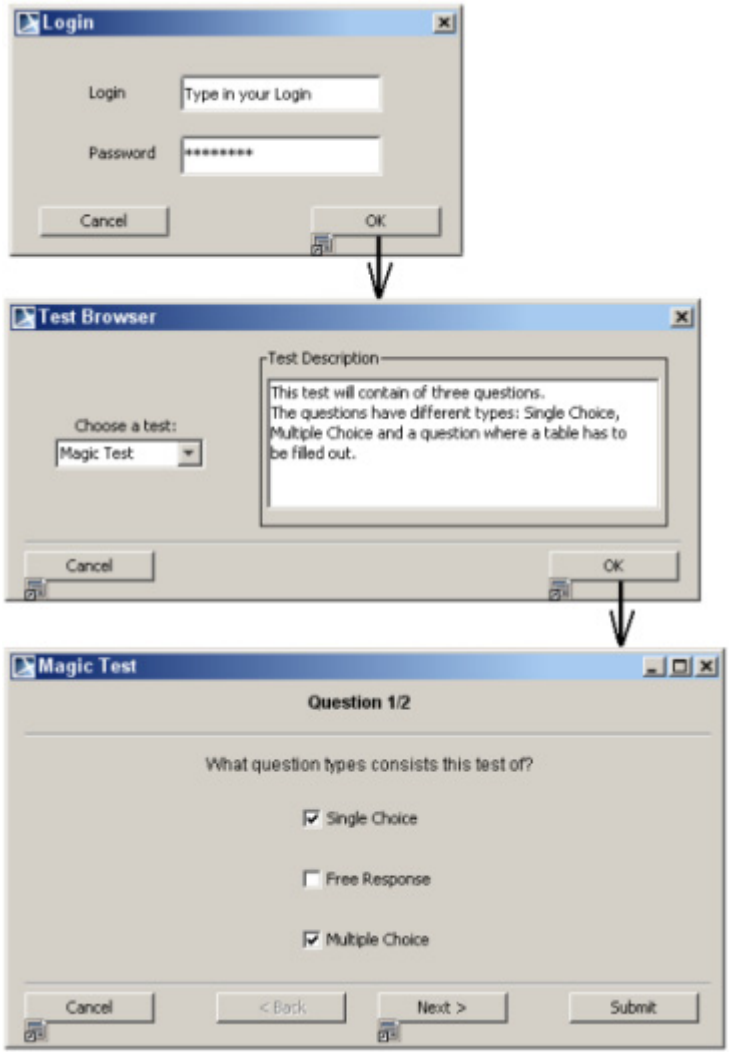


Figure 453 -- Example of User Interface Models and their Workflow

Case Studies for User Interface Modeling

This section includes three case studies on how to create User Interface models:

- "Case Study 1 - Modeling user interface for the Report Wizard dialog" on page 744.
- "Case Study 2 - Slider Example" on page 747.
- "Case Study 3 - User Interface Prototyping Example" on page 749.

Copyright © 1998-2015 No Magic, Inc.

The sample projects can be found in <MagicDraw installation folder>\samples\diagrams\User Interface Modeling\UI Modeling Samples.mdzip.

Case Study 1 - Modeling user interface for the Report Wizard dialog

This case study provides step-by-step instructions for modeling the **Report Wizard** dialog.

Step #1 Create a new Project

1. Choose **New Project** from the **File** menu.
2. Name it *Report Wizard*.

Step #2 Create a new Diagram

1. Click the **User Interface Diagram**  button in the main diagram toolbar.
2. Name the new diagram *Report Wizard*.

Step #3 Create Container components

1. Click the **Frame** button in the User Interface diagram toolbar and drag-drop it on the diagram pane. A Frame component will be created.
2. Name the Frame component. Select Frame on the diagram and type its name *Report Wizard.* Or you can define the Frame component name in the **Frame** specification dialog box:
    2.1 Double-click the Frame component on diagram to open the Frame specification dialog box.
    2.2. Type *Report Wizard* in the Title field.
3. Clear the check box properties **Maximize** and **Minimize** of the Frame in the **Frame** specification dialog box.
4. Define Icon for the Frame component:
    4.1. Select the **custom** option in the **Icon** property in the **Frame** dialog box. The **Open** dialog box will open.
    4.2. Open <MagicDraw installation>\plugins\com.nomagic.magicdraw.uimodeling and select the *nomagic.png* image to set the frame icon.
5. Create two **Group Boxes** and one **Separator**.
    5.1. Name one **Group Box** as *Select Template*
    5.2. For the other Group Box clear the check box property **Titled**.
    5.3. Drag all three components to the **Frame** and arrange everything like in Figure 455 on page 746.

Copyright © 1998-2015 No Magic, Inc.

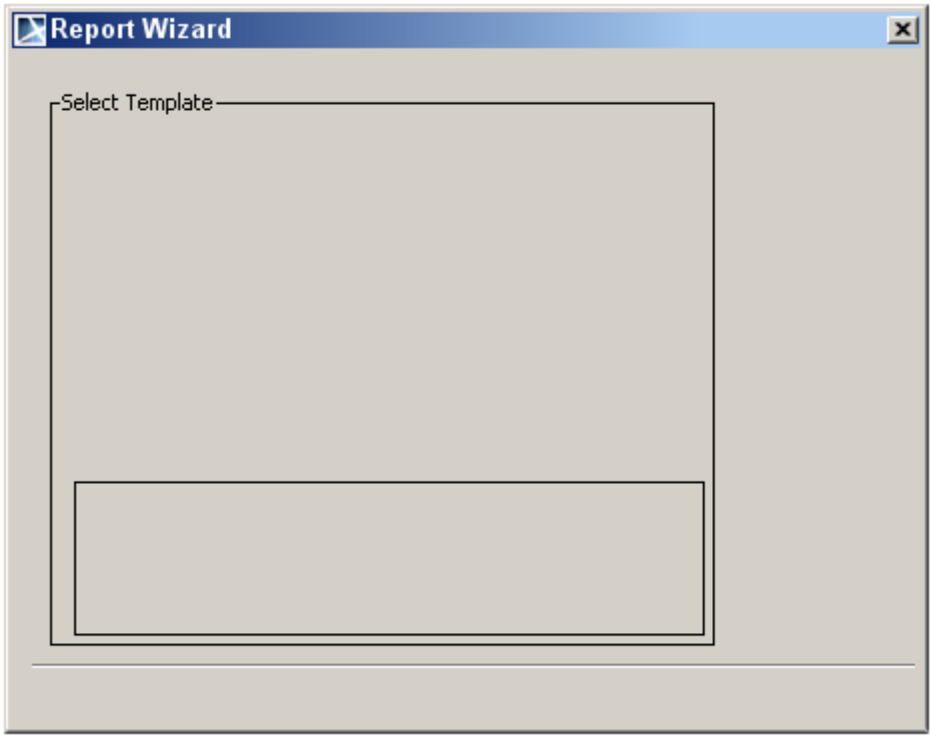


Figure 454 -- Report Wizard Frame with Group Boxes and Separator

Step #4 Create a Tree

1. Create a new Tree component:
    1.1. Click the **Other** group in the diagram toolbar to expand the toolbar.
    1.2. Click the **Tree** button in the diagram toolbar and drag-drop it on the *Select Template* Group Box on diagram pane.
2. Delete old nodes from the Tree component.
3. Create new nodes to the Tree component.
    3.1. Right-click on the Tree in the Containment tree to invoke its shortcut menu and choose **Create Element** > **Node**.
    3.2. Create five new **Nodes** in this way and name them as shown in Figure 456 on page 747.
4. Add Leaves to Nodes.
    4.1. You can add a **Leaf** by right-clicking on a **Node** in the containment tree and then choose **Create Element > Leaf**.
    4.2. Add at least one **Leaf** to every **Node** to indicate that the nodes have internal elements.

Copyright © 1998-2015 No Magic, Inc.

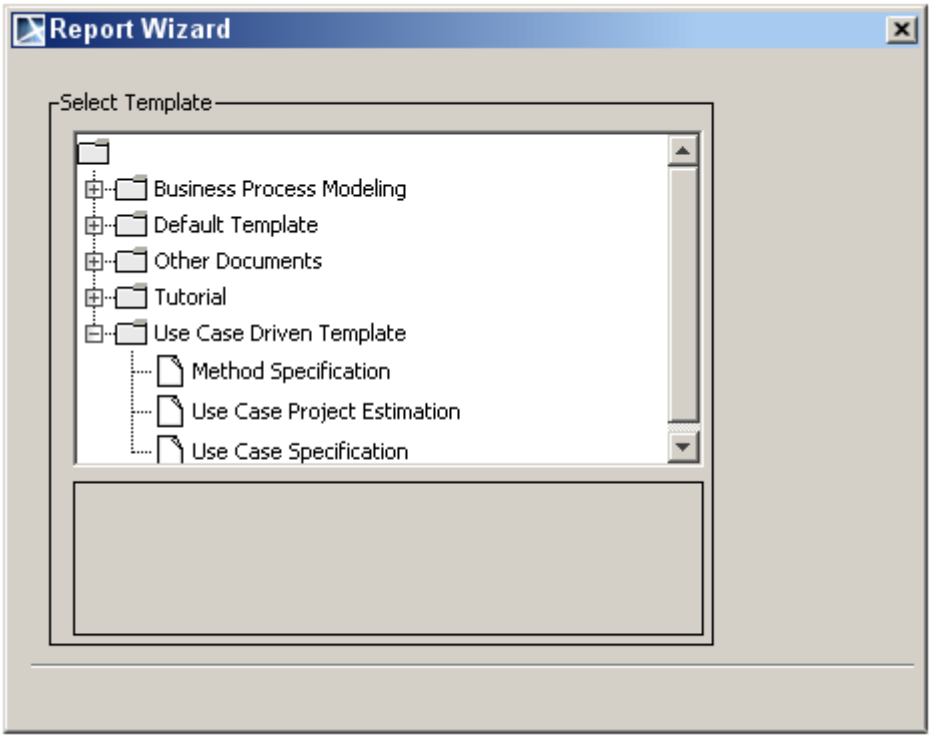


Figure 455 -- The Report Wizard Window with Added Tree

Step #5 Add Buttons

1. Create predefined text buttons.
    1.1. Click the **Buttons** group in the diagram toolbar to expand the toolbar.
    1.2. Right-click the **Back** button to expand the text buttons group. Create **Back**, **Next** and **Cancel** buttons (Figure 457 on page 748).
2. Create regular buttons.
    2.1. Click the **Button** button in the toolbar and drag it to the pane.
    2.2. Select the created button in the diagram and type in *New.*
    2.3. Create remaining buttons by repeating steps 2.1 and 2.2.
3. Check property **Inactive** for all buttons, except **Next>**, **Cancel**, **New** and **Import**.

Copyright © 1998-2015 No Magic, Inc.

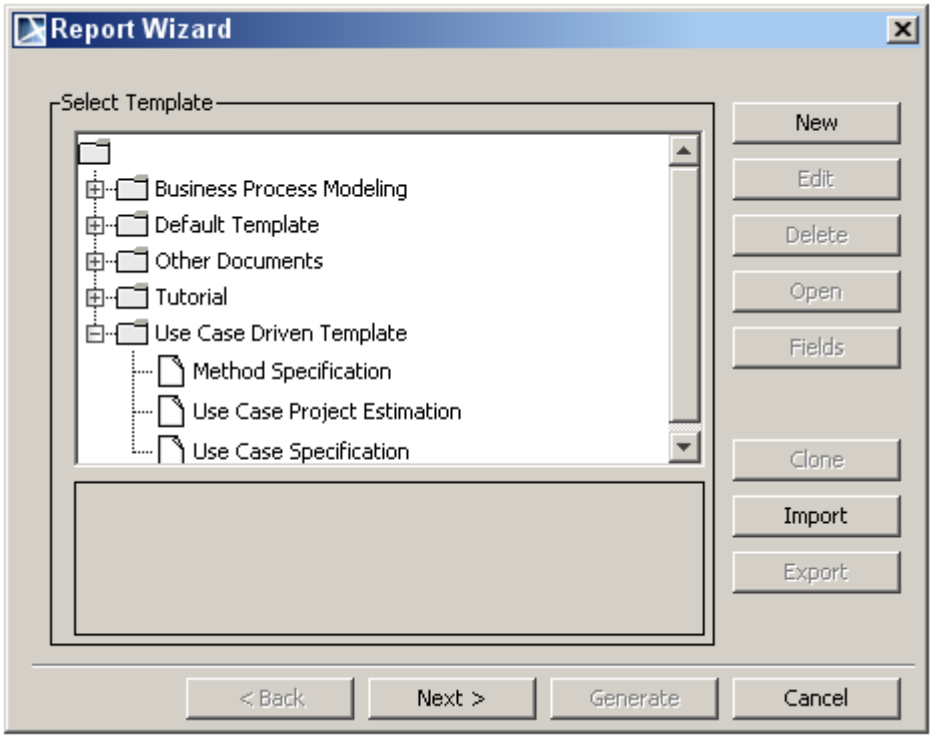


Figure 456 -- Sample of the Report Wizard window

Step #6 Using the Report Wizard window

For example this user interface model could now be exported as an image. The steps to do this are as follows:

1. Select **Save As Image** from the **File** menu.
2. In a new dialog mark **Active Diagram**.
3. In **Image File** define the location where the image should be placed.
4. Select *Joint Photographic Experts Group (*.jpg)* in **Image Format** and then press the **Save**-Button. Of course you can also take another format if you want to.

Case Study 2 - Slider Example

This case study contains step-by-step instructions showing how to create a User Interface model with Sliders. It also shows how to customize the symbol properties of User Interface components. It does not explain, however, how to create a new project and a new diagram since those are explained in “Case Study 1 - Modeling user interface for the Report Wizard dialog” on page 744.

Step #1 Create a Frame, Labels and Sliders

1. Create a new **Frame** and add title *Symbol Properties Customization.*
2. Add new **Label** to the **Frame**.
    2.1. Click the **Text** group in the diagram toolbar to expand the toolbar.
    2.2. Click the **Label** button and place the label on the diagram. Name it *Fill Color.*
    2.3. Create the remaining two labels.
3. Add **Sliders** to the **Frame.**
    3.1. Click the **Slider** button in the **Other** toolbar.
    3.2. Set property **Spacing** to 50 so as to just have three values marked in the slider.
    3.3. Set property **Knob Position** to 0 to move the knob to the left position.

Copyright © 1998-2015 No Magic, Inc.

3.4. Select given value *0 0* out of property **Values** and rename it to *0 Red*. Rename the other two given values to *50 Blue* and *100 White*.*

3.5. Draw two more sliders and repeat last steps setting values as shown in Figure 457 on page 748.

*

NOTE

With regard to setting the values of a slider it is important to know that there is no empty space between 0. 0, for example, represents a new line. So, in this case, when entering a value, it should look like this:

0

0

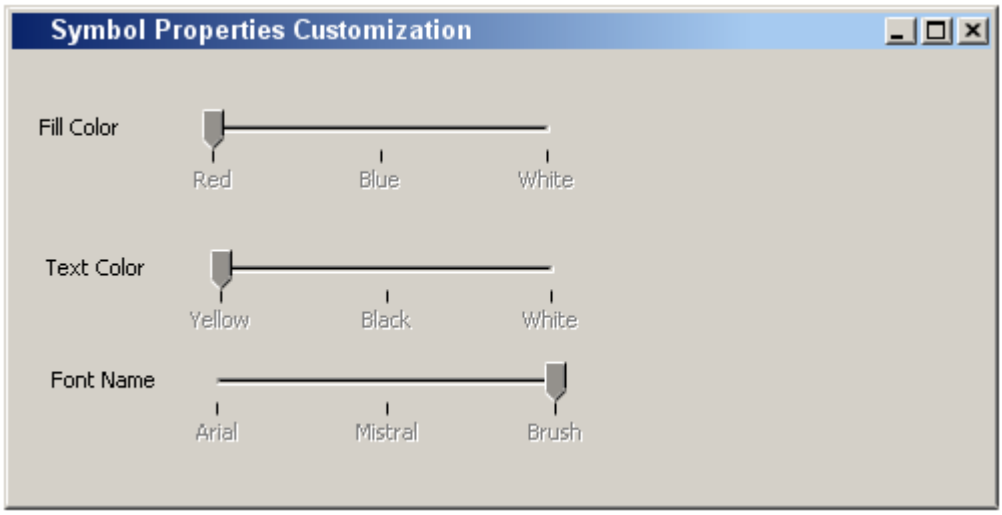


Figure 457 -- Symbol Properties Customization Frame with Added Labels and Sliders

Step #2 Add Text Fields

1. Create four **Text Fields.**

1.1. Click the **Text Field** button in the **Text** toolbar.

1.2. Name it *Red Background*.

1.3. Repeat the previous two steps for the remaining two text fields (Figure 458 on page 748).

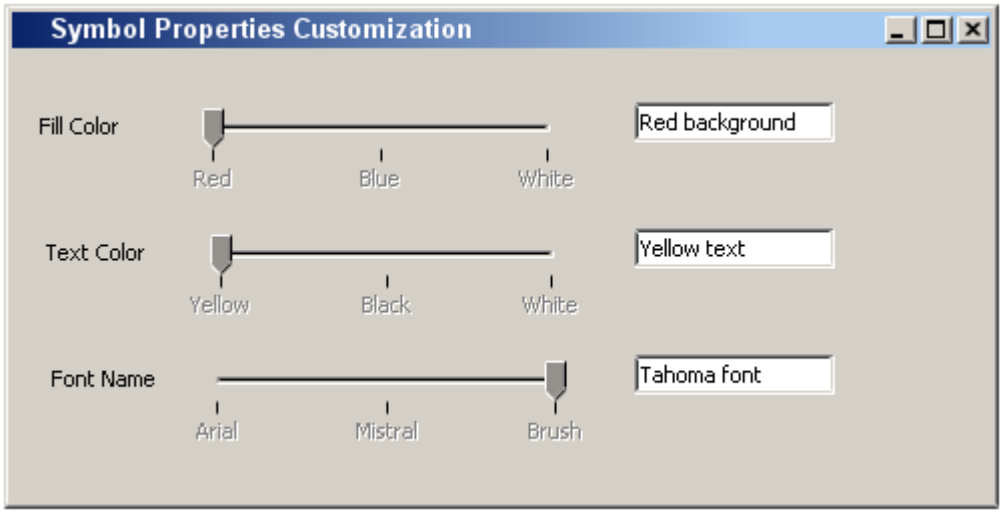


Figure 458 -- Added Text Fields to the Frame

Copyright © 1998-2015 No Magic, Inc.

Step #3 Edit Symbol Properties for the Text Fields

1. Edit Symbol Properties for the first text field.
    1.1. Select the first text field on the diagram surface.
    1.2. Right click and select **Symbol Properties** in the shortcut menu.
    1.3. Check property **Use Fill Color** and change **Fill Color** to red.
2. Edit Symbol Properties for the remaining fields.
    2.1. For the second text field, select yellow in **Text Color**.
    2.2. For the third one, select the font name Tahoma in **Font** (Figure 459 on page 749).

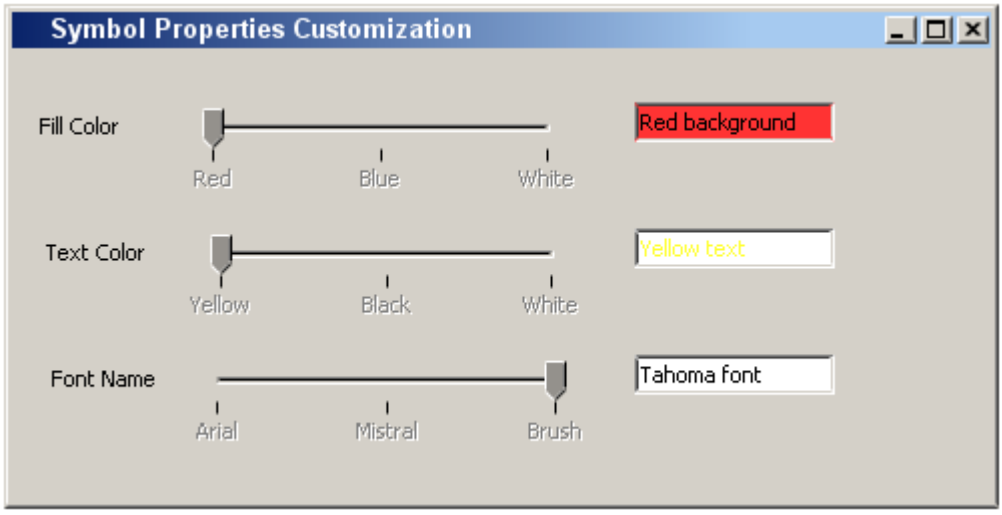


Figure 459 -- Finalized Slider Example

Case Study 3 - User Interface Prototyping Example

This case study shows how to connect several user interface models with one another and create a browsable report out of them to display the wildfowl of an application. The models should represent a test application with a Login Dialog, a Test Browser and a test with several questions. Here are step-by-step instructions for adding hyperlinks and creating browsable reports.

Step #1 Create first Model

1. Create a new **Package** for the model.
2. Create a model similar to the one in Figure 460 on page 749.

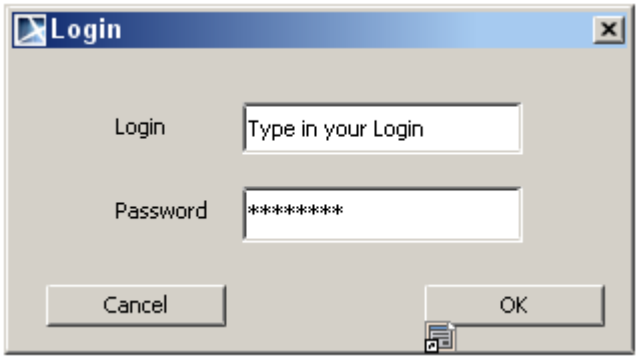


Figure 460 -- First Model of Prototyping Example - Login Dialog

Step #2 Create second Model

1. Create again a new **Package** for this model.

Copyright © 1998-2015 No Magic, Inc.

2. Create a model similar to the one in Figure 461 on page 750.

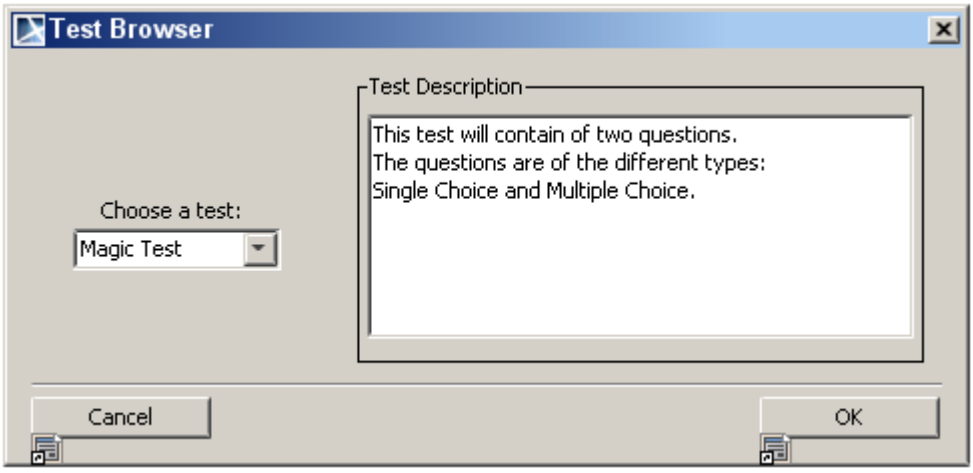


Figure 461 -- Second Model of Prototyping Example - Test Browser

Step #3 Create remaining Models

1. Create a separate **Package** for every model.
2. Build models similar to the ones shown in the full detailed sample which was mentioned in the beginning of this chapter. Or create models for questions on your own.

Step #4 Add Hyperlinks

1. Add a hyperlink.
    1.1. Open model with **Login Dialog**.
    1.2. Select the **OK** button.
    1.3. Click on smart manipulator **Hyperlinks/Go To** and select **Add Hyperlink** in the popup menu.
    1.4. Select **Element/Symbol** and click the "**...**" button. Browse to the **Package Test Browser**, select the **User Interface Diagram** in it and confirm two times with OK - a diagram symbol will appear next to the **OK** button. Double clicking on the **OK** button will lead to the other diagram.
2. Connect now all the buttons in the other diagrams with hyperlinks - you will see which component has an hyperlink because of the diagram symbol next to it.

Step #5 Create a browsable Report

1. Create a new report.
    1.1. Select **Report Wizard** from the **Tools** menu.
    1.2. Open the node **Default Template** in the tree and select **Web Publisher 2.0**.
    1.3. Confirm three times by clicking **Next** until dialog appears where to add data to the report. Add then all **Packages** that contain the created diagrams and click **Next**.
    1.4. Give a name to the output file by entering **Report file** and check the box **Display in viewer after generating report**. After clicking **Generate**, report will be built and shown in your default browser.

Copyright © 1998-2015 No Magic, Inc.

Content Diagram



This feature is available in Standard, Professional, Architect, and Enterprise editions.

The content diagram is an extension of UML notation. The purpose of the content diagram is to generate or represent a project structure (diagrams) and relations between them. The content table works as a table of contents for a project.

All content diagrams have their own specifications, wherein you can specify their name, documentation, and view the relationships in which they participate. You can also add stereotypes, tagged values, and constraints.

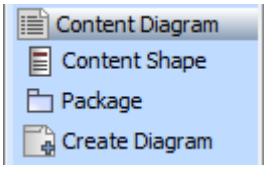
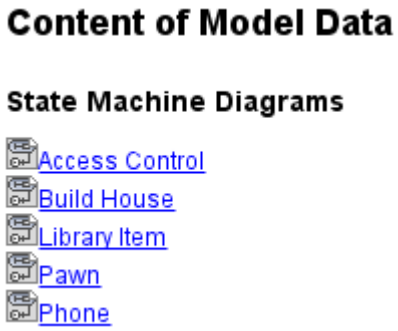
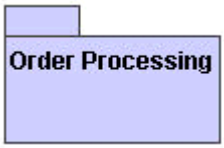


Figure 462 -- Content diagram pallet

Copyright © 1998-2015 No Magic, Inc.

Content Diagram Elements

Diagram Pallet Button	Button (Hot key)	Notation
Content Shape Creates a table of contents of all diagrams of the project.	(C)	
Package	(P)	
Create Diagram	(Ctrl+N)	

TIP! For more information about diagrams creation, see "Creating Diagrams" on page 198.

Robustness Diagram

NOTE This feature is available in Standard, Professional, Architect, and Enterprise editions.

The Robustness Analysis involves analyzing the narrative text of use cases, identifying a first-guess set of objects that will participate in those use cases, and classifying these objects based on the roles they play.

- Boundary or Interface objects are what actors use in communicating with the system.
- Entity objects are usually objects from the domain model.
- Control objects (are usually called controllers because they often are not real objects), serve as the “glue” between boundary objects and entity objects.

The Robustness analysis serves as a preliminary design within the project life cycle and provides the missing link between an analysis and a detailed design.

Four basic rules apply:

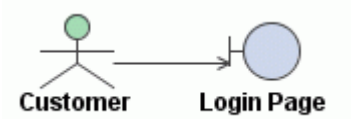
1. Actors can only talk to the boundary objects.
2. The boundary objects can only talk to the controllers and actors.

Copyright © 1998-2015 No Magic, Inc.

3. The entity objects can only talk to the controllers.
4. The controllers can talk to the boundary objects and entity objects, and to other controllers, but not to the actors.

Both the boundary objects and entity objects are nouns, and the controllers are verbs. Nouns cannot talk to other nouns, but verbs can talk to either nouns or verbs.

Robustness Diagram Elements

Diagram Pallet Button/Model Element	Button (Hot key)	Notation
Actor An actor represents a role played by an external person, a process or a thing interacting with a system. One physical object can play several roles.	 (A)	
Boundary Actors use the boundary objects to communicate with the system (sometimes called the interface objects). A class with a stereotype “boundary”	 (B)	
Control Serves as the “glue” between the boundary objects and the entity objects. A class with a stereotype “control”	 (C)	
Entity The entity objects usually are objects from the domain model. A class with a stereotype “entity”	 (E)	
Robustness Association The Association with a default Association End A navigability = false and Association End B navigability = true values.	 (S)	

Web Diagram



This feature is available in Standard, Professional, Architect, and Enterprise editions.

Copyright © 1998-2015 No Magic, Inc.

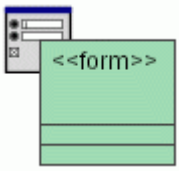
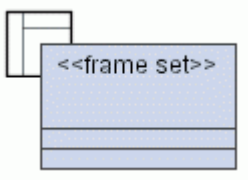
The web system consists of server applications, network, communicating protocol, and the browser. Basically, a user's requests begin from starting the browser and requesting a document through a network from the server (host computer). The web server running on the host computer, catches the user's request, locates the document and delivers it to the user.

UML is a standard language for modeling software. However, modeling specific web components cannot be done by using just a standard UML. The Web-UML diagram provides an extension to the UML model, which enables developers to model web applications.

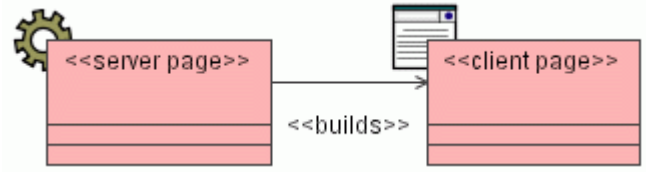
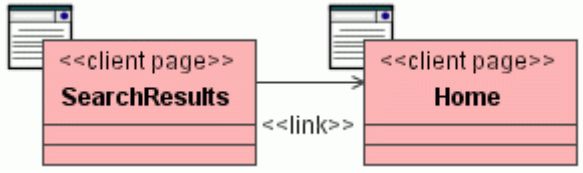
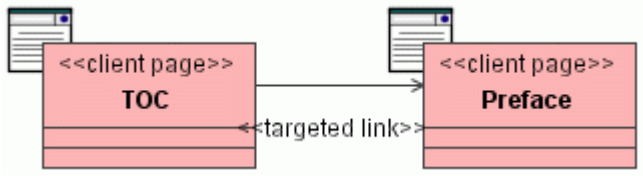
The MagicDraw Web-UML diagram includes Web-UML elements (stereotyped UML elements) for modeling: client, server pages, web form, frame classes, java script class representation and target class, and web page component.

Reference: Building Web Applications with UML by Jim Conallen Copyright ©2000 by Addison Wesley Longman, Inc.

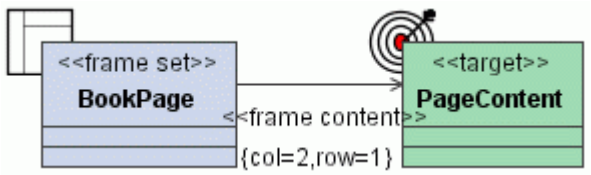
Web Diagram Elements

Diagram Pallet Button/Model Element	Button (Hot key)	Some examples
Client Page	 (SHIFT+G)	
Server Page	 (SHIFT+S)	
Form	 (SHIFT+F)	
Frame	 (SHIFT+E)	

Copyright © 1998-2015 No Magic, Inc.

Diagram Pallet Button/Model Element	Button (Hot key)	Some examples
Target	(SHIFT+T)	
Java Script	(SHIFT+J)	
Page	(SHIFT+P)	
Builds	(B)	
Link	(L)	
Redirect	(T)	
Targeted Link	(SHIFT+L)	

Copyright © 1998-2015 No Magic, Inc.

Diagram Pallet Button/Model Element	Button (Hot key)	Some examples
Frame Content	 (SHIFT+O)	
Submit	 (U)	
Object	 (O)	
RMI	 (SHIFT+R)	
IIOP	 (SHIFT+I)	

CORBA IDL Diagram



This feature is available in Architect and Enterprise editions.

The CORBA IDL diagram facilitates the creation of CORBA IDL elements. The following patterns are also available for CORBA IDL: Interface, Value Type, Type Definition, Sequence, Array, Fixed, Union, Enumeration, Struct, and Exception.



For more information about CORBA IDL usage in MagicDraw, see MagicDraw Code Engineering UserGuide.pdf, chapter “CORBA IDL Mapping to UML”.

Reference: UML™ Profile for CORBA™ Specification, Version 1.0, April 2002. http://www.omg.org/technology/documents/formal/profile_corba.htm.

Copyright © 1998-2015 No Magic, Inc.

CORBA IDL Diagram Elements

Diagram Pallet Button/ Model Element	Button (Hot key)
CORBAModule	 (M)
CORBA IDL Interface	 (U)
CORBA IDL Value	 (V)
Class by Pattern	 SHIFT+P
Generalization	 (G)
Truncatable Generalization	
Value Supports Generalization	
CORBA IDL Association	
Interface	 (I)

You can select either the UML Interface or the UML Class as a base element for the CORBA Interface



For more information about the CORBA IDL Interface implementation, see “CORBA Interface Implementation” in “ agicDraw Code Engineering UserGuide.pdf”.

Copyright © 1998-2015 No Magic, Inc.

WSDL Diagram



This feature is available in Architect and Enterprise editions.

The WSDL diagram is used to draw WSDL elements. It allows the creation of all elements used in the wsdl file, except schema. The schema elements can be created using the XMLSchema diagram. WSDL plugin provides patterns to create binding elements. The WSDL plugin requires XMLSchema plugin.

WSDL Diagram Elements

Diagram Pallet Button/Model Element	Button (Hot key)
WSDLmessage	 (M)
WSDLporttype	 (T)
WSDLbinding	 (B)
WSDLport	 (SHIFT+P)
WSDLservice	 (S)
WSDLdefinitions	 (D)
WSDLtypes	 (Y)
WSDLimport	 (I)

Copyright © 1998-2015 No Magic, Inc.

Diagram Pallet Button/Model Element	Button (Hot key)
XmlNs	 (SHIFT+L)
XSDnamespace	 (P)

Time Diagram



This feature is available in Standard, Professional, Architect, and Enterprise editions.

A Time Diagram is an extension of UML notation. The time diagram is similar to a sequence diagram, but the model elements of the time diagram have the predefined stereotypes.

Time Diagram Elements

Diagram Pallet Button/ Model Element	Button (Hot key)
Lifeline «CRconcurrent»	
Lifeline «SAshedRes»	
Message «RTevent», «CRimmediate»	
Message «CRimmediate»	
Message «SAtrigger»	)

Copyright © 1998-2015 No Magic, Inc.

Struts Diagram

NOTE This feature is available in Professional, Architect, and Enterprise editions.

A Struts Diagram is an extension of UML notation.

Struts Diagram Elements

Diagram Pallet Button/ Model Element	Button (Hot key)
Class	 (C)
Class by Pattern	 (SHIFT+P)
Interface	 (I)
ActionForm class for struts	 (G)
Action class for struts	
JavaServer Page for use with struts	
Package	 (P)
Model	 (M)
Interface Realization	 (R)

Copyright © 1998-2015 No Magic, Inc.

Diagram Pallet Button/ Model Element	Button (Hot key)
Realization	 (E)
Abstraction	 (T)
Usage	 (U)
Generalization	 (G)
Association	 (S)
Aggregation	 (A)
Composition	 (F)

Networking Diagram



This feature is available in Standard, Professional, Architect, and Enterprise editions.

This diagram allows a visual display of the network topology. The *Networking Profile* contains stereotypes for the network description. Elements with icons can be drawn on the Diagram pane.

The Networking Diagram is commonly used to depict hardware nodes as well as the connections between them.

Copyright © 1998-2015 No Magic, Inc.

Networking Diagram Elements

Diagram Pallet Button/ Model Element	Button (Hot key)
Server	
Application Server	
DB Server	
File Server	
Proxy Server	
Web Server	
PC	
Laptop	
Monitor	
Fax	
Plotter	
Printer	
Scanner	
Modem	
Router	

Copyright © 1998-2015 No Magic, Inc.

Diagram Pallet Button/ Model Element	Button (Hot key)
Switch	
Firewall	
Database	
Program	
Internet Browser	
Document	
File	
Wireless Network	
Internet	
Building	
City	
Actor	
User	
HTTP	
Ethernet	
RMI	

Copyright © 1998-2015 No Magic, Inc.

Diagram Pallet Button/ Model Element	Button (Hot key)
Communication Path	

Free Form Diagram

Free form diagram allows to draw various geometrical shapes. The diagram also includes shapes for drawing business flowcharts.

Relation Map Diagram

For more information about the relation map diagram, see section “Relation Map” on page 544.

Dependency Matrix



Dependency Matrix



This functionality is available in Standard, Professional, Architect, and Enterprise editions only.

The Dependency Matrix is a quick method for representing dependencies compactly. Diagrams, UML elements, and extended UML elements serve as row and column entries. The cells in the matrix show where these elements are associated, that is related.

Dependency matrices can show dependency criteria, such as

- UML relationships
- BPML, SysML, UPDM, and other DSL relationships
- Semantic dependencies (dependencies through properties)
- Relationships through tags (tags allow for relating UML elements that cannot be represented on the same diagram)
- Transitive relationships (through OCL or meta chain)

Dependency Matrices are useful for

- Quickly visualizing and creating dependency criteria.
- Compactly visualizing relations of a big system. Such system relations cannot be represented by a diagram on a single sheet of paper, as the diagram is very big.
- Studying relations from a particular scope and type of element by filtering the unimportant data. Hierarchical columns allows for browsing directly in the created matrix through the model scope and visualize the required area.

Copyright © 1998-2015 No Magic, Inc.

- Showing relations that cannot be represented on diagrams, such as representation (class by lifeline), behavior representation in other diagrams, operation representation by Call Behavior Action, Use Case relations with describing activities through property Owned Behavior, etc. The semantic dependency matrix is needed for deeper model analysis. The Dependency Matrix allows the representation of any kind of relations through element properties.

Get more knowledge about working with matrices in

- "Creating Dependency Matrix" on page 765
- "Using Dependency Matrix" on page 765
- "Dependency Matrix Environment" on page 778

Creating Dependency Matrix

The matrix element in the model is similar to the diagram element. After creating a new matrix, it appears in the Model Browser as a model element.

To create a Dependency Matrix

1. In the Containment tree, select an element that can be the owner of the generic table.
2. Do one of the following:.
   - From the main menu, select **Diagrams** > **Create Diagram**. Type “depe” and press **Enter**.
   - On the main toolbars, click the **Create Diagram** button. Type “depe” and press **Enter**.
   - Press **Ctrl**+**N**. Type “depe” and press **Enter**.
   - Right-click the element and from the shortcut menu select **Create Diagram** > **Analysis Diagrams** > **Dependency Matrix**.

   The newly dependency matrix opens on the right side of the application window.
3. Type the name of the matrix.
4. Select criteria for the matrix. For details, please see “Selecting criteria” on page 766.
5. Click the Refresh  button. The matrix is created.

You can also create dependency matrices using a custom dependency matrix type.



For more information about creating your own dependency matrix types, see "Creating New Dependency Matrix Type" in MagicDraw UMLProfiling&DSL UserGuide.pdf.

Using Dependency Matrix

You can modify a dependency matrix after it is created. The dependency matrix usage and modification features are described in

- "Selecting criteria" on page 766
- "Understanding cell content" on page 770
- "Modifying relationships in Dependency Matrix" on page 771
- "Navigation from cell" on page 773
- "Rearranging rows and columns" on page 774
- "Validation in Dependency Matrix" on page 778

Copyright © 1998-2015 No Magic, Inc.

Selecting criteria

To create a dependency matrix, first of all you need to define which data you wish to be displayed on its pane. Select row and column element types, row and column scope, direction of relationships as well as dependency criteria for this. Be advised that the easiest way to define all the above mentioned criteria, except the type of relationships direction is using the drag-and-drop operation.

To specify a row/column element type

Do either:

- In the Model Browser, select one or more elements which types you wish to see on your matrix and drag them to the **Row Element Type/Column Element Type** box in the **Criteria** area (see the following figure).

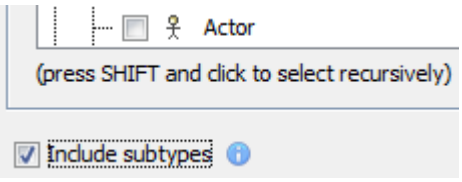
TIP!

- Hold down SHIFT to select multiple elements that are grouped together.
- Hold down CTRL to select multiple elements that are not grouped together.

- Click the **...** button next to the **Row Element Type/Column Element Type** box and in the opened dialog select what element types you wish to see on your matrix. Click **OK**.

NOTE

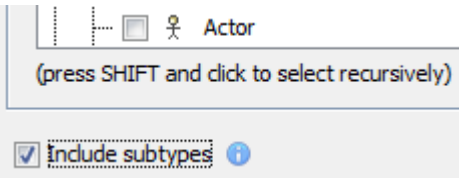
To display subtypes of selected element types, click to select the **Include subtypes** check box.



- Open the matrix Specification window, click the **Row Element Type/Column Element Type** property value cell, then click the button, and in the opened dialog select what element types you wish to see on your matrix. Click **Close**.

NOTE

To display subtypes of selected element types, click to select the **Include subtypes** check box.



TIP!

You can open the matrix Specification window in one of the following ways:

- From the matrix shortcut menu (see "Dependency Matrix shortcut menu" on page 780)
- From the matrix toolbar (see "Dependency Matrix toolbar" on page 778)
- Click an empty place in the matrix and press ENTER.

Copyright © 1998-2015 No Magic, Inc.

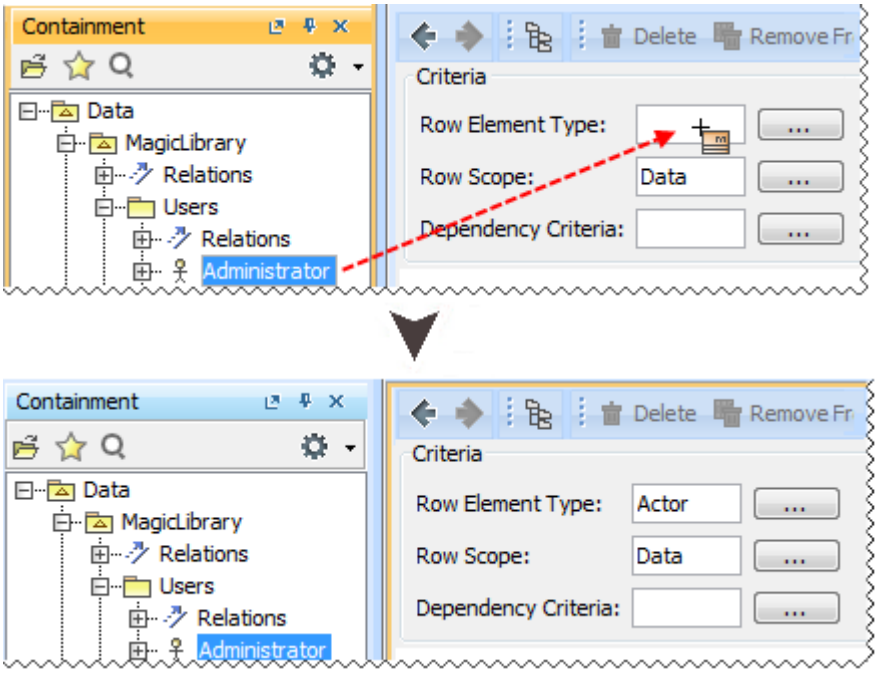


Figure 463 -- Using drag-and-drop operation to specify row element type

To specify a row/column scope

Do either:

- In the Model Browser, select one or more elements that you wish to see on your matrix and drag them to the **Row Scope/Column Scope** box in the **Criteria** area (see the following figure).

TIP!

- Hold down SHIFT to select multiple elements that are grouped together.
- Hold down CTRL to select multiple elements that are not grouped together.

- Click the **...** button next to the **Row Scope/Column Scope** box and in the opened dialog select what elements you wish to see on your matrix. Click **OK**.

TIP!

For more information about elements multiple selection, see "Elements Multiple Selection" on page 354.

- Open the matrix Specification window, click the **Row Scope/Column Scope** property value cell, then click the ... button, and in the opened dialog select what elements you wish to see on your matrix. Click **Close.**

TIP!

You can open the matrix Specification window in one of the following ways:

- From the matrix shortcut menu (see "Dependency Matrix shortcut menu" on page 780)
- From the matrix toolbar (see "Dependency Matrix toolbar" on page 778)
- Click an empty place in the matrix and press ENTER.

Copyright © 1998-2015 No Magic, Inc.

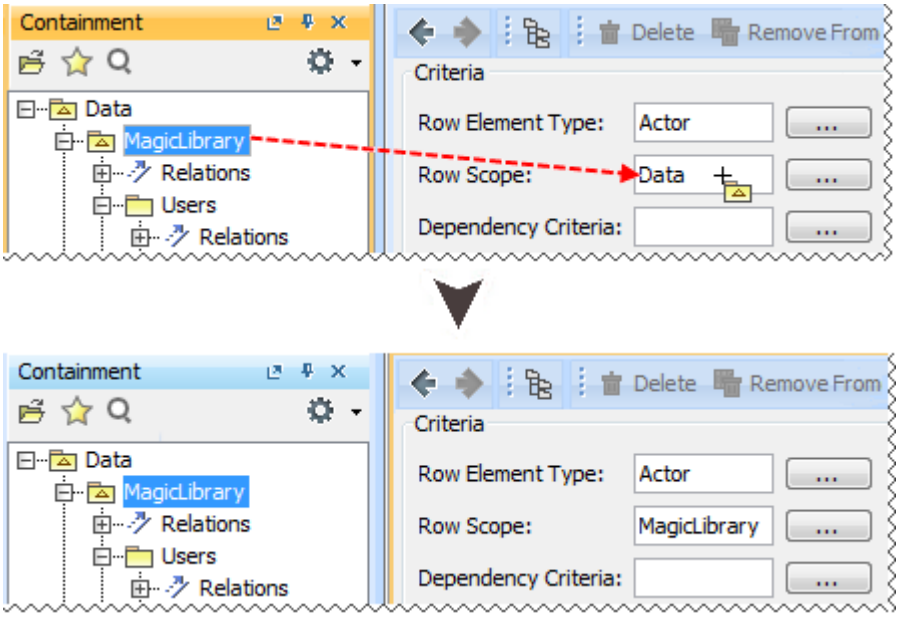


Figure 464 -- Using drag-and-drop operation to specify row scope

To specify a dependency criteria

Do either:

- In the Model Browser, select one or more relationships which types you wish to see on your matrix and drag them to the **Dependency Criteria** box in the **Criteria** area (see the following figure).

TIP!

- Hold down SHIFT to select multiple relationships that are grouped together.
- Hold down CTRL to select multiple relationships that are not grouped together.

- Click the **...** button next to the **Dependency Criteria** box and in the open dialog select what relationships you wish to see on your matrix. Click **OK.**

TIP!

More information about selecting relationship criteria, see "Specifying Criteria for Querying Model" on page 551.

- Open the matrix Specification window, click the **Dependency Criteria** property value cell, then click the ... button, and in the open dialog select what relationships you wish to see on your matrix. Click **Close**.

TIP!

You can open the matrix Specification window in one of the following ways:

- From the matrix shortcut menu (see "Dependency Matrix shortcut menu" on page 780)
- From the matrix toolbar (see "Dependency Matrix toolbar" on page 778)
- Click an empty place in the matrix and press ENTER.

Copyright © 1998-2015 No Magic, Inc.

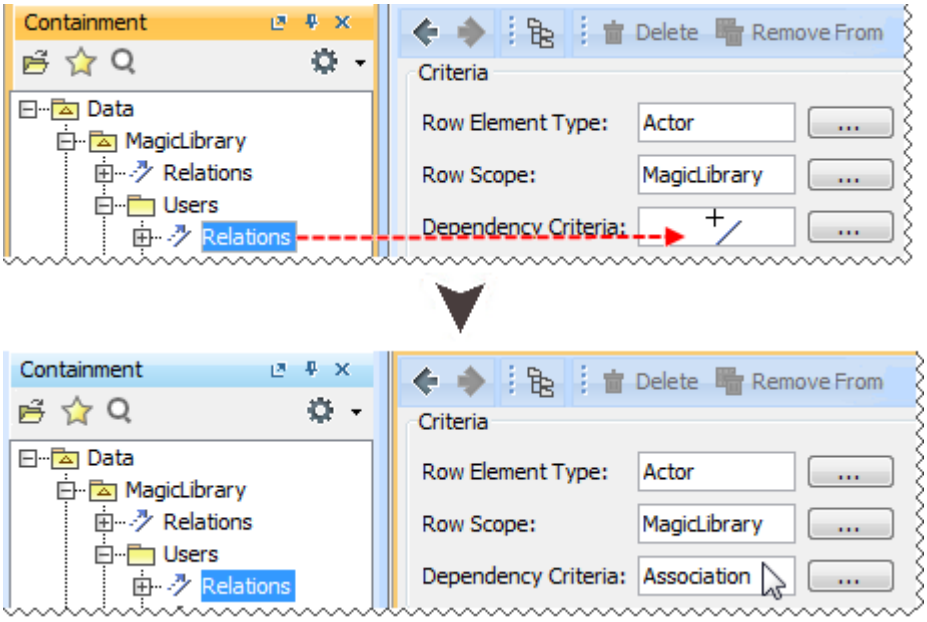


Figure 465 -- Using drag-and-drop operation to specify dependency criteria

To specify a relationship direction

Do either:

- Select a direction from the **Direction** drop-down list in the **Criteria** area.
- Open the matrix Specification window, click the **Direction** property value cell and select what direction relationships you wish to see on your matrix. Click **Close.**

TIP!

You can open the matrix Specification window in one of the following ways:

- From the matrix shortcut menu (see "Dependency Matrix shortcut menu" on page 780)
- From the matrix toolbar (see "Dependency Matrix toolbar" on page 778)
- Click an empty place in the matrix and press ENTER.

To specify what elements you wish to see on the matrix: related, non-related, or all

Do either:

- From the **Show Elements** drop-down list in the **Criteria** area, select
  - **With relations** if you need to see only related elements from the selected scope.
  - **Without relations** if you need to see only non-related elements from the selected scope.
  - **All** if you need to see both related and non-related elements from the selected scope.

Copyright © 1998-2015 No Magic, Inc.

- Open the matrix Specification window, click the **Show Elements** property value cell and choose one of the preceding described values. Click **Close**.

TIP!

You can open the matrix Specification window in one of the following ways:

- From the matrix shortcut menu (see "Dependency Matrix shortcut menu" on page 780)
- From the matrix toolbar (see "Dependency Matrix toolbar" on page 778)
- Click an empty place in the matrix and press ENTER.

NOTE

After the **Show Elements** value is changed, you should always refresh a matrix.

Once criteria are specified, you can create the matrix. Click  on the dependency matrix toolbar.

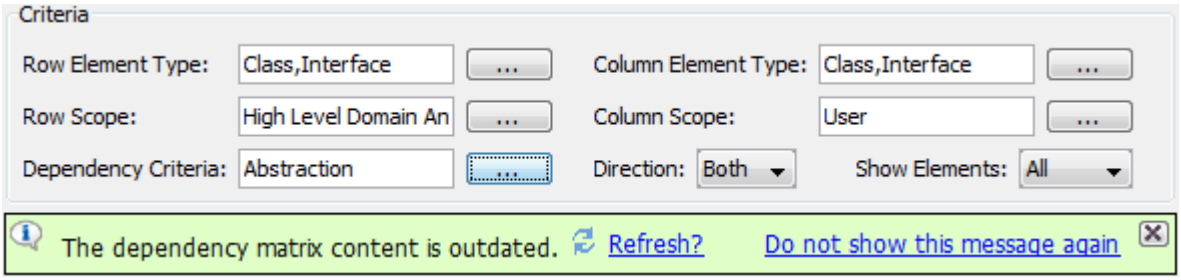


Figure 466 -- Notification to refresh dependency matrix after changing dependency criteria

TIP!

For more information about the dependency matrix toolbar, see "Dependency Matrix toolbar" on page 778.

Understanding cell content

Dependencies between elements are displayed in cells. Rows and columns display elements, which were specified as criteria.

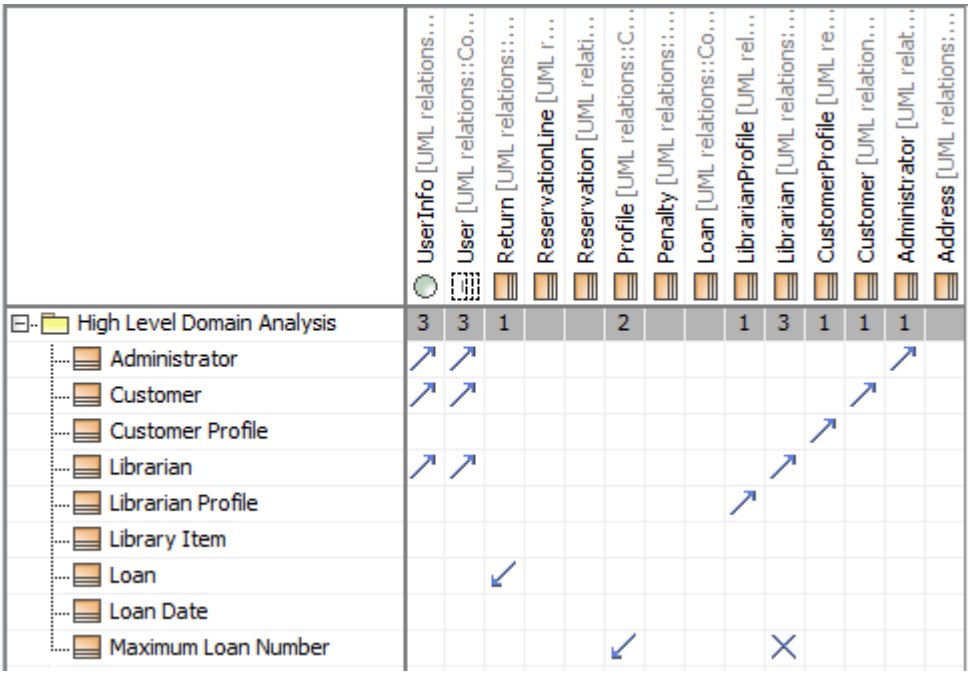


Figure 467 -- Variety of dependencies displayed on matrix

Copyright © 1998-2015 No Magic, Inc.

Dependencies can be represented as one of the following:

- **Arrowed diagonal.** It means that the element from which the direction points, depends on the element to which it points.
- **Diagonal.** It means that elements depend on each other.
- **Diagonal cross.** It means that multiple dependencies are represented in the cell.

Number of dependencies between owner elements is displayed in a cell where owners are intersecting with any element.

To see details of a dependency (relationship) that is represented in a cell

- Move mouse over the cell. Information about the relationship will be displayed on a ToolTip.

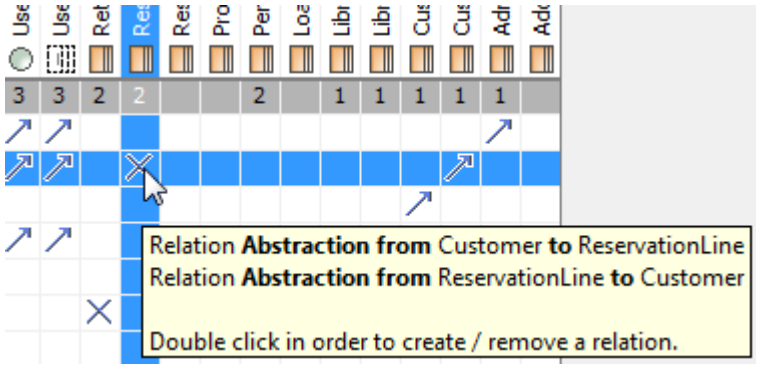


Figure 468 -- Displaying detailed information about dependency (relationship)

Modifying relationships in Dependency Matrix

You can edit relationships and element properties directly in matrices. Clicking the selected cell allows for

- Creating or deleting relationships between elements.
- Defining or removing element properties or tags.

This feature not only facilitates management of ordinary relationships between elements, but also allows for a faster creation of traceability links between elements, for example, between requirements and the architecture or requirements and test cases. Such an improvement saves huge amounts of time in comparison to linking elements in diagrams. It significantly increases applicability and usability of matrices.

To create a new relationship between model elements in the dependency matrix

NOTE

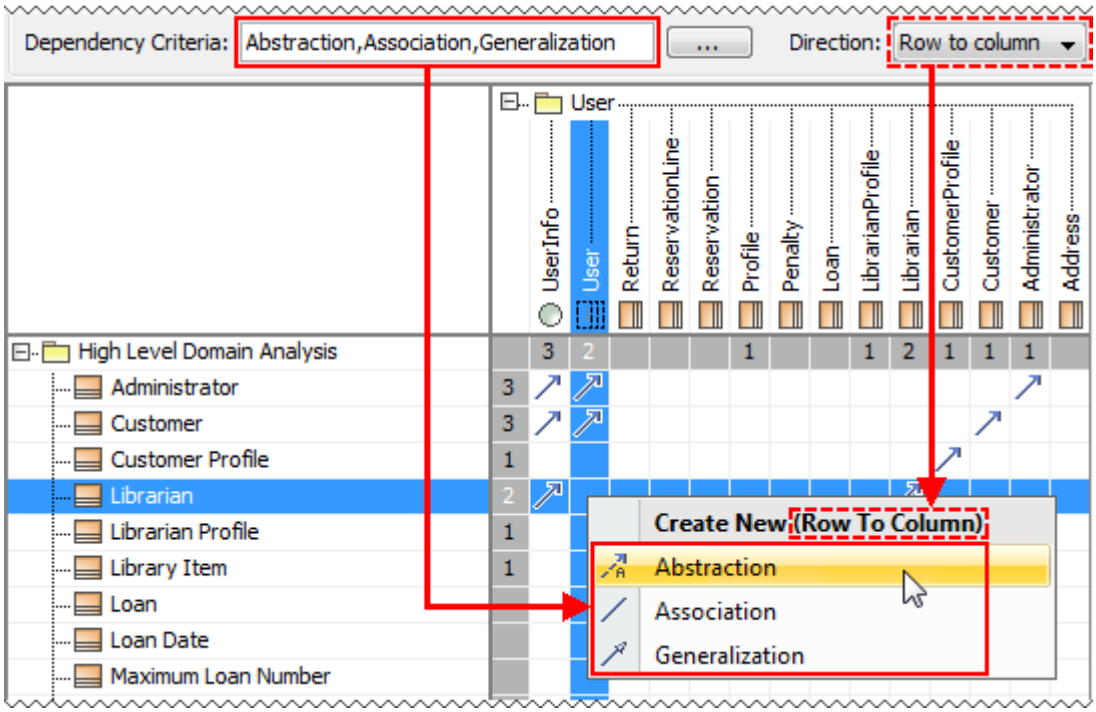
Commands on the shortcut menu of the cell, for creating a new relationship, depends on criteria specified in the **Criteria** area of the matrix:

- Possible relationship types correspond to the types specified as dependency criteria (see solid-line boxes in the following figure).
- Possible directions correspond to the directions specified in the **Direction** drop-down list (see dashed-line boxes in the following figure).

1. Do one of the following:
   - Double-click the cell. If only one relationship type and direction is available, the relationship is created instantly. If several relationship types or directions are available, the shortcut menu opens.

Copyright © 1998-2015 No Magic, Inc.

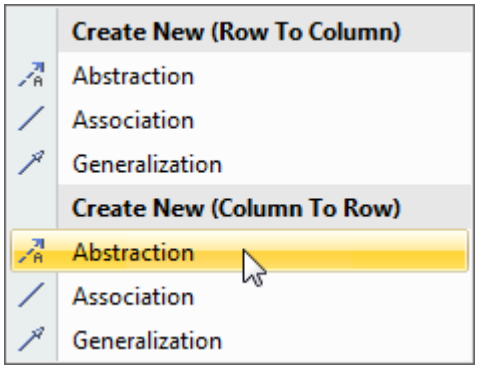
- Right-click the cell. The shortcut menu opens.



2. From the shortcut menu, under **Create New (Row To Column)** or **Create New (Column To Row)**, select a relationship type you need to create.

NOTE

If you have specified to display in the matrix, relationships with both directions, the shortcut menu of the cell will look like the one in the following figure.



Thus make sure you have selected a relationship type with relevant direction.

To remove a relationship between model elements from the dependency matrix

1. Do one of the following:
   - Double-click the cell. If only one relationship type and direction is available, the relationship is removed instantly. If several relationship types or directions are available, the shortcut menu opens.
   - Right-click the cell. The shortcut menu opens.

Copyright © 1998-2015 No Magic, Inc.

2. From the shortcut menu, under **Delete**, select the relationship you need to remove. The relationship is removed both from the matrix and the model.

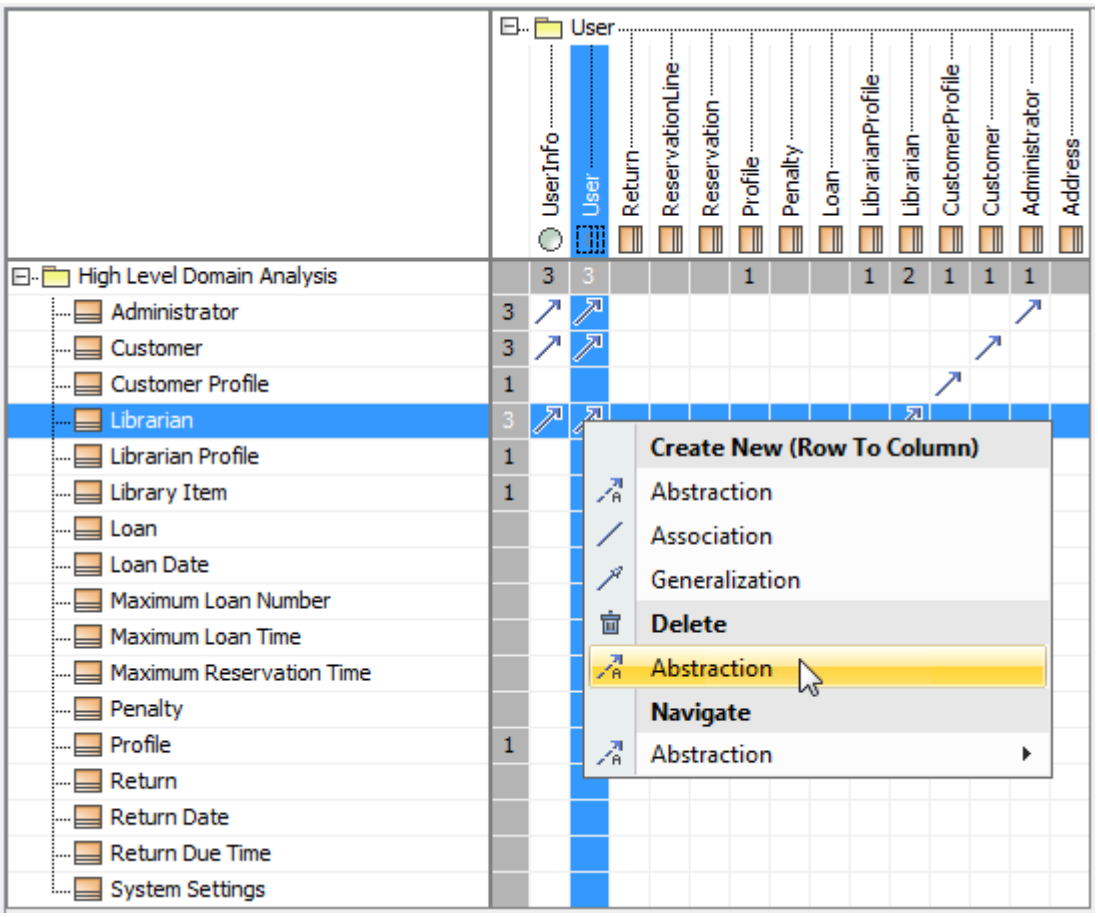


Figure 469 -- Removing relationship

To remove all relationships of a cell

1. Double-click or right-click the cell. The shortcut menu opens.
2. From the shortcut menu, under **Delete**, select **Delete All**. The relationships are removed both from the matrix and the model.

Navigation from cell

To select a relationship in the Containment tree

1. Right-click the cell. The shortcut menu opens.

Copyright © 1998-2015 No Magic, Inc.

2. On the shortcut menu, under **Navigate**, point to the relationship, and click **Select in Containment Tree**.

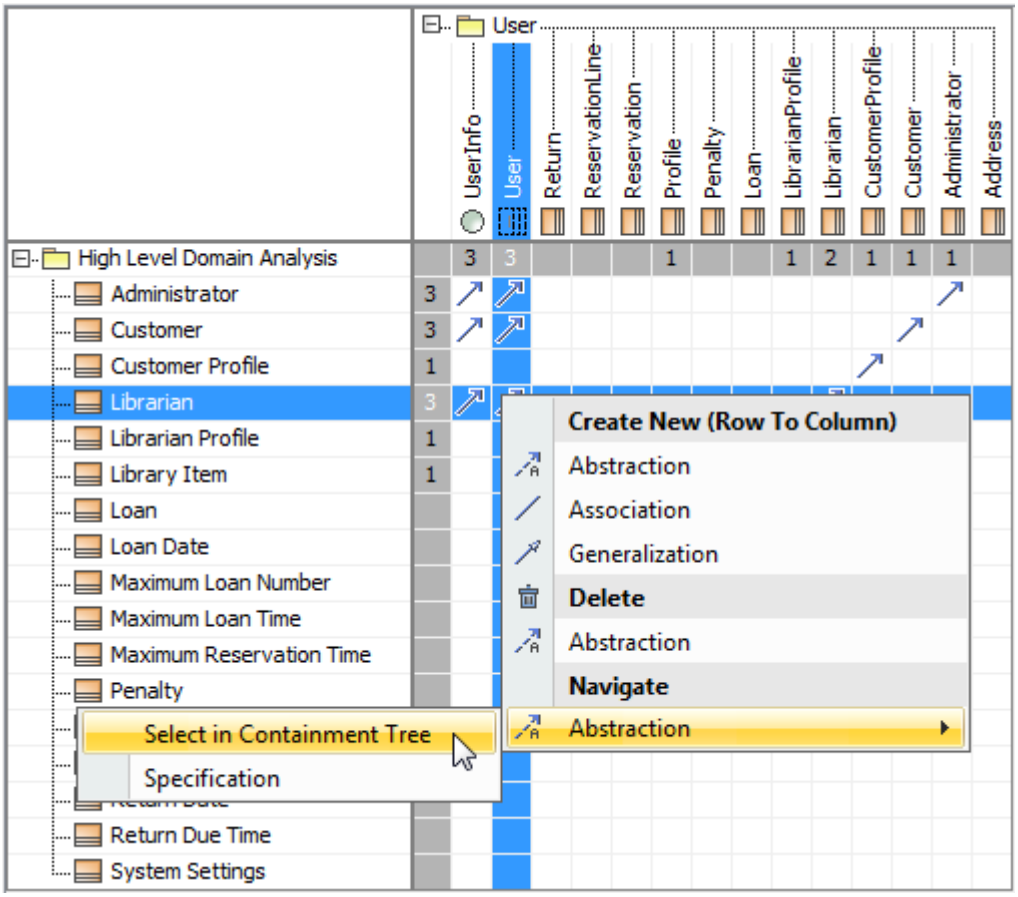


Figure 470 -- Choosing to select relationship in Containment tree

To open the Specification window of a relationship in the cell

1. Right-click the cell. The shortcut menu opens.
2. On the shortcut menu, under **Navigate**, point to the relationship, and click **Specification**.

Rearranging rows and columns

You can sort both rows and columns either in ascending or descending order (see the following figure) or create your own order by shifting selected items (either grouped or non-grouped) from one place to another. The sort

Copyright © 1998-2015 No Magic, Inc.

order persists after changing the row or column owner's display mode (see the Row Owner Display Mode or Column Owner Display Mode property description).

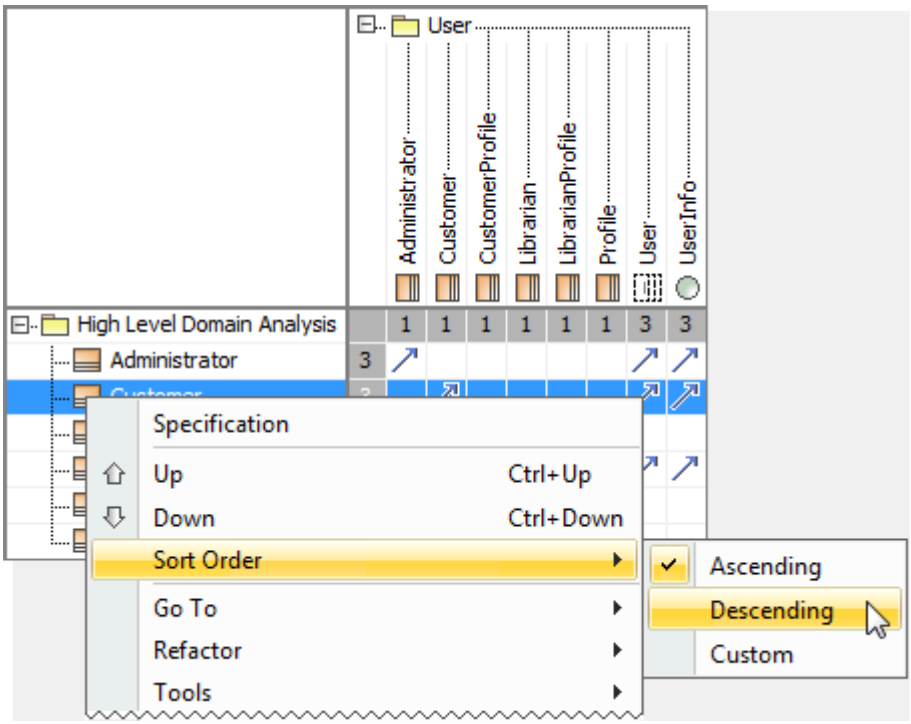


Figure 471 -- Selecting order for sorting rows

To shift rows up or down

1. Select rows you need to move.

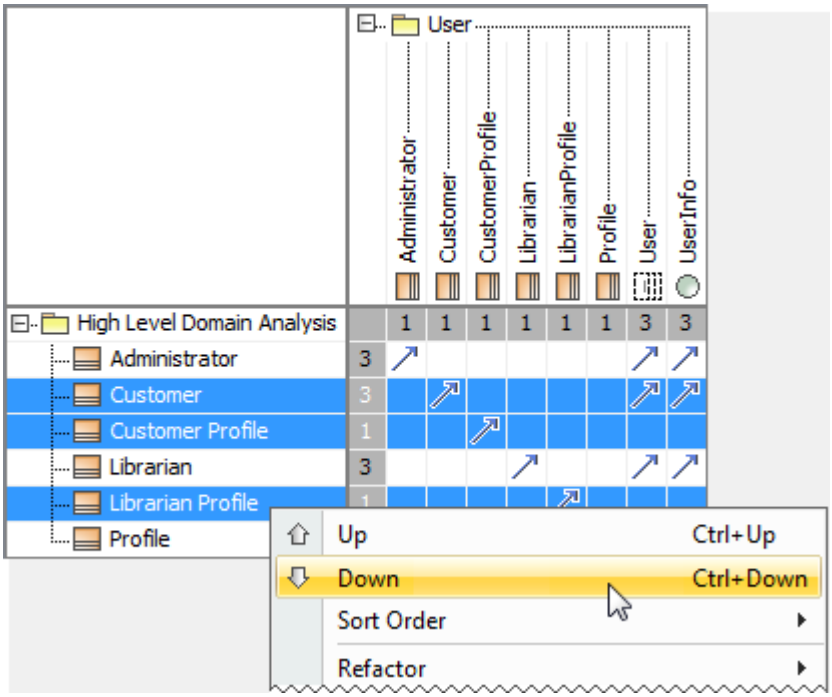
TIP!

- To select *adjacent rows*, select a single row, and then hold down the Shift key while you click other rows that you want to select.
- To select *nonadjacent rows*, select a single row, and then hold down the Ctrl key while you click other rows that you want to select.

2. Do either:

Copyright © 1998-2015 No Magic, Inc.

- Right-click the selected rows and on the shortcut menu, click **Up** or **Down**.



- Press Ctrl+Up Arrow or Ctrl+Down Arrow.



NOTE

After the manual rearrangement of rows, the sort order of the rows automatically changes to **Custom**.

To shift columns left or right

1. Select columns you need to move.

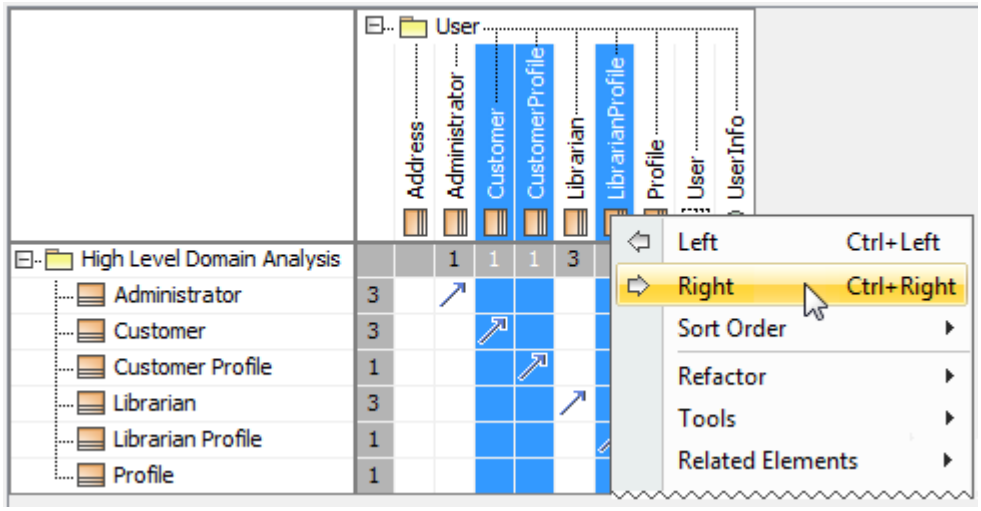
TIP!

- To select *adjacent columns*, select a single column, and then hold down the Shift key while you click other columns that you want to select.
- To select *nonadjacent columns*, select a single column, and then hold down the Ctrl key while you click other columns that you want to select.

2. Do either:

Copyright © 1998-2015 No Magic, Inc.

- Right-click the selected columns and on the shortcut menu, click **Left** or **Right**.



- Press Ctrl+Left Arrow or Ctrl+Right Arrow.



NOTE

After the manual rearrangement of columns, the sort order of the columns automatically changes to **Custom**.

Copyright © 1998-2015 No Magic, Inc.

Validation in Dependency Matrix

The validation results for relationships are visualized in matrices. Matrix cells are highlighted whenever relationships violate validation rules. This feature is useful, for example, to check which test cases for requirements verification fail, to check if dependencies are valid or up-to-date when doing gap analysis.

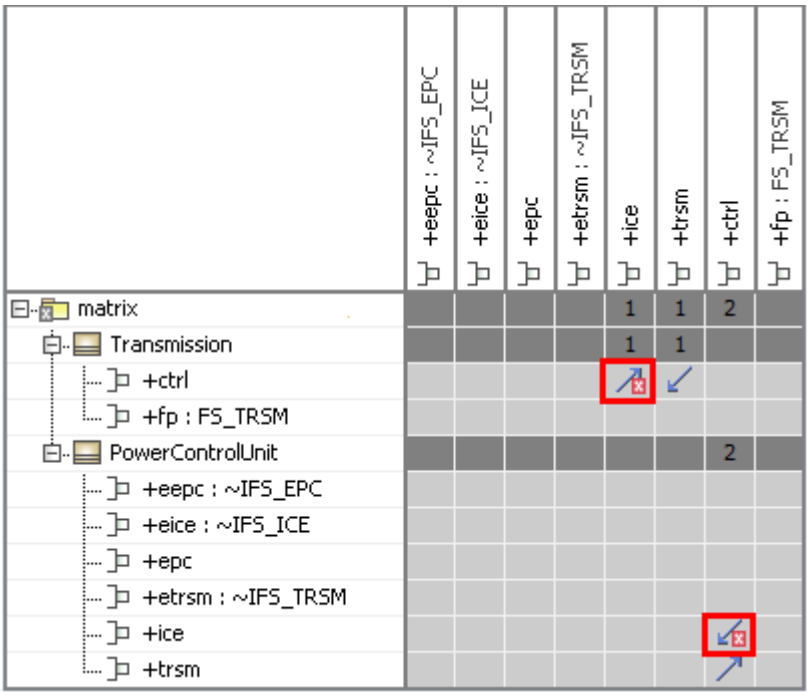


Figure 472 -- Active validation results visualization on dependency matrix

TIP!

For more information on validation, see "Active Validation" on page 632 and "Validation" on page 612.

Dependency Matrix Environment

The dependency matrix environment elements, such as toolbar, **Criteria** area, and shortcut menus are described in the following sections:

- "Dependency Matrix toolbar" on page 778
- "Dependency Matrix Criteria area" on page 779
- "Dependency Matrix shortcut menu" on page 780
- "Row/Column element shortcut menu" on page 782
- "Dependency Matrix Specification window" on page 783

Dependency Matrix toolbar

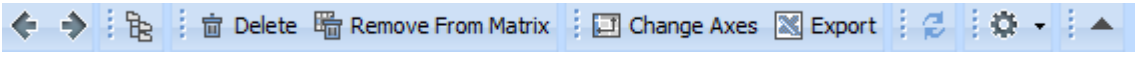


Figure 473 -- Dependency matrix toolbar

Copyright © 1998-2015 No Magic, Inc.

See the detailed descriptions of the dependency matrix toolbar buttons in the following table.

Button	Button name	Description
	Back (Alt+Left)	Navigates back to the previous diagram that was open in the current diagram tab.
	Forward (Atl+Right)	Navigates forward to the diagram in the current diagram tab.
	Select in Containment Tree (Alt+B)	Opens the Containment tab in the Model Browser and select the current diagram or symbol in the Containment tree.
	Delete (Ctrl+D)	Deletes the selected element from the matrix and model. **NOTE** If you undo the deletion, you should refresh the matrix to make the deleted elements appear in the matrix again.
	Remove From Matrix (Delete)	Removes the selected element from the matrix. The selected element is not removed from the model.
	Change Axes	Transposes rows and columns.
	Export	Exports a matrix as a Comma Separated Values (.csv) file. The file can be opened with Microsoft Excel.
	Refresh	Refreshes the matrix after updating the criteria or data.
	Options	Expands the menu with other options.
	• Specification	Opens the matrix Specification window wherein you can specify matrix properties.
	Suppress/ Expand Criteria Area	Suppresses or expands the matrix **Criteria** area.

Dependency Matrix Criteria area

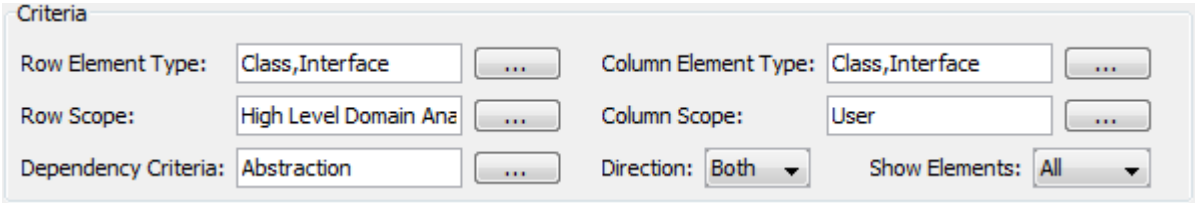


Figure 474 -- Dependency matrix criteria area

Copyright © 1998-2015 No Magic, Inc.

See the descriptions of the dependency matrix **Criteria** area in the following table.

Element Name	Element Type	Description
Row Element Type	Text box with ...	Click the **...** button to select an element type or multiple element types to show in rows of the dependency matrix. For more information about selecting the row element types, see the procedure "To specify a row/column element type" on page 766.
Column Element Type	Text box with ...	Click the **...** button to select an element or multiple element types to show in columns of the dependency matrix. For more information about selecting the column element types, see the procedure "To specify a row/column element type" on page 766.
Row Scope	Text box with ...	Click the **...** button to define a scope of the model (packages/ profiles) from which elements should be displayed in rows of the dependency matrix. For more information about selecting the row scope, see the procedure "To specify a row/column scope" on page 767.
Column Scope	Text box with ...	Click the **...** button to define a scope of the model (packages/ profiles) from which elements should be displayed in columns of the dependency matrix. For more information about selecting the row scope, see the procedure "To specify a row/column scope" on page 767.
Dependency Criteria	Text box with ...	Click the **...** button to define what relationships between row and column elements you wish to display in the dependency matrix cells. For more information about selecting the dependency criteria, see the procedure "To specify a dependency criteria" on page 768.
Direction	Drop-down list	Select a direction of relationships for the relationship analysis and representation in the dependency matrix. Be aware that new relationships will be created with the selected direction. For more information about selecting the direction of relationships, see the procedure "To specify a relationship direction" on page 769.
Show Elements	Drop-down list	Select to show only related (by a selected dependency criteria), only non-related, or all elements.

Dependency Matrix shortcut menu

To open the dependency matrix shortcut menu

- Right-click an empty space on one of the following
  - Dependency Matrix Criteria area

Copyright © 1998-2015 No Magic, Inc.

- Matrix pane

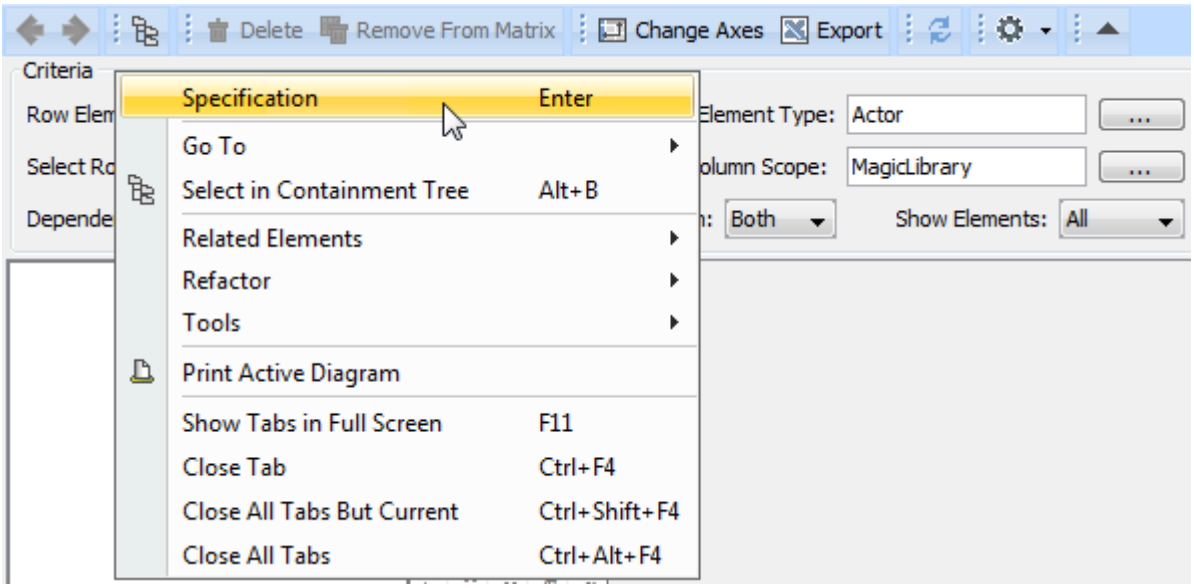


Figure 475 -- Dependency Matrix shortcut menu

Command	Description
Specification (Enter)	Opens the Specification window of a dependency matrix.
Go To	Navigates back or forward to the diagram that was open in the current diagram tab, finds and navigates to diagrams wherein the matrix is used, or to the matrix owner in the Containment tree, or to a hyperlinked object, or to an element to which the selected element is related through a traceability relationship. For more information, see • "Symbol Usage in Diagrams" on page 576 • "To add a hyperlink from the Model Browser" on page 343 • "Traceability properties in Go To submenu" on page 521
Select in Containment Tree (Alt+B)	Selects the dependency matrix in the Containment tree of the Model Browser.
Related Elements	Allows selecting whether to create a Relation Map for the matrix or search for model elements which depends on the matrix or are used by the matrix. For more information about the Relation Map, see “Relation Map” on page 544. For more information about usages and dependencies, see "Analyzing Usages and Dependencies" on page 509.
Refactor	Converts or replaces the matrix to a diagram indicated by the user. For more information, see "Refactoring" on page 359.
Tools	Opens a list of tools that are available for the matrix. For more information, see "Tools" on page 434.
Print Active Diagram	Prints a matrix.
Show Tabs in Full Screen (F11)	Shows a matrix in the full screen mode. Click **Close Full Screen** to return to the previous view.
Close Tab (CTRL+F4)	Closes an active matrix.

Copyright © 1998-2015 No Magic, Inc.

Close All Tabs But Current (CTRL+SHIFT+F4)	Closes all diagrams except an active one.
Close All Tabs (CTRL+ALT+F4)	Closes all opened diagrams.

Row/Column element shortcut menu

To open a row/column element shortcut menu

- Right-click the appropriate row/column.

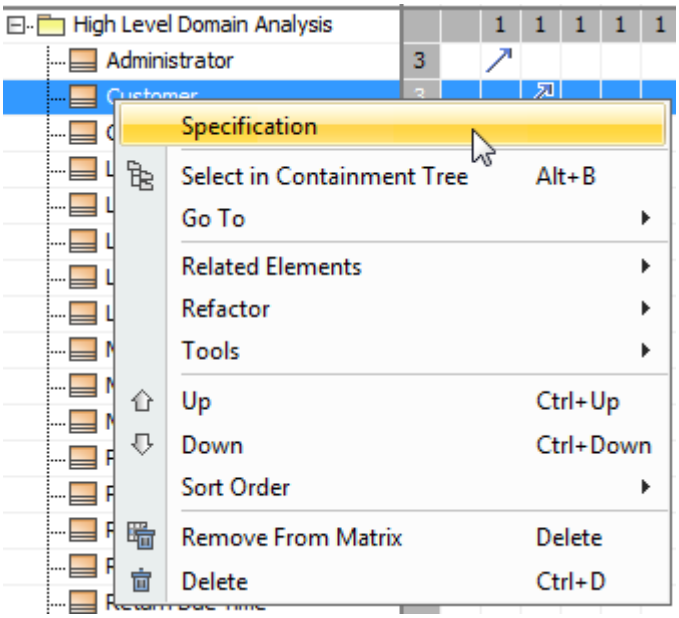


Figure 476 -- Shortcut menu of Dependency Matrix row element

Command (Shortcut key)	Description
Specification	Opens the element's Specification window. For more information, see "Specification Window" on page 273.
Select in Containment Tree	Selects the element in the Containment tree of the Model Browser.
Go To	Finds and navigates to a diagram which is owned by the selected element, or to the one, wherein the element shape is drawn, or to a hyperlinked object, or to an element to which the selected element is related through a traceability relationship. For more information, see • "Symbol Usage in Diagrams" on page 576 • "To add a hyperlink from the Model Browser" on page 343 • "Traceability properties in Go To submenu" on page 521
Related Elements	Opens the dialog for defining options to search for specified usages or dependencies of the selected element. For more information, "Analyzing Usages and Dependencies" on page 509.
Refactor	Converts or replaces the selected element to an element indicated by the user. For more information, see "Refactoring" on page 359.

Copyright © 1998-2015 No Magic, Inc.

Command (Shortcut key)	Description
Tools	Opens a list of tools that are available for the selected element. For more information, see "Tools" on page 434.
Up (Ctrl+Up Arrow)	Moves selected rows (either grouped or non-grouped) up. The command is available on the shortcut menu of one or more rows. For more information, see "To shift rows up or down" on page 775.
Down (Ctrl+Down Arrow)	Moves selected rows (either grouped or non-grouped) down. The command is available on the shortcut menu of one or more rows. For more information, see "To shift rows up or down" on page 775.
Left (Ctrl+Left Arrow)	Moves selected columns (either grouped or non-grouped) left. The command is available on the shortcut menu of one or more columns. For more information, see "To shift columns left or right" on page 776.
Right (Ctrl+Right Arrow)	Moves selected columns (either grouped or non-grouped) right. The command is available on the shortcut menu of one or more columns. For more information, see "To shift columns left or right" on page 776.
Sort Order	Point to the command and select the order for sorting rows or columns. It can be either **Ascending** or **Descending**. Selecting **Custom** does not affect the order of rows/columns. The **Custom** order is automatically selected after shifting at least one row or column to from one place to another. The sort order persists after changing the row or column owner's display mode (see the Row Owner Display Mode or Column Owner Display Mode property description).
Remove from Matrix	Removes the selected element from the matrix.
Delete	Deletes the selected element from both the matrix and the model. **NOTE** If you undo the deletion, you should refresh the matrix to make the deleted elements appear in the matrix again.

Dependency Matrix Specification window

See the descriptions of the dependency matrix Specification window properties in the following table.



Some properties are available only in the Expert mode. Hence if you do not see a desired property, change the property display mode.

Property name	Description
Dependency Matrix	
Direction	Select a direction of relationships for the relationship analysis and representation in the Dependency Matrix. Be aware that new relationships will be created with the selected direction.
Dependency Criteria	Specify what relationships between row and column elements you need to display in the dependency matrix cells. For more information, see "To specify a dependency criteria" on page 768.
Show Elements	Select to show only related (by a selected dependency criteria), only non-related, or all elements. For more information, see "To specify what elements you wish to see on the matrix: related, non-related, or all" on page 769.

Copyright © 1998-2015 No Magic, Inc.

Property name	Description
Show Inner Dependencies	Set to *true* to show number of relationships in every owning element cell.
Suppress Criteria Area	Set to *true* to suppress the **Criteria** area toolbar.
Read Only	Set to *true* to make the matrix read-only. You will not be able to create or delete relationships.
Description Area	Type the description for the dependency matrix. The text will be displayed in the description area of the matrix.
Hide Types	Set to *true* to hide the **Row Element Type** and **Column Element Type** boxes from **Criteria** area.
Hide Scope	Set to *true* to hide the **Row Element Scope** and **Column Element Scope** boxes from **Criteria** area.
Hide Dependency Criteria	Set to *true* to hide the **Dependency Criteria** and **Direction** boxes from **Criteria** area.
Column	
Column Owner Display Mode	Select the **Compact tree** mode to display elements with their direct and common owners in the column header. The data will be represented as a tree. Select the **Complete tree** mode to display elements with all their owners in the column header. The data will be represented as a tree. Select the **Hidden** mode to display elements without any owners in the column header. The data will be represented as a list. Select the **Full qualified name** mode to display elements with their owners in the column header. The data will be represented as a list.
Column Text Direction	Specify the direction of the text in column header. Be aware that this property value can be applied only when the **Column Owner Display Mode** property value is **Hidden** or **Full qualified name**.
Column Element Type	Specify element types to show in the columns of the dependency matrix. For more information, see "To specify a row/column element type" on page 766.
Column Scope	Specify what elements you wish to see in the columns of the dependency matrix. For more information, see "To specify a row/column scope" on page 767.
Column Property Filter	Select properties and their values to create more specific filter for column elements.
Removed Column Elements	Select elements to exclude from the relationship analysis scope in the matrix columns. If a selected element owns other elements, they will be excluded from the matrix as well.
Column Header Height	Specify the height of column header in pixels.
Column Types Include Subtypes	Set to *true* to display subtypes of selected element types. For example, if a class is selected, then all its subtypes, such as component or custom subtypes like SysML block and requirement will be displayed. For more information, see "To specify a row/column element type" on page 766.
Row	

Copyright © 1998-2015 No Magic, Inc.

Property name	Description
Row Owner Display Mode	Select the **Compact tree** mode to display elements with their direct and common owners in the row header. The data will be represented as a tree. Select the **Complete tree** mode to display elements with all their owners in the row header. The data will be represented as a tree. Select the **Hidden** mode to display elements without any owners in the row header. The data will be represented as a list. Select the **Full qualified name** mode to display elements with their owners in the row header. The data will be represented as a list.
Row Element Type	Specify element types to show in the rows of the dependency matrix. For more information, see "To specify a row/column element type" on page 766.
Row Scope	Specify what elements you wish to see in the row of the dependency matrix. For more information, see "To specify a row/column scope" on page 767.
Row Property Filter	Select properties and their values to create more specific filter for row elements.
Removed Row Elements	Select elements to exclude from the relationship analysis scope in the matrix rows. If a selected element owns other elements, they will be excluded from the matrix as well.
Row Header Width	Specify the width of row header in pixels.
Row Types Include Subtypes	Set to *true* to display subtypes of selected element types. For example, if a class is selected, then all its subtypes, such as component or custom subtypes like SysML block and requirement will be displayed. For more information, see "To specify a row/column element type" on page 766.

Generic Table



This feature is available in Standard, Professional, Architect, and Enterprise editions.

A generic table displays a list of selected elements and their properties in the single place. With the help of generic table you can do the following:

- Edit property values of the listed model elements.
- Create selected type elements.
- Create new derived properties.



For more information, see "Derived Properties" on page 537.

Copyright © 1998-2015 No Magic, Inc.

#	Name	PreCondition	Goal	Basic Flow of Events
1	Create Course	The user has permmisions to create/modify.	Create a new course	1. Type the course information 2. Select the class where the course take place 3. Specify the schedule of the course
2	Create Class	The user has permmisions to create/modify.	Create a new class	1. New class creation form is opened 2. Enter all class information (class name, schedule)
3	Create User	The user has permmisions to create/modify.	Create a new user	1.Type the information of the user 2. Select the role type 3. Specify the permissions for the new user
4	Modify User	The user has permmisions to create/modify.	Modify selected user	1. Open the user information form 2. Modify the information of the selected user
5	Modify Course	The user has permmisions to create/modify.	Modify information of the selected course	1. Open course list 2. Select course 3. Edit information of the selected course
6	Modify Class	The user has permmisions to create/modify.	Modify selected class	1. Open class list 2. Select class 3. Edit information of the selected class

Figure 477 -- Example of generic table



You can also read about the generic table feature and analyze given examples within the Generic table project sample.

To open the sample, do either:

- On the Welcome screen, select **Samples** > **Diagrams** > **Generic table**.
- Go to <*MagicDraw installation directory*>*samples**diagrams* and open the *generic table.mdzip* file.

All the examples given in this section are based on the data from this sample.

Creating Generic Tables

You could create a generic table in one of the ways that are suitable for creating any diagram in MagicDraw (see the procedure "Opening Diagrams", which is described in "Creating Diagrams" on page 198).

This section describes several most popular ways of creating a generic table:

- **Creating a generic table without help of any wizard.** Once created the table can be filled with data using the generic table environment capabilities.

  For more information see "To create a generic table without help of any wizard" on page 787.

- **Creating a generic table with help of the diagram creation wizard.** Once created the table is already filled with data.

  For more information see "To create a generic table with help of the diagram creation wizard" on page 789.

- **Creating a generic table for a set of selected elements using the diagram creation wizard.** Note that element types, elements, and columns of the generic table will be selected automatically according to the set of selected elements.

  For more information, see "To create a new generic table for a set of selected elements" on page 791.

Copyright © 1998-2015 No Magic, Inc.

When creating a generic table you should specify the following:

1. The name of a table and the owner package should be denoted first of all. This is mandatory.
2. Element types that will be used in the table should be defined. This is mandatory.
3. Elements corresponding to the element types should be selected. This is optional.
4. Table columns should be specified. Columns represent element properties. This action is also optional. The creation of the table could be finished without specifying columns.

You can modify the table after it is created.



For more information, see "Modifying generic table" on page 794.

To create a generic table without help of any wizard

1. In the Containment tree, select an element that can be the owner of the generic table.
2. Do one of the following:

   - From the main menu, select **Diagrams** > **Create Diagram**. Type “gen” and press **Enter**.
   - On the main toolbars, click the **Create Diagram** button. Type “gen” and press **Enter**.
   - Press **Ctrl**+**N**. Type “gen” and press **Enter**.
   - Right-click the element and from the shortcut menu select **Create Diagram** > **Other Diagrams** > **Generic Table**.

   The newly created generic table opens on the right side of the application window.
3. Type the name of the table.
4. Select element types.

Copyright © 1998-2015 No Magic, Inc.

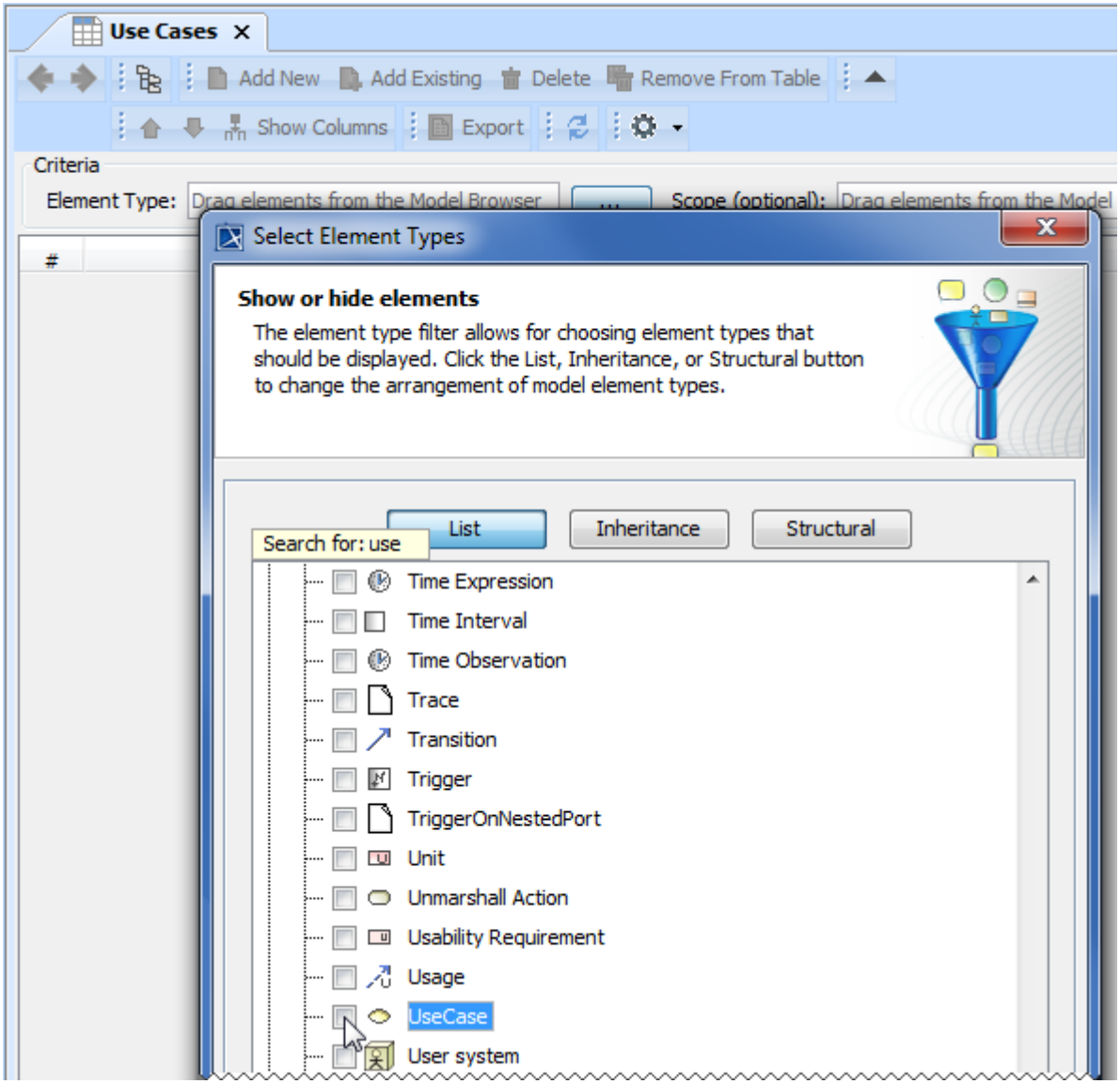


Figure 478 -- Selecting element types for generic table

Select desired element types and/or stereotypes in the **Select Element Types** dialog. The **Element Type** box in the **Criteria** area of the generic table will display the selected items. This action is optional, so you can continue the creation without selecting any type or stereotype. You will be able to modify selected element types when the generic table is created.

TIP!

If you drag an element from the Containment tree to the table, the type of this element will be automatically set as the **Element Type** box value.

5. Add elements to the table in the following ways:
   - Specify the scope, from which the elements should be added to the table. All the elements afterwards created in that scope will be automatically added to the table.
   - Create new elements directly in the table.

Copyright © 1998-2015 No Magic, Inc.

- Add existing elements to the table one by one.

TIP!

For more information about

- Defining a scope, see "To specify the scope for the generic table" on page 795
- Creating a new element, see "To add a new element" on page 796
- Adding already existing element, see "To add an existing element from the model" on page 796

6. Specify the set of columns for showing on the table.

TIP!

For more information about specifying columns, see "To add or remove columns" on page 798.

To create a generic table with help of the diagram creation wizard

1. Open the **Generic Table Wizard** dialog. It could be done in one of the following ways:
   - On the **Diagrams** menu, click **Diagram Wizards** > **Generic Table Wizard**.
   - On the **Analyze** menu, click **Model Visualizer**.
2. Type the name of the table and select the owner for it.

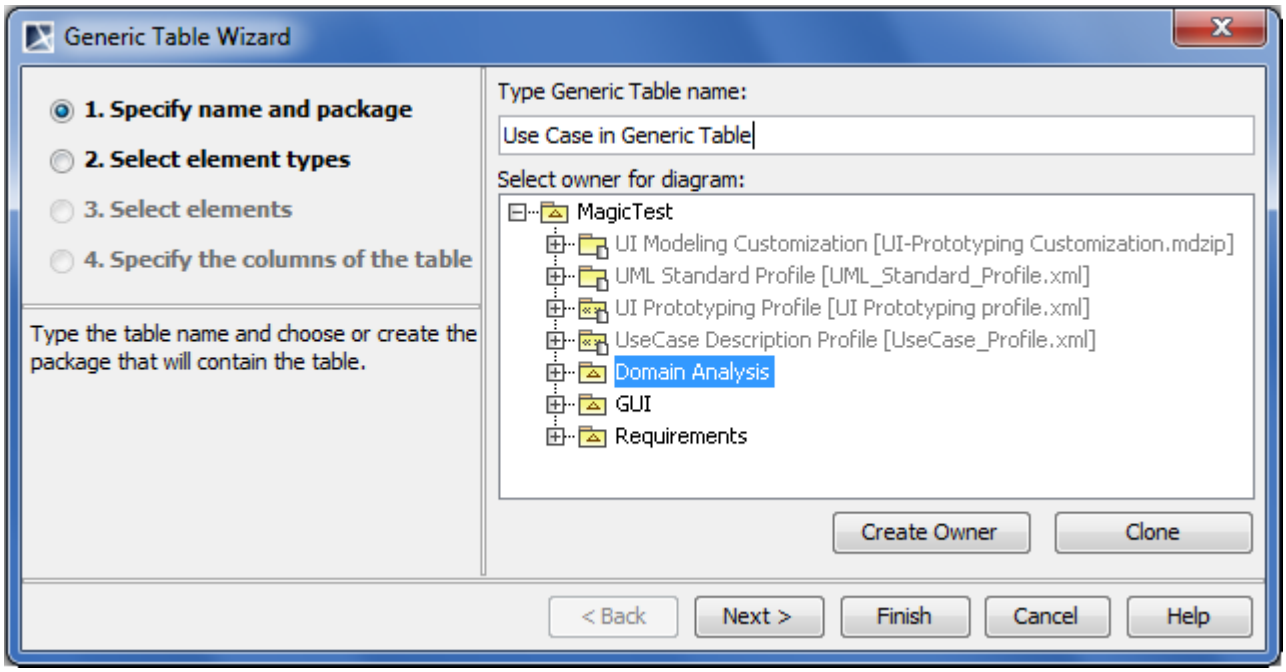


Figure 479 -- Generic table creation wizard. Specifying name for new generic table and selecting its owner

Copyright © 1998-2015 No Magic, Inc.

3. Select element types.

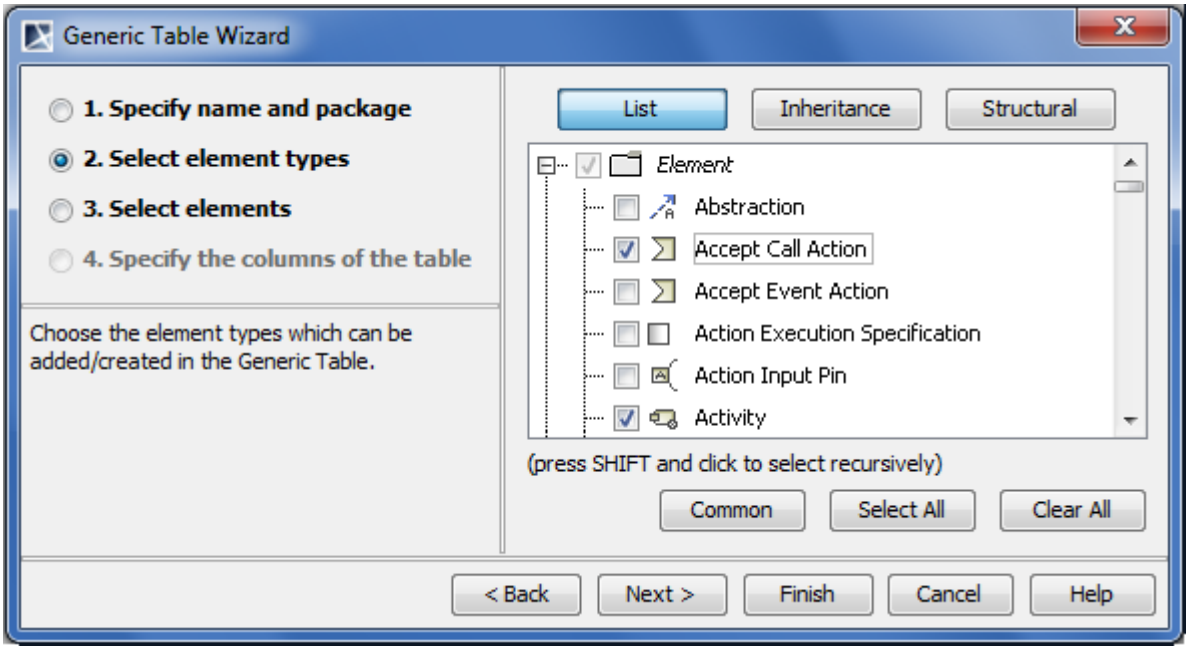


Figure 480 -- Generic table creation wizard. Selecting element types

TIP!

For the instruction on how to select element types see step #3 of the procedure "To create a generic table without help of any wizard" on page 787.

4. Select elements.

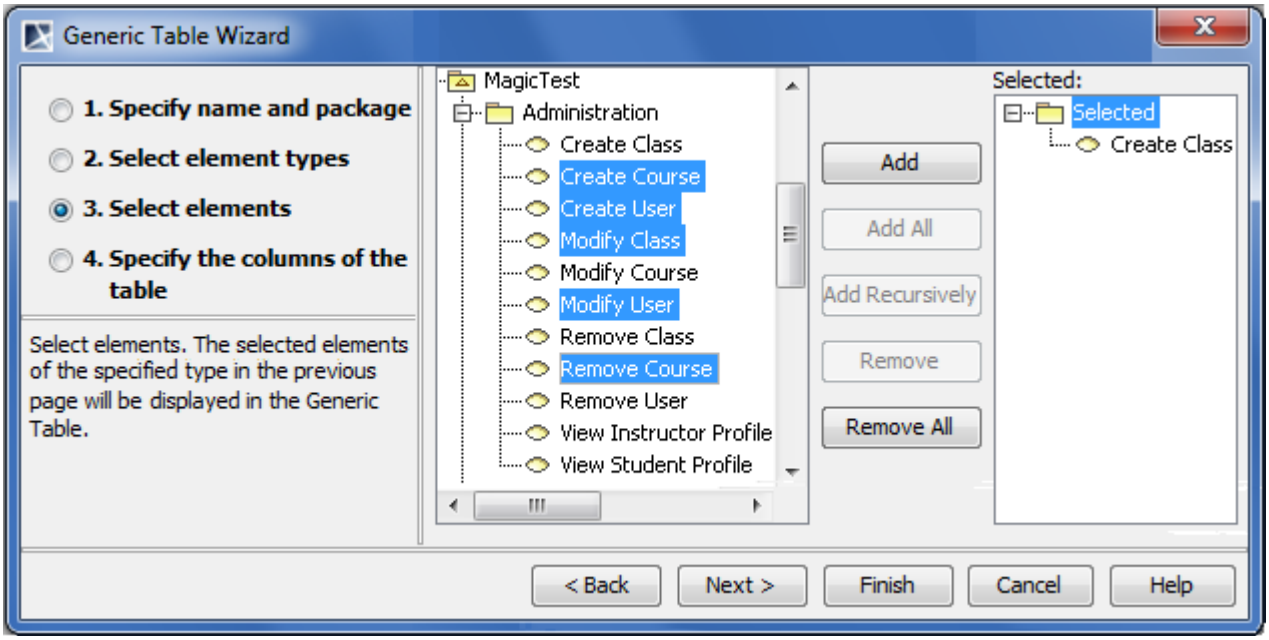


Figure 481 -- Generic table creation wizard. Selecting elements

Select elements from the model tree and add them into the **Selected** list.

NOTE

Thought you can select any element from the Model Browser, only the elements of selected types will be added into the generic table.

If the generic table creating wizard is opened from one or more elements' shortcut menu, the element or the set of elements will be added to the **Selected** list automatically.

Copyright © 1998-2015 No Magic, Inc.

TIP!

For more information about the element selection and dialog buttons see the section "Elements Multiple Selection" on page 354.

5. Specify the set of columns for showing on the table.

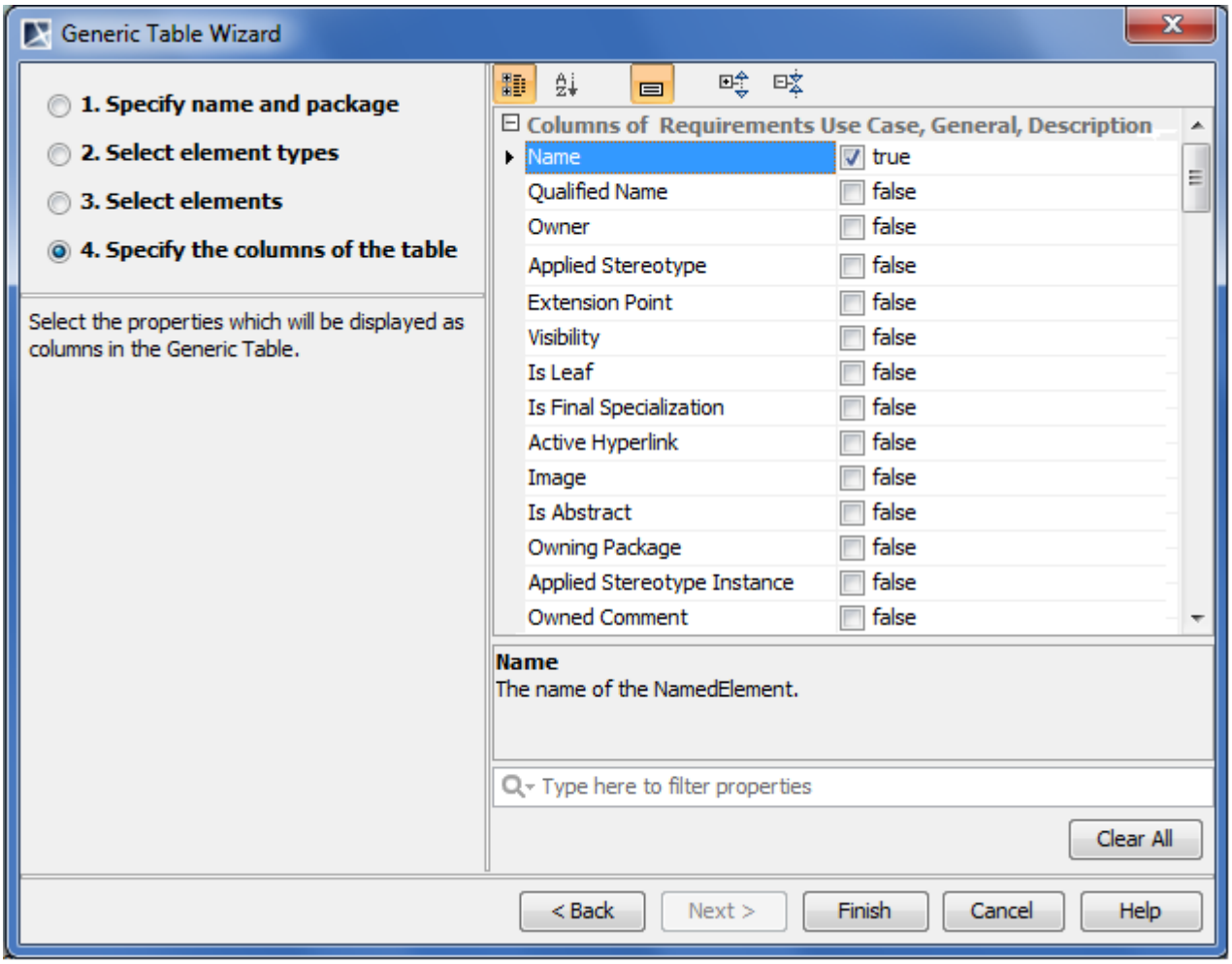


Figure 482 -- Generic table creation wizard. Selecting columns

The list of all available properties corresponding to selected element types is presented in the dialog. If selected element types have tags, they are also displayed in the list. All selected properties will be displayed as columns in the generic table. The **Name** property is selected automatically, all other properties should be selected by the user.

Select properties you need to see as columns to finish the table.

To create a new generic table for a set of selected elements

1. Select a set of elements you want to add to the generic table.
2. From the shortcut menu select **Tools** > **Generic Table Wizard**.
3. Follow the steps described in the procedure "To create a generic table with help of the diagram creation wizard" on page 789 but note that element types, elements, and columns will be selected automatically according to the set of selected elements.

NEW! To create elements in the table from another resource

1. Create a table.
2. Select element type.

Copyright © 1998-2015 No Magic, Inc.

3. Choose the columns that you want to fill with copied information.



The cells must be editable in the generic table.

4. Copy table from another resource.



- The number of columns in that table should be the same as the number of columns in the generic table
- The data types of columns in the generic table must be compatible with copied information.

5. Press Ctrl+V (Cmd+V on OS X).

Each row displays one element with its properties.



If element already exists in the model, the information updates.

Copyright © 1998-2015 No Magic, Inc.

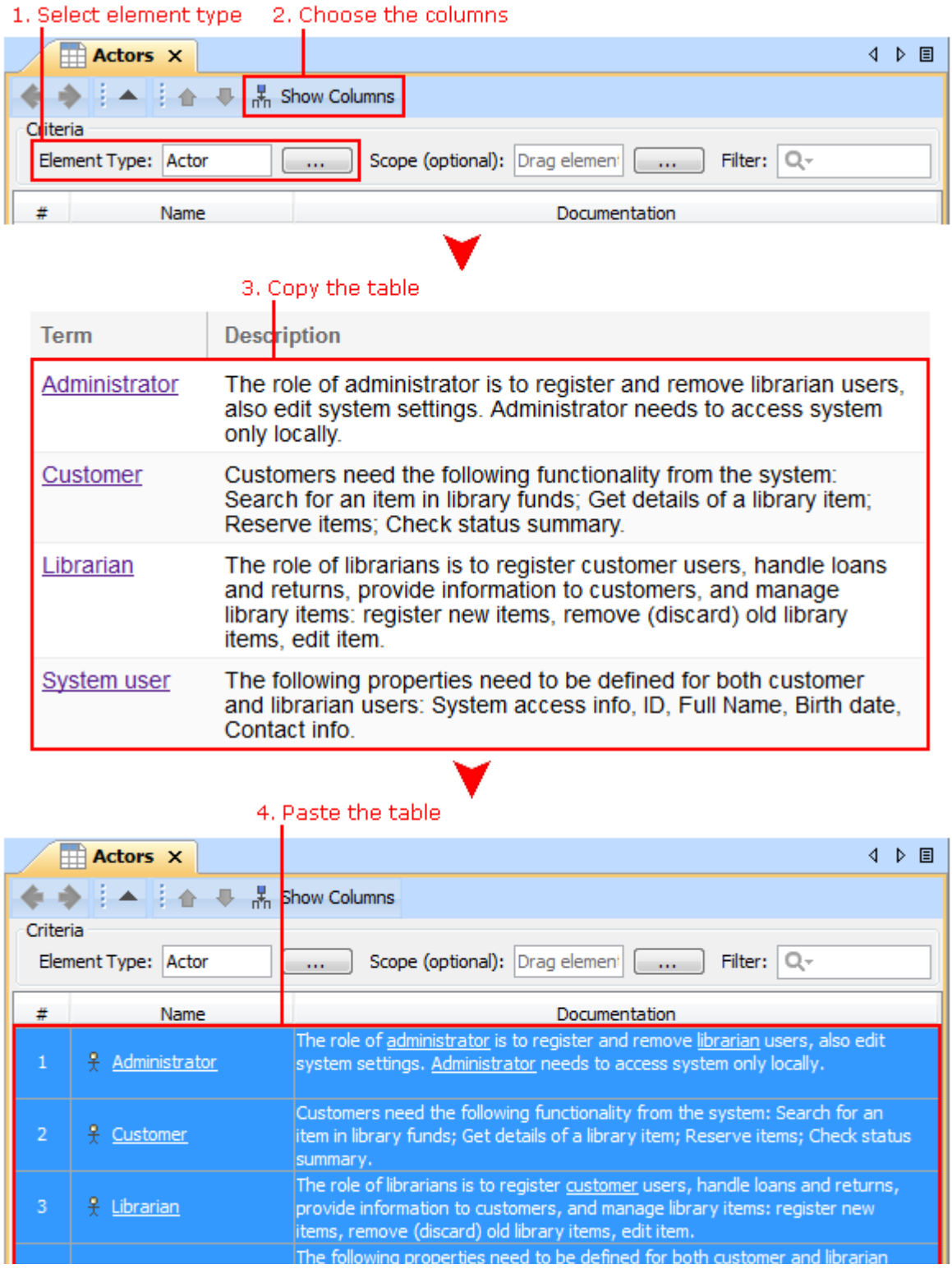


Figure 483 -- Creating elements in generic table from another resource

Using Generic Tables

There is an ability to modify a generic table after it is created. Features for a table modification and working with a generic table are described in the following subsections:

- "Setting detailed column names" on page 794

Copyright © 1998-2015 No Magic, Inc.

- "Modifying generic table" on page 794
- "Using Quick Filter" on page 798
- "Exporting generic table" on page 799
- "Manipulations in generic table" on page 800

Setting detailed column names

Column names in the generic table header are set automatically, and they are element property names. You cannot change column names.

If an element has some properties with the same name (for example, in associations or if a tag of stereotypes is selected as a column), the detailed column names could provide more precise information.

To see detailed column names

- On the Options toolbar, click and then select **Show Detailed Column Name**.

  The group name (between brackets), to which the selected property belongs, and/or the stereotype name (just before the property name) will be displayed in the column heading.

The following picture gives the example of three different columns with the same name for the association element: **Name**, **Name** (Role of A), and **Name** (Role of B) and a column for the class element with the stereotype «Teacher».**Name** (Tags). If the command **Show Detailed Column Name** were not selected, there would be four columns with same headings, i.e., **Name**.

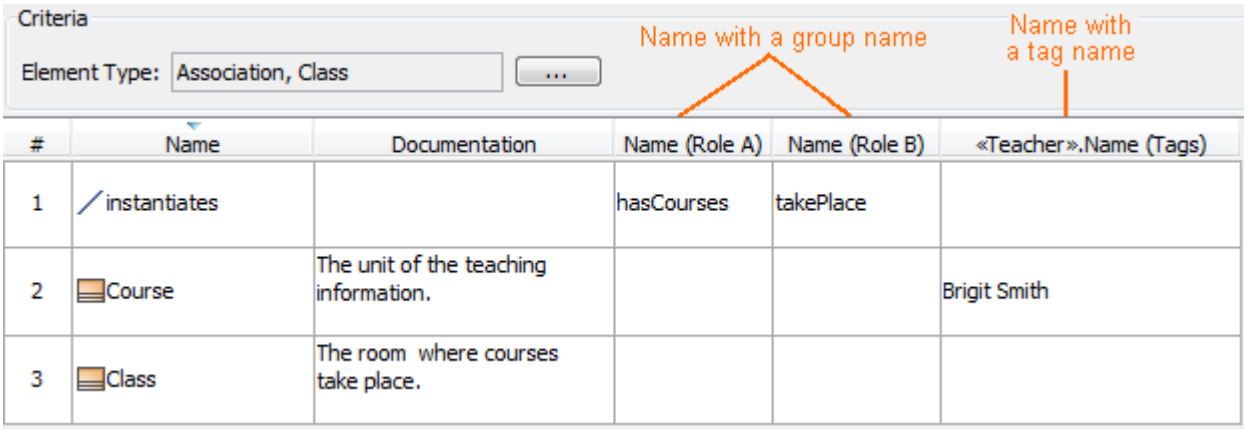


Figure 484 -- Example of detailed column names in generic table

Modifying generic table

To sort data

- Click the header of the column by which you want to sort table data. A small arrow appears on that column header. This arrow shows, how records are sorted: ascending or descending.

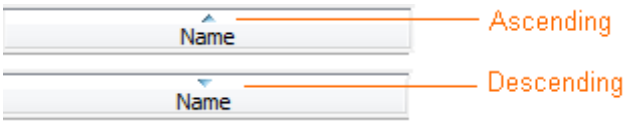


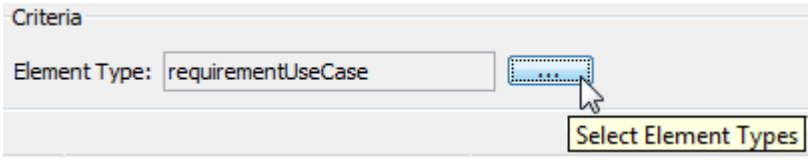
Figure 485 -- Orders for sorting data in generic table

Rows are renumbered automatically after the sorting.

Copyright © 1998-2015 No Magic, Inc.

To add or remove element types

1. In the **Criteria** area, click the **...** button next to the **Element Type** box. The dialog with the element and stereotype list opens.



2. Do one of the following:
   - To add element types select the appropriate check boxes in the list.

IMPORTANT!

Properties corresponding to the selected types will be added to the list of available columns.

   - To remove element types click to clear appropriate check boxes in the list.

IMPORTANT!

Properties corresponding to the deselected types will be removed from the list of available columns.

TIP!

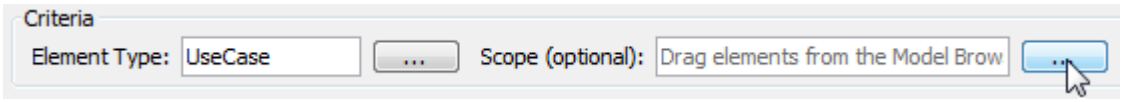
To add an element type, you can also drag a selected element to the **Element Type** box directly from the Model Browser.

- To select *adjacent* elements, select a single element and then hold down the Shift key while you click other elements.
- To select *nonadjacent* elements, select a single element and then hold down the Ctrl key while you click other elements.

The new value of the **Element Type** box replaces the old value.

To specify the scope for the generic table

1. In the **Criteria** area of the generic table, click the **...** button next to the **Scope (optional)** box.



The **Select Scope** dialog opens.

2. In the tree or list on the left side of the dialog, select one or more containers, for example, packages or smart packages, and add them to the **Selected Elements** list.

TIP!

The **Select Scope** dialog is a modification of the element Selection dialog. For more information about the manipulations in the dialog, see "Elements Multiple Selection" page 354.

3. Click **OK** to both confirm the addition and close the dialog.

The scope is defined, and the contents of the generic table becomes updated with the elements from the selected scope. In future, all the elements created in that scope will be automatically added to the table.

IMPORTANT!

If the contents of the table has not been updated, try the following:

- Click  on the View toolbar. This helps in case a smart package is specified as the scope criteria.
- Make sure the specified scope contains elements, which types are specified in the **Element Type** box. Element types must correspond to the selected element types!
- Make sure at least one element type is specified for the table.

Copyright © 1998-2015 No Magic, Inc.

TIP!

To define a scope, you can also drag a selected container to the **Scope (optional)** box directly from the Model Browser (see the following figure).

- To select *adjacent* elements, select a single element and then hold down the Shift key while you click other elements.
- To select *nonadjacent* elements, select a single element and then hold down the Ctrl key while you click other elements.

The new value of the **Scope (optional)** box replaces the old value.

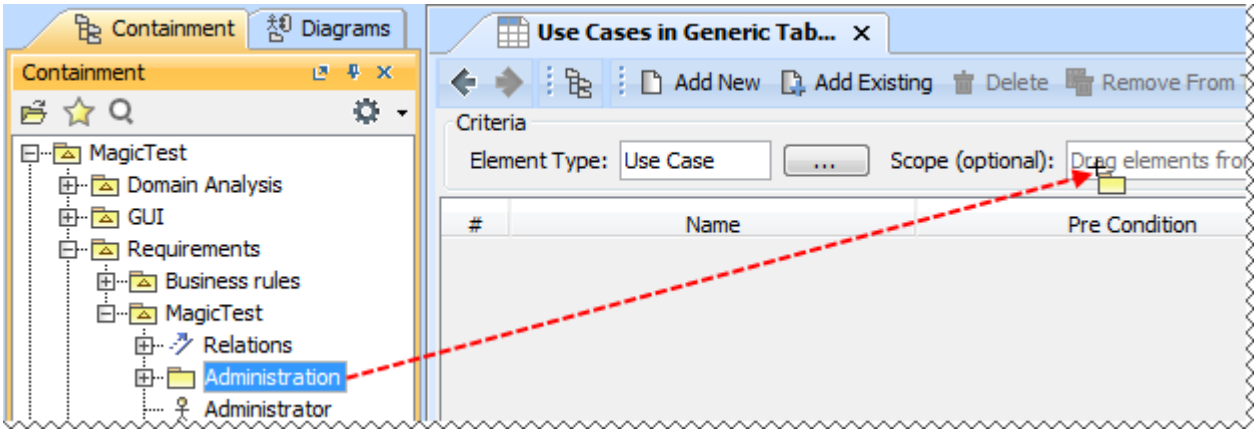


Figure 486 -- Using drag-and-drop operation to specify scope for generic table

To add a new element

IMPORTANT!

At least one element type must be selected in the **Element Type** box before adding a new element.

1. Click the **Add New** button on the Edit toolbar. If there is more than the one element type selected, a submenu with the list of available element types opens.

   NOTE

   Only element types available to create in a possible diagram owner which contains the generic table will be displayed on the submenu.

2. Select an element type. The element of the selected type will be added to the last row of the table and to the model.
3. Name the new element in the table.

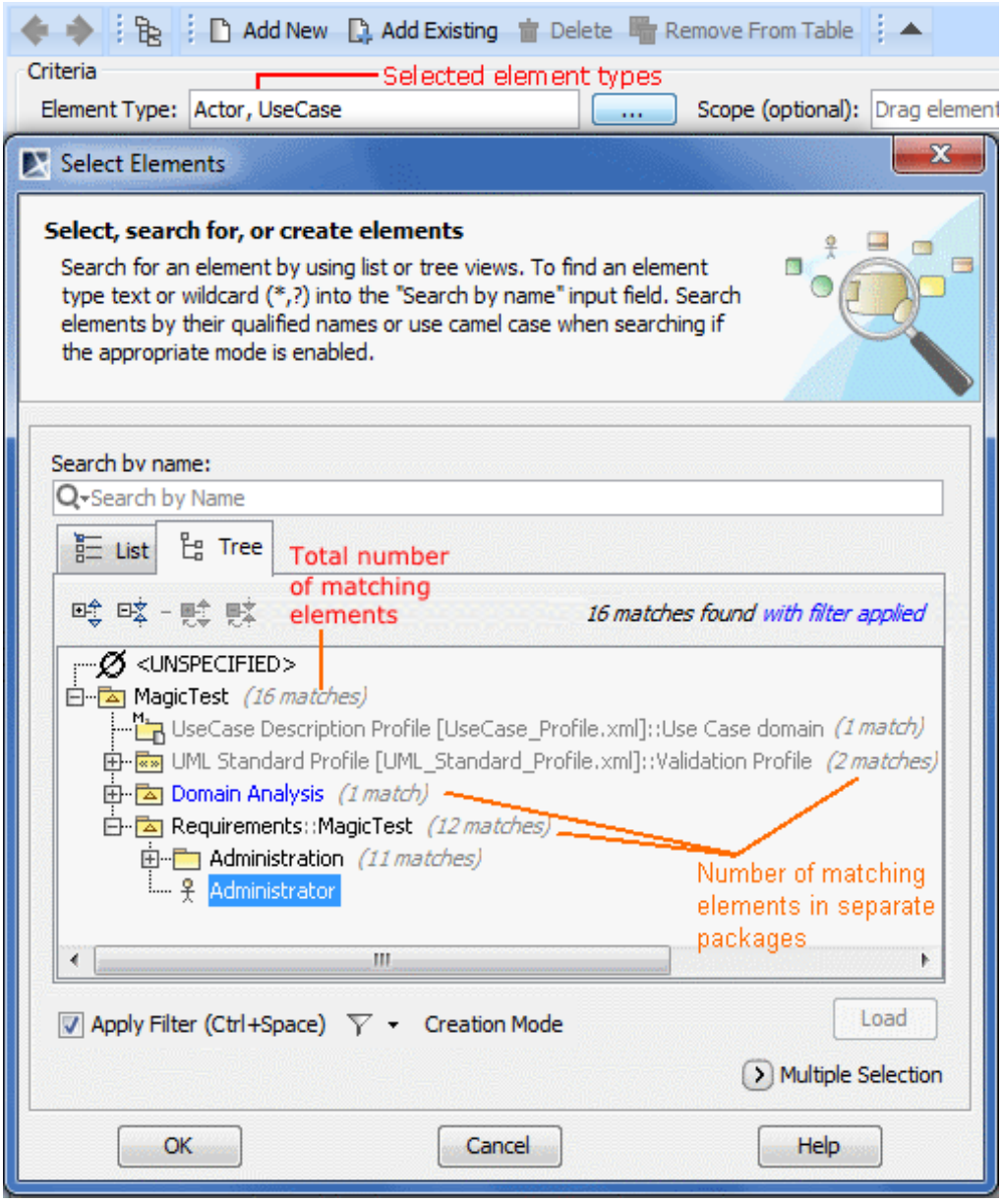
Copyright © 1998-2015 No Magic, Inc.

To add an existing element from the model

IMPORTANT!

Elements you want to add should correspond to selected element types.

1. Click the **Add Existing** button on the Edit toolbar. The **Select Element** dialog opens.



2. Select the element you want to add to the generic table. This element and its name will be added to the table as the last row.

TIP!

Use the **Multiple Selection** mode to add more than one element at a time.

NOTE

If you have not defined the element type before starting to add elements to the table, it is automatically set after the first element is added to the table. The type of this element becomes the value of the **Element Type** box.

TIP!

To add an existing element, you can also drag a selected element to the contents of the generic table directly from the Model Browser.

If you drag a container, for example, a package or a smart package, the generic table will display its contents, but not the container itself.

Copyright © 1998-2015 No Magic, Inc.

To add or remove columns

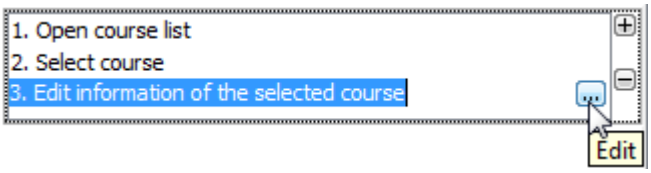
1. To open the available columns list:
   - On the Layout toolbar, click **Show Columns** to open the submenu with common properties corresponding to element types that have been selected for the generic table.
   - On the Layout toolbar, click **Show Columns** > **Select Columns** to open the dialog with all properties, including tags of stereotypes corresponding to selected element types.

   NOTE Using the **Select Columns** dialog, you can choose to display not only the properties of the specified element type, but also the properties that are subtypes of the specified element type.
2. Do either:
   - Select the properties you want to see as columns in the table.
   - Unselect the properties you do not want to see as columns is the table.

To edit element's property in a cell

NOTE The property can be edited in the element's Specification window if it is not locked.

1. Click a cell you want to edit.



2. Do one of the following:
   - Edit the value directly in the selected cell.
   - Click the **...** button. The property value editor opens. For more information about various property value editors, see "Editing Property Values" on page 298.

IMPORTANT! All changes made in the generic table are saved in the model.

Using Quick Filter

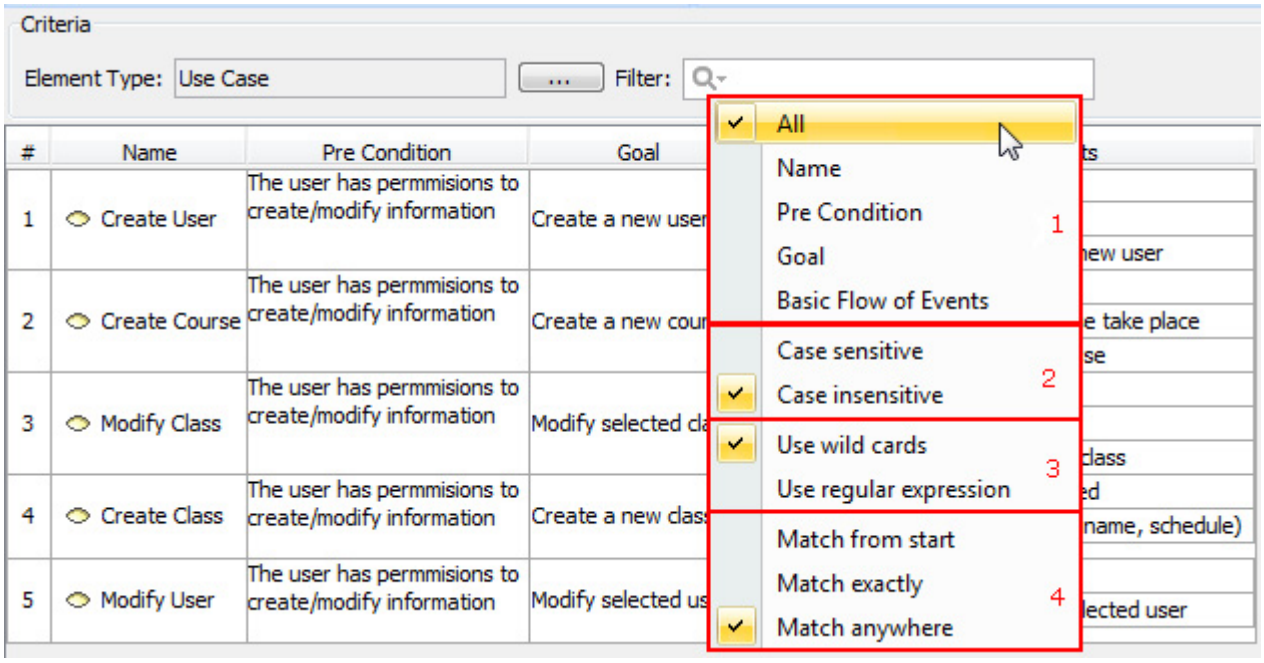
Using the Quick filter box you can quickly find the required rows in the table. This is especially handy when you are working with a large table containing many rows and columns. Rows can be filtered by the text in the **Filter** box.

When using a Quick Filter, you can specify several criteria:

- Specify columns wherein you want to search (1).

  NOTE You can search in the entire table or select one or several columns.
- Specify case sensitivity (2)
- Specify if you want to use a wild card or regular expression (3)
- Specify how the results should match your key words (4)

Copyright © 1998-2015 No Magic, Inc.



You can save the filter criteria for a particular table. Even after reloading the project or restarting MagicDraw, the filter criteria will remain saved.

To save the filter criteria

- On the Options toolbar, click and then select **Save Filter Criteria**.

NOTE

When exporting the table, only the filtered out columns and rows will be included in the result.

Related concepts

Quick Filter

Exporting generic table

You can export the selected generic table to an **.html*, **.csv,* or **.xlsx* file.

To export a generic table

1. On the Publish toolbar, click **Export**.
2. In the opened dialog, do the following:
    2.1. Browse for a location to save a table.
    2.2. Type the exported table name.
    2.3. Select the exported table format.
3. Click **Save** when you are done. The table is exported to the location you have specified.

The style of **.html* tables is specified in *template.html* which can be found in *<MagicDraw installation directory>\data\table*. The file stores cascading style sheets (CSS) that define the appearance of an exported generic table. Before exporting a table, you can customize the *template.html* according to your needs.

Copyright © 1998-2015 No Magic, Inc.

To customize the t*emplate.html*

1. Open <MagicDraw installation directory>\data\table\template.html for editing.
2. Edit cascading style sheets (CSS) according to your needs.
3. Save changes.

Manipulations in generic table

To open the element's Specification window

Do either:

- Double-click a non-editable cell in the element's row, if there is such cell.
- Right-click on a cell and from the shortcut menu select **Specification**. Then:
  - If the selected cell does not refer to any element, the Specification window of the element in the row will open.
  - If the selected cell refers to some elements, the submenu with the list of appropriate elements will open. Choose the element whose Specification window you want to open.

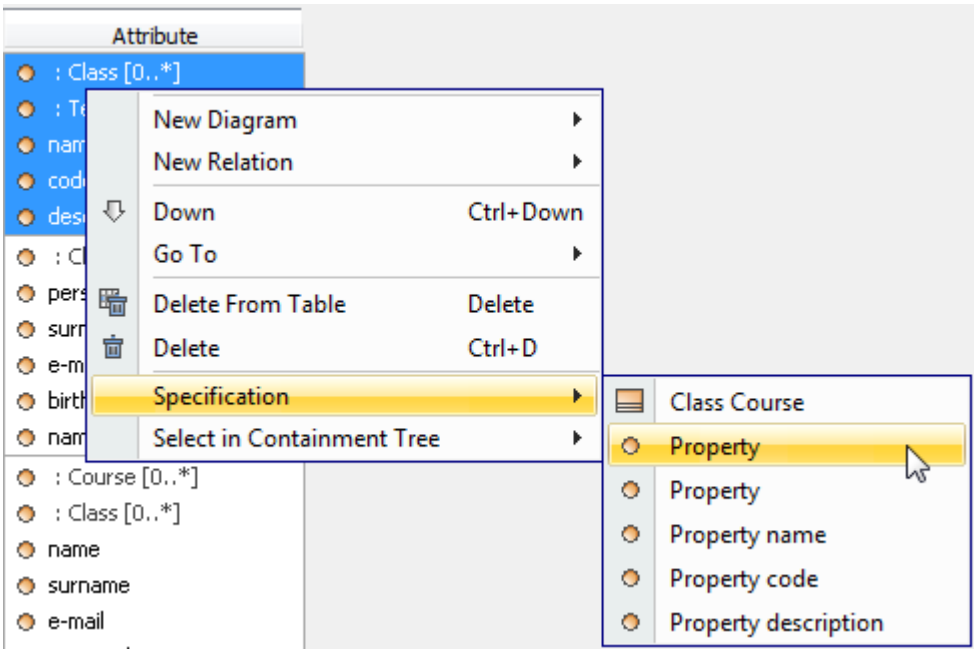


Figure 487 -- Submenu of Open Specification command

To reorder the columns

- Click the column header and drag it to a desired place. Only the first column cannot be moved.

Generic Table Environment

The generic table environment elements, such as toolbars and the **Criteria** area, are described in this section:

- "Table toolbars" on page 801
- "Criteria area" on page 802

Copyright © 1998-2015 No Magic, Inc.

Table toolbars

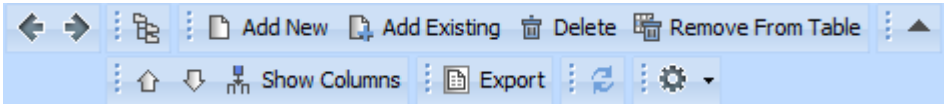


Figure 488 -- Generic table toolbars

NOTE If the generic table toolbars are not available in a server project, try to lock the table for edit (make sure you have the right to edit model of this project).

The following table describes the generic table toolbar buttons.

Button	Shortcut keys	Description
Navigation toolbar:		
	Alt+B	Click to select in the Containment tree the element of the selected row.
Edit toolbar:		
Add New	Insert	Click to create a new element in the table. This element will be added to the model also. If there are available several element types, a shortcut menu with available types will open. You will be able to select an element of the type you need. For more information, see the procedure "To add a new element" on page 796.
Add Existing	Ctrl+Insert	Click to add an element from a model. The **Select Element** dialog will open. Only elements of the type defined in a generic table will be listed. For more information, see the procedure "To add an existing element from the model" on page 797.
Delete	Ctrl+D	Click to remove selected elements from both the table and the model.
Remove From Table	Delete	Click to remove selected elements only from the table.
Layout toolbar:		
Up	Ctrl+Up Arrow	Click to shift selected elements (either grouped or non-grouped) up a row. The elements are automatically renumbered after moving.
Down	Ctrl+Down Arrow	Click to shift selected elements (either grouped or non-grouped) down a row. The elements are automatically renumbered after moving.
Show Columns	N/A	Click to change the set of columns for showing on the generic table. For more information, see the procedure "To add or remove columns" on page 798.
Publish toolbar:		
Export	N/A	Click to export the contents of the generic table to an **.html*, **.csv,* or **.xlsx* file.
View toolbar:		
	N/A	Click to update the contents of the generic table after specifying the smart package as the scope criteria. For more information, see the procedure "To specify the scope for the generic table" on page 795.

Copyright © 1998-2015 No Magic, Inc.

Button	Shortcut keys	Description
Options toolbar:		
	N/A	Click to open the Options menu and then select: • **Show full paths** – to display the full paths of elements (where owner information can be displayed). • **Save Filter Criteria** – to keep the filter criteria unchanged after reloading the project or even after restarting MagicDraw. For more information about specifying the filter criteria, see "Using Quick Filter" on page 798. • **Show Detailed Column Name** – to show property group and/or stereotype names in the table header. This is helpful when the table displays several columns with the same name. For more information, see "Setting detailed column names" on page 794. • **Strip Multiline Text** – to strip the text that covers more than five rows. Three dots will be added at the end of the stripped text.
Suppress/Expand Criteria Area toolbar:		
	N/A	Click to hide the **Criteria** area.
	N/A	Click to show the **Criteria** area.
Other buttons:		
	Alt+Left Arrow	Click to navigate to the previously opened diagram, table, dependency matrix, or relation map.
	Alt+Right Arrow	Click to navigate to the previously opened diagram, table, dependency matrix, or relation map.

Criteria area

The **Criteria** area appears below the generic table toolbars. If it is not displayed, click the button on the Suppress/Expand Criteria Area toolbar.

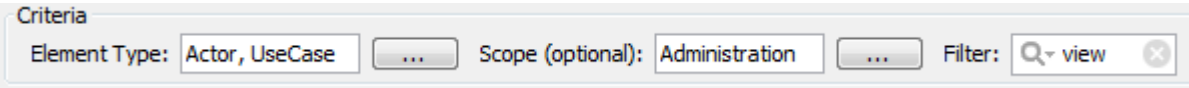


Figure 489 -- Generic table Criteria area

The following table describes the contents of the **Criteria** area

Element Name	Element Type	Description
Element Type	Text box with 	Click the **...** button to select one or more types of elements that you need to display in the generic table. For more information, see "To add or remove element types" on page 795.

Copyright © 1998-2015 No Magic, Inc.

Element Name	Element Type	Description
Scope (optional)	Text box with ...	Click the **...** button to select one or more packages that contain the elements you need to display in the generic table. For more information, see the procedure "To specify the scope for the generic table" on page 795.
Filter	Text box	Type the phrase by which the element should be filtered. To open the menu with more filter options, click in the **Filter** box. To reset the filter options, click in the **Filter** box. For more information, see "Using Quick Filter" on page 798.

Instance Table

NOTE This feature is available in Standard, Professional, Architect, and Enterprise editions.

An instance table allows for managing the instance specifications (instances) of your model in a spreadsheet-like form. Each row in the table represents an instance.

With help of this table you can easily

- Review the instances of one or more of the classifiers in a single place.
- Create instances for the classifiers.
- Edit slot values of the instances displayed in the table.
- Customize the representation of the table.
- Export the data into an **.html*, **.csv,* or **.xlsx* file.

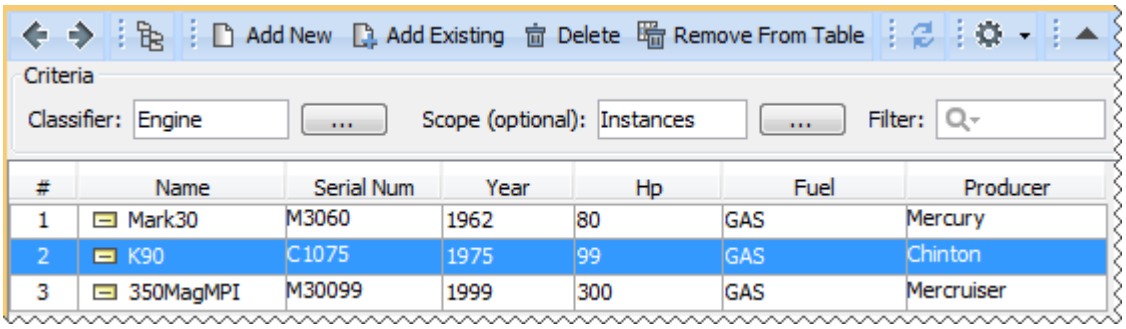


Figure 490 -- Example of instance table

Creating Instance Tables

The creation of an instance table includes

1. Creation of an empty one
2. Specification of the classifier whose instances should be displayed in the table
3. Selection of columns to display on the table
4. Filling the table with instances
5. Specifying instances as slot values (optional)
6. NEW! Creating instances in the table from another resource

Copyright © 1998-2015 No Magic, Inc.

Creating an empty instance table

To create an empty instance table

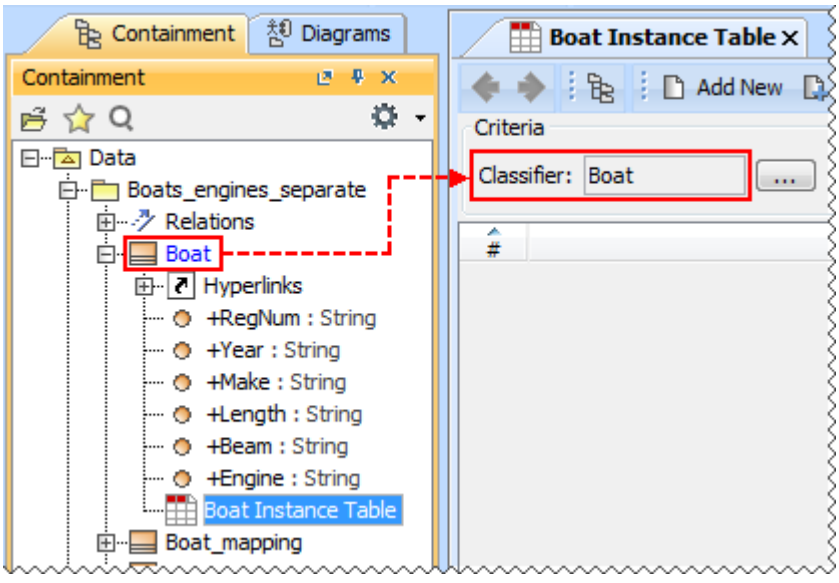
1. In the Containment tree, select an element that can be the owner of the instance table.
2. Do one of the following:

   - From the main menu, select **Diagrams** > **Create Diagram**. Type “ins” and press **Enter**.
   - On the main toolbars, click the **Create Diagram** button. Type “ins” and press **Enter**.
   - Press Ctrl+N. Type “ins” and press **Enter**.
   - Right-click the element and from the shortcut menu select **Create Diagram** > **Other Diagrams** > **Instance Table**.

The newly created instance table opens on the right side of the application window.

NOTE

If the *owner* of the instance table is also a *classifier*, it becomes the default value of the **Classifier** box in this instance table.



3. If the default name of the instance table is not appropriate, type a new one directly in the Containment tree.

TIP!

For more information about creating diagrams, refer to "Creating Diagrams" on page 198.

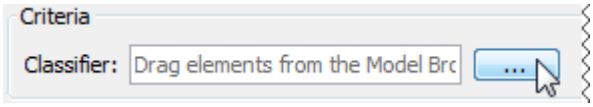
Specifying a classifier

Specifying the classifier criteria is necessary, because you will not be able to add a single instance to the table until the classifier is unknown. Hence, before starting to fill the newly created instance table with data, you must specify the classifier, whose instances you need to display in this table, unless the owner of the instance table is also a classifier (see the note in step#2 of the procedure "To create an empty instance table" on page 804).

Copyright © 1998-2015 No Magic, Inc.

To specify the classifier for the instance table

1. In the **Criteria** area of the instance table, click the **...** button next to the **Classifier** box.



The **Select Classifier** dialog opens.

2. In the tree or list on the left side of the dialog, select one or more elements and add them to the **Selected elements** list. These are the elements that can be selected:
   - Classifier
   - Classifier that owns other classifiers. In this case, the classifier and all the classifiers owned by it will be specified as criteria.
   - Package that contains classifiers. In this case, the contents of the package, but not the package itself will be specified as criteria.



The **Select Classifier** dialog is a modification of the element Selection dialog. For more information about the manipulations in the dialog, see "Elements Multiple Selection" on page 354.

3. Click **OK** when you are done to close the dialog.

Once the classifier is specified, the following columns are displayed in the instance table (on the right of the row number **#** column):

- **Name** which is up to display the names of the instances of the specified classifier.

Copyright © 1998-2015 No Magic, Inc.

- Columns corresponding to the Data type attributes of the specified classifier and named after these attributes using the sentence style capitalization. The columns are up to display the slot values of the instances of the specified classifier.

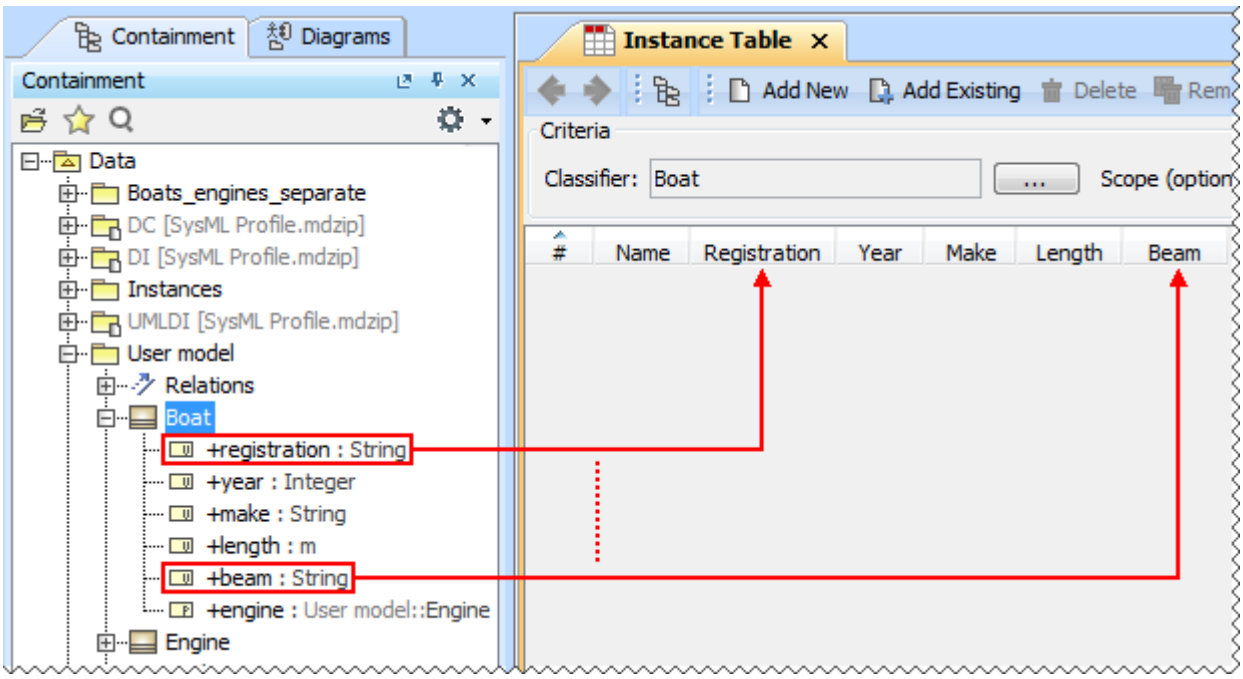


Figure 491 -- Instance table column names

NOTE

The instance table by default shows nine columns, including **#**, **Name**, and the columns that correspond to the *first seven* Data type attributes of the specified classifier. The rest of the attributes are by default hidden.

You can show the hidden columns. For more information, see "Selecting columns to display" on page 807.

TIP!

To specify the classifier, you can also drag a selected element to the **Classifier** box directly from the Model Browser (see the following figure).

- To select *adjacent* elements, select a single element and then hold down the Shift key while you click other elements.
- To select *nonadjacent* elements, select a single element and then hold down the Ctrl key while you click other elements.

Keep in mind that the new value of the **Classifier** box replaces the old value.

Copyright © 1998-2015 No Magic, Inc.

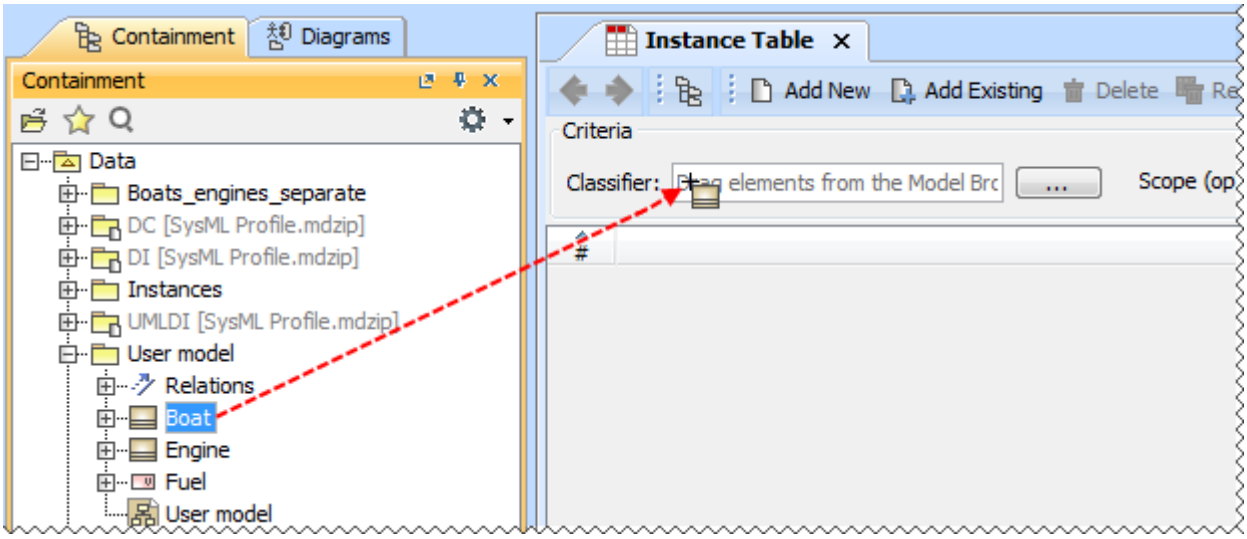


Figure 492 -- Using drag-and-drop operation to specify classifier for instance table

Selecting columns to display

The instance table by default shows nine columns, including **#** (row number), **Name** (instance name), and the columns for instance slot values, that correspond to the *first seven* Data type attributes of the specified classifier. The rest of the columns are by default hidden, though you can show them if there is a need.

In overall, you can select to show any column that corresponds to one of the following:

- Attribute of the classifier that is specified for the instance table.
- Property of the instance of the specified classifier.
- Tag of a stereotype that can be applied on the instance of the specified classifier.

To show hidden columns in the instance table

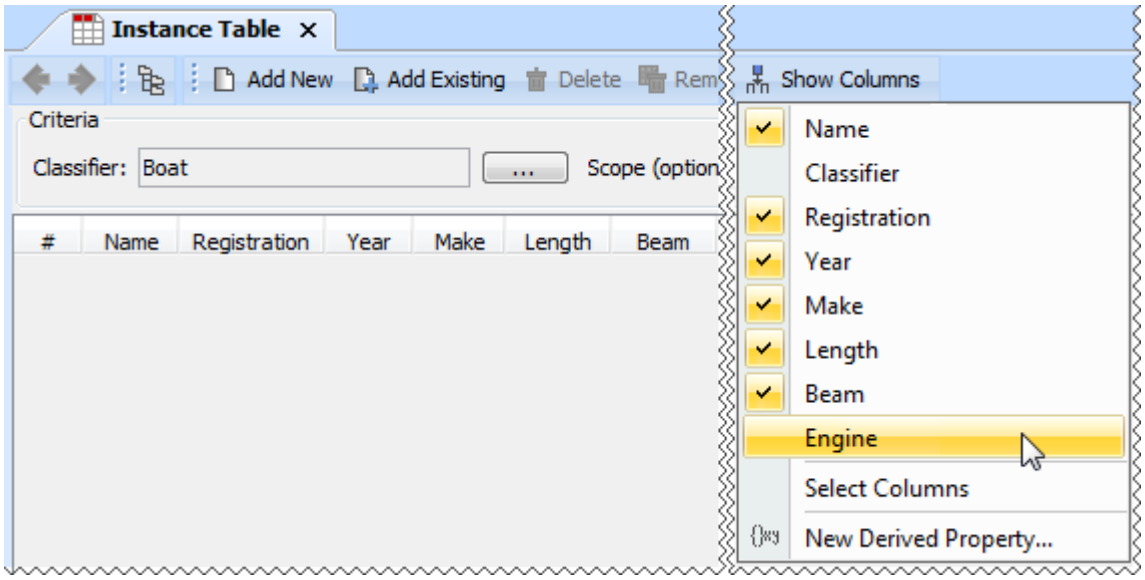
Do either:

- On the Layout toolbar, click the **Show Columns** button and then click to select the check boxes nearby the columns you need to show.



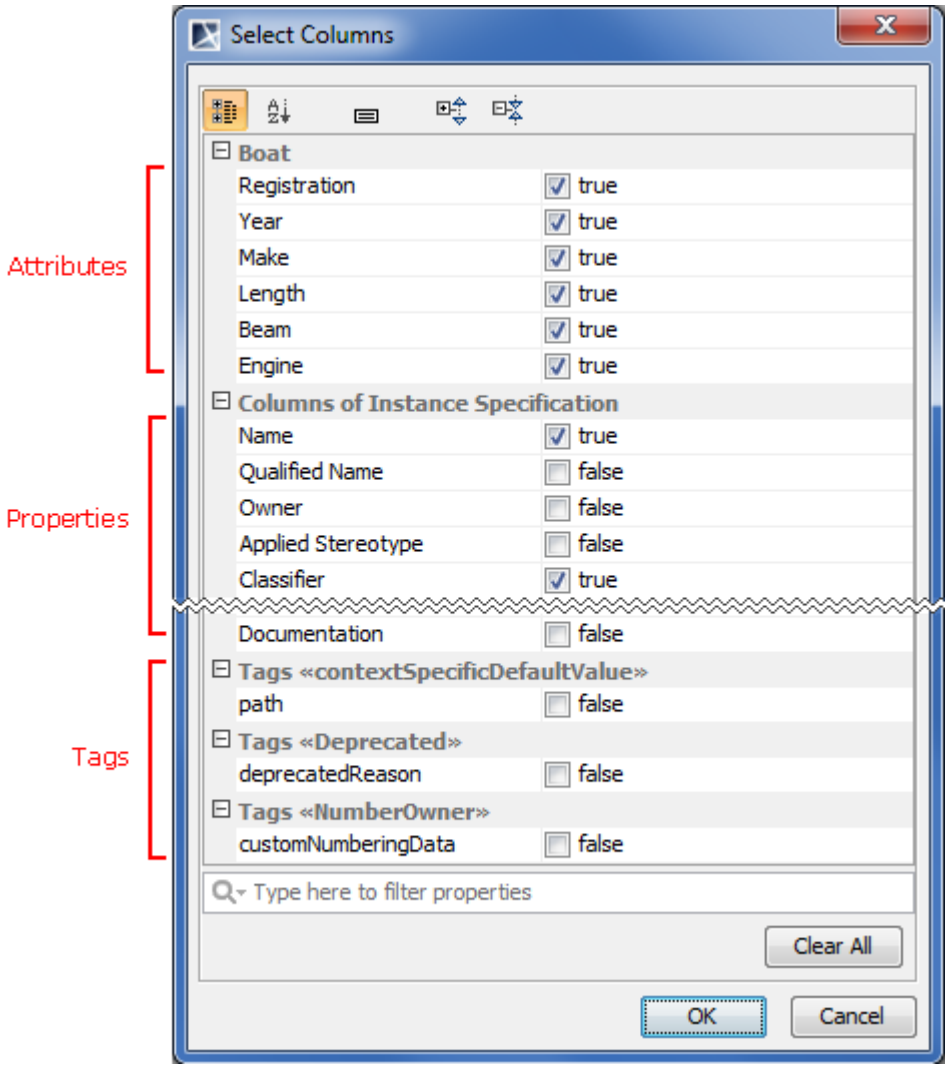
The column appears on the menu of **Show Columns** button only if it is selected in the **Select Columns** dialog.

Copyright © 1998-2015 No Magic, Inc.



Copyright © 1998-2015 No Magic, Inc.

- On the Layout toolbar, click **Show Columns** > **Select Columns** and in the open dialog, which displays the columns that correspond to above described attributes, properties, and tags, click to select the check boxes next to the columns you need to show.



NOTE

The selected to show column also appears on the menu of the **Show Columns** button.

IMPORTANT!

If you add a new attribute to the classifier and need to display the corresponding column in the instance table, open the **Select Columns** dialog and click to select the appropriate check box.

Keep in mind that columns corresponding to newly created attributes will at first be available only in the **Select Columns** dialog (until you select to show them).

Copyright © 1998-2015 No Magic, Inc.

Filling the table with instances

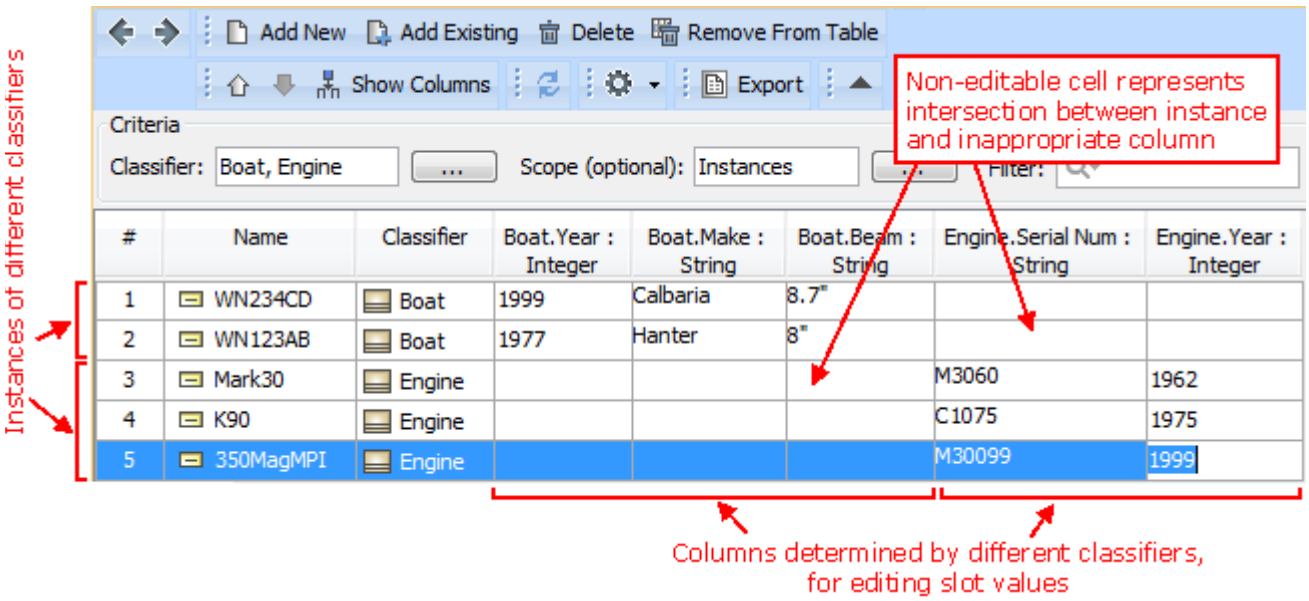
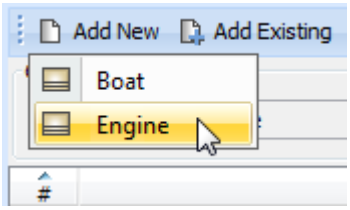


Figure 493 -- Example of instance table filled with instances of two different classifiers

To fill the instance table for the specified classifier, you can either create new instances directly in the table or add to it already created ones. Already created instances can be added one by one or you can specify a package, which contains instances. If you need the contents of your instance table to update automatically, you should specify the package as the scope criteria.

To add a new instance

1. On the Edit toolbar, click the **Add New** button. If there is more than the one classifier specified, from the menu of the button, select a classifier for that you need to create an instance.



The new instance is added to the instance table as the last row.



The new instance can be created only if the classifier is already specified for the instance table. Otherwise the **Add New** button is not available.

2. Type the instance name and specify slot values.

To add an instance from the model

1. On the Edit toolbar, click the **Add Existing** button. The **Select Instance Specification** dialog opens.
2. In the tree or list on the left side of the dialog, select an element and add it to the **Selected elements** list. Actually, you can select more than one element. Elements available for selection:
    - Instances of one or more classifiers specified for the table. An instance is available, if it belongs to at least one of the classifiers.

Copyright © 1998-2015 No Magic, Inc.

- Packages that contain these instances. In this case, the contents of a selected package, but not the package itself will be added to the table.

TIP!

The **Select Instance Specification** dialog is a modification of the element Selection dialog. For more information about the manipulations in the dialog, see "Elements Multiple Selection" on page 354.

3. Click **OK** when you are done to close the dialog.
The selected instance is added to the table as the last row.

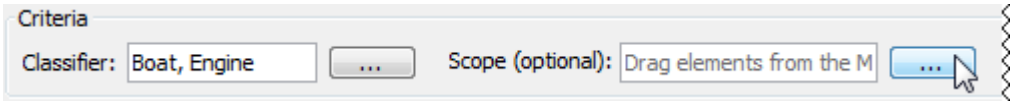
TIP!

To add an instance from the model, you can also drag a selected element to the contents of the instance table directly from the Model Browser.

- To select *adjacent* elements, select a single element and then hold down the Shift key while you click other elements.
- To select *nonadjacent* elements, select a single element and then hold down the Ctrl key while you click other elements.

To specify the scope for the instance table

1. In the **Criteria** area of the instance table, click the **...** button next to the **Scope (optional)** box.



The **Select Scope** dialog opens.

2. In the tree or list on the left side of the dialog, select one or more packages and add them to the **Selected Elements** list.

TIP!

The **Select Scope** dialog is a modification of the element Selection dialog. For more information about the manipulations in the dialog, see "Elements Multiple Selection" on page 354.

3. Click **OK** both to confirm the addition and close the dialog.
The scope is defined, and the contents of the instance table becomes updated with the elements from the selected scope. In future, the contents of the table will be automatically updated after making changes in the selected scope.

IMPORTANT!

If the contents of the table has not been updated, try the following:

- Click on the View toolbar of the table. This helps in case a smart package is specified as the scope criteria.
- Make sure the specified scope contains instances of the classifiers specified for this table.
- Make sure at least one classifier is specified for the table.

TIP!

To define a scope, you can also drag a selected package to the **Scope (optional)** box directly from the Model Browser (see the following figure).

- To select *adjacent* elements, select a single element and then hold down the Shift key while you click other elements.
- To select *nonadjacent* elements, select a single element and then hold down the Ctrl key while you click other elements.

The new value of the **Scope (optional)** box replaces the old value.

Copyright © 1998-2015 No Magic, Inc.

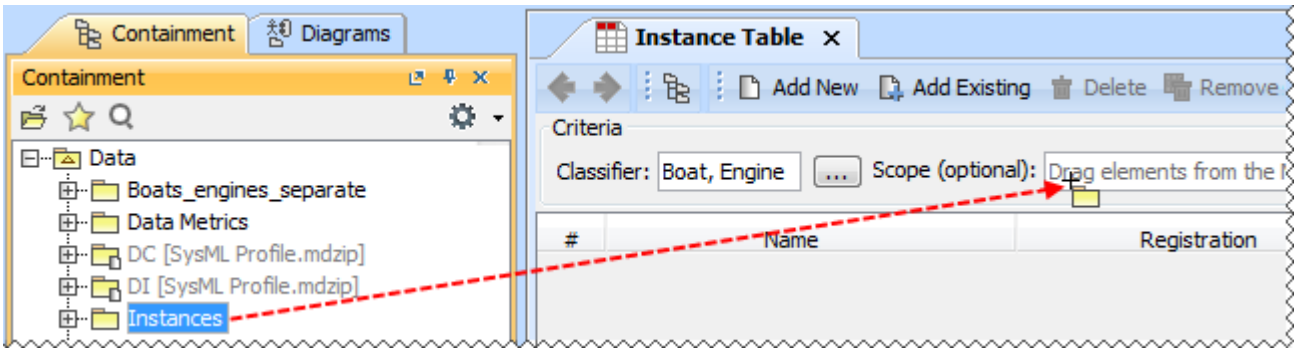


Figure 494 -- Using drag-and-drop operation to specify scope for instance table

Specifying instances as slot values

If the slot type is a classifier, you can specify an instance of this classifier as the slot value. Do this in one of the following ways:

- By selecting an instance from the model.
- **NEW!** By creating an instance directly in the instance table.

To select an instance from the model, simply click ... in the slot value cell and choose the instance from the tree or list in the open dialog.

NEW! To create an instance as slot value in the instance table

1. Click the slot value cell. ⊕ appears in the cell.
2. Click ⊕. If the slot type has subtypes, choose one of them (see the following figure).
The new instance is simultaneously created in the model and set as the slot value. You can see this instance in the Model Browser, where it is owned by the classifier, which is a type or a subtype of the selected slot.

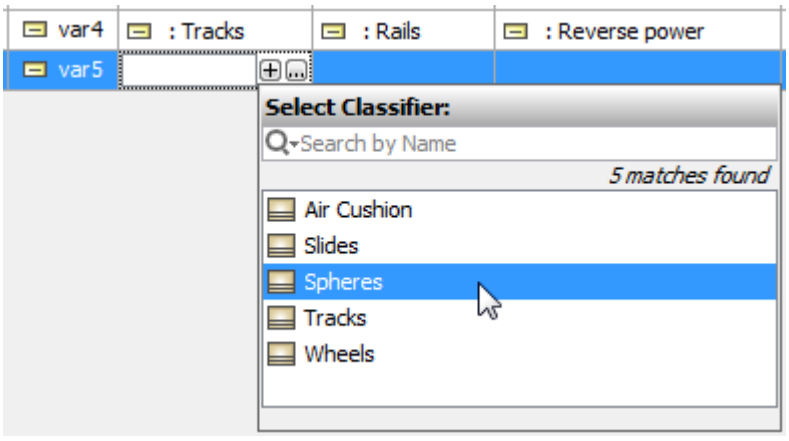


Figure 495 -- Choosing subtype for new instance

If you need to remove the instance from the cell, click that cell and then click ⊖. Keep in mind that the instance will be removed from the instance table, but not from the model.

Copyright © 1998-2015 No Magic, Inc.

NEW! Creating instances in the table from another resource

It is possible to create instances in the table from another resource (HTML, Excel, Word, etc.) using copy-and-paste command.

NEW! To create instances in the table from another resource

1. Create a table.
2. Specify the classifier.
3. Choose the columns of instance slot values that you want to fill with copied information.



The cells must be editable in the instance table.

4. Copy table from another resource.



- The number of columns in that table should be the same as the number of columns in the instance table
- The data types of columns in the instance table must be compatible with copied information.

5. Press Ctrl+V (Cmd+V on OS X).

Each row displays one instance with its slot values.



If instance already exists in the model, the information updates.

Copyright © 1998-2015 No Magic, Inc.

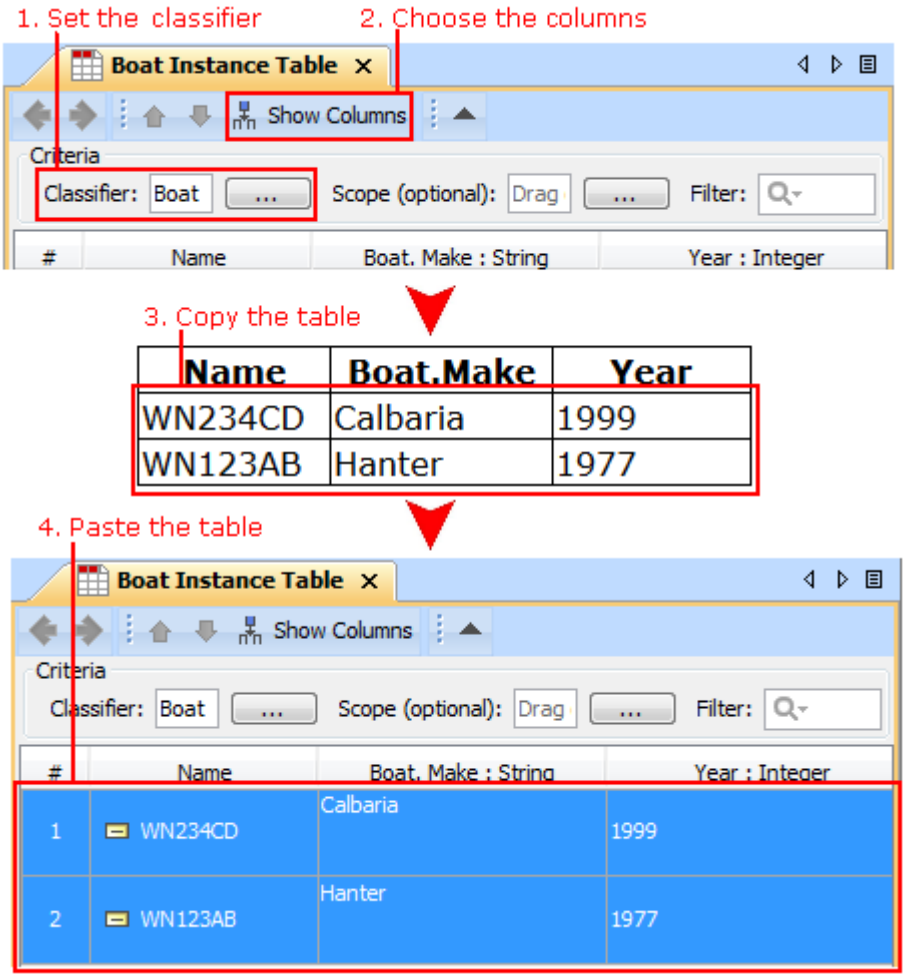


Figure 496 -- Creating instances in the instance table from another resource

Manipulations in Instance table

In this section, you will find the information how to modify the contents of an instance table. The section includes

- "Sorting table data" on page 814
- "Reordering columns" on page 815
- "Hiding single columns" on page 815
- "Filtering table data" on page 815

Sorting table data

Data in the instance table are by default sorted by the first column from smallest to largest number, though you can change the order of your instances in the following ways:

- Sort the rows by another column
- Move the selected rows up or down

Copyright © 1998-2015 No Magic, Inc.

To sort data by selected column

- Click the header of the column by which you want to sort the instance table data. A small arrow showing the automatic sort order appears on the column header. Clicking the header once makes the rows sorted in the ascending order. Clicking the header twice makes the rows sorted in the descending order.

All the rows in the table become automatically renumbered after sorting.



Figure 497 -- Sort order identification on columns

To move a row up or down

1. Select the row.
2. Do either:
   - On the Layout toolbar, click the **Up** or **Down** button appropriately.
   - Right-click the row and from the shortcut menu, select the **Up** or **Down** command appropriately.
   - Press Ctrl+Up Arrow or Ctrl+Down Arrow appropriately.

The selected row as well as associated rows becomes automatically renumbered after moving.

Reordering columns

To reorder the columns

- Click the column header and drag it to a desired place. Only the first column cannot be moved.

Hiding single columns

To hide a single column

- Right-click the column and from the shortcut menu select **Hide**.

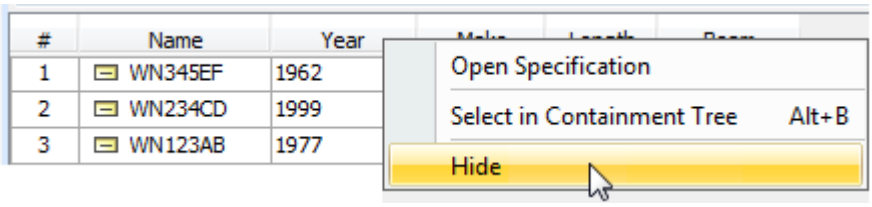


Figure 498 -- Hiding single column

Filtering table data

By filtering data in the table you can quickly find the needed instances. This is especially handy when you are working with a large table containing many rows and columns.

Copyright © 1998-2015 No Magic, Inc.

Instances can be filtered by a phrase specified in the **Filter** box. The search for the phrase can be performed either in the whole table or in the selected columns. You can also specify the case sensitivity, whether you want to use the wild cards or regular expression, and how the results should match the specified phrase.

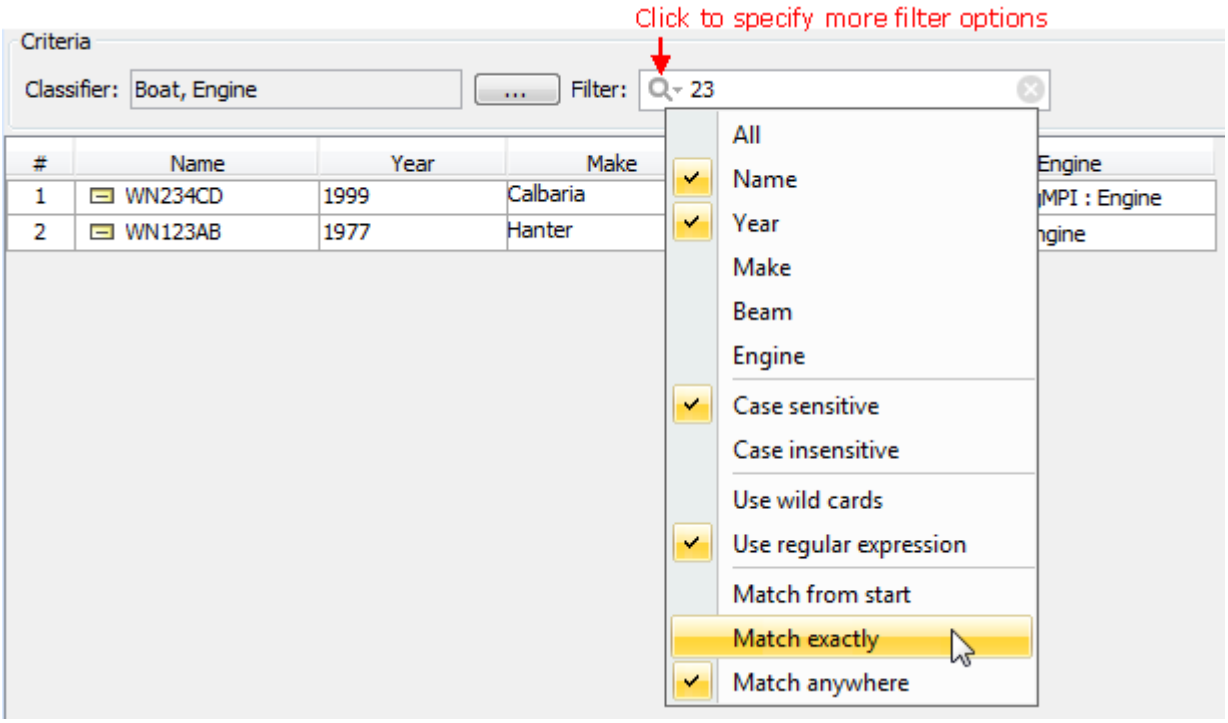


Figure 499 -- Specifying options for filtering data

Instance Table Environment

In this section, you will find the brief information on each button and command that can be useful while working with instance tables. The section includes

- "Table toolbars" on page 816
- "Criteria area" on page 818

Table toolbars

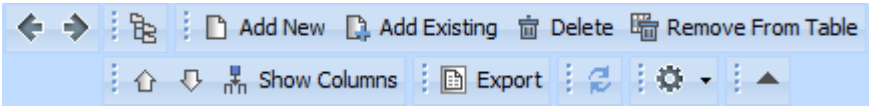


Figure 500 -- Instance table toolbars



If the instance table toolbar buttons are not available in a server project, try to lock the table for edit (make sure you have the right to edit model of this project).

The following table describes the instance table toolbar buttons.

Button	Shortcut key	Description
Navigation toolbar:		
	Alt+B	Click to select in the Containment tree the instance of the selected row.
Edit toolbar:		

Copyright © 1998-2015 No Magic, Inc.

Button	Shortcut key	Description
Add New	Insert	Click to create a new instance in the table. For more information, see the procedure "To add a new instance" on page 810.
Add Existing	Ctrl+Insert	Click to add an instance from the model. For more information, see the procedure "To add an instance from the model" on page 810.
Delete	Ctrl+D	Click to remove selected instances both from the table and the model.
Remove From Table	Delete	Click to remove selected instances only from the table.
Layout toolbar:		
Up	Ctrl+Up Arrow	Click to shift selected instances (either grouped or non-grouped) up a row. The instances are automatically renumbered after moving.
Down	Ctrl+Down Arrow	Click to shift selected instances (either grouped or non-grouped) down a row. The instances are automatically renumbered after moving.
Show Columns	N/A	Click to change the set of columns for showing on the instance table. For more information, see "Selecting columns to display" on page 807.
Publish toolbar:		
Export	N/A	Click to export the contents of the instance table to an **.html*, **.csv,* or **.xlsx* file.
View toolbar:		
	N/A	Click to update the contents of the instance table after specifying the smart package as the scope criteria. For more information, see the procedure "To specify the scope for the instance table" on page 811.
Options toolbar:		

Copyright © 1998-2015 No Magic, Inc.

Button	Shortcut key	Description
	N/A	Click to open the Options menu and then select: • **Show full paths** – to display the full paths of elements (where owner information can be displayed) in the instance table. • **Save Filter Criteria** – to keep the filter criteria unchanged after reloading the project or even after restarting MagicDraw. For more information about specifying the filter criteria, see "Filtering table data" on page 815. • **Show Detailed Column Name** – to show classifier names in column headers. For example, in the **Boat.Year** column name, **Boat** is the classifier. This option works, when the instance table has two or more classifiers specified. • **Show Column Type** – to show types of slot values in column headers. For example, **Year : Integer** or **Boat.Beam : String**. • **Strip Multiline Text** – to strip the text that covers more than five rows. Three dots will be added at the end of the stripped text.
Suppress/Expand Criteria Area toolbar:		
	N/A	Click to hide the **Criteria** area.
	N/A	Click to show the **Criteria** area.
Other buttons:		
	Alt+Left Arrow	Click to navigate to the previously opened diagram, table, dependency matrix, or relation map.
	Alt+Right Arrow	Click to navigate to the previously opened diagram, table, dependency matrix, or relation map.

Criteria area

The **Criteria** area appears below the instance table toolbars. If it is not displayed, click the button on the Suppress/Expand Criteria Area toolbar.

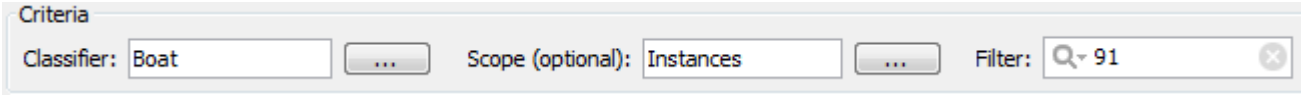


Figure 501 -- Instance table Criteria area

The following table describes the contents of the **Criteria** area

Element Name	Element Type	Description
Classifier	Text box with 	Click the **...** button to select one or more classifiers whose instances you need to display in the instance table. For more information, see "Specifying a classifier" on page 804.

Copyright © 1998-2015 No Magic, Inc.

Element Name	Element Type	Description
Scope (optional)	Text box with ...	Click the **...** button to select one or more packages, which contain the instances you need to display in the instance table. For more information, see the procedure "To specify the scope for the instance table" on page 811.
Filter	Text box	Type the phrase by which the instances should be filtered. To open the menu with more filter options, click in the **Filter** box. To reset the filter options, click in the **Filter** box. For more information, see "Filtering table data" on page 815.

Glossary

DEMO Glossary

Glossary saves time by ensuring consistent usage of terminology in the organization. It also improves the communication between team members since terms are understood in the same way and definitions become visible everywhere the terms are used.

This capability allows you create a list of all the specific terms used in the project. You can define your terms in one place and afterwards use them everywhere in your project. Furthermore, you can export your glossary to new project and use it in other projects. Also, you can have several glossaries in one project.

Words and phrases defined as terms are underlined. That is how you can identify terms in your model.

NOTE The word or phrase is underlined only if it matches completely the defined term. For example, you have defined a word Project in your glossary. So when you type a word Projects, it would not be underlines, because it is used in plural.

For your convenience, the term descriptions are visible directly on the diagrams.

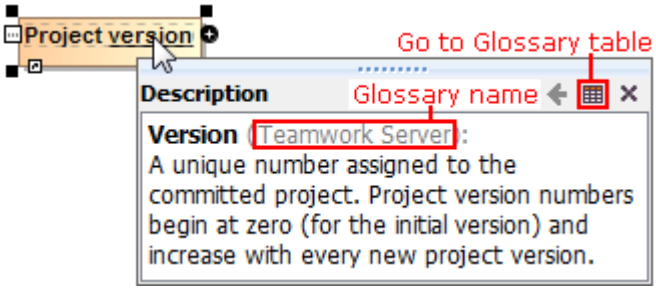


Figure 502 -- Term description window

When a term is a whole phrase and one or more words are also defined as terms, you can easily navigate through all those terms in the description window.

Copyright © 1998-2015 No Magic, Inc.

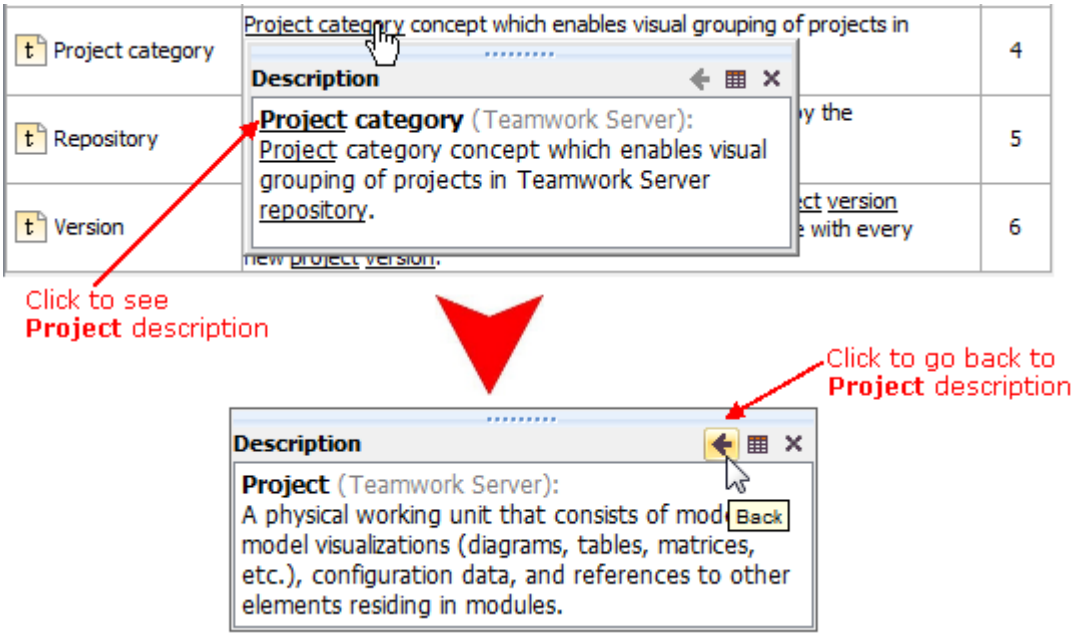


Figure 503 -- Navigation in description window

So, how do you start working with the glossary? You can choose one of the following ways:

- Select a word or phrase while you are typing and select to add it to the new glossary. The new package holding a new glossary table and all terms is created in the model.
- We recommend creating a new package where the glossary table and all terms will be stored. In the created package, create a glossary table.

#	Term	Description
1	Author	A user who has committed a new project version.
2	Comment	Optional description of changes in the committed version.
3	Project	A physical working unit that consists of model, model visualizations (diagrams, tables, matrices, etc.), configuration data, and references to other elements residing in modules.
4	Project category	Project category concept which enables visual grouping of projects in Teamwork Server repository.
5	Repository	A storage place for projects and their versions managed by the Teamwork Server.
6	Version	A unique number assigned to the committed project. Project version numbers begin at zero (for the initial version) and increase with every new project version.

Figure 504 -- Glossary table

The glossary table allows for managing the terms of your model in a spreadsheet-like form. Each row in the table represents a term. A word, phrase, or any element of the model can be a term.

With the help of this table you can easily:

- Create and manage all terms of the model in a single place
- Customize the representation of the table
- Export the data into an *.html, *.csv, or *.xlsx file

Working with Glossary

Working with glossary is quick and easy. You can create a term at any moment and in several ways:

Copyright © 1998-2015 No Magic, Inc.

- Add any word or phrase to the glossary while typing or editing
- Drag any element you wish to be a term to the glossary table
- Click the **Add New** or **Add Existing** button in the glossary table toolbar
- **NEW!** Paste terms in the glossary table from another resource

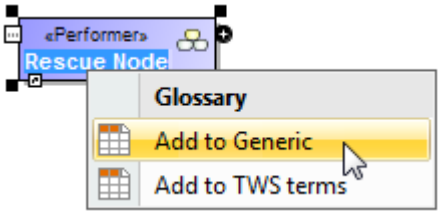
To add a word or phrase to a glossary while typing or editing

1. Select a word or phrase you want to be a term.



Even if you mark only a part of a word, the whole word will be added to the glossary as a term.

2. From the shortcut menu, select **Add to <glossary table name>**. The word or phrase is converted to term and added to the glossary table.

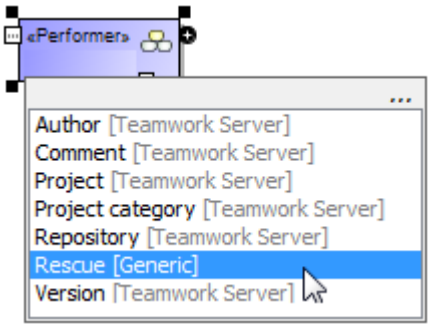


To add an element to the glossary table

- Do one of the following
  - In the Containment tree, select an element you wish to be a term and drag it to the glossary table.
  - In the glossary table toolbar, click the **Add Existing** button, and from the **Select Element** dialog, choose the element you wish to be a term.



After terms are defined in the glossary, it is easy to use them in your project. Just press Ctrl+spacebar while typing to get a list of available terms.



NEW! To create terms in the table from another resource

1. Create a table.



The cells must be editable in the glossary table.

2. Copy table from another resource.



- The number of columns in that table should be the same as the number of columns in the glossary table
- The data types of columns in the glossary table must be compatible with copied information.

Copyright © 1998-2015 No Magic, Inc.

3. Press Ctrl+V (Cmd+V on OS X).
Each row displays one term with its description.



If the term already exists in the model, the information updates.

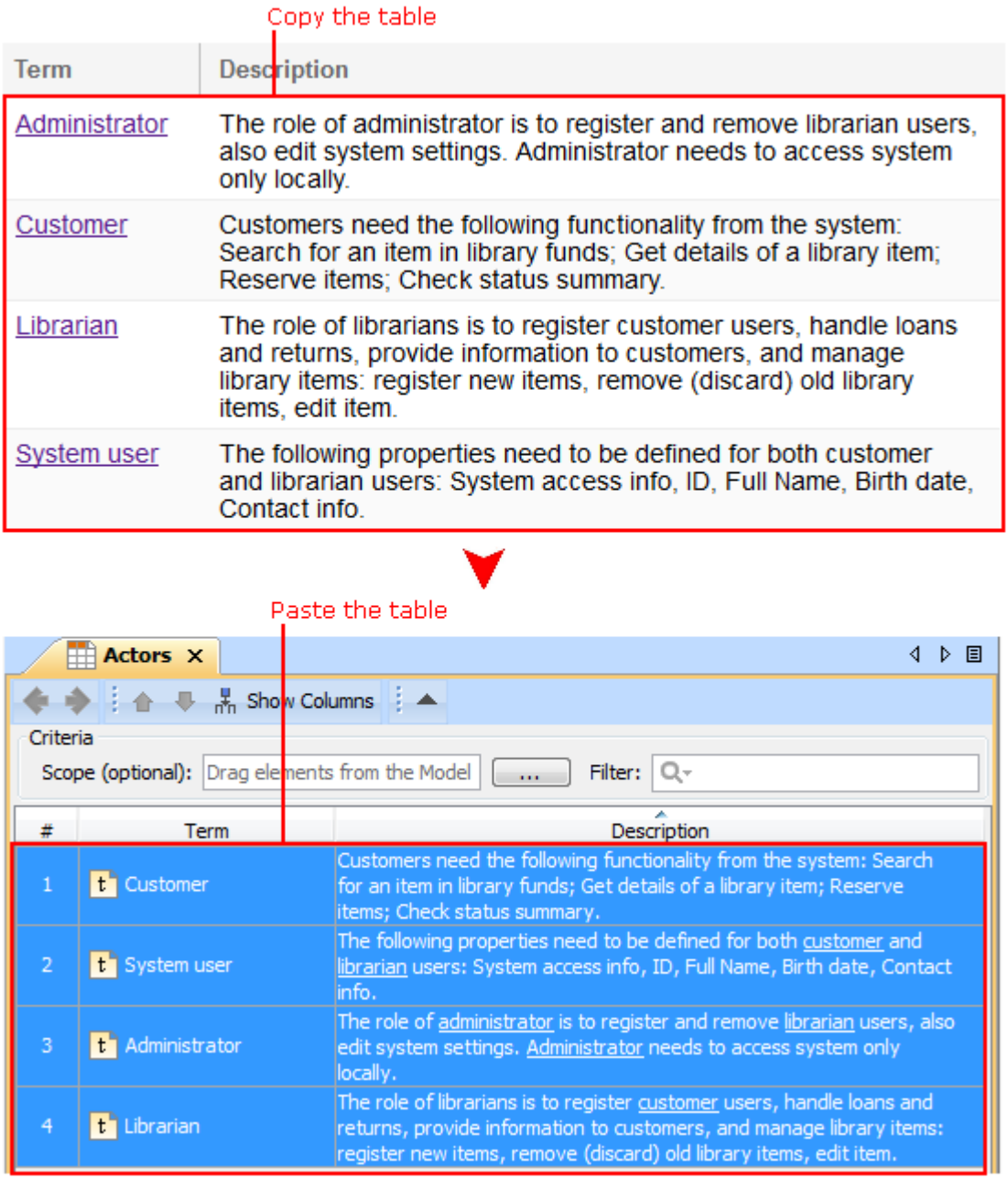


Figure 505 -- Creating terms in glossary table from another resource

Metric Table

For the instructions on how to create a metric table and use it for managing metrics, see "Calculating Metrics" on page 597.

Copyright © 1998-2015 No Magic, Inc.

11 MODELING ELEMENTS

Action

An action is a named element that is the fundamental unit of an executable functionality. The execution of an action represents some transformations or processing in the modeled system, be it a computer system or otherwise. An action execution represents the run-time behavior of executing an action within a specific behavior execution. As action is an abstract class, all action executions will be the executions of a specific kind of action. When the action will be executed and what its actual inputs will be are determined by the concrete action and the behaviors in which it is used.

The following are types of actions:

- "Accept Event Action" on page 825.
- "Call Behavior Action" on page 827.
- "Call Operation Action" on page 829.
- "Opaque Action" on page 831.
- "Send Signal Action" on page 831.

To remove an action from diagram by keeping incoming and outgoing control flows connected

Do one of the following:

- On a diagram select an action and press **Delete**. Then, in the opened message, click **Yes**.

Copyright © 1998-2015 No Magic, Inc.

- On a diagram, from an action shortcut menu, select **Refactor** and then **Remove**.

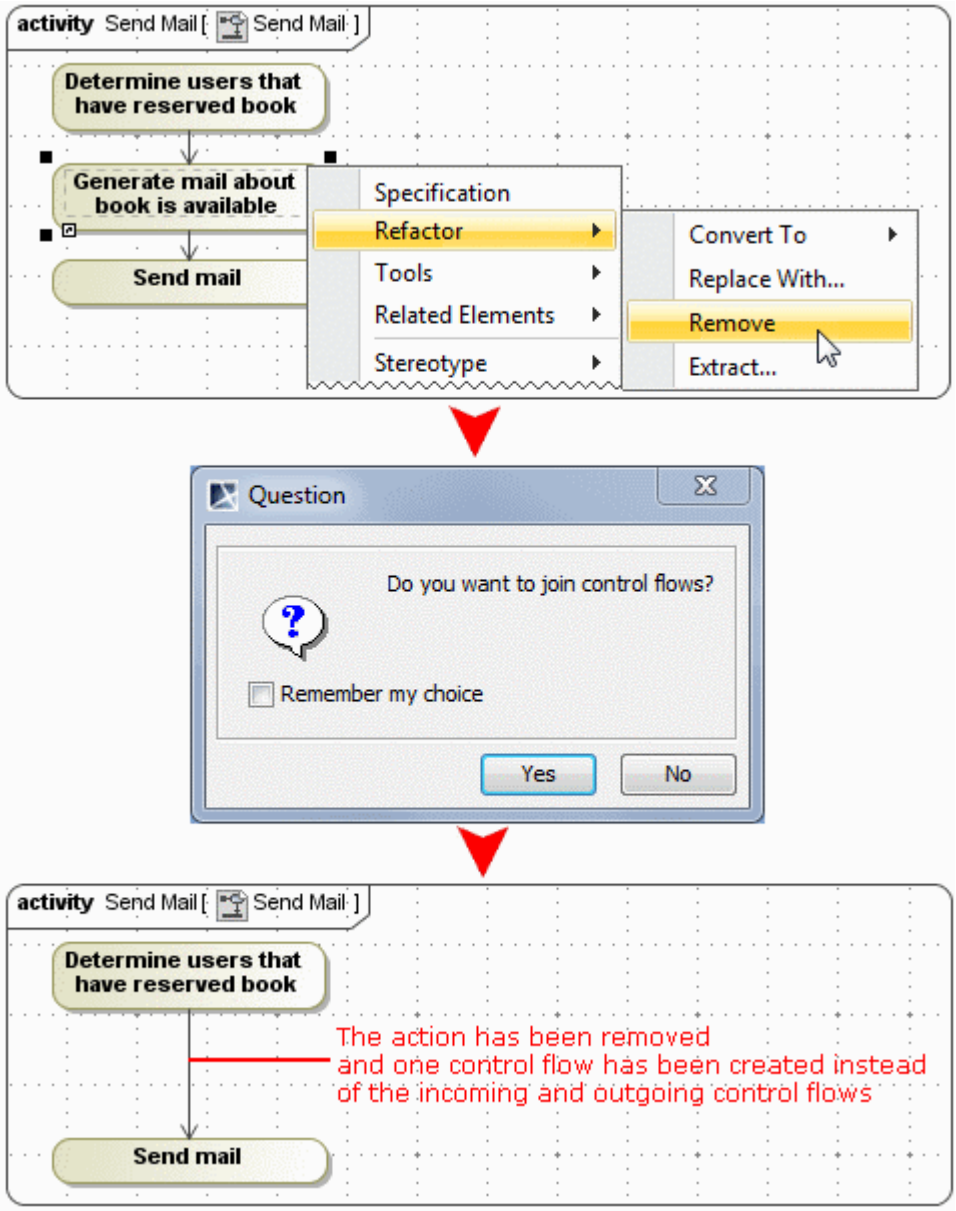


Figure 506 -- Example of Action remove

TIP!

- Flows can be joined only if removed action has one incoming and one outgoing control flows.
- You can select more than one action to remove, if only the selected part has one incoming and one outgoing control flow.
- In the **Environment Options** dialog, in the **Diagram** options group, under the **Editing** category, you can select one of the **Join Control Flows** option value: *Ask to join control flows*, *Yes* or *No*.

Copyright © 1998-2015 No Magic, Inc.

Action Properties

You can specify action properties in the action Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

See also: "Working with Actions" on page 834.

Usage in diagrams: "Activity Diagram" on page 697.

Parent topic: "Modeling Elements" on page 823.

Related topics

"Specification Window" on page 273.

"Formatting Symbols" on page 317.

Accept Event Action

The Accept event action is an action that waits for the occurrence of an event that meets specified conditions. If an accept event action has no incoming edges, then the action starts when the containing activity or structured node starts, whichever most immediately contains the action. In addition, an accept event action with no incoming edges remains enabled after it accepts an event. It does not terminate after accepting an event and outputting a value, but continues to wait for other events. An accept event action with no incoming edges and contained by a structured node is terminated when its container is terminated.

Notation	Description
Accept event action 	The *Order cancel request* accept event action is connected to *Cancel order* with the exception handler relationship.
Rotated accept event action 	The symbol of accept event action is rotated.
Accept time event action 	The time event is connected to the call behavior action.

Assigning signals

To assign a signal to an accept event action, you can use on the following procedures:

- To assign a signal using a drag-and-drop operation

Copyright © 1998-2015 No Magic, Inc.

- To assign a signal using the accept event action's shortcut menu

To assign a signal using a drag-and-drop operation

IMPORTANT!

To assign a signal to an accept event action, at least the one signal element should exist in the project.

1. Select the signal in the Model Browser.
2. Drag the signal to the accept event action as it is shown in the following picture.

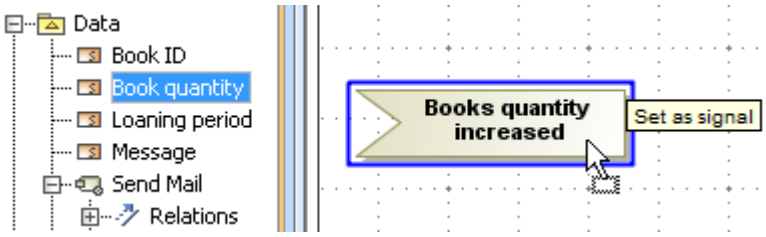


Figure 507 -- Signal dragging to accept event action

To assign a signal using the accept event action's shortcut menu

1. Right-click the accept event action to open the shortcut menu.
2. On the shortcut menu, click **Signal** and then click to select a signal that you wish to assign to the element. Or, click the **Create Signal** button.

TIP!

For more information about creating a new element see Section "Elements Multiple Selection" on page 354.

Once the signal is assigned to the accept event action, a signal event and a trigger for this action are created automatically. If you change the signal, the signal event and the trigger will change accordingly to a new signal. The following figure shows the signal event and the trigger positions in the Containment tree.

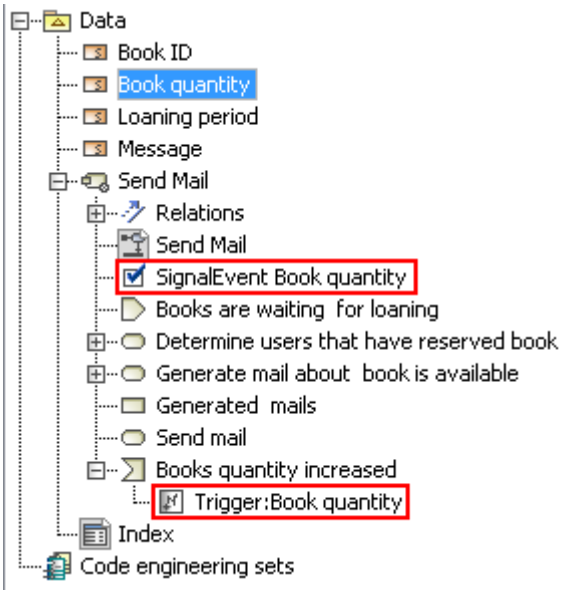


Figure 508 -- Signal event and trigger in the Containment tree

Copyright © 1998-2015 No Magic, Inc.

Using Time Event Shape

To draw the accept event action with the time event shape

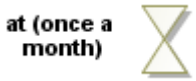
- In the activity diagram panel select the **Time Event** button and then click the diagram pane.

To change the position of the accept event action name

1. Select the accept event action shape on the diagram pane.



2. Click the **Rotate State** button on the upper-right side of the shape. The name of the accept event action will appear on the opposite side of the shape.



You can specify for an accept event action, whether there is a single output pin for the event, or multiple output pins for the attributes of the event.

Specifying the Is Unmarshall property

To specify the **Is Unmarshall** property

1. Select the accept event action and open its Specification window.
2. Select or clear the **Is Unmarshall** check box. If the value is set to:
   - *false*, then there is a single output pin for the event, and a real-world instance of the signal is placed on this output pin.
   - *true*, then there are multiple output pins for the attributes of the event, and feature values of the signal instance are placed on the corresponding output pins.

Call Behavior Action

The call behavior action invokes a behavior directly rather than invoking a behavioral feature that, in turn, causes the behavior. The argument values of the action are available to the execution of the invoked behavior. The execution of the call behavior action waits until the execution of the invoked behavior completes and a result is returned on its output pin. In particular, the invoked behavior can be an activity.



Figure 509 -- The Receive Order and Fill Order call behavior actions

To assign a behavior to the call behavior action

Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

- On the activity diagram pallet, click the *Automatic Behavior Creation* button. An activity will be created automatically on a call behavior action creation.

TIP! For more information about the *Automatic Behavior Creation* mode, see "Creating Behavior Automatically in Activity Diagram" on page 703.

- Open the Specification window of the call behavior action and click the cell of the **Behavior** property value. Then click and in the open dialog, select the behavior. Click **OK** when you are done.
- On the activity diagram, right-click the call behavior action shape. On the shortcut menu, click Behavior and then click to select a behavior that you wish to assign on the element.
- On the activity diagram, click a call behavior action shape, and then on the smart manipulator toolbar, click . In the open list, select the behavior.
- On the activity diagram, click a call behavior action shape, and then press **Ctrl+T**. In the open list, select the behavior.
- On the diagram, click a call behavior action shape, then click the name area. Type the ":" colon and then type the existing behavior name. Press **Enter** or click anywhere on the diagram. The behavior is created and assigned.

TIP! Type the ":" colon, then press **Ctrl+Space** or **Ctrl+Backspace** and then in the open list, select the behavior.

- Drag a behavior from the Containment tree onto the call behavior action shape on the diagram. Behavior is assigned to the call behavior action.

NOTE

- Double click the call behavior action with the assigned behavior - the behavior Specification window opens or if the assigned behavior is a diagram, the diagram opens in the same diagram tab.
- The rake icon is displayed on the shape if the call behavior action refers to another activity diagram. The rake icon isn't shown by default therefore you need to set the **Show Rake Icon** property value to *true* in the **Property** dialog. For more information see "NEW! Displaying rake icon on symbol" on page 322.
- A parameter is created automatically for each newly created pin on the call behavior action. The parameter's name, type, direction and multiplicity are synchronized with the pin name, type, direction, and multiplicity.

  Note that a parameter for a pin is created automatically if:
  - the call behavior action has the behavior assigned
  - the behavior is editable
  - the behavior has only one usage and it is not used in another project (it is not in a shared packaged and the project is not used other projects).

To create a new diagram for the call behavior action

Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

- On the activity diagram, click the call behavior action shape, and then on the smart manipulator toolbar, click .The new activity diagram opens in the same diagram tab.



- Name of the new activity diagram is the same as the name of the call behavior action.
- If a behavior was assigned for the call behavior action already, a new activity diagram under the behavior will be created. This is valid, if the behavior is editable and behavior's type is Activity.
- If pin(s) are created on the call behavior action, then parameters for these pins are created in a newly created activity. Pins are synchronized with the created parameters, that is, the pin name, type and multiplicity are synchronized with the parameter name, type, and multiplicity.
- If parameters are created for a new activity, then activity parameter nodes are created in a new activity diagram and displayed on the activity diagram frame.

- On the activity diagram, right-click the call behavior action shape, point to **Create Diagram** and then point to a diagram type.

To change the name display mode on the call behavior action

- In the call behavior action **Symbol Properties** dialog, select the **Name Display Mode** property and then select one of the options: *Show Action Name* (default), *Show Behavior Name*, or *Show Both*.



If the *Automatic Behavior Creation* mode is turned on, the default value of the **Name Display Mode** property is *Show Behavior Name* for the newly created call behavior action.
For more information about the *Automatic Behavior Creation* mode, see "Creating Behavior Automatically in Activity Diagram" on page 703.

Call Operation Action

The call operation action transmits an operation call request to the target object, where it can cause the invocation of an associated behavior.

You can display an action name and/or name of the operation on the call operation action shape. For example, if you have two call operation action elements calling the same operation, you can specify their names to distinguish which action means what.

Notation	Description
The call operation action with *getOrder* operation 	In this example the call operation action has been assigned *getOrder* operation whose type is *OrdersDB* class.
The call operation action named Get supplementary order 	In this example the call operation action is named *Get supplementary order.* It has been assigned *getOrder* operation whose type is *OrdersDB* class.

Copyright © 1998-2015 No Magic, Inc.

Call operation action with hidden classified name	
	The classified name of operation is hidden from the call operation action shape.

To assign an operation for the call operation action

Do one of the following:

- On the diagram, right-click the call operation action and on thee shortcut menu, click **Select Operation**. Then click to select an operation that you wish to assign to the element. Or click the **Creation Mode** button to create an operation. Click **OK** then you are done.
- In the call operation action Specification window, click to select the **Operation** property specification cell. Then click ... button. In the **Select Operation** dialog, click to select an operation that you wish to assign to the element. Or click the **Creation Mode** button to create an operation. Click **OK** then you are done.

MagicDraw allows for displaying the operation name and class of the operation on the call operation action shape.

When an operation is assigned to the call operation action, there are three name/operation display options available:

- If the call operation action is not named, the name of the class is displayed under the operation name.



- If the name of the call operation action is the same as the assigned operation name, then the name of the class is displayed under the operation name.



- If the call operation action name differs from the assigned operation name, then <*class of the operation*>::<*operation name*> is displayed under the call operation action name.



To hide the operation name and class of the operation from the call operation action shape

- On the call operation action shortcut menu, click to clear the **Show Qualified Name for Operation** check box. You can also customize this property in the **Symbol Properties** dialog of the call operation action.



When opening a project from a version earlier than MagicDraw 15.0, by default the class of the operation is not displayed on the diagram pane. The **Show Qualified Name for Operation** option is unchecked.

To open the behavior diagram of the call operation action

- Double-click a call operation action to open the specified behavior diagram. The diagram opens in the same diagram tab.

Copyright © 1998-2015 No Magic, Inc.

Opaque Action

The opaque action is introduced for implementation-specific actions or for use as a temporary placeholder before some other actions are chosen.

The opaque action has no specific notation.

There is an additional property named **Body and Language** in the Specification window of an opaque action..

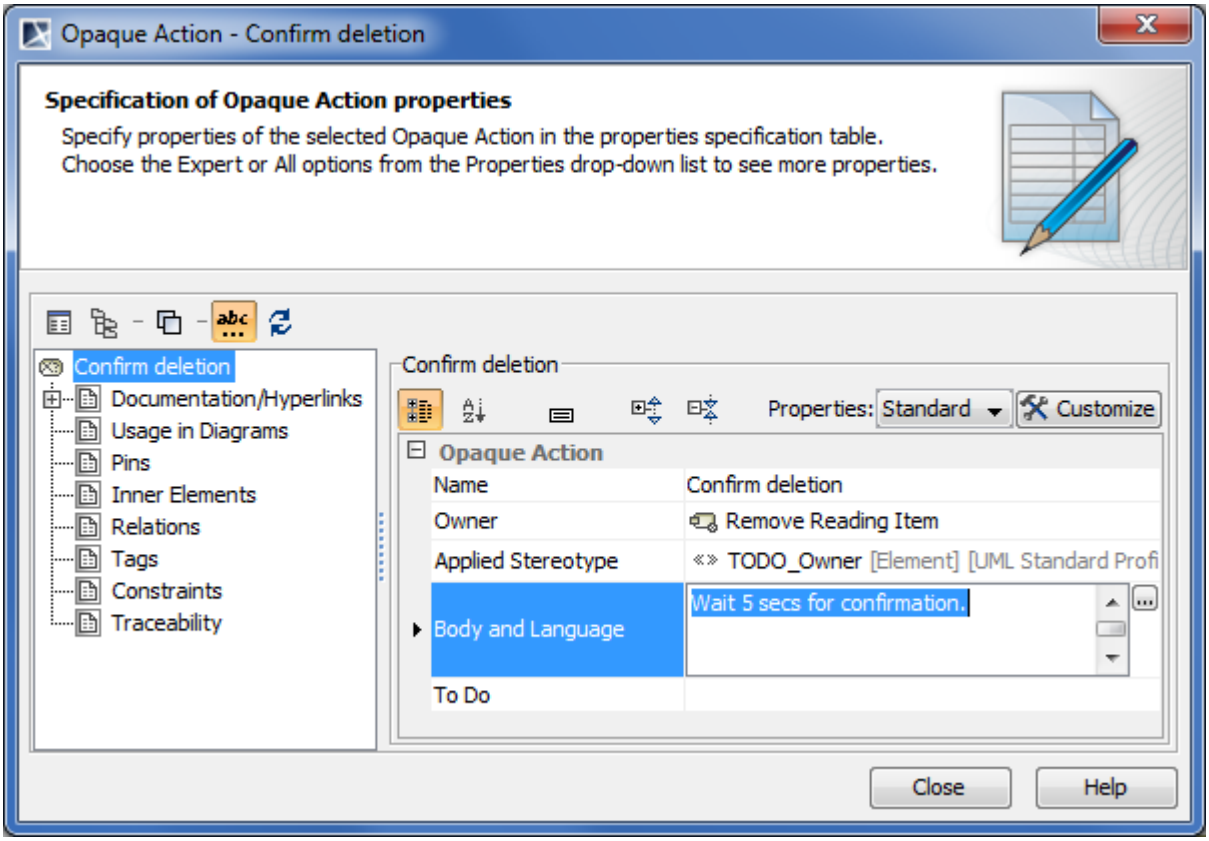


Figure 510 -- Opaque Action's Specification window

Send Signal Action

The send signal actions creates a signal instance from its inputs and transmits it to the target object, where it can cause the start of a state machine transition or the execution of an activity. The argument values are available to the execution of associated behaviors. The requester continues the execution immediately. Any reply message is ignored and is not transmitted to the requester. If the input is already a signal instance, use the Send object action.

Notation	Description

Copyright © 1998-2015 No Magic, Inc.

Send signal action



This example describes an order process.

1. First, an order is created (*Create order* call behavior action).
2. Then, a signal to fill the order request is sent to the warehouse (*Fill order request* send a signal action).
3. Finally, an invoice is created (*Create invoice* call behavior action).

The relationships are represented with control flow paths.

Assigning signals

To assign a signal to a send signal action you can use any of the following features:

- Send signal action's Specification window
- Drag-and-drop operation
- Send signal action's shortcut menu

To assign a signal via the send signal action's Specification window

1. Open the Specification window for the send signal action.
2. In the **Signal** property value cell do any of the following:
   - Click the "..." button. The **Select Signal** dialog will open. Select the signal from the list or create a new one.



For more information about creating a new element see Section "Creating New Elements" on page 356.

   - Click the button to open the list of signals available in the project. Select the signal from the list. The example of the signal list is shown in the following picture.

Copyright © 1998-2015 No Magic, Inc.

3. Click **Close** once the signal is selected.

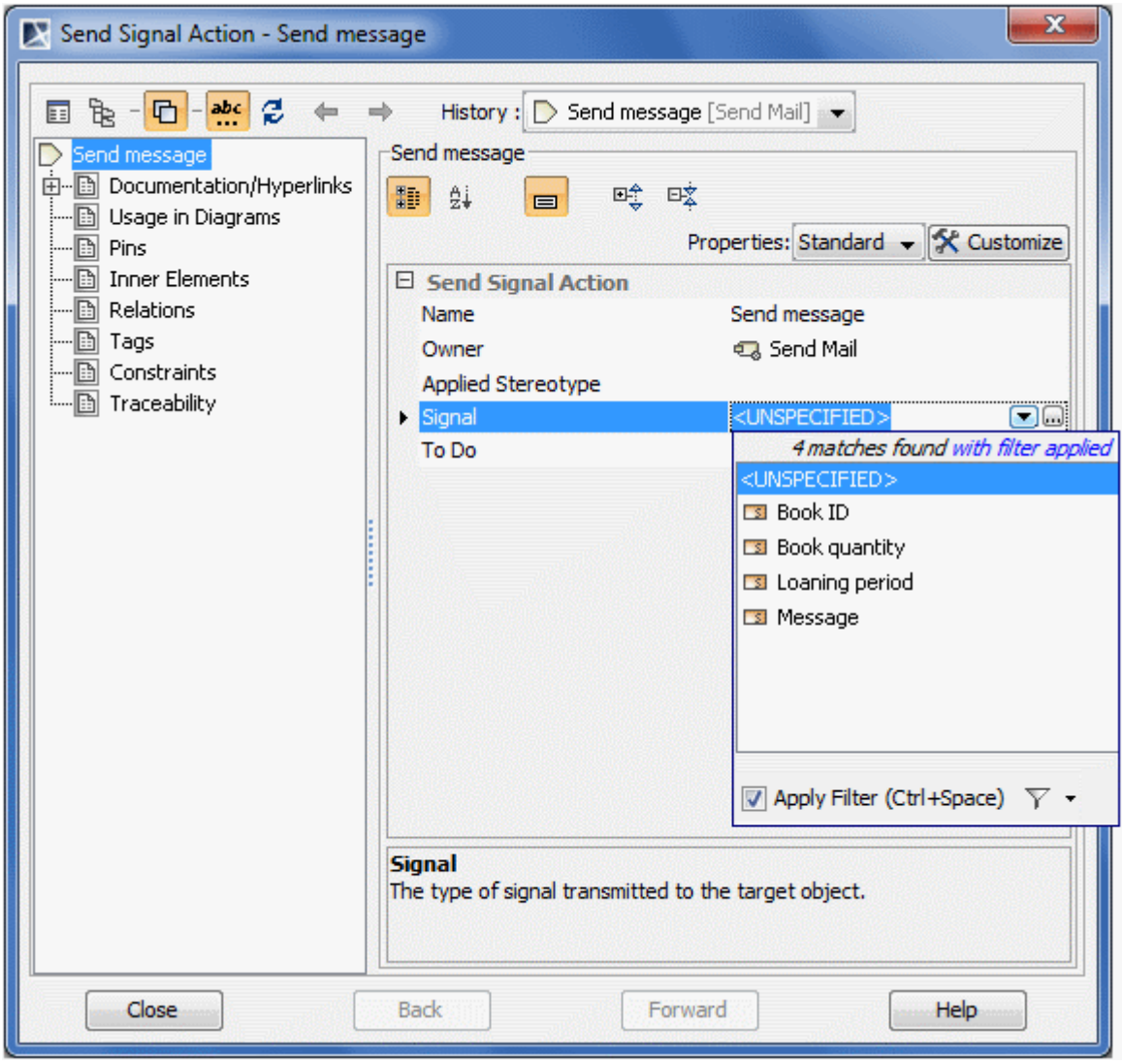


Figure 511 -- Send signal action's Specification window. Signal selection

To assign a signal using a drag-and-drop operation

1. Select the signal in the Containment tree.
2. Drag the signal to the send signal action as it is shown in the following picture.

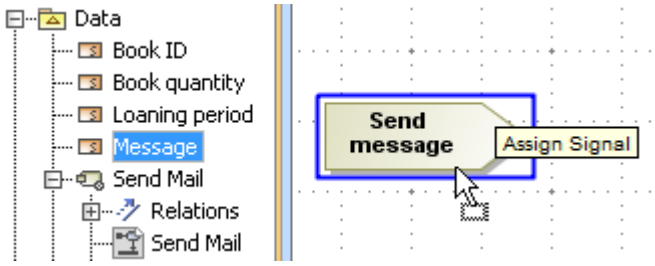


Figure 512 -- Signal dragging to send signal action

To assign a signal using the send signal action shortcut menu

1. Right-click the send signal action to open the shortcut menu.
2. Click **Signal**. The list of signals available in the project will open. The example of the dialog is shown in the following picture.

Copyright © 1998-2015 No Magic, Inc.

3. Select the signal from the list or click **New Signal** and create a new one.

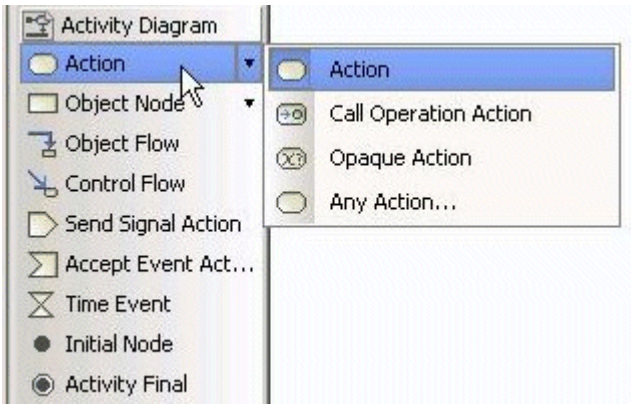


For more information about creating a new element see Section "Creating New Elements" on page 356.

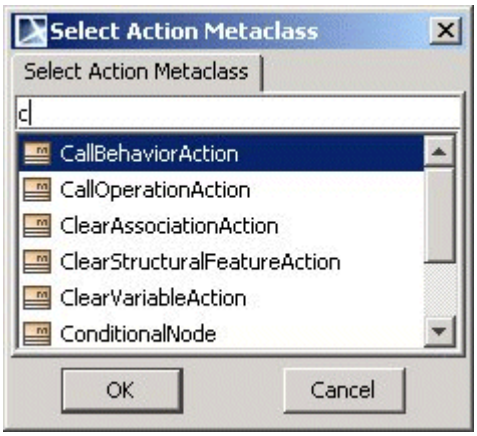
Working with Actions

To quickly create any action

1. In the activity diagram toolbar, right-click the **Action** button. The menu opens.



2. Select the **Any Action** command. The **Select Action Metaclass** dialog opens.



3. Select an action metaclass from the list or type the first letter of the metaclass. Click **OK**. The action is created.
4. Click on the diagram pane. An action symbol is drawn.

Applying duration constraint on the action

You can create and apply a time duration constraint on an action that states that the output must occur after delay from the input.

To create and apply a time duration constraint

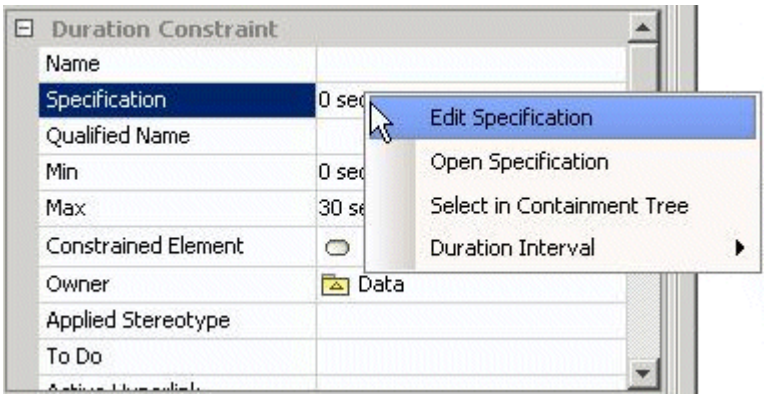
1. Select an action and create the input and output pins to specify the events.
2. Create a duration constraint for the action:
    2.1 In the **Call Behavior Action** Specification window, click the **Constraints** tab.

Copyright © 1998-2015 No Magic, Inc.

2.1 Click the **Apply** button. The **Select Constraint** dialog opens.

2.1 Select the constraint storage place and click the **Create** button. In the menu that opens, select the **Duration Constraint** command. The **Duration Constraint** Specification window opens.

3. Specify a duration interval. In the **Duration Constraint** dialog, select the **Expert** property display mode. Type the minimum and maximum duration for holding an activity in the **Min** and **Max** fields, e.g. type the following values: *0 sec* and *30 sec*.
4. Assign the events for the input and output pins:
   4.1 In the open **Duration Constraint** Specification window, right-click the **Specification** property specification cell. The following menu opens:



4.1 Select the **Open Specification** command. The **Duration Interval** Specification window opens.

4.1 Right-click the **Min** property and select the **Open Specification** command. The **Duration** Specification window opens.

4.1 In the **Event** field, click the "..." button. The **Select Element** dialog opens. Select the activity input pin.

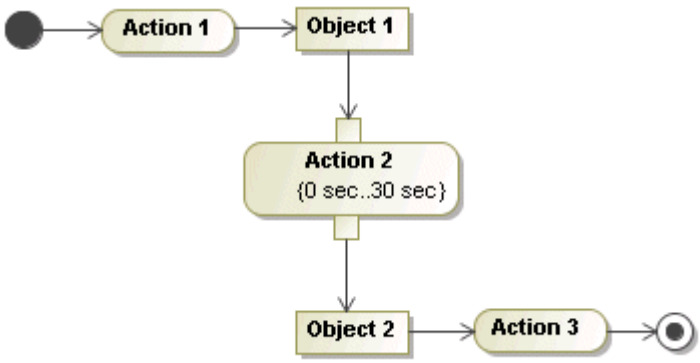
4.1 Repeat steps c and d for the **Max** property - select the action output pin as an event.

5. Apply the created duration constraint on the action.



To return to the former dialog in the Specification window, click the **Back** button.

The sample below depicts an activity diagram with a duration constraint applied on an action.



Copyright © 1998-2015 No Magic, Inc.

Actor

Actors represent roles played by human users, external hardware, and other subjects. An actor does not necessarily represent a specific physical entity but merely a particular “role” of some entities that is relevant to an action performed by an actor. Typically, such actions are called use cases.

An actor is shown as a “stick man” figure with the name below the figure.



Figure 513 -- Example of actor connected to use case

You can format the actor symbol properties in the **Symbol Properties** dialog.

TIP!

For more information about symbol representation properties, see "Formatting Symbols" on page 317.

You can specify actor properties in the **Actor** Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Actors can be represented in other diagrams, such as sequence or activity diagrams.

TIP!

For more information,
see "Actor usage in other diagrams" on page 836.

Actor usage in other diagrams

The actor can be used in other diagrams in various ways. Most common cases are as follows:

- Actors can be depicted as lifelines in a sequence diagram.
- Actors can be depicted as swimlane headers in an activity diagram.

These cases are described in the following procedures.

To create a lifeline for an actor in a sequence diagram

1. Open the sequence diagram wherein you want to create a lifeline.
2. Select an actor in the Model Browser.

Copyright © 1998-2015 No Magic, Inc.

3. Drag and drop the selected actor on the sequence diagram pane. The lifeline is created. In the following figure, see an example wherein two lifelines are depicted.

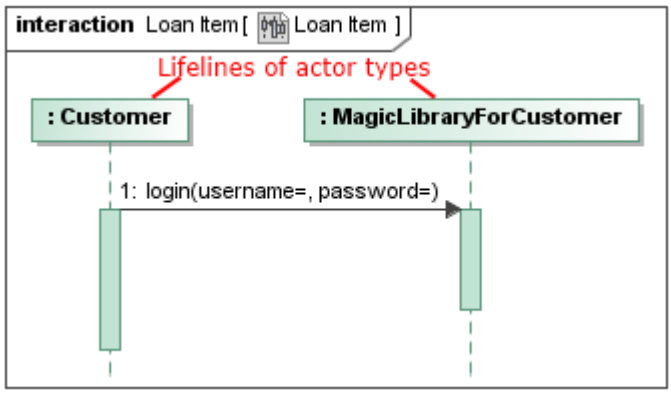


Figure 514 -- Lifelines of particular actor types in sequence diagram

To create a swimlane header for an actor in an activity diagram

1. Open an activity diagram.
2. Create a swimlane.
3. Select an actor in the Model Browser.
4. Drag and drop the actor on the swimlane header in the activity diagram. The actor is assigned to the header of the swimlane. See the example in the following figure.

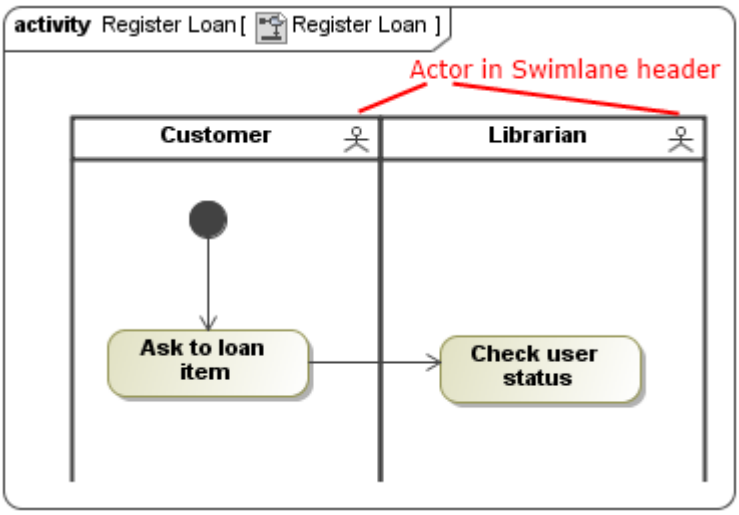


Figure 515 -- Example of swimlanes that represents actors

Related diagrams

Use Case Diagram
Sequence Diagram
Activity Diagram

Related procedures

Formatting Symbols
Editing Property Values

Related references

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Related resources

Example of Use Case diagram in *<MagicDraw installation directory>\samples\diagrams\use case diagram.mdzip*.

Anchor

Use an anchor to relate a symbol to a note or comment. By default, newly created anchor links an element with a note.

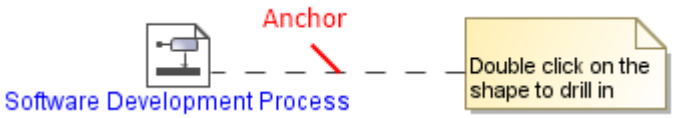


Figure 516 -- Example of anchor

You can format the anchor symbol properties in the **Symbol Properties** dialog.

TIP!

For more information about symbol representation properties, see "Formatting Symbols" on page 317.

To draw an anchor from a symbol

1. On the diagram pane, select a symbol.
2. In the appeared smart manipulator toolbar, click the Anchor button as depicted in the following figure:



3. Click on the diagram pane to create a new anchored note or click the note or comment to which you want to anchor the element symbol. The anchor is created from the element symbol.

To change an anchor path style

1. On the diagram pane, select an anchor and open its shortcut menu.
2. Select the **Path Style** command and then choose one of the following path style: Rectilinear, Oblique, or Bezier.

Related elements

Note

Comment

Related procedures

Formatting Symbols

Copyright © 1998-2015 No Magic, Inc.

Association

In the class diagrams, an association represents the semantic relationship between two or more classifiers which specifies connections between their instances. An association relationship is the most general of all relationships and the most semantically weak.

In the use case diagrams, an association represents the participation of an actor in a use case, for example, when instances of an actor and instances of a use case communicate with each other. This is the only relationship between actors and use cases. Sometimes an association relationship is called communication association.

An association is drawn as a solid path connecting two classifier symbols.



For a general information about working with symbols, see "Diagramming" on page 198.

Working with Associations

Association Properties

You can specify association properties in the association Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To show the direction arrow near the association name

- From the association shortcut menu, select **Show Direction Arrow**.



This command is available in the **Full Featured** perspective. For more information about perspectives, see "Customizing and Selecting Perspective" on page 65.

Default Direction Arrow direction is displayed according path creation direction.

To change the Direction Arrow direction

- From the association shortcut menu, select **Reverse Direction Arrow**.

The Direction Arrow is a graphical representation that most often is used in top level domain class diagrams. The Direction Arrow helps to read a diagram and explains diagram semantics. The Direction Arrow has no meaning in a model.

Usually, Direction Arrow is used on a diagram where navigability is not defined yet. Direction Arrows are usually displayed for named associations. When you move on with your modeling and create more detailed diagrams with specified navigability, direction arrows and associations names usually are not displayed in this type of diagrams.

As it is depicted in the following figure, *User* and *Account* classes are connected with the association. A navigation arrow can be displayed to either side depending on the association name. If the association name is

Copyright © 1998-2015 No Magic, Inc.

"*belongs to*", the Direction Arrow should point from the *Account* class to the *User* class. If association name is "*has*", the Direction Arrow should point from the *User* class to the *Account* class.



Figure 517 -- Examples of Direction Arrows

In the class diagram, you can add attributes to an association using an association class. The association class is a simple class that has a dashed line connected to the association.

To draw an association class

1. Draw two classes.
2. Click the **Association Class** button on the diagram pallet.



If you do not see the **Association Class** button in the diagram pallet, click the black arrow button near the **Association** button to see the list of associations and select **Association Class** from that list.

3. On the diagram pane click the first class shape (path source).
4. Drag the path to the second class (path destination) and drop it there.

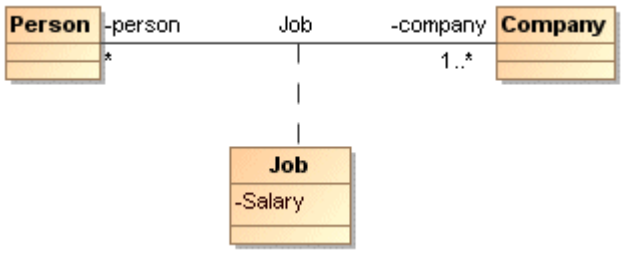


Figure 518 -- Sample of association class

If you need to model a relationship among a number of classes, the N-ary association should be used. The N-ary association is drawn as a big diamond with all the associations attached to its points. Every involved class can have a role name and multiplicity.

To draw an N-ary association class

1. Draw several classes on the diagram pane.
2. Draw the **N-ary Association** connector shape.



If you do not see the **N-ary Association** button in the diagram pallet, click the black arrow button near the **Association** button to see the list of associations and select **N-ary Association** from that list.

3. Connect all classes to the N-ary association using an association relationship.

Adding Association between Read-only Classifiers

Adding a new Association always creates two roles or properties at both ends that are owned by the attached Classifier by default. However, when one or both ends of the Association are not editable for some reasons (for example, locked in a server project or stored in a read-only profile or used project), the properties will be owned

Copyright © 1998-2015 No Magic, Inc.

by the Association itself. In this case, MagicDraw will display a warning informing you about the sometimes-unexpected issue of a model creation (see the following figures).

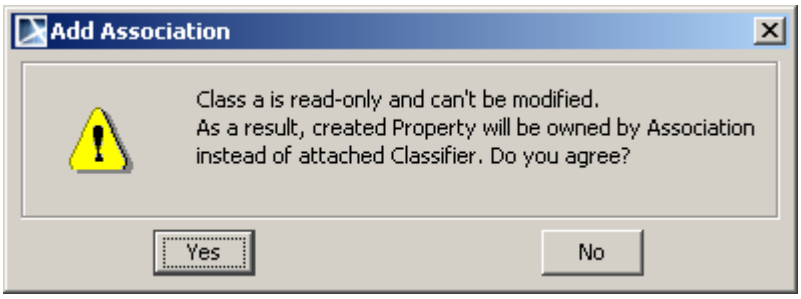


Figure 519 -- The Add Association Dialog for a Read-Only Classifier

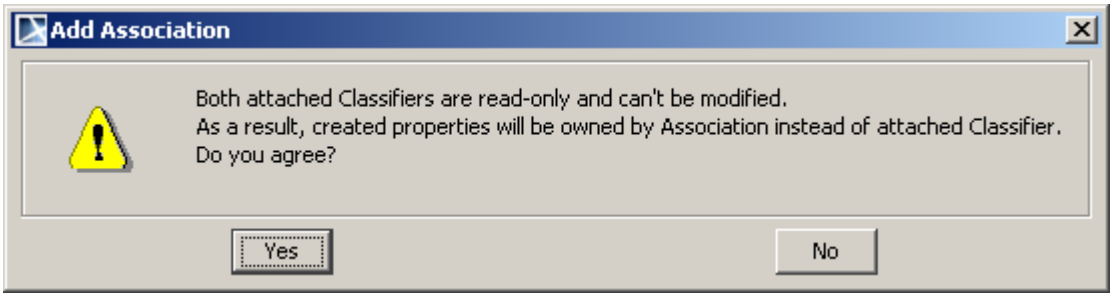


Figure 520 -- The Add Association Dialog for Both Read-Only Classifiers

Association End

Association ends are represented by properties each of which is connected to the type of the end. When a property is an association end, the value or values are related to the instance or instances at the other end(s) of the association.

An association end is the connection between lines depicting an association and the shape.

Association end properties

The following properties of the association end can be specified: name, association end type, visibility, multiplicity, qualifier, aggregation kind.

Name

Association end has other name - role. A role indicates a role played by the class in terms of an association. The role name is placed at the association end near the class playing that role. The role name at the implementation level maps to the reference name of the opposite class. Roles can have visibility (public, package, protected, and private).

Association end type

Changing the association end type, changes the target of the association or in other words the classifier to which the association is connected.

Qualifier

A qualifier is an attribute or a list of attributes whose values serve to partition the set of instances associated with an instance across an association. Qualifiers are attributes of an association. It is represented as a small rectangle attached to the end of an association path between the final path segment and the symbol of the

Copyright © 1998-2015 No Magic, Inc.

classifier that it connects to. The qualifier rectangle is part of the association path, not part of the classifier. The qualifier rectangle drags with the path segments. The qualifier is attached to the source end of the association.

To change the association end type

Do one of the following:

- On the diagram, select the association end and move it to the other target. The association is connected to the other target and the type of the association end (that is, property) is changed.
- In the Containment tree, select the property which represents the association end and drag it onto the available classifier (association target). If the association is represented on the diagram pane, the association is redrawn (connected) to the changed target automatically.
- In the property (that is, association end) Specification window, change the **Type** property value.

To define the association end visibility

1. Open the association end Specification window.
2. From the **Visibility** list, select one of the visibility type.

The association end multiplicity describes how many entities are participating at each association end:

- 0 – zero and only zero.
- 1 – one and only one.
- 0..1 – zero or one.
- 0..* – from zero to any positive integer.
- 1..* – from one to any positive integer.
- * – any positive integer.

To place multiplicity values in association path ends

- Open the shortcut menu of a selected association end and click a desired multiplicity.
- Open association's shortcut menu, point to one of a desired association end (**Role of <class name>**), and then click the desired multiplicity value.
- Open the association **Specification** window and, from the **Multiplicity** list, select or type the multiplicity value for the desired association end.
- Perform the following steps:
    1. Open the association end Specification window.
    2. In the **Multiplicity** property value cell, type or select from the list a multiplicity value.
- **NEW!** Edit directly on the diagram pane:
    1. Select multiplicity area and press F2 to switch it to edit mode.
    2. Press Ctrl + Spacebar or Ctrl + Backspace to see available suggestions.

  Choose one of the suggestions or type the multiplicity value.

A qualifier is an attribute or a list of attributes whose values serve to partition the set of instances associated with an instance across an association. Qualifiers are attributes of an association. It is represented as a small rectangle attached to the end of an association path between the final path segment and the symbol of the classifier that it connects to. The qualifier rectangle is part of the association path, not part of the classifier. The qualifier rectangle drags with the path segments. The qualifier is attached to the source end of the association.

Copyright © 1998-2015 No Magic, Inc.

To add, edit, or remove a qualifier to/from an association end

1. Open the association end Specification window.
2. In the property group list, click the **Qualifiers** group and do one of the following:
   - To add a new qualifier, click the **Create** button. In the opened qualifier Specification window, define qualifier properties and click **Back** to return to the association end Specification window or click **Close** to exit the Specification window.
   - To edit a qualifier, select a qualifier and edit its properties in the association end Specification window or click the  button at the end of a qualifier row to edit qualifier properties in the qualifier Specification window.



Existing qualifiers are also displayed in the property group list under the **Qualifiers** group. Click the selected qualifier to open its Specification window, and define qualifier properties there.

   - To remove a qualifier, select a qualifier and click **Delete**.
3. After you have finished working with qualifiers, click **Close** to exit the Specification window.

If two classes are linked with an association path, both classes have an attribute of an opposite class type. This property can be displayed on the class shape as well as an association link.

To show association ends as attributes on linked class shapes

1. From the class shortcut menu, select **Symbol Properties** or press Alt+Enter. The **Symbol Properties** dialog opens.
2. In the **Attribute** category, click the **Show Association End as Attributes** property value cell.
3. Click one of the following display modes:
   - **All**. Properties and association paths will be displayed on the class shape.
   - **Without Association Symbol**. If an association symbol is deleted, the property will be displayed on the class shape.
   - **Do Not Show**. Neither property, nor association path will be displayed on the class shape.

An association end is defined as a property. It has attribute properties defined in the Specification window.

To define an association end name

- Open the shortcut menu of a selected association end and click **Edit Name**. The association end is marked for editing. Type or edit the name directly on the Diagram pane.
- Open association's shortcut menu, point to one of a desired association end (**Role of <class name>**), and then click **Edit Name**. The association end is marked for editing. Type or edit the name directly on the Diagram pane.
- Perform the following steps:
  1. Open the selected association end Specification window.
  2. Type an association end name in the **Name** property value cell.

Copyright © 1998-2015 No Magic, Inc.

The shared or composite aggregation kind can be assigned to the association end. A composite aggregation is represented as a filled diamond. A shared aggregation is represented as a hollow diamond.



Figure 521 -- Representation of composite and shared aggregations

To assign an aggregation kind to the association end

- Open association's shortcut menu, point to one of a desired association end (**Role of <class name>**), and then select one of the following:
  - **None**. No aggregation is assigned to the selected association end.
  - **Shared**.
  - **Composite**.
- On the Diagram pallet, click the **Composition** or **Aggregation** button and draw an appropriate path on the diagram.
- Right click the association path end and select **Shared** or **Composite** command from the shortcut menu.
- Perform the following steps:
  1. Open the selected association or association end Specification window.
  2. For the selected association end, click **Aggregation** property value cell to select an aggregation kind (**None**, **Shared**, or **Composite**).

NOTE

The **Aggregation** property is in the **All** properties display mode by default. For more information about properties display mode, see "Property Group Toolbar" on page 278.

To convert an association role to an attribute

1. On a diagram pane, select an association with a role.
2. From the shortcut menu, select the **Refactor** command, then **Convert To**, and **Attribute(s)**. The association is converted to the attribute.



Figure 522 -- Example of association role (on the left side) conversion to attribute (on the right side)

You can specify association end properties in the association end Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To convert association ends to ports

1. Right- click an association end(s).

Copyright © 1998-2015 No Magic, Inc.

2. On the shortcut menu, click **Refactor**, **Convert To**, and then **Port**.

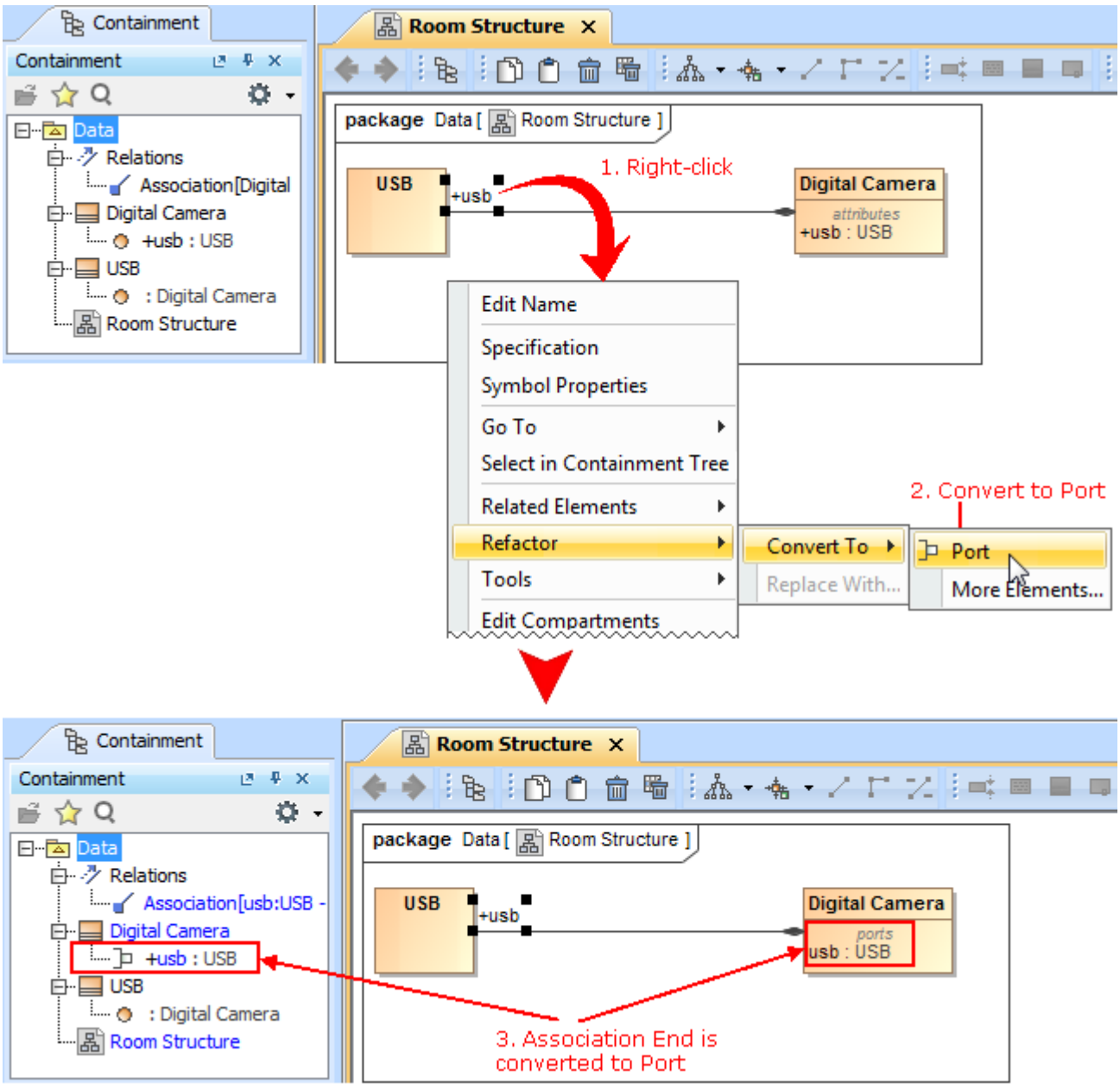


Figure 523 -- Example of converting Association End to Port

To show association ends as ports on the shape

1. On the diagram, right-click a shape, on which you want to display association ends as ports.
2. On the shortcut menu, click **Symbol Properties**.
3. In the **Symbol Properties** dialog, at the right-top corner, click to expand the menu and select the *All* mode.

Copyright © 1998-2015 No Magic, Inc.

4. Under the **Ports** category, click the **Show Association Ends as Ports** property cell and select *All*.

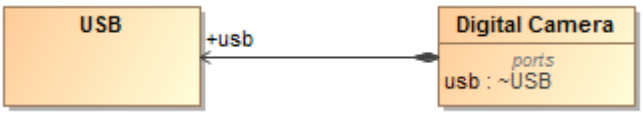


Figure 524 -- Example of displaying association end as port on shape

Association navigability

The association navigability indicates whether it is possible to traverse an association within an expression of a classifier to obtain the object or a set of objects associated with instances. The navigability is shown as an arrow that can be attached to the end of the path to indicate that the navigation is supported toward the classifier attached to the arrow.



By default, an association is navigable on both sides and its navigability is not visible.

A role indicates the role played by the class in terms of an association. The role name is placed at the association end, near the class playing that role. The role name at the implementation level maps to the reference name of the opposite class. Roles can have visibility (public, package, protected, and private).

To change the association navigability

Do one of the following:

- Open the association end Specification window, turn on the **Expert** property display mode, and do one of the following:
  - Set the **Navigable** property value to *true* to mark the association end as navigable.
  - Set the **Navigable** property value to *false* to mark the association end as not navigable.
- From the association shortcut menu, select **Role of <class name>** and then do one of the following:
  - Select **Navigable** to mark the association end as navigable.
  - Click to clear **Navigable** to mark the association end as not navigable.
- Open the association end shortcut menu and do one of the following:
  - Select **Navigable** to mark the association end as navigable.
  - Click to clear **Navigable** to mark the association end as not navigable.

To display the association navigability

- From both association ends shortcut menu, select **Show Navigability**.

In the following figure, the association is navigable on both sides and its navigability is visible.



Copyright © 1998-2015 No Magic, Inc.

Advancing actions: navigable owned association ends

The navigability describes the need for an object to access another object by navigating across the link. The association end can be owned by a classifier or an association. The association end owned by a classifier can be decorated with the dot. The absence of the dot signifies the ownership by the association.



In MagicDraw, the dot notation is not enabled by default. Please pay attention to this before making decisions about the association end ownership just from the model representation on a diagram.

To enable a dot notation

1. On the **Options** menu, click **Options**. The **Project Options** dialog opens.
2. In **General project options**, set the **Enable Dot Notation for Associations** value to *true.*

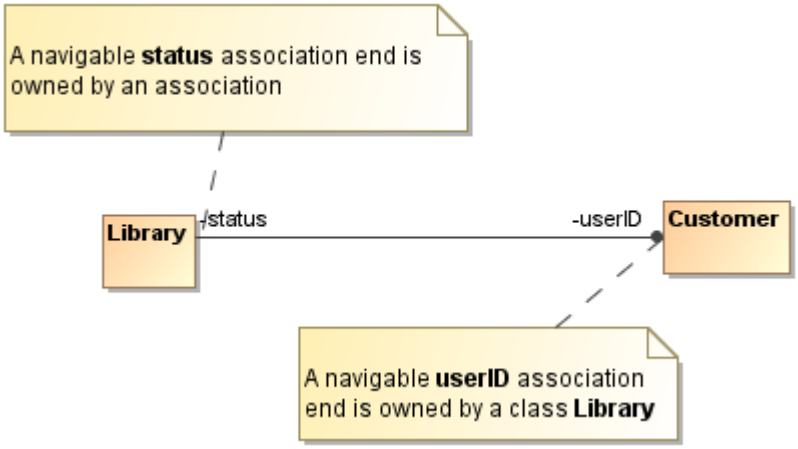


Figure 525 -- Example of dot notation

According to the UML 2.4.1 specification, the association ends owned by classes and associations is navigable. This improved functionality allows a proper management of the navigableOwnedEnd property for associations:

- The ownership of an association end can be changed manually.
- The navigability of association ends owned by the associations can be changed keeping the ownership.

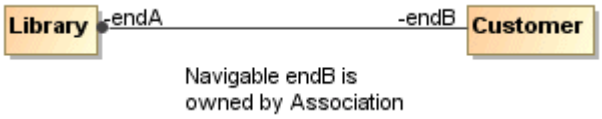
To change the ownership of an association end

Do one of the following:

- On the association end shortcut menu, click **Owned By** and then select the desired owner.
- In the association end Specification window, click the **Owned By** property value cell and then select the desired owner from the list.
- From the association shortcut menu, select the role of the desired association end, click **Owned By** and then select the desired owner.
- In the association Specification window, click the **Owned By** property value cell of the role with the desired association end and then select the desired owner from the list.

Copyright © 1998-2015 No Magic, Inc.

Association in Use Case Diagrams



An association shows the participation of an actor in a use case, for example, instances where an actor and a use case communicate with each other. This is the only relationship between actors and use cases. The association relationships are also known as communication associations.

Attribute

An attribute is a named property of a class that describes a range of values that can be held by the instances of that class.

You can specify attribute properties in the attribute Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



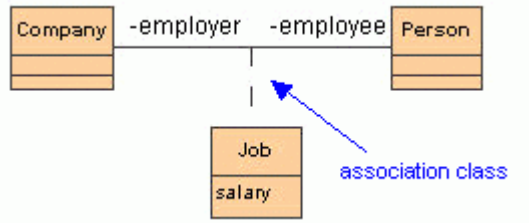
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To create a new attribute

- In the **Class** Specification window, click the **Attributes** property group and then click the **Create** button. The **Attribute** Specification window opens.
- In the Containment tree, right-click a class, from the shortcut menu, select **Create Element** and then **Property**.
- On the diagram, click the class shape, press the Ctrl+Alt+A shortcut keys and type the attribute's name.
- On the diagram, click the class shape and then click the *Create Property* button.
- On the diagram, click the class shape, then click the *Create Element* button. From the menu, select the **Property** command.

To create an association class

1. Draw two classes (for example A and B).
2. From the class diagram toolbar, expand the **Association** elements group and select the **Association Class** to draw.
3. Link the previously drawn classes with this path. An additional class with a dashed line will be created on the association between classes.



Copyright © 1998-2015 No Magic, Inc.

If an attribute type is another model class, this attribute can be represented as an association with a role (attribute name) between the owner class and the class of attribute type.

To change an attribute name

The attribute name must be unique in the class scope.

- Click the attribute in the selected class on the diagram pane or in the Browser tree. and type a new name.
- Change an attribute name in the **Attribute Specification** dialog.

To define the type of an attribute



The attribute type can be the other class, interface, or a primitive class, such as *int* or *double*.

- Open the **Specification** window of the attribute and click the cell of the **Type** property value. Then click . In the open dialog, select the type. Click **OK** when you are done.
- On the diagram, right-click the attribute, and then point to **Type**. In the open list, select the type or point to **New** to create a type.
- On the class shape on the diagram, click the attribute, and then on the smart manipulator toolbar, click the *Specify type* button. In the open list, select the type.
- On the class shape on the diagram, click the attribute, and then click the attribute the second time. After the attribute name, type the colon ":" and then type the name for the new type. Press **Enter** or click anywhere on the diagram. The type is created and assigned.



Type the colon ":", press **Ctrl+Space** or **Ctrl+Backspace**, and then in the open list, select the type.

- In the Containment tree, select an element and drag it to the attribute on the diagram.

To add additional information about the type of an attribute

1. Open the **Attribute Specification** window.
2. Click the **Show Expert Properties** button to enlarge a list of general available attribute properties.
3. In the **Type Modifier** list, select a particular sign in the:
   - & - one class has a reference to other model elements.
   - * - one class has a pointer to other model elements.
   - [] - one class has an array of other model elements.

To set the attribute visibility

Visibility Name	Function
Public '+'	An attribute can be accessed by any other elements.
Package '~'	An attribute can be accessed by elements from the same package.
Protected '#'	An attribute can be accessed from the inside of the selected class and classes derived from that class.
Private '-'	An attribute can be accessed only from inside of that class.

- Type '+', '~', '-', or '#' visibility marks just before an attribute name directly on a diagram.

Copyright © 1998-2015 No Magic, Inc.

1. Open the **Attribute** Specification window.
2. From the **Visibility** drop-down list, select the desired item (public, package, protected, and private.).



The attribute visibility is shown at the attribute signature.

To set an attribute scope

1. Open the **Attribute Specification** window.
2. Click the **Show Expert Properties** button to enlarge the list of available attribute properties.
3. Select the **Is Static** check box.

To set the attribute multiplicity

1. Open the **Attribute Specification** window.
2. Click the **Show Expert Properties** button to enlarge the list of available attribute properties.
3. Select or set the multiplicity value for the **Multiplicity** value.

To set the attribute changeability

The attribute changeability controls the access by operations on the class on the opposite end.

Name	Function
Is Read Only	When *false* - no restrictions on modifications. When *true* - the value can not be altered after the object is instantiated and its values initialized. No additional values can be added to a set.

1. Open the **Attribute Specification** window.
2. Click the **Show Expert Properties** button to enlarge the list of available attribute properties.
3. Select or click to clear the **Is Read Only** check box.

To convert an attribute to an association

1. On a diagram pane, select a class with an attribute or select an attribute itself.
2. From the class or attribute shortcut menu, select the **Refactor** command, then **Convert To**, and **Association**. On the diagram pane, the attribute is represented as an association with a role.



Figure 526 -- Example of attribute (on the left side) conversion to the association with the association role (on the right side)



You can move attributes back to the class by dragging and dropping them on the class shape.



To convert the attribute to the association:

- the attribute has to have a type specified.
- the attribute is not a member end of some association.

Copyright © 1998-2015 No Magic, Inc.

To convert an attribute to a port

1. Right- click an attribute or attributes.
2. On the shortcut menu, click **Refactor**, **Convert To**, and then **Port**.

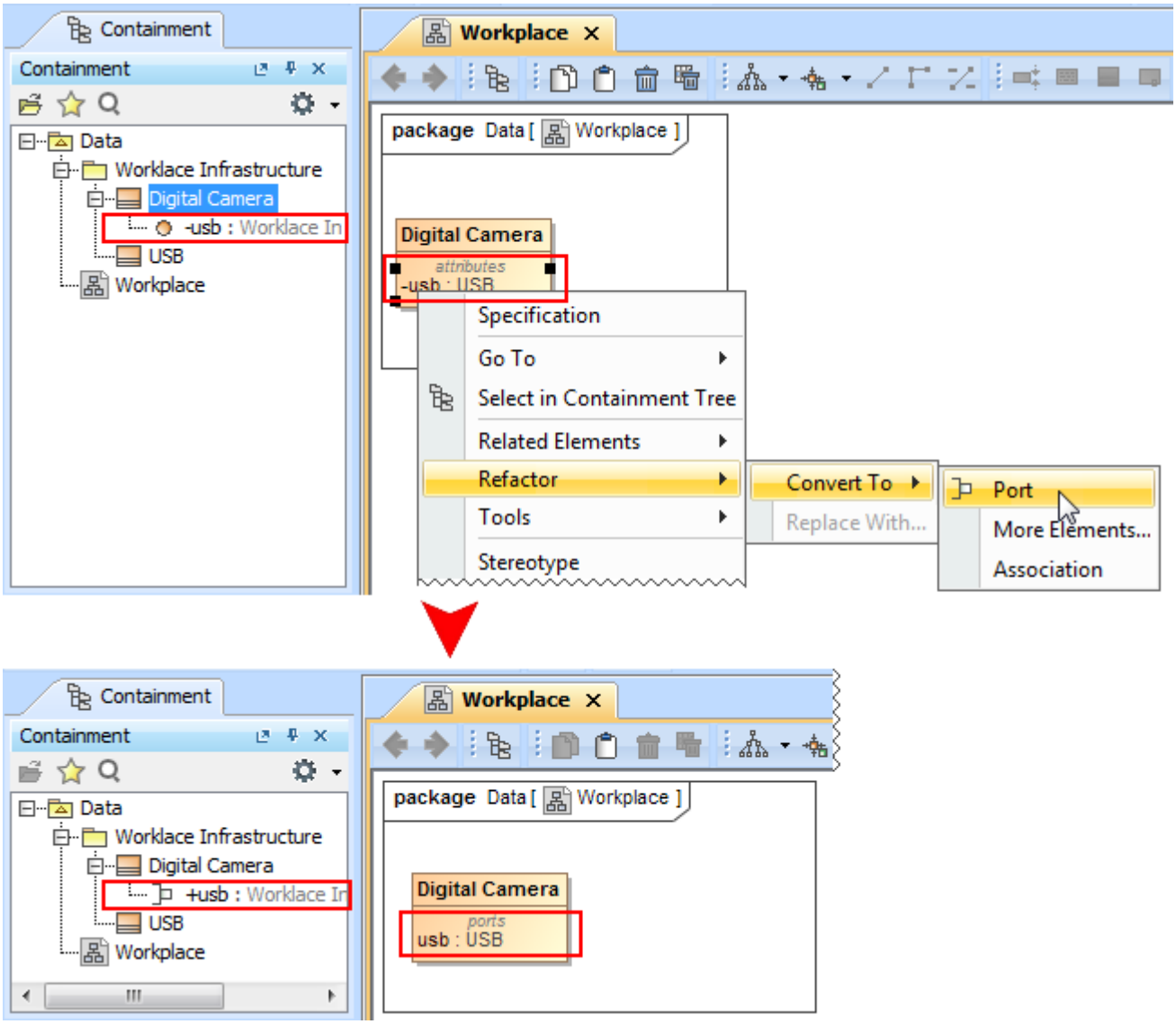


Figure 527 -- Example of converting Attribute to Port

Behavior

A behavior describes how the states of objects changes over time. Behavior types are as follows:

- Activity
- FunctionBehavior
- Interaction
- OpaqueBehavior
- Protocol State Machine
- State Machine

Copyright © 1998-2015 No Magic, Inc.

Behaviors are used:

- To specify an Entry, Do, and Exit activities of a State.



For more information,
see "Assigning behavior to state" on page 984.

- To specify an Effect of a Transition.



For more information,
see "Assigning Behavior Type" on page 1003.

The behaviors are displayed in the compartments of the following elements:

- Class
- Component
- Stereotype
- Node
- Actor
- Use Case
- Collaboration
- Interaction
- State Machine
- Activity
- Opaque Behavior
- Association Class

Behaviors are divided into two groups: classifier behavior and owned behaviors. Be informed, that the classifier behavior group can have only one behavior.



Figure 528 -- Two groups of behavior compartment

Assigning Behaviors

See references on how to assign the behavior for the following elements:

- Action. For more information, see "To assign a behavior to the call behavior action" on page 827 and "Creating Behavior Automatically in Activity Diagram" on page 703.
- State. For more information, see "Assigning behavior to state" on page 984.
- Transition. For more information, see "Assigning Behavior Type" on page 1003.
- Operation. For more information, see "To quickly create a behavior or a diagram for an operation" on page 955.
- See the following procedure, on how to assign the behavior for the rest of the elements (Class, Component, Stereotype, Node, Actor, Use Case, Collaboration, Interaction, State Machine, Activity, Opaque Behavior, Association Class).

Copyright © 1998-2015 No Magic, Inc.

To create and assign a behavior

Do one of the following:

- Right-click the element in the Model Browser, on the shortcut menu, click **Create Diagram** and then click to create Activity, Sequence, State Machine or other behavior diagram.
- Create a behavior in the element Specification window:
    1. Open the element Specification window and click the **Behaviors** property group.
    2. Click the **Create** button and then from the menu, select the behavior type.
    3. In the behavior Specification dialog, specify the behavior and click **Close**.



To create an operation for the behavior quickly, in the element Specification window, **Behaviors** property group, click the behavior and then click the **Create Operation** button.
Note that the **Create Operation** button is unavailable, if the **Specification** property is defined already.

Assigning Behavior Diagrams Automatically

When assigning a behavior of an Activity, Interaction, Protocol State machine, or State Machine type for a state or transition, the corresponding behavior diagram is created automatically:

- an Activity diagram for an Activity behavior
- a Sequence diagram for Interaction behavior
- a Protocol State Machine diagram for Protocol State Machine behavior
- a State Machine diagram for State Machine behavior

Double-click the assigned behavior on a state shape to open the diagram. You can also open the behavior diagram from the Model Browser.

To disable an automatic diagram creating

1. On the **Options** menu, click **Environment**. The **Environment Options** dialog opens.
2. In **General** options, **Editing** category, set the **Create a behavior diagram when selecting the Behavior Type for the Transition Effect and State Entry/Do/Exit activities** value to *false* (the default value is *true*).

You can specify behavior properties in the Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Related diagrams

Activity Diagram
Sequence Diagram
State Machine Diagram
Protocol State Machine Diagram

Related procedures

Formatting Symbols
Customizing Environment Options

Copyright © 1998-2015 No Magic, Inc.

Related references

Specification Window

Class

A class is drawn as a solid-outline rectangle with three compartments separated by horizontal lines. The top name compartment holds the class name and other general properties of the class (including stereotype); the middle list compartment holds a list of properties; the bottom list compartment holds a list of operations. The property and operation compartments are optional and you can suppress them.

A class is the descriptor for a set of objects with similar structure, behavior, and relationships. The model is concerned with describing the intention of the class, that is, the rules that define it. The run-time execution provides its extension, that is, its instances.

Classes are declared in the class diagrams and used in most of other diagrams. UML provides a graphical notation for declaring and using these classes as well as a textual notation for referencing classes within the descriptions of other model elements.

A class represents a concept within the system being modeled. It has a data structure, behavior, and relationships to other elements. The name of a class has a scope within the package in which it is declared and the name must be unique (among class names) within its package.

Working with classes

A general information about working with shapes is offered in section "Diagramming" on page 198.

You can specify class properties in the class Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To insert an inner element in the selected class

1. Double-click the selected class or select **Specification** from the class shortcut menu. The **Class** Specification window opens.
2. Click the **Inner Elements** tab and then click the **Create** button or press Insert. Select the element you wish to add from the list.
3. Click the selected element.
4. The corresponding Specification window opens. Define the class, use case or interface, and click **OK**.

To generate operations for setting or getting private data to the selected class

From the class shortcut menu, select **Tools** and then **Create Setters/Getters**. For a detailed description, see "Creating Setters/Getters" on page 464.

To control a list of operations and attributes that are visible on a diagram

Select **Edit Compartment** from the class shortcut menu. The **Compartment Edit** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

A class can be defined as active (a border to the class shape is added). An active class specifies whether an object of the class maintains its own thread of control.

To define a class as abstract and/or active

1. Open the **Class Specification** window.
2. In the **General** tab, select the **Is Abstract**, and/or **Is Active** check box.

To show members (attributes and operations) on the classifier shape according to the visibility

1. On the diagram, right-click the shape and then from the shortcut menu, select the **Symbol Properties**.
2. In the **Symbol Properties** dialog, change the **Show Members** property.

Creating A Structured Class

There is a way to create a piece of model with a single click: class, with a composite structure diagram inside it and hyperlink them. It can be useful for architects and system engineers.

The same applies to SysML Block and IBD.

To create a structured class:

3. Create a class diagram.
4. In the class diagram toolbar, right-click the **Class** button. A menu with available options opens. Select the **Structured Class** element.
5. Click on the diagram. The class element with a hyperlink to a composite structure diagram is created.
6. Select the class in the Browser to see its structure. To do this, choose **Select in Containment Tree** on the diagram from the class shortcut menu.

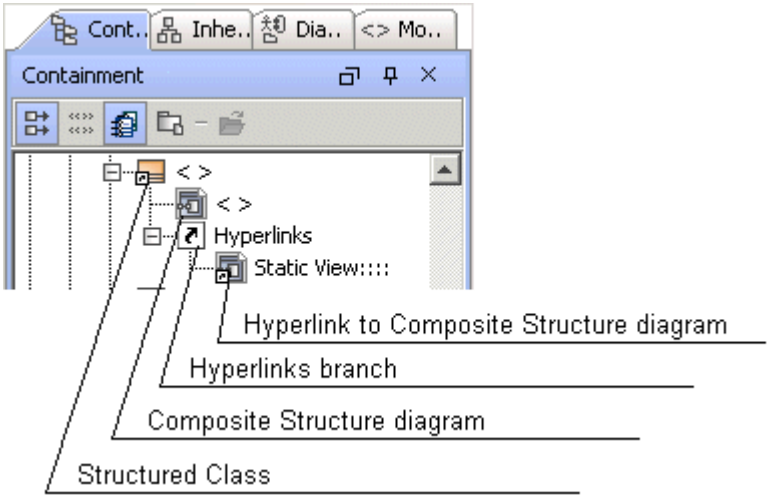


Figure 529 -- Example of structured class

See the above example of a structured class. In the Browser you can see the created class with the composite structure diagram inside it. The class is linked with the diagram. It means that after double click the diagram, or the Browser, the composite structure diagram will be opened.

TIP!

For more information about hyperlink, see "Defining Hyperlinks" on page 340.

Copyright © 1998-2015 No Magic, Inc.

The names of the class and the composite structure diagram are synchronized. Type a name for the class, for example, Order, and the name of the diagram is automatically changed to Order and vice versa. This is synchronization of a diagram name and its context name.

TIP! For more information about diagram name and its context synchronization, see "Diagram Name and its Context Name Synchronization" on page 206.

Design Patterns

You can create and edit the design patterns for the selected class. A detailed description of templates can be found in the Design Patterns of Reusable Object-Oriented Software.

To create the design pattern for the selected class

1. From the class shortcut menu, select **Tools**, and then **Apply Pattern**. The **Pattern Wizard** dialog opens.
2. Select the design pattern you want to apply and select the desired options. Click **OK**.

- Select the class and then **Apply Pattern** from the **Tools** menu.

TIP! For a detailed description of this dialog, see "Pattern Wizard" on page 459.

Collaboration

A collaboration describes the structure of elements that implement a certain behavior. In the collaboration the elements are the properties connected with the connectors.

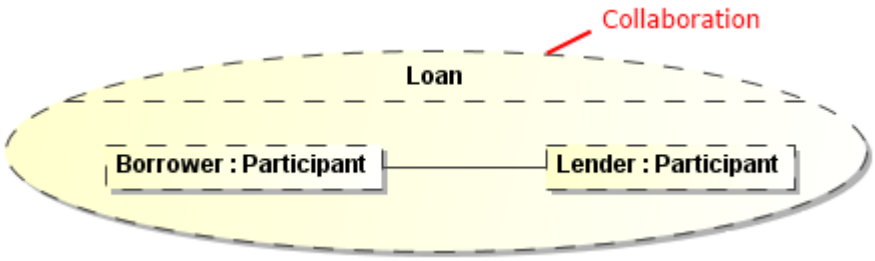


Figure 530 -- Example of collaboration

In the example, you can see the *Loan* collaboration. The collaboration contains two properties - *Borrower* and *Lender*. These are the properties (attributes) and the *Participant* is the type of the property. The *Borrower* and the *Lender* are connected with the connector relationship.

TIP! For more information on how to assign a behavior to a collaboration use, see "Behavior" on page 851.

To create the property and connector in the Collaboration

1. Create a collaboration symbol in a class diagram.
2. On the diagram, select the collaboration and open its shortcut menu.
3. Select the **Symbol Properties** command. The **Symbol Properties** dialog opens.
4. Click to clear the **Suppress Structure** check box.
5. Click **OK** to close the **Symbol Properties** dialog.

Copyright © 1998-2015 No Magic, Inc.

6. On the diagram pane or in the Model Browser select the element and drag-and-drop it to the collaboration symbol on diagram. The new property is created. The type of the property is the element that was dragged and dropped.
7. To create a connector between properties, expand the Composite Structure diagram pallet and click the Connector button. Draw the connector relationship between properties. Now you have the structure represented in the collaboration.

Collaboration properties

You can format collaboration symbol properties in the **Symbol Properties** dialog.

TIP!

For more information about symbol representation properties, see "Formatting Symbols" on page 317.

You can specify collaboration properties in the collaboration Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Related diagrams

Class Diagram

Composite Structure Diagram

Collaboration Use

A collaboration use represents the implementation of a pattern described by a collaboration in a specific situation. In this context, role binding means that the roles in the collaboration are bound by concrete elements. A collaboration can be used many times in different collaboration uses.

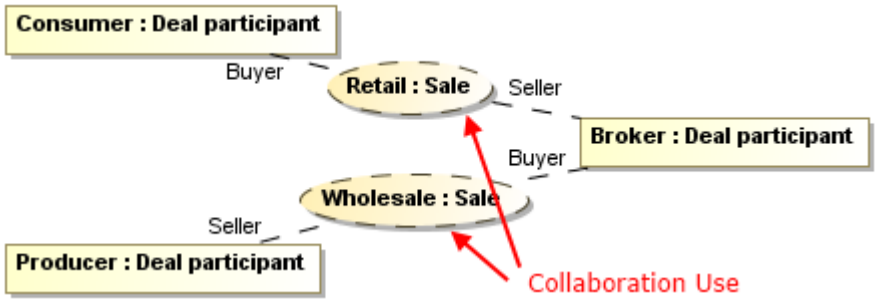


Figure 531 -- Collaboration Use example

To assign a collaboration to a collaboration use

Do one of the following:

- On the composite structure diagram pallet, click Collaboration Use, and then in the open dialog, select the collaboration or create a new one. Click **OK** when you are done.

Copyright © 1998-2015 No Magic, Inc.

- Open the Specification window of the collaboration use and click the cell of the **Type** property value. Then click and in the open dialog, select the collaboration. Click **OK** when you are done.
- On the diagram, right-click the collaboration use shape, and then point to **Type**. In the open list, select the collaboration or point to **New Collaboration** to create a collaboration.
- On the diagram, click a collaboration use shape, and then on the smart manipulator toolbar, click . Type the name of the collaboration and press **Enter** or click anywhere on the diagram. The collaboration is created and assigned.
- On the diagram, click an collaboration use shape, and then press **Ctrl+T**. Type the name of the new collaboration and press **Enter** or click anywhere on the diagram. The collaboration is created and assigned.
- On the diagram, click a collaboration use shape, then click the name area. Type the ":" colon and then type the collaboration name. Press **Enter** or click anywhere on the diagram pane. The collaboration is created and assigned.
- In the Containment tree, select the collaboration and drag it to the diagram pane. The collaboration use with the assigned collaboration is created.

Collaboration use properties

You can format collaboration use symbol properties in the **Symbol Properties** dialog.



For more information about symbol representation properties, see "Formatting Symbols" on page 317.

You can specify collaboration use properties in the collaboration use Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Related diagrams

Composite Structure Diagram

Combined Fragment

A fragment is an abstract notion of the most general interaction unit. It is a piece of an interaction. Each interaction fragment is conceptually like an interaction by itself. Using the Combined Fragment, a fragment of a sequence diagram can be separated.

MagicDraw represents twelve kinds of fragments: Alternatives, Loop, Option, Parallel, Break, Negative, Critical Region, Consider, Ignore, Weak Sequencing, Strict Sequencing, and Assertion.

You can specify association end properties in the association end Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Copyright © 1998-2015 No Magic, Inc.

Comment

A comment is a UML element used to specify various remarks of the element.

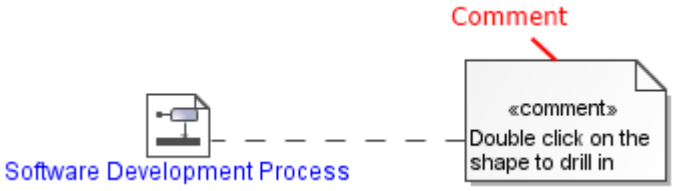


Figure 532 -- Example of comment

In the diagram pallet, you can find the **Comment** command under the **Note** command.

A text in the comment can be displayed as a plain text or as an HTML text.

You can format comment symbol properties in the **Symbol Properties** dialog.



For more information about symbol representation properties, see "Formatting Symbols" on page 317.

You can specify a comment in the **Comment** Specification window. In the same window, you can find the description of each comment property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To create a comment in the element Specification window

1. Open the Specification window of a selected element.
2. Expand the **Documentation/Hyperlinks** property group and select **Comment**.

Copyright © 1998-2015 No Magic, Inc.

3. In the **Comment** pane, click the **Create** button. The **Comment** specification window opens.

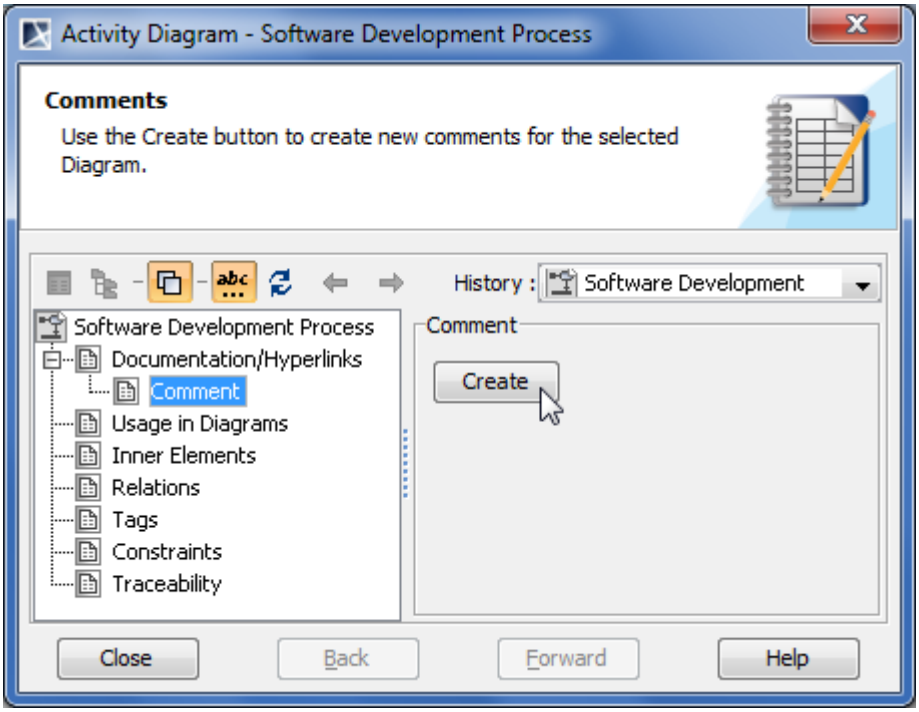


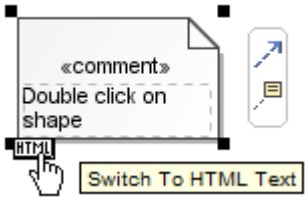
Figure 533 -- Creating Comment in element Specification window

4. Type the comment content as the **Body** property value.
5. To create the comment symbol on the diagram pane, select the created comment in **Comment** specification window property group list and drag it on the diagram pane. The comment symbol is created.
6. Close the Specification window.

To switch a comment text to HTML or to Plain text

Do one of the following:

- On the diagram pane, select the comment symbol and click the button that appears on the lower-left corner of the shape:
  - The **HTML** button - to switch the text in the comment to the HTML text.
  - The **Plain** button - to switch the text in the comment to the plain text.



- Open the selected comment's shortcut menu and select or click to clear the **HTML Text** command.

You can edit the HTML text in the comment using the HTML editor.



For more information about working with HTML editor, see "HTML Editor" on page 422.

Copyright © 1998-2015 No Magic, Inc.

To display an anchored element documentation in a comment

1. Anchor the comment to the element symbol.
2. Select the comment on the diagram pane and open the comment's shortcut menu.
3. Select the **Retrieve Documentation** command. The anchored element documentation is displayed on the comment's symbol.

TIP! You can edit element documentation directly on the comment symbol.

NOTE For more information about drawing the anchor to the comment, see the procedure "To draw an anchor from a symbol" on page 838.

To display constraints and tagged values on a comment symbol

1. Anchor the comment to the element symbol. Select the comment on the diagram pane and open comment's shortcut menu.
2. To expand the comment shortcut menu, at the end of the comment's shortcut menu click the little arrow.
3. Select the **Show Constraints,** or **Show Tagged Values** command. Element properties, constraints and tagged values is displayed on the comment symbol on the diagram pane in the separated compartments.

NOTE You can also display owner and stereotype on the comment symbol.

To choose which constraints, or tagged values to display on a comment symbol

1. Anchor the comment to the element symbol.
2. Select the comment on the diagram pane and open the comment's shortcut menu.
3. To expand the comment shortcut menu at the end of comment's shortcut menu, click the little arrow.
4. Select the **Edit Compartment** command and choose one of the following command: Constraints or Tagged Values. The **Compartment Edit** dialog opens (see the following figure).

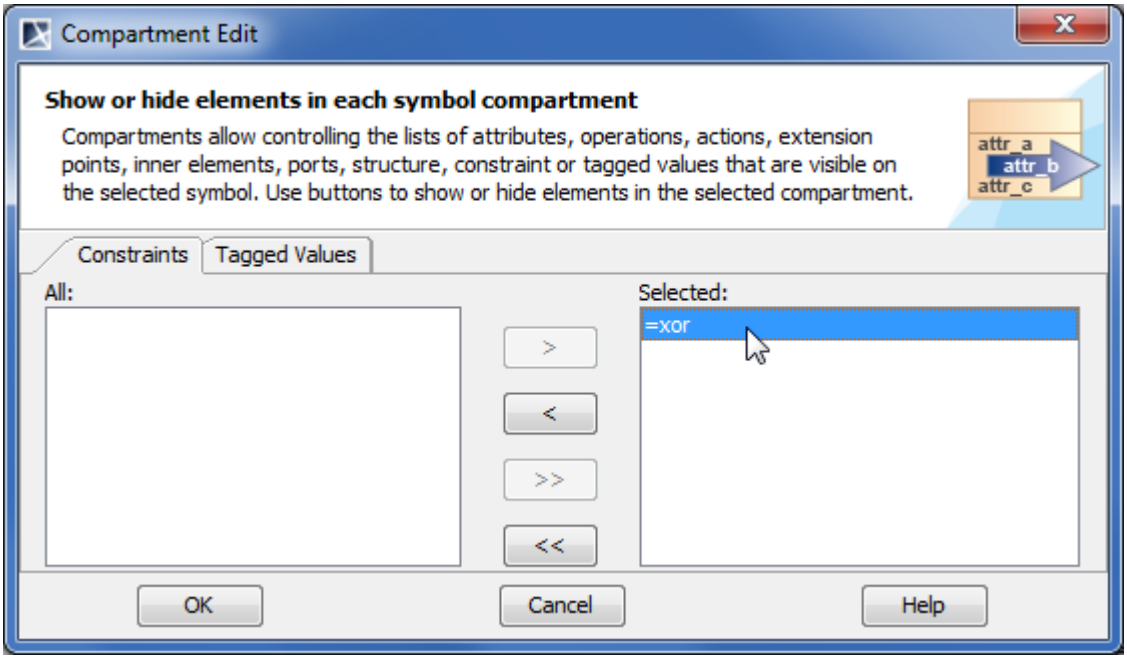


Figure 534 -- Compartment Edit dialog

Copyright © 1998-2015 No Magic, Inc.

5. In the **All** list select the property you want to display on the comment.
6. Click the > button to move the property to the **Selected** list.
7. Click **OK**. The selected property or properties is displayed on the comment.

Related procedures

Formatting Symbols
HTML Editor
To draw an anchor from a symbol

Related dialogs

Specification Window

Component

A component represents all kinds of elements that make up the system. A component can always be considered as an autonomous unit within a system or subsystem. It is a module having the following features:

- A component is a replaceable and independent part of the system performing a specific action.
- A component acts in a context of a well-defined architecture.
- Components interact among each other using interfaces.

Components provides compartments for listing its provided and required interfaces, realizations, and artifacts.

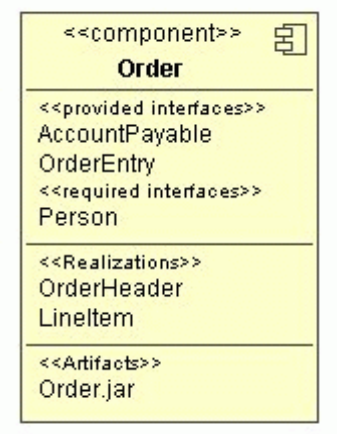


Figure 535 -- An example of a component

TIP!

For more information about working with symbols, see “Diagramming” on page 198.

You can specify component properties in the component Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Copyright © 1998-2015 No Magic, Inc.

To show/hide the interfaces, realizations, and artifacts on the component's shape

Do one of the following:

- On the diagram, right-click the component's shape and then from the shortcut menu, select **Symbol Properties**. In the **Symbol Properties** dialog, click to select or clear the **Suppress Interfaces**, **Suppress Realizations**, or **Suppress Artifacts** check box.
- On the diagram, click the component's shape and then click the *Compartments* button. Then in the menu, click to select or clear the **Interfaces**, **Realizations**, or **Artifacts** check box.

Connector

Specifies a link that enables communication between two or more instances. This link can be an instance of an association, or it can represent the possibility of the instances being able to communicate because their identities are known by virtue of being passed on as parameters, held in variables or slots, or because the communicating instances are the same instance.

The link can be realized by something as simple as a pointer or by something as complex as a network connection. In contrast to the associations, which specify the links between any instance of the associated classifiers, the connectors specify the links between instances playing the connected parts only.

Each connector can be attached to two or more connectable elements, each representing a set of instances.

You can specify connector and connector end properties in the connector and connector end Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

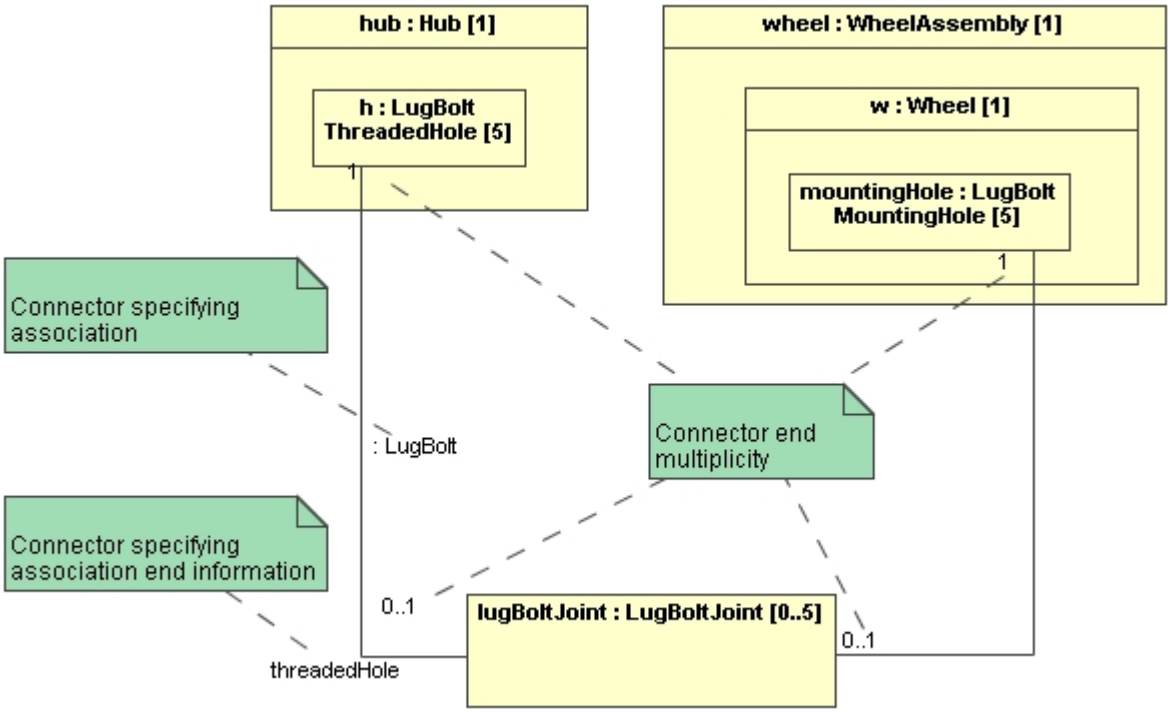


- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Copyright © 1998-2015 No Magic, Inc.

Drawing a connector end

The connector end information can be displayed on the diagram.



To select a port automatically when drawing a connector from a port to a part

1. Select a port on the diagram.
2. On the smart manipulator toolbar, click the Connector button.
3. Click a part on the diagram. The **Select Port** menu appears.

NOTE The menu appears if a new port can be created on the part or the part has hidden ports.

4. Click one of the following commands:
   - **None** - if you want to connect connector straight to the part.
   - On a port in the list. Hidden ports of the part are listed. The selected port will be represented on the diagram and the connector will be connected to it.
   - **New Port** - to create a new port with the same name, type and multiplicity as the port from which the connector is drawn.
   - **New Nested Port** - to create a new nested port. Note that the command exists only if the connector is drawn from the nested port.

TIP! You can draw a connector to a port too and the menu will appear. As a result, the nested port will be created.

Copyright © 1998-2015 No Magic, Inc.

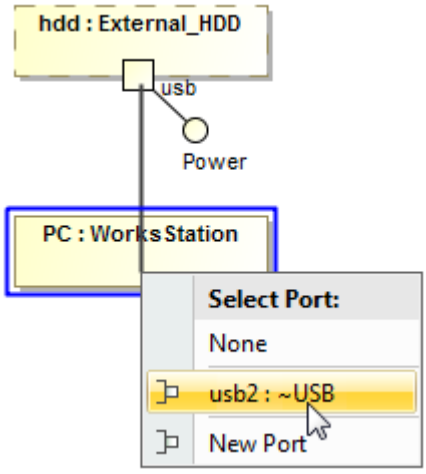


Figure 536 -- Example of menu

TIP! See the "*composite structure diagram.mdzip"* sample that can be found in *<MagicDraw installation folder>/samples/diagrams*.

Related diagrams

Composite Structure Diagram

Constraint

A Constraint represents additional semantic information attached to the constrained elements. It is an assertion that indicates a restriction that must be satisfied by a correct design of the system. The constrained elements are those elements required to evaluate the constraint specification. In addition, the context of the Constraint can be accessed, and can be used as the namespace for interpreting names used in the specification.

A Constraint is represented as a linguistic, enclosed in braces ({}), statement in some formal (OCL, C++, and other), or a natural language. There are 14 standard constraints in UML such as association, global, and parameter. You can also define your own constraints.

Time and duration constraints

The Time Constraint specifies the combination of min and max timing interval values.

Copyright © 1998-2015 No Magic, Inc.

The Duration Constraint defines a value specification that specifies the temporal distance between two time instants.

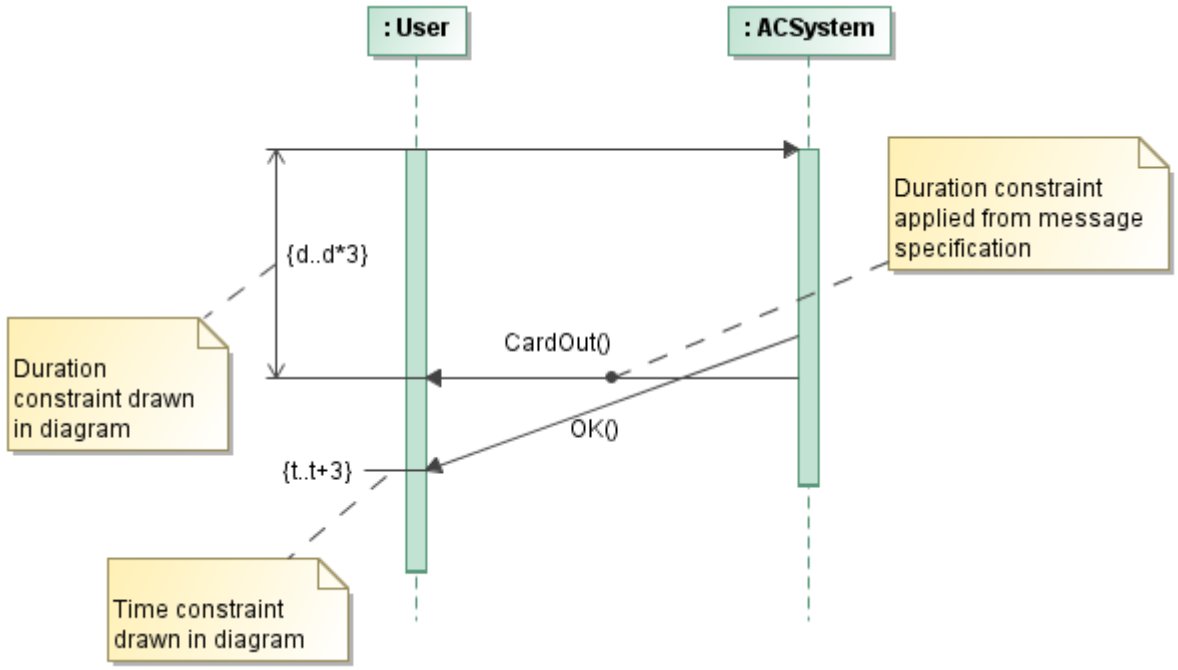


Figure 537 -- Example of constraints

Working with Constraints

You can edit constraints in the **Constraints** property group of the containing element Specification window, in the constraint's Specification window, or double click the selected constraint and edit it on the diagram pane directly.

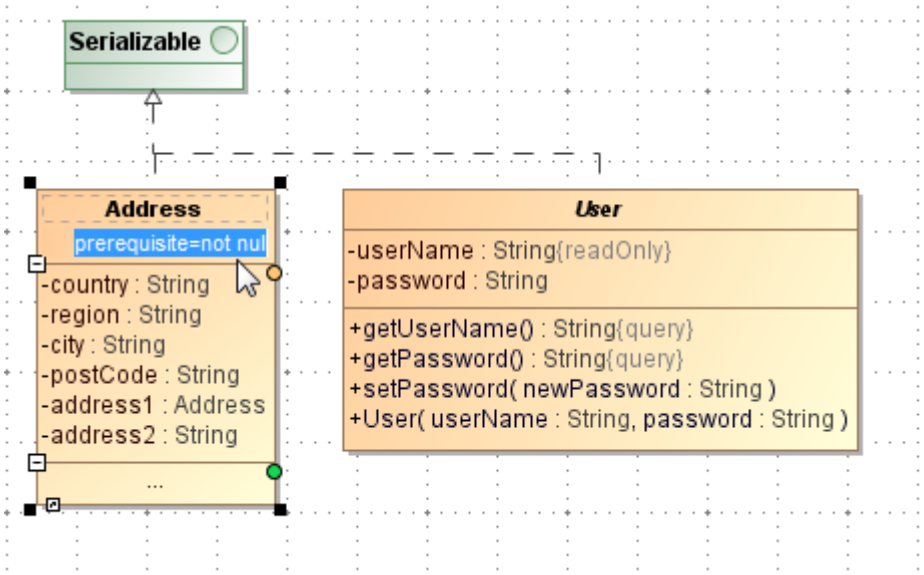


Figure 538 -- Constraint selected for editing

NOTE

To see the constraint expression, make sure the **Show Constraint** value is set to true, in the **Symbol Properties** dialog.

In this section, see the following procedures:

- To define a new constraint

Copyright © 1998-2015 No Magic, Inc.

- To apply a constraint to an element

To define a new constraint

Do one of the following:

- On the element shortcut menu in the Browser, click **Create Element** and then **Constraint**.
- Click the **Inner Elements** property group in the Specification window and then click the **Create** button (select **Constraint** from the list, if needed). Specify the constraint in the constraint Specification window.
- Click the **Constraints** property group in the Specification window and then click the **Create** button. On the menu, click a duration, interaction, time, or simple constraint. The constraint Specification window appears.



If a constraint is displayed in the **Constraints** property group of the element Specification window, it means this constraint is applied for the element. If it is displayed in the **Inner Elements** list, this constraint is only owned by the element.

To apply a constraint to an element

1. Click the **Constraints** property group in the Specification window for each model element and click the **Apply** button.
2. The **Select Elements** dialog opens. Select a constraint existing in the model from the **All Data** tree and click the **Add** button to move it to the **Selected Objects** list.
3. Click **OK** when you are done.

The Constraint properties

You can specify constraint properties in the constraint Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

OCL

Object Constraint Language (OCL) is a formal language used to express constraints. These typically specify the invariant conditions that must hold for the system being modeled.

Expressions can be used in a number of places in a UML model:

- To specify the initial value of an attribute or association end.
- To specify the derivation rule for an attribute or association end.
- To specify the body of an operation.
- To indicate an instance in a dynamic diagram.
- To indicate a condition in a dynamic diagram.
- To indicate the actual parameter values in a dynamic diagram.

There are four types of constraints:

- An *invariant* is a constraint that states a condition that must always be met by all instances of the class, type, or interface. The invariant is described using an expression that evaluates to true if the invariant is met. Invariants must be true all the time.

Copyright © 1998-2015 No Magic, Inc.

- A *precondition* to an operation is a restriction that must be true at the moment the operation is going to be executed. The obligations are specified by the postconditions.
- A *postcondition* to an operation is a restriction that must be true at the moment the operation has just been executed.
- A *guard* is a constraint that must be true before a state transition discharged.

Invariants on attributes

The simplest constraint is an invariant on an attribute. Suppose a model contains a class *Customer* with an attribute age, then the following constraint restricts the value of the attribute:

```
context Customer inv:
age >= 18
```

Invariants on associations

One may also put constraints on the associated objects. Suppose a model contains the class *Customer* that has an association to the class *Salesperson* with the role name *salesrep* and multiplicity 1, then the following constraint restricts the value of the attribute knowledge level of the associated instance of *Salesperson*:

```
context Customer inv:
salesrep.knowledgelevel >= 5
```

Collections of objects

In most cases the multiplicity of an association is not 1, but more than 1. Evaluating a constraint in these cases will result in a collection of instances of the associated class. Constraints can be put on either the collection itself, e.g. limiting the size, or on the elements of the collection. Suppose in a model the association between *Salesperson* and *Customer* has the role name *clients* and multiplicity 1..* on the side of the *Customer* class, then we might restrict this relationship by the following constraints:

```
context Salesperson inv:
clients->size() <= 100 and clients->forAll(c: Customer | c.age >= 40)
```

Pre- and postconditions

In the pre- and postconditions the parameters of the operation can be used. Furthermore, there is a special keyword result which denotes the return value of the operation. It can be used in the postcondition only. For example an operation *sell* was added to the *Salesperson* class.

```
context Salesperson::sell( item: Thing ): Real
pre: self.sellableItems->includes( item )
post: not self.sellableItems->includes( item ) and result = item.price
```

Derivation Rules

Models often define derived attributes and associations. A derived element does not stand alone. The value of a derived element must always be determined from other (base) values in the model. Omitting the way to derive the element value results in an incomplete model. Using OCL, the derivation can be expressed in a derivation rule. In the following example, the value of a derived element *usedServices* is defined to be all services that have generated transactions on the account:

```
context LoyaltyAccount::usedServices : Set(Services)
derive: transactions.service->asSet()
```

Initial Values

In the model information, the initial value of an attribute or association role can be specified by an OCL expression. In the following examples, the initial value for the attribute points is 0, and for the association end transactions, it is an empty set:

```
context LoyaltyAccount::points : Integer
init: 0
```


Copyright © 1998-2015 No Magic, Inc.

```
context LoyaltyAccount::transactions : Set(Transaction)
init: Set{}
```

Body of Query Operations

The class diagram can introduce a number of query operations. The query operations are operations that have no side effects, i.e. do not change the state of any instance in the system. The execution of a query operation results in a value or set of values without any alterations in the state of the system. The query operations can be introduced in the class diagram, but can only be fully defined by specifying the result of the operation. Using OCL, the result can be given in a single expression, called a body expression. In fact, OCL is a full query language, comparable to SQL. The use of body expressions is an illustration thereof.

The next example states that the operation *getCustomerName* will always result in the name of the card owner associated with the loyalty account:

```
context LoyaltyAccount::getCustomerName() : String
body: Membership.card.owner.name
```

To check OCL syntax according to OCL grammar

1. Open the constraint Specification window.
2. Click the **Specification** property value.
3. Click the "..." button in the property value cell. The **Edit** Specification window opens.
4. In the **Language** list, click **OCL** and select the **Check OCL syntax** check box. In the **Body** box, incorrect expression will be underlined in red.

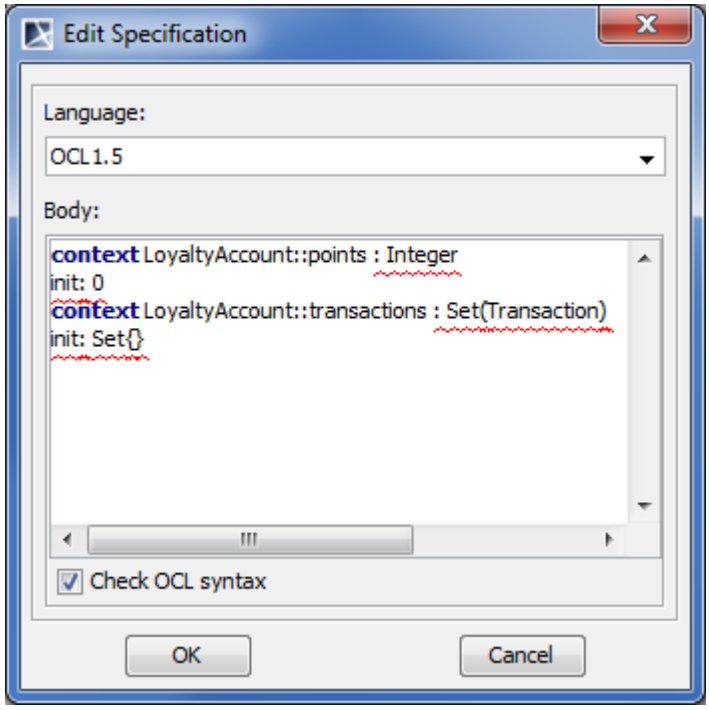


Figure 539 -- Checking OCL syntax

Constraint path

A constraint path is a graphical representation of a constraint element. The constraint path is shown on the diagram pane as a dashed arrow from one shape to other. To the constraint path you can assign a particular

Copyright © 1998-2015 No Magic, Inc.

constraint. The constraint assigned to the constraint path is labeled in braces: *{}*. See an example of the constraint representation in the following figure.

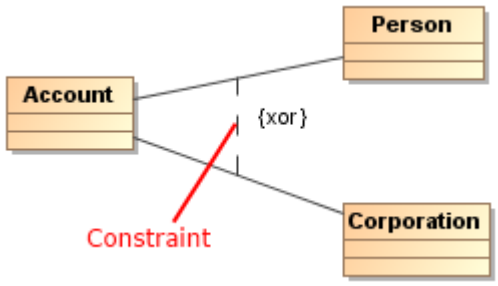


Figure 540 -- The Constraint Path example

You can display on the constraint path the constraint direction arrow. The direction of the arrow represents a relevant information within the constraint. The client (the tail of the arrow) is mapped to the first position and the supplier (arrowhead) is mapped to the second position in the constraint.

You can format the constraint symbol properties in the **Symbol Properties** dialog.

TIP!

For more information about symbol representation properties, see "Formatting Symbols" on page 317.

After applying the constraint to the constraint path, you can specify the constraint in the **Constraint** specification window. In the same window, you can find the description of each constraint property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To draw a constraint path

1. In the diagram pallet, the **Common** category, expand the **Anchor** command and select the **Constraint** command.
2. On the diagram pane click the shape from which you want to draw the constraint path.
3. Click the shape to which you want to create the constraint path.

To add a constraint expression to the constraint path

From the constraint path shortcut menu, choose **Select Constraint** and the choose one of the following:

- Select **New Constraint** to create a new constraint.
- Choose an already created constraint from the list.

To display a constraint direction arrow

1. Select the constraint path on the diagram pane and invoke the constraint's path shortcut menu.

Copyright © 1998-2015 No Magic, Inc.

2. Select the **Show Arrow** command. The direction of the constraint path will be displayed. See an example in the following figure.

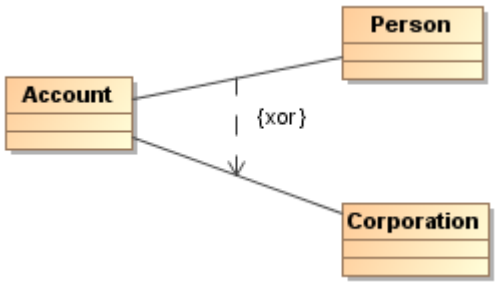


Figure 541 -- Example of constraint path with direction arrow

Related procedures

Formatting Symbols

Related dialogs

Specification Window

Containment

The containment relationship is used to represent containment of elements on the diagram pane - a containment path replaces nesting representation. For example, instead of drawing on the diagram pane one package nested into the other, you can draw a containment relationship from one package to another.

That is, the containment relationship is not only a graphical representation - the containment relationship means nesting of elements. See an example in the following figure.

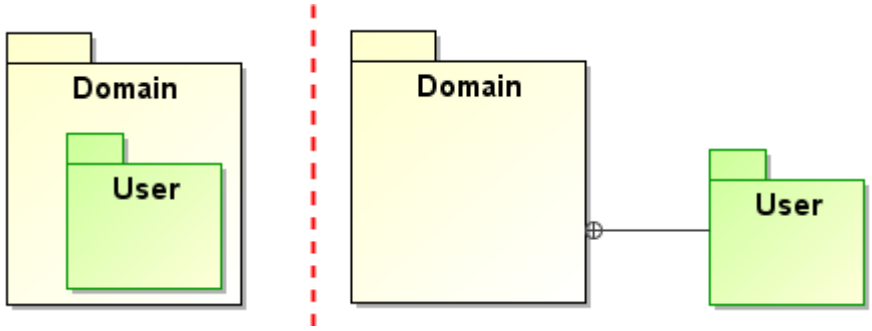


Figure 542 -- Example of package nesting (on the left) and package nesting using containment relationship (on the right)

In the example above, the containment end with a circle plus shows a containing element. So, the *User* package is a contained element and the *Domain* package is a containing element - the owner of the *User* package is the *Domain* package.

You can format the containment symbol properties in the **Symbol Properties** dialog.



For more information about symbol representation properties, see "Formatting Symbols" on page 317.

Related concepts

Containment Tree

Copyright © 1998-2015 No Magic, Inc.

Related diagrams

Class Diagram
Use Case Diagram
Component Diagram
Object Diagram
Package Diagram
Composite Structure Diagram

Related procedures

Containment Tree
Formatting Symbols

Containment Tree

When drawing more than one containment relationship, the containment tree can be created (see the following figure).

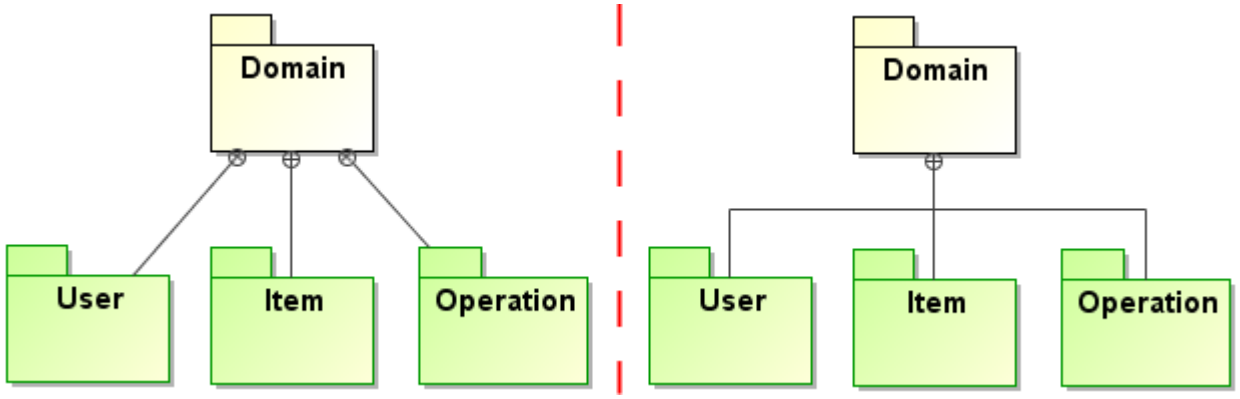
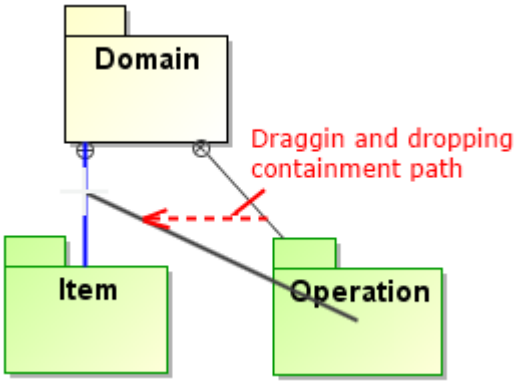


Figure 543 -- Example of more than one containment (on the left) and example of containment tree (on the right)

To group containment paths into a tree

1. Draw a containment path from each contained element to the containing element.
2. Do one of the following:
   - Select the containment relationship. Click and hold the little square (the end of the containment path) which is next to the containment end with circular plus. Drag and drop the selected containment end on the other containment path. While dragging, the containment path becomes blue that indicates the allowable connection (see the following figure). Containment paths merge together.

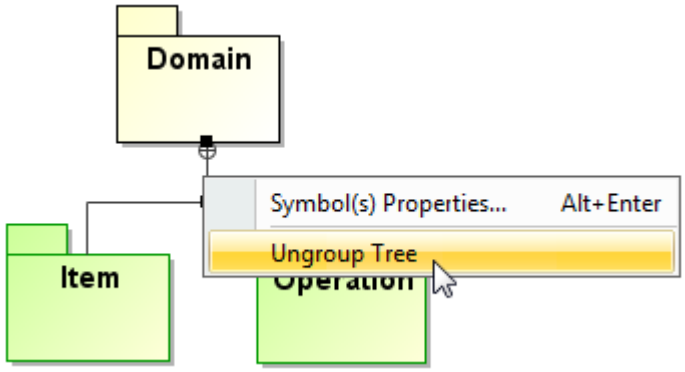


Copyright © 1998-2015 No Magic, Inc.

- Open the shortcut menu of the containing element and click **Refactor** > **Make Sub Tree**.

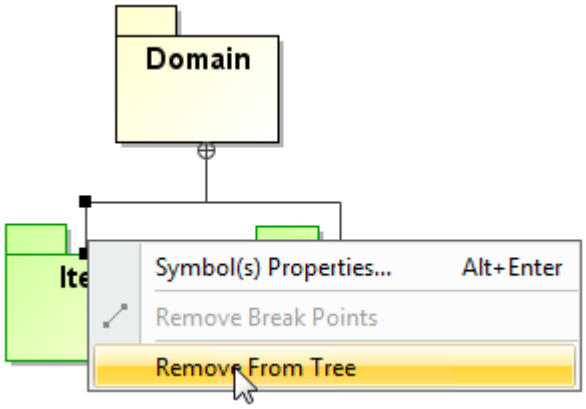
To ungroup a containment tree

- Right-click the containment tree's end with circled plus pointing to a containing element and, in the opened shortcut menu click the **Ungroup Tree** command.



To separate the containment path from the containment tree, do one of the following

- Select the containment path that you want to remove from the tree. Open its shortcut menu and click the **Remove From Tree** command. The contained element is added to the containing element separately.



- Drag one containment path to another element:
  1. On the diagram pane, select the vertical part of the particular containment path.
  2. Click and hold the upper little square (see the following figure).
  3. Drag and drop end of the containment path to the other containing element.The containment path is removed from the containment tree and owner of the containing element has been changed.

Related concepts

Containment

Related procedures

Formatting Symbols

Copyright © 1998-2015 No Magic, Inc.

Data type

A data type is a type whose values have no identity; that is, they are pure values. It is a classifier and inherits the general features of the classifier: visibility, generalizable element properties, and operations.

MagicDraw provides the following predefined data types: boolean, byte, char, date, double, float, int, Integer, Real, long, short, void, and String.

You can also create Enumeration or Primitive data types.

To create a new Data Type, including Enumeration or Primitive

- From the Browser, select **New** from the class or **Create Element** from the package, subsystem, or model shortcut menu, and then select **Data Type**, **Enumeration**, or **Primitive**.
- In the **Class**, **Package**, **Subsystem**, and **Model** Specification widows, **Inner Elements** tab, click **Create** and select a data type.

You can specify data type properties in the data type Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Enumeration

The enumeration defines a kind of data type whose range is a list of predefined values, called enumeration literals. An Enumeration can contain operations, but they must be pure functions (this is the rule for all data type elements).

You can specify enumeration properties in the enumeration Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window..



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

An enumeration literal defines an element of the run-time extension of an Enumeration data type. It has no relevant substructure, therefore, it is atomic.

To add an enumeration literal

Do one of the following:

- In the **Enumeration** Specification window, click the **Enumeration Literals** property group and then click the **Create** button. The **Enumeration Literal** Specification window appears.
- In the Containment tree, right-click an enumeration and then from the shortcut menu, select **Create Element** and **Enumeration Literal**.
- On the diagram, click the enumeration shape and then click the *Create Enumeration Literal* smart manipulator.
- On the diagram, click the enumeration shape, then click the *Create Element* button. From the menu, select the **Enumeration Literal** command.

Copyright © 1998-2015 No Magic, Inc.

To suppress the enumeration literals

Do one of the following:

- On the diagram, select the enumeration symbol, and then click the ⊟ *Suppress Enumeration Literals* button.
- On the diagram, right-click the enumeration and from the shortcut menu, select **Symbol Properties**. Then in the **Symbol Properties** dialog, click to select the **Suppress Enumeration Literals** check box.

To open the **Enumeration Literal Specification** window

1. Open the **Enumeration** Specification window.
2. Click the **Enumeration Literals** property group, double-click the desired literal, or click the **Edit**, or **Create** button.

Primitive

A primitive defines a predefined data type without possessing any relevant UML substructure; that is, it has no UML parts. A primitive data type can have an algebra as well as operations defined outside of UML (for example, mathematically). The primitive data types used in UML include Integer, Unlimited Integer, Real, and String.

Decision Node

Decisions are made using guard conditions. They help protect transitions that depend on a guarding condition. The symbol used for the decision is a large diamond shape, which can have one or more incoming transitions and two or more outgoing transitions.

A decision in an activity diagram is used much like a choice or junction point in the state diagrams. Decision points allow to separate the transition paths. Merges allow to merge the transition paths back together. The symbol used for the merge is the same as for the decision.

Dependency

A dependency is a relationship signifying that a single or a set of model elements requires other model elements for their specification or implementation. This means that the complete semantics of the depending elements is either semantically or structurally dependent on the definition of the supplier element(s).

A dependency is shown as a dashed arrow between classes or packages. The model element at the tail of the arrow (the client element) depends on the model element at the arrowhead (the supplier element). The arrow can be labeled with an optional stereotype and an optional individual name.



You can also draw a dependency between a class and other class elements, such as attributes and operations.

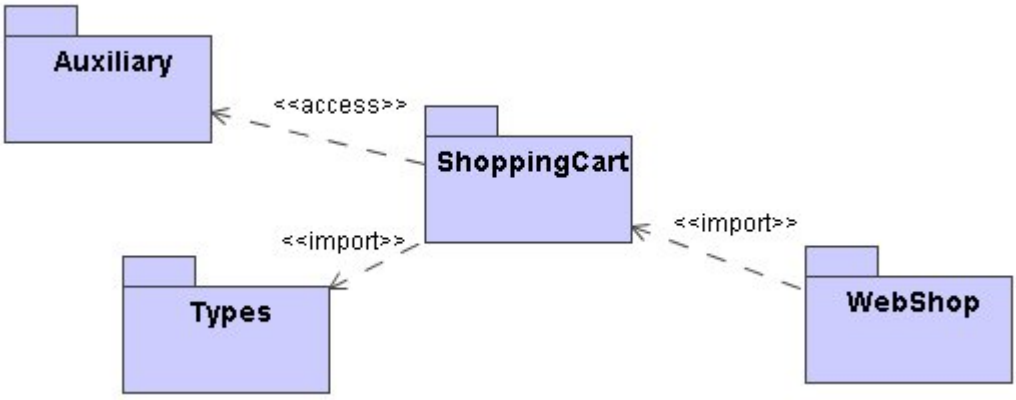


For more information about working with the symbols, see "Diagramming" on page 198.

Copyright © 1998-2015 No Magic, Inc.

Example of the dependency relationships

See the example of dependency relationships in the Figure below.



The Dependency and Its Kinds Specification windows

Dependency, abstraction, and usage relationships defined in the dialog of the same structure. They differ from one another only by the corresponding Specification name.

You can specify dependency properties in the dependency Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Template Binding dependency

A template binding represents a relationship between a templateable element and a template. A template binding specifies the substitutions of actual parameters for the formal parameters of the template.

You can specify template binding dependency properties in the template binding dependency Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Abstraction

An abstraction is a relationship that relates two elements or sets of elements that represent the same concept at different levels of abstraction or from different viewpoints. In the metamodel, an abstraction is a dependency in which there is a mapping between the supplier and the client.

Define an abstraction relationship in the **Abstraction Specification** window.

TIP!

For a detailed description of this dialog, see "The Dependency and Its Kinds Specification windows" on page 876.

Copyright © 1998-2015 No Magic, Inc.

Usage

A usage is a relationship in which one element requires another element (or set of elements) for its full implementation or operation. In the metamodel, a usage is a dependency in which the client requires the presence of the supplier.

Define a usage relationship in the **Usage Specification** window.



For a detailed description of this dialog,
see "The Dependency and Its Kinds Specification windows" on page 876.

Package Merge

A package merge is a directed relationship between two packages, that indicates that the contents of the two packages are to be combined. It has a dependency relation with the applied stereotype «*merge*».

Define a merge relationship in the **Dependency Specification** window.



For a detailed description of this dialog,
see "The Dependency and Its Kinds Specification windows" on page 876.

Package Import

A package import is defined as a directed relationship that identifies a package whose members are to be imported by a namespace. It is a relationship between an importing namespace and a package, indicating that the importing namespace adds the names of the members of the package to its own namespace. It is dependency relation with applied stereotype «*import*».

Define an import relationship in the **Dependency Specification** window.



For a detailed description of this dialog,
see "The Dependency and Its Kinds Specification windows" on page 876.

To draw the Package Import link, select the Package Import path to draw in the Class diagram toolbar, from the Abstraction group.

Element Import

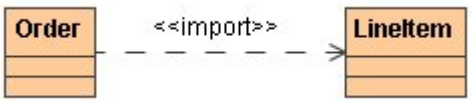
An element import is defined as a directed relationship between an importing namespace and a packageable element. The name of the packageable element or its alias is to be added to the namespace of the importing namespace. It has a dependency relation with the applied stereotype «*import*».

Define an import relationship in the **Dependency Specification** window.



For a detailed description of this dialog,
see "The Dependency and Its Kinds Specification windows" on page 876.

To draw the Element Import link, select the Element Import path to draw in the Class diagram toolbar, from the Abstraction group.



Copyright © 1998-2015 No Magic, Inc.

Access

An access relationship shows that elements can only be accessed from a package, and it cannot be referenced.

To draw an Access link

1. In the Class diagram toolbar, from the Abstraction group, select the Package Import path to draw.
2. Open the **Package Import Specification** window and set the **Visibility** property to *private.*

Deployment

To draw a deployment relationship

- In the Deployment diagram, click the **Deployment** button and draw a deployment link from a node to an artifact.

To display the deployed artifacts on the node instance shape

Do one of the following:

- On the diagram, select the node instance symbol, and then click the *Compartments* button. In the menu, click to select the **Deployed Elements** check box.
- On the diagram, right-click the node instance and from the shortcut menu, select **Symbol Properties**. Then in the **Symbol Properties** dialog, click to clear the **Suppress Deployed Elements** check box.

Deployment Specification

The Deployment Specification is a type of Artifact.

To draw the Deployment Specification on the diagram pane

In the Component (or Deployment) diagram toolbar, right-click the Artifact button group. In the open list, select the Deployment Specification to draw.

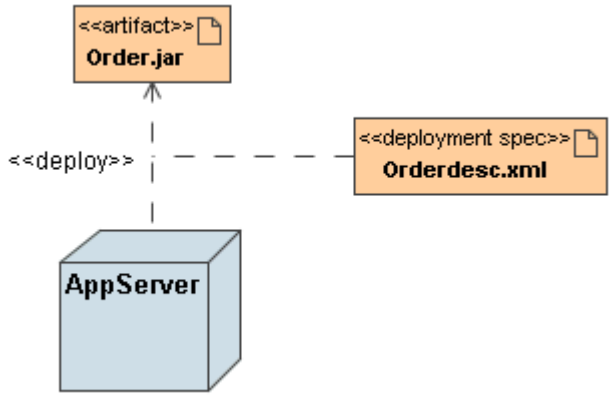
The Deployment Specification is a general mechanism to parameterize a Deployment relationship.

To specify the Deployment relationship

1. Create the Deployment Specification.

Copyright © 1998-2015 No Magic, Inc.

2. Between the Deployment relationship and Deployment Specification draw the Dependency relationship. The Dependency is drawn without an arrow.



Event

An event is the specification of some occurrence that may potentially trigger effects by an object, that is, an event shows what should happen to change a particular state in a system. There are the following event types: Any Receive Event, Call Event, Change Event, Signal Event, and Time Event.

Events are important in diagrams which represent a behavior of a system. These diagrams are listed in the following table.

Diagram name	Event usage description
Activity Diagram	To specify an event type for the Accept Event Action.
State Machine Diagram	To specify a event type for the Transition, or transition to self.
Protocol State Machine Diagram	To specify an event type for the protocol transition, or protocol transition to self.

When specifying an event type for a transition, you can type the command straight on the transition path on the diagram pane. The same assignment is valid for a transition to self, protocol transition, and protocol transition to self.

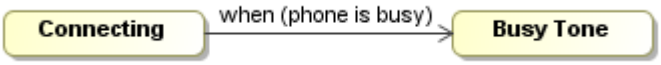
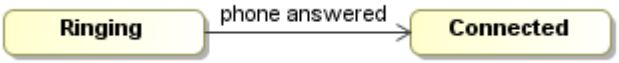
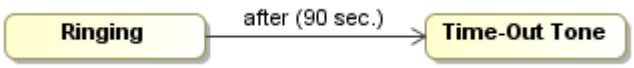


For more information, see "To assign an event type straight on a transition path" on page 1002.

Event types, their functions, samples, and command syntax are described in the following table.

Name	Function and Sample	Command syntax
Any Receive Event	A trigger for an AnyReceiveEvent is triggered by the receipt of any message that is not explicitly handled by any related trigger.	*all*
Call Event	A call event specifies the receipt by an object of a message invoking a call of an operation. See an example:  In this example, the call event type is specified with the *create()* operation.	<*operation* ()>

Copyright © 1998-2015 No Magic, Inc.

Name	Function and Sample	Command syntax
Change Event	A change event specifies a change in the system configuration that makes a condition true. See an example:  In this example, the change event type is specified, and its Change expression property is specified as *phone is busy*.	*when* (<*expression*>)
Signal Event	A signal event represents the receipt of an asynchronous signal instance. A signal event may, for example, cause a state machine to trigger a transition. See an example:  In this example, the signal event type is specified, and its Signal property is specified as phone answered.	<*signal name*>
Time Event	A time event specifies a point in time. At the specified time, the event occurs. There are two possible types of event occurrences: at a relative time and at an absolute time. See an example:  In this example, the relative time event is specified, and its *When* property is specified as *90 sec*. For more information, see "Specifying a time for a time event" on page 881.	• *after* (<*time*>) - an event occurrence at a relative time • *at* (<*time*>) - an event occurrence at an absolute time

Related diagrams

Activity Diagram

State Machine Diagram

Protocol State Machine Diagram

Related procedures

Formatting Symbols

To assign an event type straight on a transition path

Specifying a time for a time event

Related references

Specification Window

Related resources

Example of the State Machine diagram in *<MagicDraw installation directory>\samples\diagrams\state machine diagram.mdzip*

Example of the Activity diagram in *<MagicDraw installation directory>\samples\diagrams\activity diagram.mdzip*

Specifying a time for a time event

A time event specifies an instance in time by an expression. The expression might be absolute or it might be relative to some other point in time. Relative time events must always be used in the context of a trigger and the starting point is the time at which the trigger becomes active.

Copyright © 1998-2015 No Magic, Inc.

A relative time trigger is specified with the keyword “after” followed by an expression that evaluates to a time value, such as “after (5 seconds).” An absolute time trigger is specified with the keyword “at” followed by an expression that evaluates to a time value, such as “Jan. 1, 2012, Noon.”

You can change the time expression by changing the **Is Relative** value (see the following procedure).

To specify a time for a time event

1. Open the selected **Transition** Specification window or **Accept Event Action** Specification window having the TimeEvent type assigned.
2. In the **Trigger** category, do one of the following:



If you do not see the **Event Type** property, click the **+** button near the **Trigger** category to expand its content.

- To specify the event occurrence at an absolute time, set **Is Relative** value to *false*.The time event is specified as absolute, and the command syntax changes from *at ()* to *after* ().
- To specify the event occurrence at a relative time, set **Is Relative** value to *true*.The time event is specified as relative, and the command syntax changes from *after()* to *at* ().



Figure 544 -- Specifying time for TimeEvent in Transition Specification window

Related diagrams

Activity Diagram

State Machine Diagram

Protocol State Machine Diagram

Related procedures

Formatting Symbols

To assign an event type straight on a transition path

Related references

Event

Specification Window

Related resources

Example of the State Machine diagram in *<MagicDraw installation directory>\samples\diagrams\state machine diagram.mdzip*

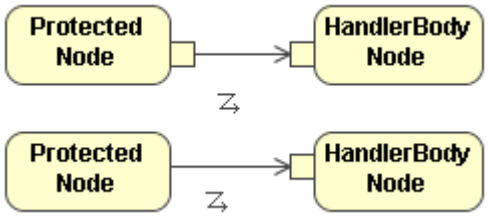
Example of the Activity diagram in *<MagicDraw installation directory>\samples\diagrams\activity diagram.mdzip*

Copyright © 1998-2015 No Magic, Inc.

Exception Handler

An Exception Handler is an element that specifies a body to execute in case the specified exception occurs during the execution of the protected node.

The Exception Handler may be drawn from an Output Pin to an Input Pin or from an Action to an Input Pin:



Extend

An extend use case indicates how and when the behavior defined in the extending use case can be inserted into the behavior defined in the extended use case. The extend relationship is a relationship from an extending use case to the extended use case.

The extended use case owns one or more specific extension points. An extension point identifies a point in the behavior of a use case where that behavior can be extended.



For more information about adding an extension point, see "Adding extension points" on page 1005.

An extend relationship can have a condition. If the condition is true, the extend action is performed.

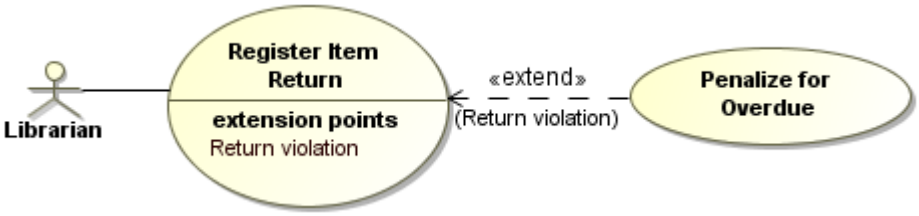


Figure 545 -- Example of extend relationship

In the preceding example, the *Penalize for Overdue* use case extends the *Register Return* use case. The *Register Return* use case owns the *Return violation* extension point.

You can format the extend symbol properties in the **Symbol Properties** dialog.



For more information about symbol representation properties, see "Formatting Symbols" on page 317.

You can specify extend properties in the extend Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Copyright © 1998-2015 No Magic, Inc.

Related diagrams

Use Case Diagram

Related procedures

Formatting Symbols

Related dialogs

Specification Window

Related resources

Example of the Use Case diagram in *<MagicDraw installation directory>\samples\diagrams\use case diagram.mdzip*.

Flow Final Node

It is a final node that terminates a flow and destroys all tokens that arrive at it. It has no effect on other flows in the activity.

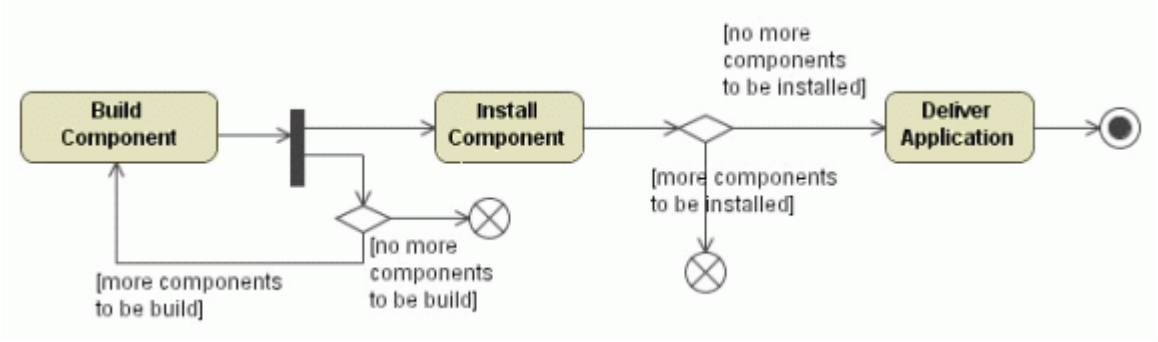


Figure 546 -- Example of flow final node element notation

Fragment

MagicDraw represents twelve kinds of fragments: Alternatives, Loop, Option, Parallel, Break, Negative, Critical Region, Consider, Ignore, Weak Sequencing, Strict Sequencing, and Assertion.

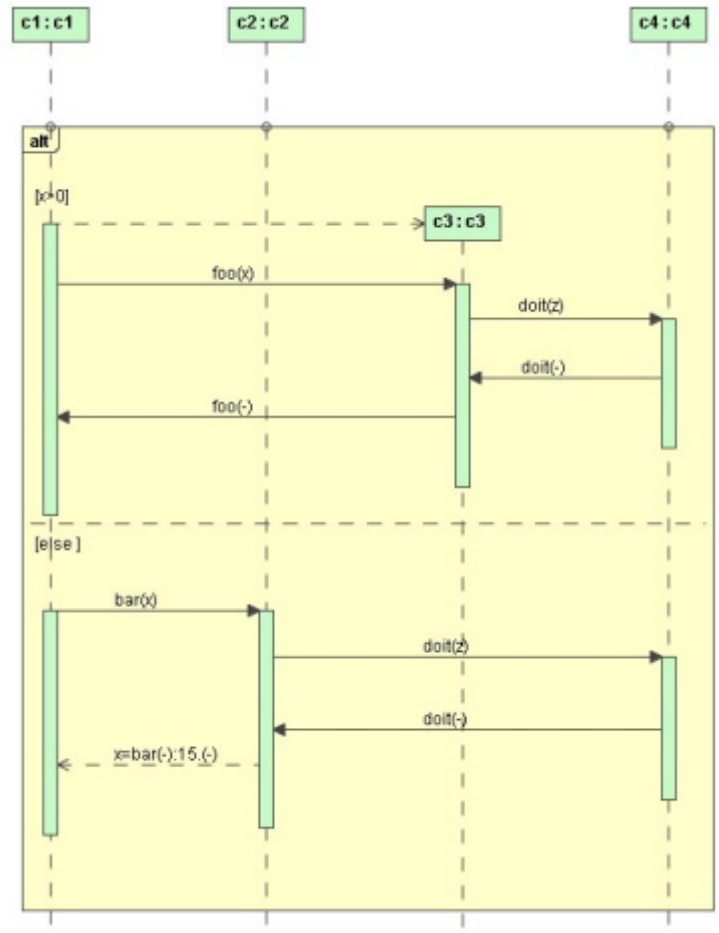
Alternative Fragment

The alternative fragment models *if…then…else* constructions.

Copyright © 1998-2015 No Magic, Inc.

To draw an Alternative Fragment

- In the Sequence diagram toolbar, select the Alternatives element to draw.



Combined Fragment

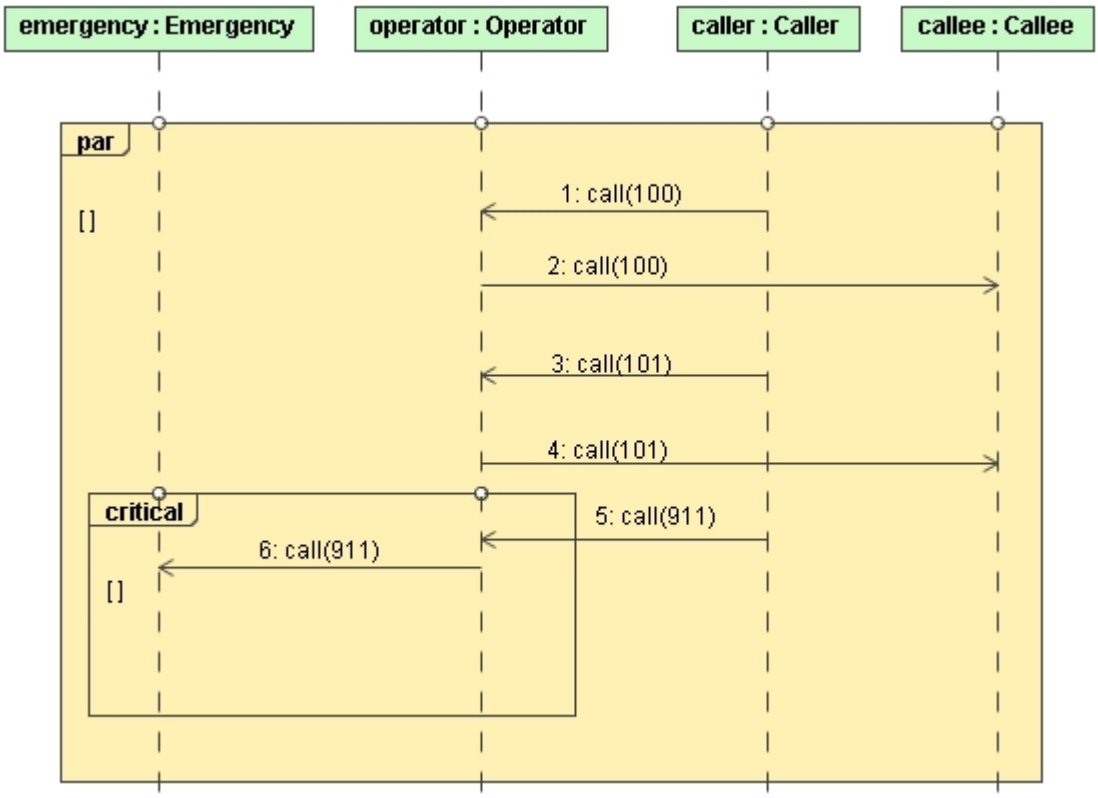
The UML Combined Fragment element allows the expressions of interaction fragments to be defined in the sequence diagram. The combined fragments provide a means to define special conditions and subprocesses for any sections of lifelines in the sequence diagram by specifying an area where the conditions or subprocesses apply. Using the Combined Fragment, a fragment of the sequence diagram can be separated.

To draw a Combined Fragment

- In the Sequence diagram toolbar, select the Alternatives element to draw.
- From the combined fragment shortcut menu select the **Covered Lifelines** command and in the **Covered Lifelines** dialog, select the lifelines to display.
- In the Sequence diagram toolbar, **Options** group, select the Parallel combined fragment to draw.

Copyright © 1998-2015 No Magic, Inc.

- Draw the Critical Region combined fragment from the Sequence diagram toolbar, **Option** button group.



Function Behavior

The function behavior allows modeling external functions that take only inputs and produce outputs. It has no effect on the specified system.

To create the function behavior

1. In the Model Browser, right-click the root package *Data*.
2. In the shortcut menu, select **Create Element** > **Function Behavior** .

Gate

MagicDraw allows the display of messages leaving or entering a sequence diagram, interaction use, or combined fragment. The Gate is a connection point for representing a message from the outside to the current sequence diagram, interaction use, or combined fragment.

Gates can be used in three cases:

- For passing and returning arguments to InteractionUse, which calls some Interaction.
- For displaying “exceptions“ as messages that stops an interaction execution and leaves it.
- For “calling“ sequence blocks represented as CombinedFragments.

Copyright © 1998-2015 No Magic, Inc.

Gate has no notation. Gates are created as message ends when drawing messages to/from a diagram frame, an interaction use, or a combined fragment.

Parent topic: "Modeling Elements" on page 823.

Usage in diagrams: "Sequence Diagram" on page 683.

Related topics:

"Model" on page 935.

"Working with Lifelines in Sequence Diagrams" on page 910.

"Combined Fragment" on page 858.

"Interaction Use" on page 904.

To create a formal gate

- Draw a call, send, create, or delete a message from the diagram frame.
- Draw a reply message to the diagram frame.
- Draw a call, send, create, or delete a message from the combined fragment (inside combined fragment).



- You can view the created gates of message in the **Message** Specification window. In the **Send Event** list box you can see formal gate and the **Receive Event** lists the actual gate.
- The Gate uses text from the message as an identification name. For example, a message name or a message operation.

To draw create an actual gate of the formal gate

- Draw a call, send, create, or delete a message to the interaction use, which refers to the diagram with the formal gates. The **Select Formal Gate** dialog opens.
- Draw a reply message from the interaction use, which refers to the diagram with the formal gates. The **Select Formal Gate** dialog opens.
- Draw a call, send, create, or delete a message to the combined fragment (outside combined fragment). The **Select Formal Gate** dialog appears.



You can also view the formal and actual gates in the gates **Interaction** Specification window, **Interaction Use** dialog, **Combined Fragment** dialog, and the **Actual Gates** and **Formal Gates** panes.

To select a formal gate for the actual gate

1. Draw a message to invoke the **Select Formal Gate** dialog (see "To draw create an actual gate of the formal gate" on page 886).
2. Select one of the listed formal gates and click OK. An actual gate is created.

- Or from the message shortcut menu, select the **Select Formal Gate** command. The **Select Formal Gate** dialog opens.



The **Select Formal Gates** command exists only if there are formal gates.

Copyright © 1998-2015 No Magic, Inc.

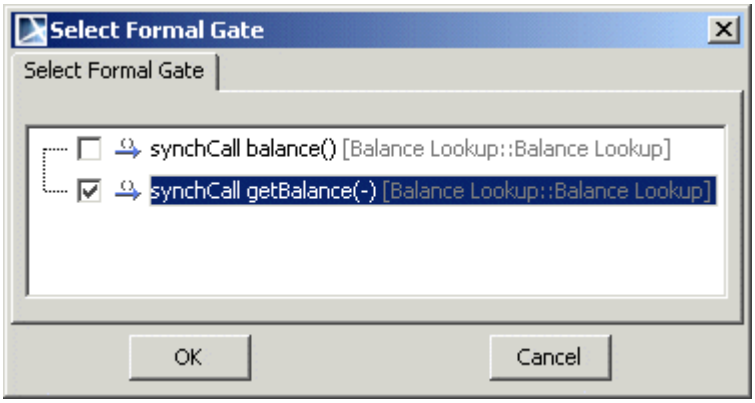


Figure 547 -- Select Formal Gate dialog

The formal gate and actual gate usage in the sequence diagram

See the following figure where the *getBalance* message is drawn from the diagram frame to the *theirBank* lifeline. The *getBalance* message has a gate.

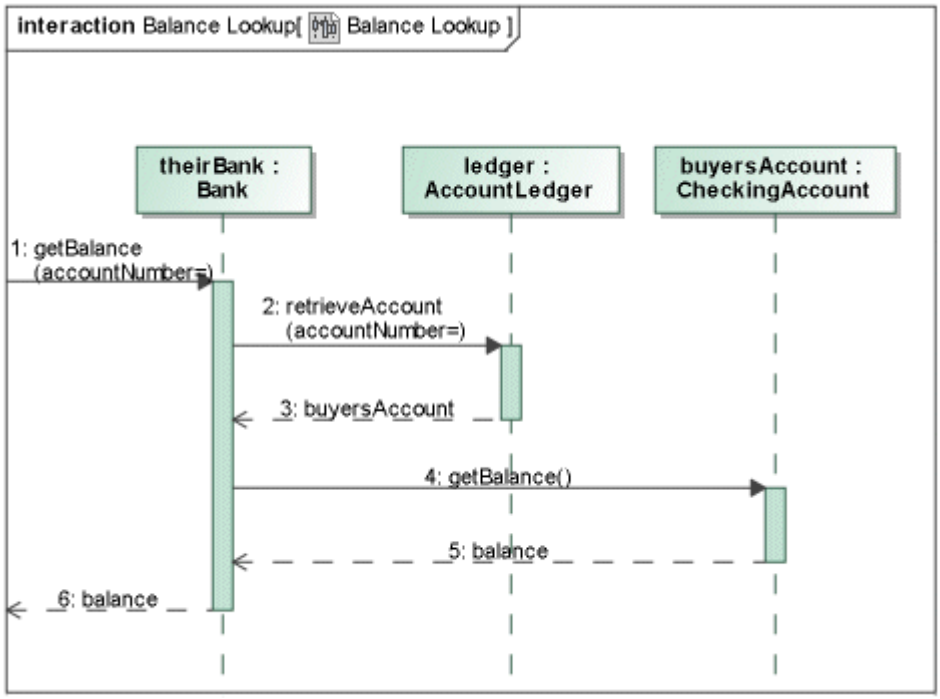


Figure 548 -- Formal gates usage in Sequence diagram

Copyright © 1998-2015 No Magic, Inc.

See the following figure where the actual gate is presented. The *Balance Lookup* interaction use refers to the *Balance Lookup* sequence diagram. The *getBalance* message (see the 2nd message) has selected the formal gate and automatically repeats the data of the *getBalance* message from the *Balance Lookup* diagram.

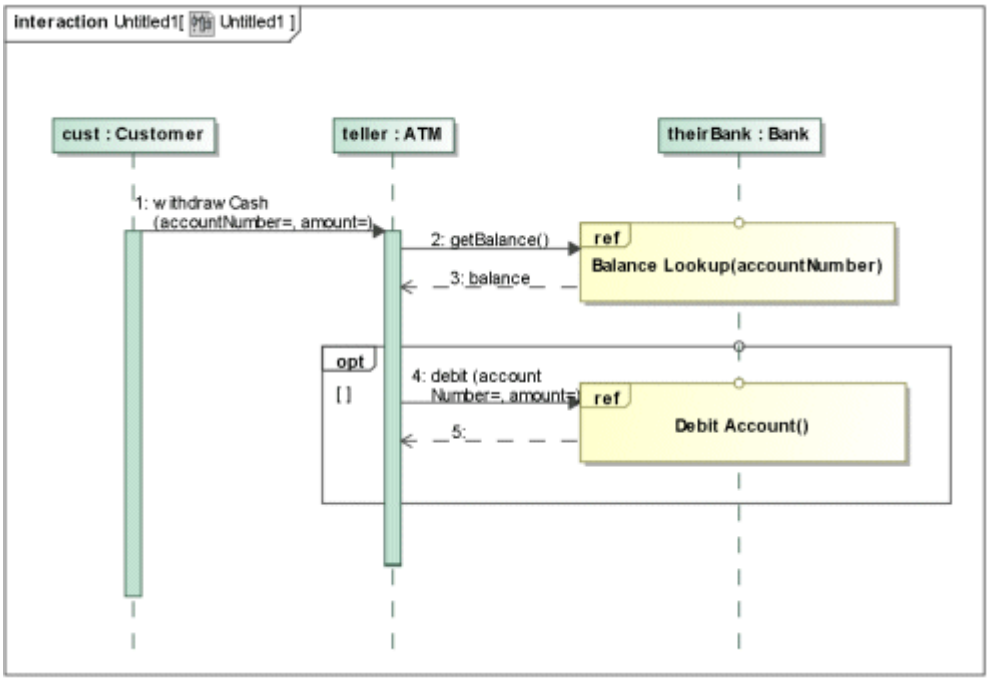


Figure 549 -- Actual gates usage in sequence diagram

Generalization

A generalization is a taxonomic relationship between a more general classifier and a more specific classifier. Each instance of the specific classifier is also an indirect instance of the general classifier. Thus, the specific classifier inherits the features of the more general classifier.

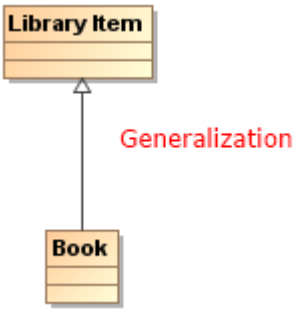


Figure 550 -- Example of generalization relationship

TIP!

- Generalizations can be grouped to the generalization tree. For more information, see "Generalization Tree" on page 889.
- Generalization may be assigned to the generalization set. For more information, see "Generalization Set" on page 893.

You can format the generalization symbol properties in the **Symbol Properties** dialog.

TIP!

For more information about symbol representation properties, see "Formatting Symbols" on page 317.

Copyright © 1998-2015 No Magic, Inc.

You can specify generalization properties in the **Generalization** Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Related diagrams

Class Diagram
Use Case Diagram
Component Diagram
Deployment Diagram
Profile Diagram

Related procedures

Generalization Tree
Formatting Symbols

Related dialogs

Specification Window

Related resources

You can see a generalization relationship sample in *<MagicDraw installation directory>\samples\case studies\Magic Library.mdzip*.

Generalization Tree

Usually the general element has more than one specific elements (see the Figure 551 on page 889) and sometimes specific elements may have their specific elements - the diagram with these generalizations may become messy. To display the generalization relationships and specific elements orderly, the generalization tree can be created (see the Figure 552 on page 890).

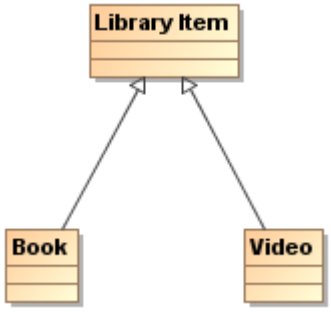


Figure 551 -- Example of more than one generalization relationships

Copyright © 1998-2015 No Magic, Inc.

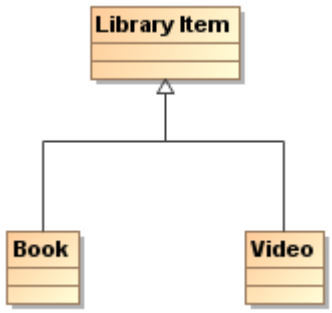
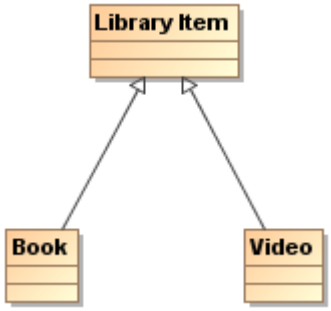


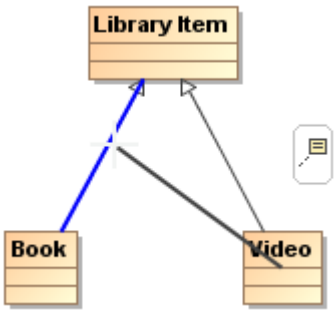
Figure 552 -- Example of generalization tree

To group generalization paths into a tree

1. Draw a generalization path between the specific element and the general element.
2. Draw a generalization path from the second specific element to the general element.

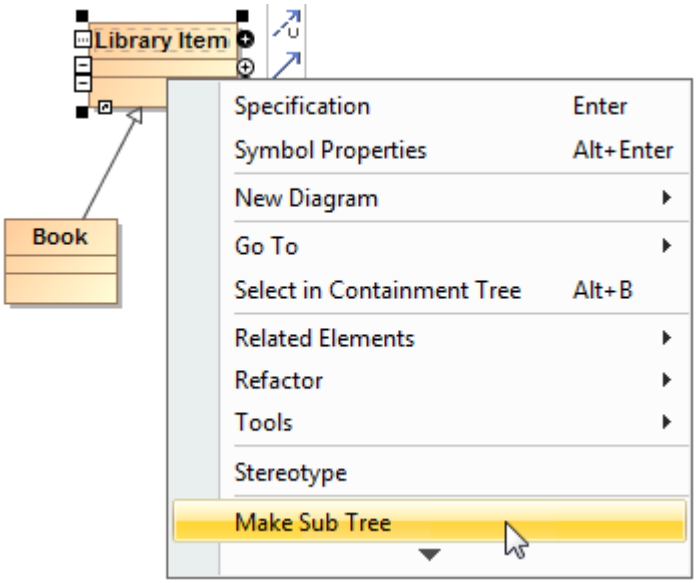


3. Do one of the following:
   - Drag the path to the other generalization path and drop it there.

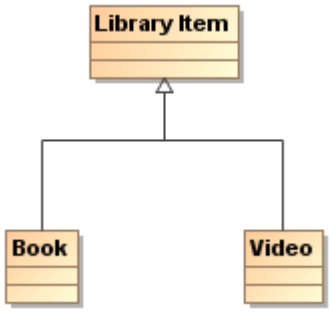


Copyright © 1998-2015 No Magic, Inc.

- Select **Refactor** > **Make Sub Tree** from the general element shortcut menu.

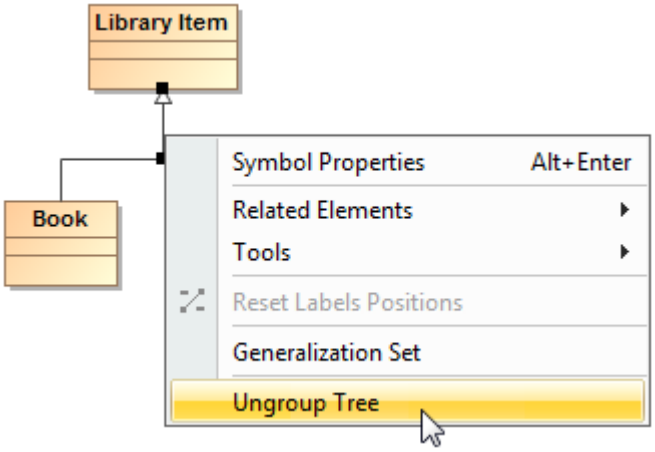


4. The generalization tree is created. See an example in the following figure.



To ungroup a generalization tree

1. Click the generalization tree's hollow triangle pointing to a general element.
2. From the tree shortcut menu, select the **Ungroup Tree** command (see the following figure). The generalization tree is ungrouped.

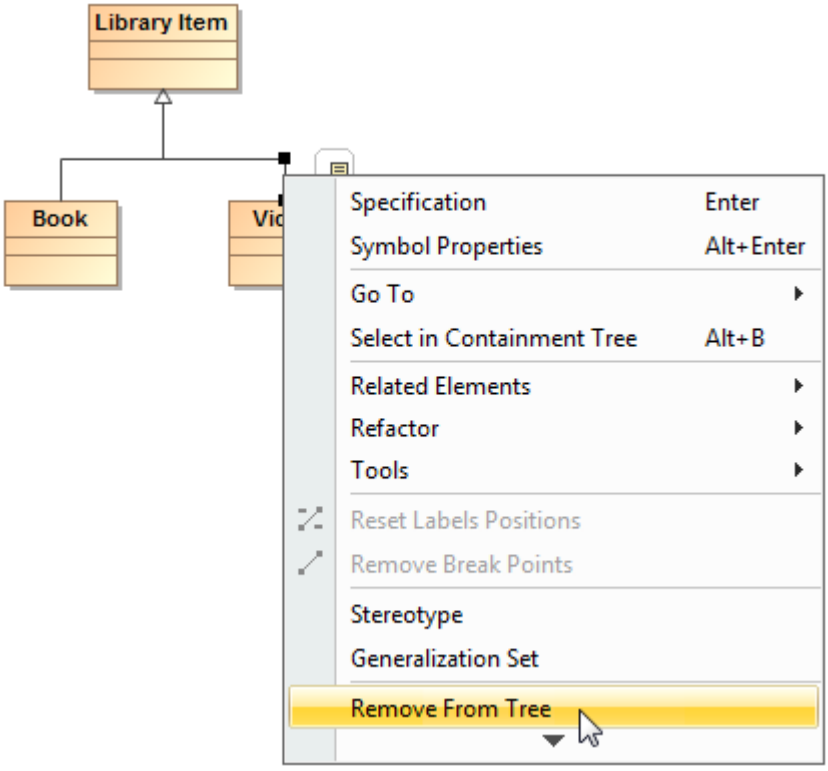


To separate the generalization path from the generalization tree

Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

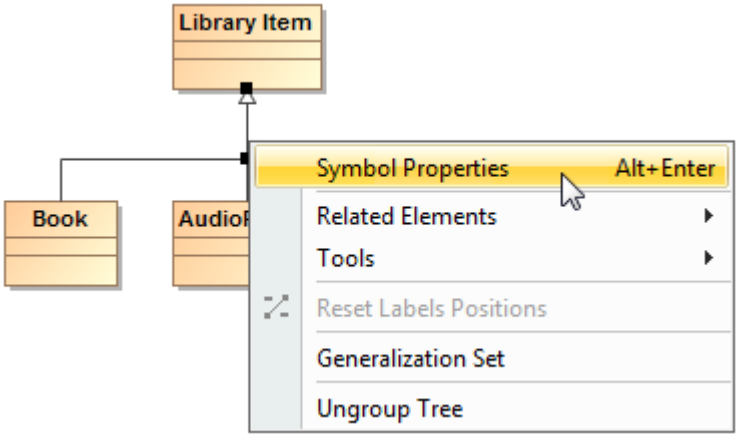
- From the generalization path shortcut menu, select **Remove From Tree**. The generalization is removed from the generalization tree.



- Drag one generalization path to another general or specific element. The generalization path is removed from the generalization tree and its general or specific element is changed.

To open the generalization tree **Symbol Properties** dialog

- On the diagram pane select the generalization tree and from its shortcut menu, select the **Symbol Properties** command (see the following figure). The **Symbol Properties** dialog opens.



Related concepts

Generalization

Related procedures

Formatting Symbols

Copyright © 1998-2015 No Magic, Inc.

Generalization Set

A generalization set is a packageable element (from Kernel) whose instances define collections of subsets of generalization relationships. Each generalization set defines a particular set of generalization relationships that describe the way which a general classifier may be divided using specific subtypes.

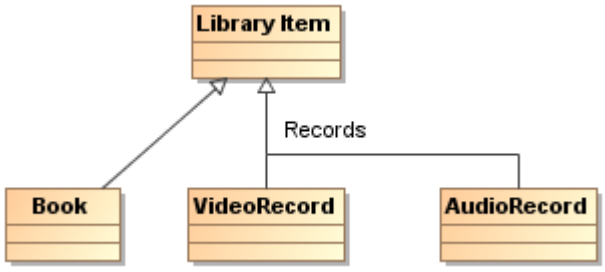


Figure 553 -- Example of Records generalization set

You can format the generalization set symbol properties in the **Symbol Properties** dialog. For more information on how to open the generalization set **Symbol Properties** dialog, see "To open the generalization set Symbol Properties dialog" on page 898. For more information about symbol representation properties, see "Formatting Symbols" on page 317.

You can specify generalization set properties in the **Generalization Set** Specification window. For more information on how to open the **Generalization Set** Specification window, see "To open the Generalization Set Specification window" on page 895. In the Specification window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To create a new generalization set

Do one of the following:

- Create a generalization set in the **Generalization** Specification window:
  1. Open the **Generalization** Specification window.
  2. Select the **Generalization Set** property cell and then click the "..." button. The **Select Generalization Set** dialog opens.

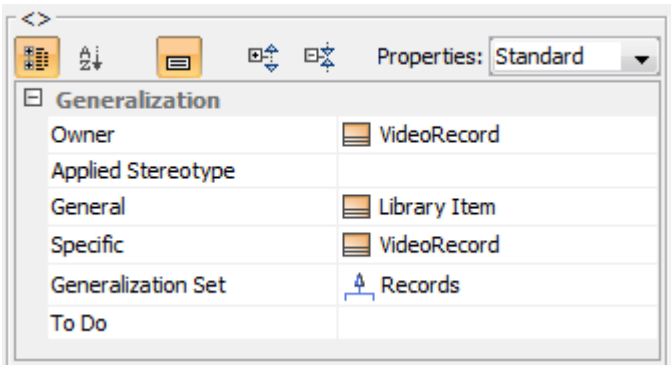
TIP!

For more information about working with the elements Selection dialog, see "Selecting an Element" on page 352.

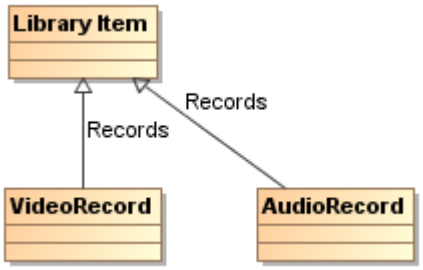
  3. Click the **Create** button. The list appears with the available to create elements.
  4. Click the **Generalization Set** command. The **Generalization Set** Specification window opens.
  5. Type the generalization set name and specify other properties.
  6. Click **Close**. The **Generalization Set** Specification window is closed. The created generalization set is selected in the **Select Generalization Set** dialog.

Copyright © 1998-2015 No Magic, Inc.

7. Click **OK**. The generalization set is created and assigned for the current generalization.



- Create a generalization set from the generalization shortcut menu:
  1. Select the generalization relationship on the diagram pane.
  2. Open the generalization shortcut menu.
  3. Select the **Generalization Set** command and then select **New Generalization Set**. The **Select Owner** dialog opens.
  4. Select an owner in which you want to create a new generalization set.
  5. Click **OK**. The **Generalization Set** Specification window opens.
  6. Type the generalization set name and specify other properties.
  7. Click **Close**. The new generalization set is created and assigned to the current generalization. Name of the generalization set is displayed on the diagram pane next to the generalization.



NOTE In the Generalization Set list many generalization sets can be listed, but on the diagram pane, near the generalization, only the first in the list is displayed.

To create a new generalization set for more than one generalization

1. On the diagram panes, select some generalizations with the same general element.
2. Right click to invoke the generalizations shortcut menu.
3. Select the **Generalization Set** command. See the next steps of a new generalization set creation in the following procedure.

NOTE The **Generalization Set** command exists only if all selected generalizations have the same general element.

To assign an existing generalization set to a generalization

Do one of the following:

- Assign a generalization set in the **Generalization** Specification window:
  1. Open the **Generalization** Specification window.

Copyright © 1998-2015 No Magic, Inc.

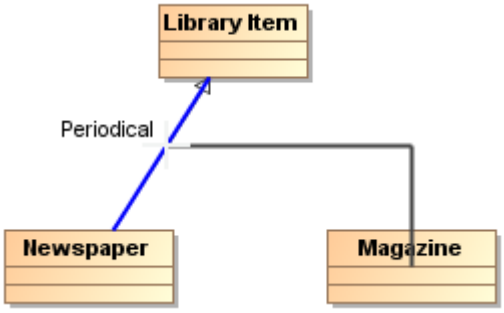
2. Select the **Generalization Set** property cell and then click the "..." button. The **Select Generalization Set** dialog opens.



For more information about working with the elements Selection dialog, see "Selecting an Element" on page 352.

3. Select the generalization set in the list.
4. Click **OK**. The generalization set is assigned to the current generalization.

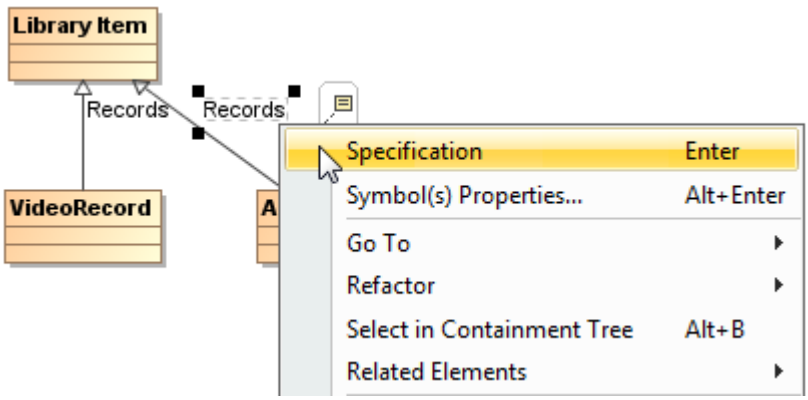
- Assign a generalization set from the generalization shortcut menu:
  1. Select the generalization relationship on the diagram pane.
  2. Open the generalization shortcut menu.
  3. Select the **Generalization Set** command. The list with the available generalization sets opens.
  4. Click the generalization set you want to assign.
  5. The generalization set is assigned.
- Draw or move a generalization line to a generalization set tree or to a generalization that belongs to a generalization set (see the following figure). The newly created generalization is assigned to the same generalization set.



To open the **Generalization Set** Specification window

Do one of the following:

- Invoke the **Generalization Set** Specification window from the diagram pane shortcut menu:
  1. Right click the generalization set name on the diagram to open its shortcut menu.
  2. Select the **Specification** command (see the following figure). The **Generalization Set** Specification window opens.

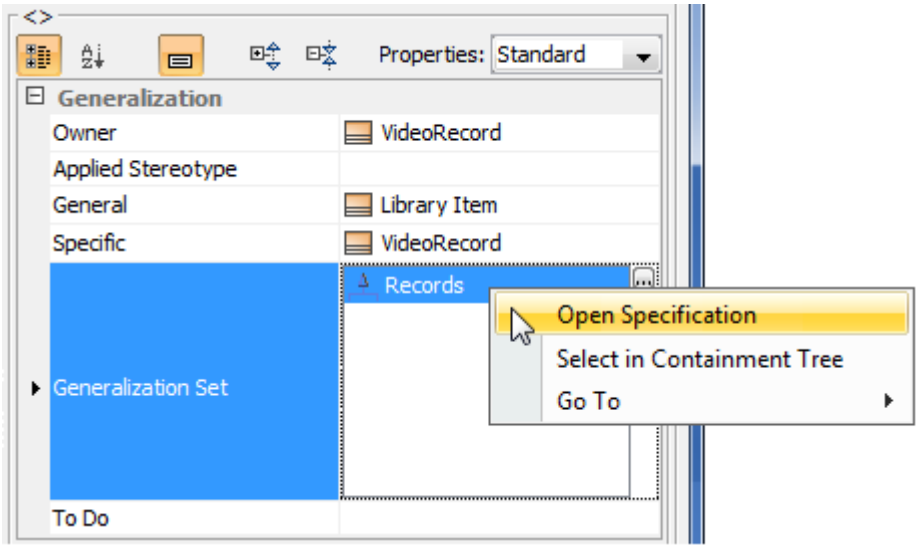


- Double click on the generalization set name on the diagram pane. The **General Set** Specification window opens.

Copyright © 1998-2015 No Magic, Inc.

- Open the **Generalization Set** Specification window from the **Generalization** Specification window:

    3. Open the **Generalization** Specification window.
    4. Select the **Generalization Set** property cell.
    5. Select the generalization set which specification window you want to open.
    6. Right click to open the generalization set shortcut menu.
    7. Select the **Open Specification** command (see the following figure). The **Generalization Set** Specification window opens.

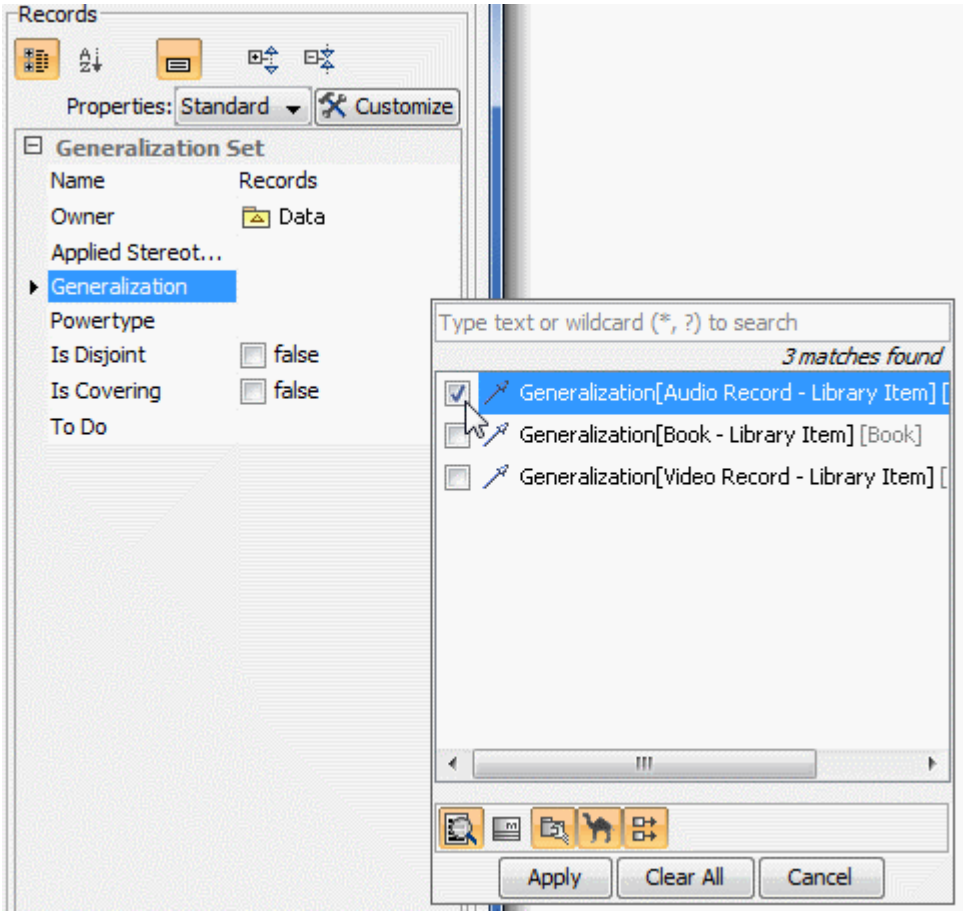


To assign a generalization to the particular generalization set

1. Open the **Generalization Set** Specification window.
2. Select the **Generalization** property cell.
3. Click the "..." button. The list with the available to select generalization opens.
4. Click to select the generalization.

Copyright © 1998-2015 No Magic, Inc.

5. Click the **Apply** button (see the following figure). The selected generalization or generalizations are assigned to the particular generalization set.



NOTE

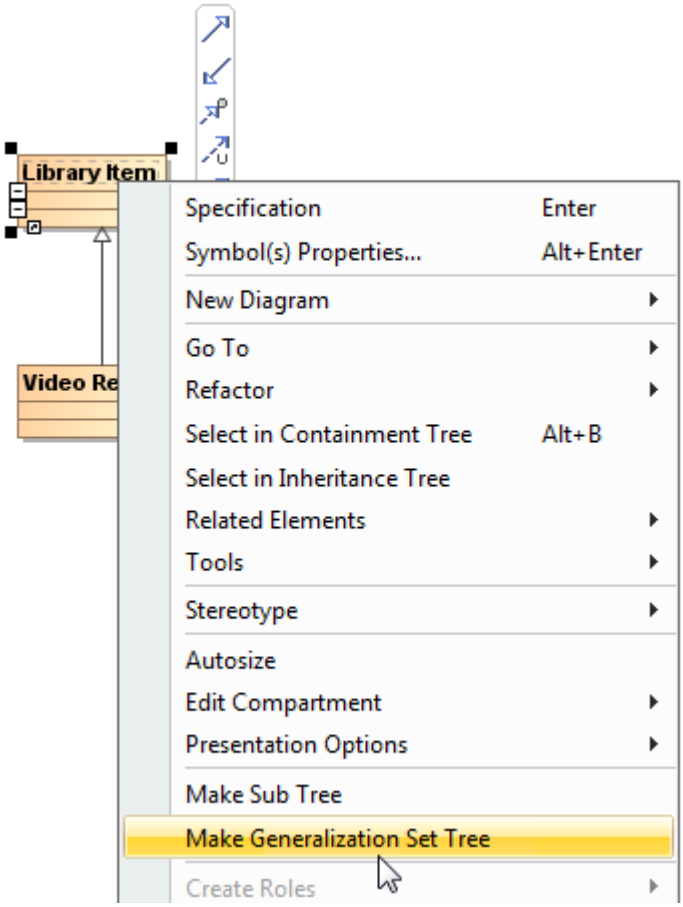
The generalizations that belong to other sets are allowed to be selected. Selecting such a generalization removes it from the previous set and adds it to the current one.

To group the generalizations to the generalization set trees

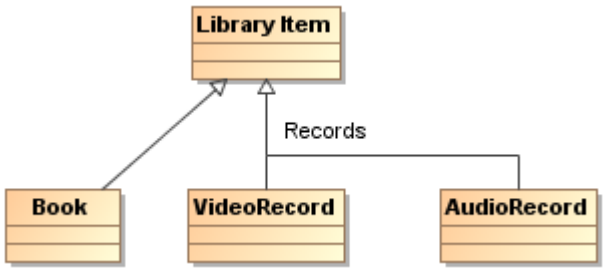
1. Select a general element on the diagram pane and invoke its shortcut menu.
2. Click the little arrow to expand the general element shortcut menu.

Copyright © 1998-2015 No Magic, Inc.

3. Select the **Make Generalization Set Tree** command. See the following figure.



4. The generalizations are grouped to trees, according to the generalization sets. See an example in the following figure.



NOTE

The **Make Generalization Set Tree** command is available and enabled only if the general element has at least two generalizations and at least two generalizations has assigned the same generalization set.

To open the generalization set **Symbol Properties** dialog

- On the diagram pane select the generalization set name and from its shortcut menu, select the **Symbol Properties** command (see the following figure). The **Symbol Properties** dialog opens.

Related elements

Generalization

Related procedures

Formatting Symbols

Copyright © 1998-2015 No Magic, Inc.

Selecting an Element

Related dialogs

Specification Window

Include

The include relationship is used when there are common parts of the behavior among two or more use cases. Each common part is then extracted to a separate use case, to be included by all base use cases having this part in common. The include relationship is represented from a base use case to an inclusion use case.

An include relationship between use cases indicates that an instance of the including use case will also contain the behavior as specified by the included use case.

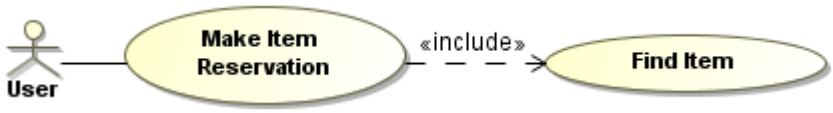


Figure 554 -- Example of include relationship

In the example above the *Register Item Reservation* use case is a base use case. It includes the *CFind Item* use case. The *Make Item Reservation* use case is the including use case and the *Find Item* use case is the included use case.

You can format the include symbol properties in the **Symbol Properties** dialog.



For more information about symbol representation properties, see "Formatting Symbols" on page 317.

You can specify include properties in the include Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Related diagrams

Use Case Diagram

Related procedures

Formatting Symbols

Related dialogs

Specification Window

Related resources

Example of the Use Case diagram in *<MagicDraw installation directory>\samples\diagrams\use case diagram.mdzip*.

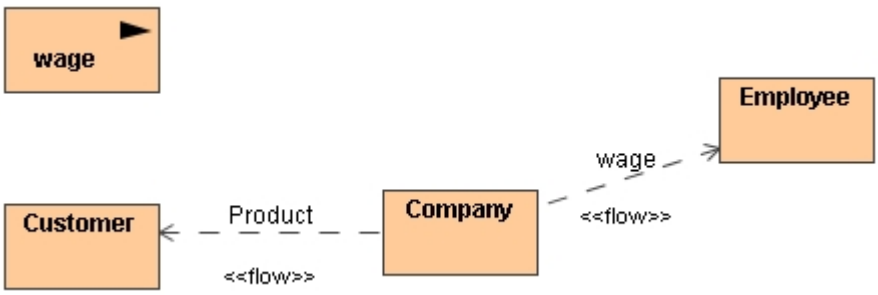
Copyright © 1998-2015 No Magic, Inc.

Information Flow

An Information Flow specifies that one or more information items circulates from its sources to its targets. Information flows require some kind of “information channel” for transmitting information items from the source to the destination.

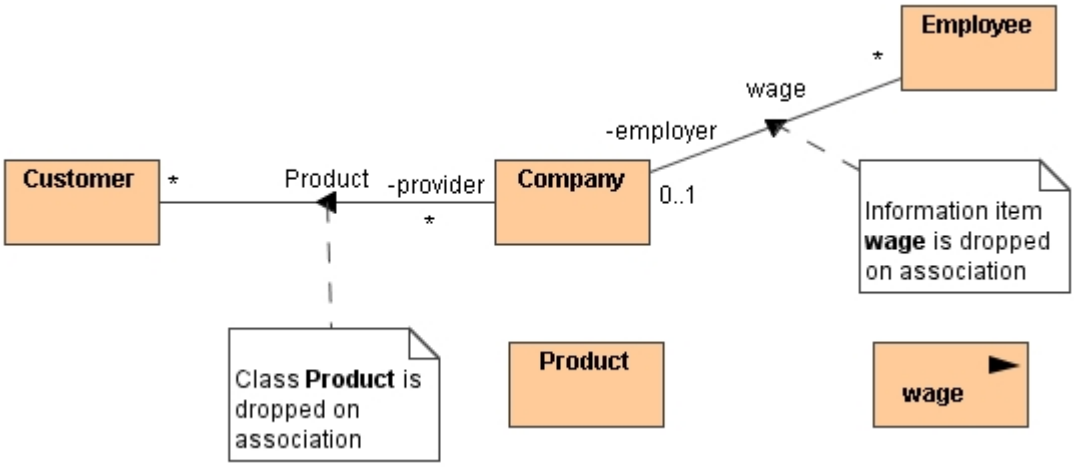
An information channel is represented in various ways depending on the nature of its sources and targets. It can be represented by connectors, links, associations, or even dependencies. For example, if the source and destination are parts in some composite structure diagrams such as a collaboration, then the information channel is likely to be represented by a connector between them. Or, if the source and target are objects (which are a kind of InstanceSpecification), they can be represented by a link that joins the two, and so on.

The information flow and the information item notation are added. You can draw them using the Information Flows toolbar in the class or composite structure diagram:



You can also create information flows in the associations in the class diagram and on the connectors in the composite structure diagrams:

1. Draw classes and associations.
2. From the association shortcut menu select command **Symbol Properties** and select the **Show conveyed information A** and **Show conveyed information B** check boxes.
3. Drag the class or information item on an association. An information flow is created.



Information Item

The Information Flows package provides mechanisms for specifying the exchange of information between entities of a system at a high level of abstraction.

Copyright © 1998-2015 No Magic, Inc.

The Information flows describe a circulation of information in a system in a general manner. They do not specify the nature of the information nor the mechanisms by which this information is conveyed (message passing, signal, common data store, parameter of operation, etc.). They also do not specify sequences or any control conditions. It is intended that, while modeling in detail, representation and realization links will be able to specify which model element implements the specified information flow, and how the information will be conveyed.

An information item is an abstraction of all kinds of information that can be exchanged between objects. It is a kind of classifier intended for representing information in a very abstract way, the one which cannot be instantiated.

One purpose of information items is to be able to define preliminary models, before making detailed modeling decisions on types or structures. Another purpose of information items and information flows is to abstract complex models by using a less specific but more general representation of the information exchanged between entities of a system.

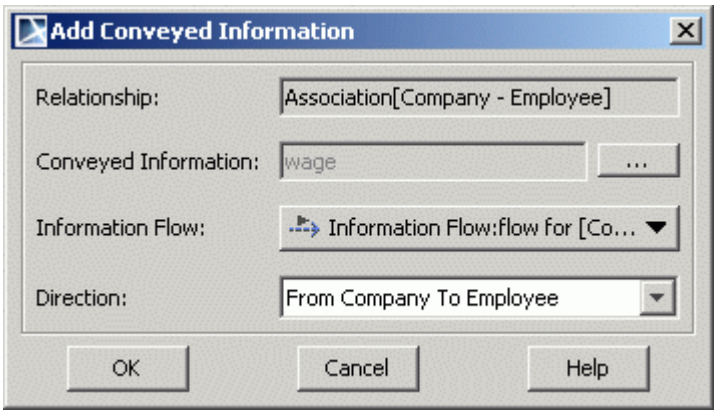
In a classifier, the information item can be represented as a name inside a rectangle. The black triangle icon on top of this rectangle indicates that it is an information item.



Information Items (or any conveyed classifiers) can be displayed on any relationship.

To display information Items on relationships:

1. Select the information item on the diagram pane and drag it on the relationship shape. The **Add Conveyed Information** dialog opens.



2. After specifying information flow and direction arrow, click **OK**.

Instance Specification

An instance specification specifies the existence of an entity in a modeled system and completely or partially describes the entity.

The description may include:

- The classification of an entity by one or more classifiers of which the entity is an instance. If the only classifier specified is abstract, then the instance specification only partially describes the entity.

Copyright © 1998-2015 No Magic, Inc.

- A kind of instance based on its classifier or classifiers - for example, an instance specification whose classifier is a class describes an object of that class, while an instance specification whose classifier is an association describes a link of that association.
- A specification of values of structural features of the entity. Not all structural features of all classifiers of the instance specification need to be represented by slots, in which case the instance specification is a partial description.
- A specification of how to compute, derive, or construct the instance (optional).

MagicDraw allows you to create the instances of classifiers – class, interface, enumeration, use case, actor, node, component, artifact, and other classifiers.

In the Component (or Deployment) diagram Node Instance, Component Instance, Artifact Instance elements are the same Instance Specification elements with an assigned component, node or artifact.

The instances are shown using a rectangle by underlining the name string of an instance element. The instance of an actor is shown as an actor “stick man” figure with the actor’s name string below the symbol.



For more information about working with symbols, see "Diagramming" on page 198.

You can specify instance properties in the instance Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.



If you need to handle a huge amount of instance specifications, use instance table. For more information, refer to "Instance Table" on page 803.

To assign a classifier to an instance

Do one of the following:

- On the object diagram pallet, click the *Instance* button, and then in the open dialog, select the classifier. Click **OK** when you are done.
- Open the Specification window of the instance and click the cell of the **Classifier** property value. Then click and in the open dialog, select the classifier. Click **OK** when you are done.
- On the diagram, right-click the instance shape, and then point to **Classifier**. In the open list, select the classifier or point to **New** to create a classifier.
- On the diagram, click an instance shape, and then on the smart manipulator toolbar, click . In the open list, select the classifier.
- On the diagram, click an instance shape, and then press **Ctrl+T**. In the open list, select the classifier.
- On the diagram, click an instance shape, then click the name area. Type the “:” colon and then type the classifier name. Press **Enter** or click anywhere on the diagram. The classifier is created and assigned.

  Type the “:” colon, then press **Ctrl+Space** or **Ctrl+Backspace** and then in the open list, select the classifier.

- In the Containment tree, select a classifier and drag it to the instance shape.

Copyright © 1998-2015 No Magic, Inc.

To hide/show an assigned classifier

From the instance shortcut menu, select/clear the **Show Classifier** check box.

To create a Component Instance

1. In the Component (or Deployment) diagram toolbar, click the Component Instance button. The **Select Components** dialog opens.
2. Select a component from the list or click the **Create** button to create a new one. Click **OK**.

The same is valid for the Node Instance and Artifact Instance creation.



Use the Node Instance button to create the Instance Specification with assigned Node and the Node Instance will have a Node shape.



Assigning Instance Specification as Default Value quickly

You can drag Instance Specification element on Classifier property on a diagram to assign it as default value. Drag and drop is available only if Instance Specification classifiers are compatible with Property type and if Property is editable.

To set the initial value to an attribute of the assigned classifier

1. Double-click an instance shape or select **Specification** from the shape shortcut menu.
2. The **Instance** Specification window opens. Click the **Slots** tab.
3. Click the **Edit Value** button and type the name of the value.

To show/hide slots of the assigned classifier

Clear/select **Suppress Slots** check box in the instance shortcut menu.



By default slots of the classifier are suppressed.

To display slot type on the instance symbol

Slot type can be optionally displayed on Instance or Part shapes.

Property **Show Slot Type** is added to slot symbol properties (select command **Symbol Properties** from instance shortcut menu to invoke **Properties** dialog). Slot type name (see Figure 555 on page 904), slot type qualified name (see Figure 556 on page 904) or no slot type (see Figure 557 on page 904) may be displayed

Copyright © 1998-2015 No Magic, Inc.

next to slot.

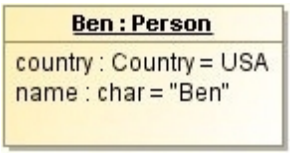


Figure 555 -- Slot type name is displayed next to slot



Figure 556 -- Slot type qualified name is displayed next to slot

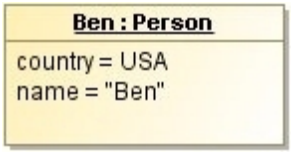


Figure 557 -- No slot type is displayed

To select slot in a diagram

Slot can be selected in a diagram. This allows the deletion of a slot straight from the diagram and to attach a note to a slot.

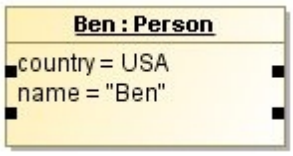


Figure 558 -- Slot selected in a diagram

To display specification value on the Instance Specification symbol

Specification value can be optionally displayed on the Instance Specification symbol. Check box **Show Specification Value** is added to Instance Specification properties (select command Symbol Properties from instance shortcut menu to invoke **Properties** dialog).

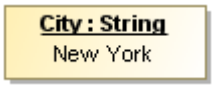


Figure 559 -- Specification value is displayed on the Instance Specification symbol

Interaction Use

Interactions are units of behavior of an enclosing Classifier. They focus on the passing of information with Messages between the Connectable Elements of the Classifier.

A reference to the interaction can be created.

Copyright © 1998-2015 No Magic, Inc.

You can specify interaction use properties in the interaction use Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To add a reference to an interaction use

Do one of the following:

- Open the Specification window of the interaction use and click the cell of the **Refers To** property value. Then click and in the open dialog, select the element. Click **OK** when you are done.
- On the diagram, right-click the interaction use shape, and then point to **Refers To**. Then click to select an element that you wish to apply on the interaction use.
- On the diagram, click an interaction use shape, and then press **Ctrl+T**. Then click to select an element that you wish to apply on the interaction use.
- On the diagram, click an interaction use shape, and then on the smart manipulator toolbar, click . Then click to select an element that you wish to apply on the interaction use.
- Drag an interaction from the Model Browser on to the interaction use shape on the diagram.

To add an actual gate

1. Add a reference to the diagram, from which the diagram frame formal message is created.
2. To the current interaction use draw an actual message with the selected formal gate.



For more information about working with gates, see "Gate" on page 885.



The rake icon is displayed on the shape if the interaction use refers to another sequence diagram. The rake icon isn't shown by default therefore you need to set the **Show Rake Icon** property value to *true* in the **Property** dialog. For more information see "NEW! Displaying rake icon on symbol" on page 322.

Related diagrams

Sequence Diagram

Interface

An interface is a specifier for the externally-visible operations of a class, component, or other classifiers (including subsystems) without a specification of the internal structure. Each interface often specifies only a limited part of the behavior of an actual class.

The set of interfaces realized by a classifier is its provided interfaces, which represent the obligations that instances of that classifier have to their clients. They describe the services that the instances of that classifier offer to their clients.

Copyright © 1998-2015 No Magic, Inc.

You can specify interface properties in the interface Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

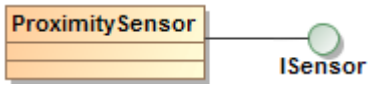
Provided and Required Interfaces

The set of interfaces realized by a classifier is its provided interfaces, which represent the obligations that instances of that classifier have to their clients. They describe the services that the instances of that classifier offer to their clients.

The interfaces may also be used to specify required interfaces, which are specified by a usage dependency between the classifier and the corresponding interfaces. Required interfaces specify services that a classifier needs in order to perform its function and fulfill its own obligations to its clients.

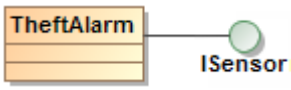
To draw a Provided Interface

1. In the class diagram, create a class.
2. Select the class shape and on the smart manipulator toolbar, click the *Interface Realization* button and then click on the diagram. The provided interface is created.
3. Click the provided interface shape and then click the *Suppress Operations* button.



To draw a Required Interface

1. In the class diagram, create a class.
2. Select the class shape and on the smart manipulator toolbar, click the *Usage* button and then click on the diagram. The required interface is created.
3. Click the required interface shape and then click the *Suppress Operations* button.



TIP!

- For more information, see "Working with classes" on page 854.
- For more information about working with shapes, see "Diagramming" on page 198.

Provided and Required Interfaces in the Composite Structure diagram

Information about provided and required interfaces is crucial in the assembly stage of complex internal structures. It helps to decide where connectors should be attached.

Provided and required interfaces are valuable parts of the UML Composite Structure Diagram and SysML Internal Block Diagram.

Copyright © 1998-2015 No Magic, Inc.

A provided interface is shown using the "lollipop" notation attached to the port and required interface is shown using the "fork" notation attached to the port (see the following figure).

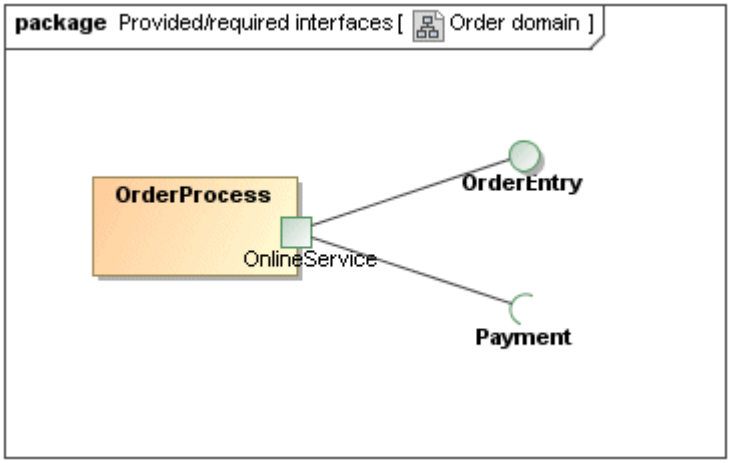


Figure 560 -- Provided and Required interface

In the Composite Structure diagram you cannot draw provided and required interfaces itself, but with the new functionality of MagicDraw you can display preexisting port with the required and provided interfaces as images.

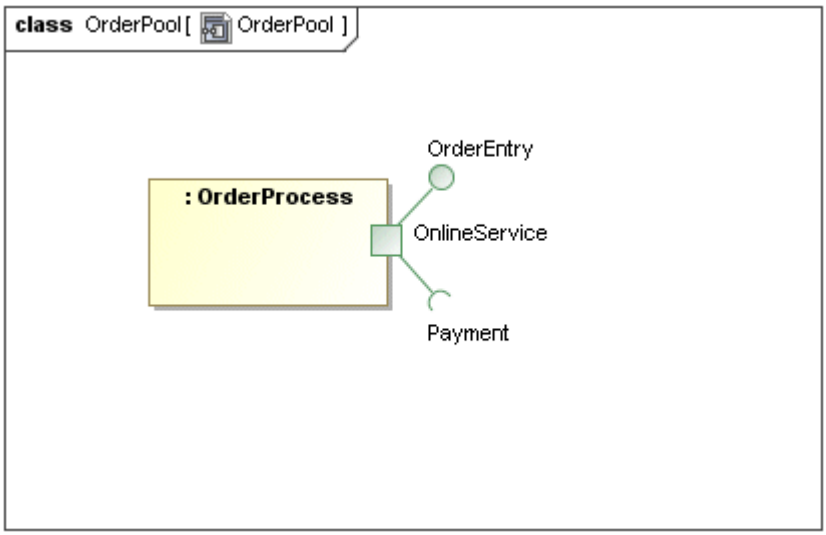


Figure 561 -- Provided and required interface in the Composite Structure diagram

Lollipop and fork symbols in the Composite Structure diagram are implemented as small attachments to a Port symbol (like name label). It is not the same as the independent standalone notation of the interface, it is only part of port symbol. It is important for Composite Structure diagrams where real Interfaces (as Classifiers) cannot be used.

It is an optional notation, a port does not display provided or required interfaces by default.

Displaying provided/required interfaces in the Composite Structure diagram

1. Create provided and required interfaces in the Class diagram. See procedures "To draw a Provided Interface" on page 906 and "To draw a Required Interface" on page 906.
2. In the Composite Structure diagram, select **Related Elements** from the port shortcut menu and then **Display Provided/Required Interfaces**. Or, in the individual Port symbol **Properties** dialog, select the **Show Provided Interfaces** and **Show Required Interfaces** check boxes.

Copyright © 1998-2015 No Magic, Inc.

As Port can provide or require many interfaces, displayed or hidden interfaces can be managed in the **Edit Compartment** dialog.

Provided/required interfaces in the Component diagram

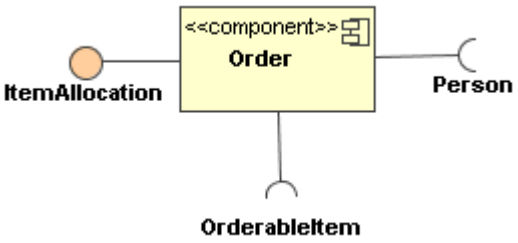
A component specifies a formal contract of the services that it provides to its clients and those that it requires from other components or services in the system in terms of its provided and required interfaces.

The required and provided interfaces may optionally be organized through ports.

To add and manage the added provided and required interfaces quickly, in the **Component** Specification window, select the **Provided/Required Interfaces** pane.



For more information about provided and required interfaces, see the procedure "Provided and Required Interfaces" on page 906.



Internal transition

In all other cases, the action label identifies the event that triggers the corresponding action expression. These events are called internal transitions and are semantically equivalent to self transitions except that the state is not exited or re-entered. This means that the corresponding exit and entry actions are not performed.



For more information on defining transitions, see "Fork and Join" on page 975.

Specify the internal transition in the **Transition** Specification window.



For a detailed description of this dialog, see "Transition" on page 1000.

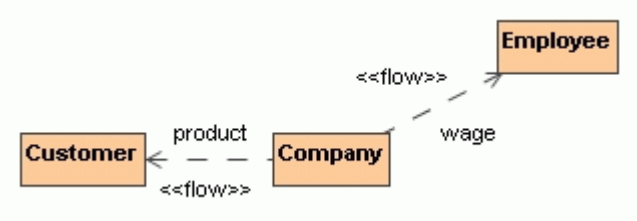
To define an internal transition

1. Double-click the state or select **Specification** from the state shortcut menu. The **State** Specification window opens.
2. Click the **Internal Transitions** group.
3. Click the **Create** button. The **Transition** Specification window opens. Specify an internal transition.

To remove the internal transition, click the **Delete** button.

Copyright © 1998-2015 No Magic, Inc.

The Information Flow can be related to any relationship.



Lifeline

A lifeline represents an individual participant in the Interaction. The lifeline represents only one interacting entity. It is shown using a rectangle symbol.

For more general information about working with symbols, see "Diagramming" on page 198.

To assign a type (classifier) to a lifeline

Do one of the following:

- Open the Specification window of the lifeline and click the cell of the **Type** property value. Then click and in the open dialog, select the type. Click **OK** when you are done.
- On the diagram, right-click the lifeline shape, and then point to **Type**. In the open list, select the type or point to **New** to create a type.
- On the diagram, click a lifeline shape, and then on the smart manipulator toolbar, click . In the open list, select the type.
- On the diagram, click a lifeline shape, and then press **Ctrl+T**. In the open list, select the type.
- On the diagram, click a lifeline shape, then click the name area. Type the ":" colon and then type the name of the type. Press **Enter** or click anywhere on the diagram. The type is created and assigned.

  TIP! Type the ":" colon, press **Ctrl+Space** or **Ctrl+Backspace** and then in the open list, select the type.
- In the Containment tree, select an element and drag it to the lifeline shape.

To hide/show a base classifier

- On the diagram, right-click the lifeline and from the shortcut menu, select the **Show Classifier** check box.

To create lifelines for existing data (the interaction properties and parameters) or display all lifelines of the interaction which are not displayed in the diagram

- Drag and drop the selected element from the Browser to the diagram pane.

Copyright © 1998-2015 No Magic, Inc.

- On the diagram shortcut menu, click **Related Elements** > **Display Lifelines** and, in the **Display Lifelines** dialog, choose the required elements you want to display or create.

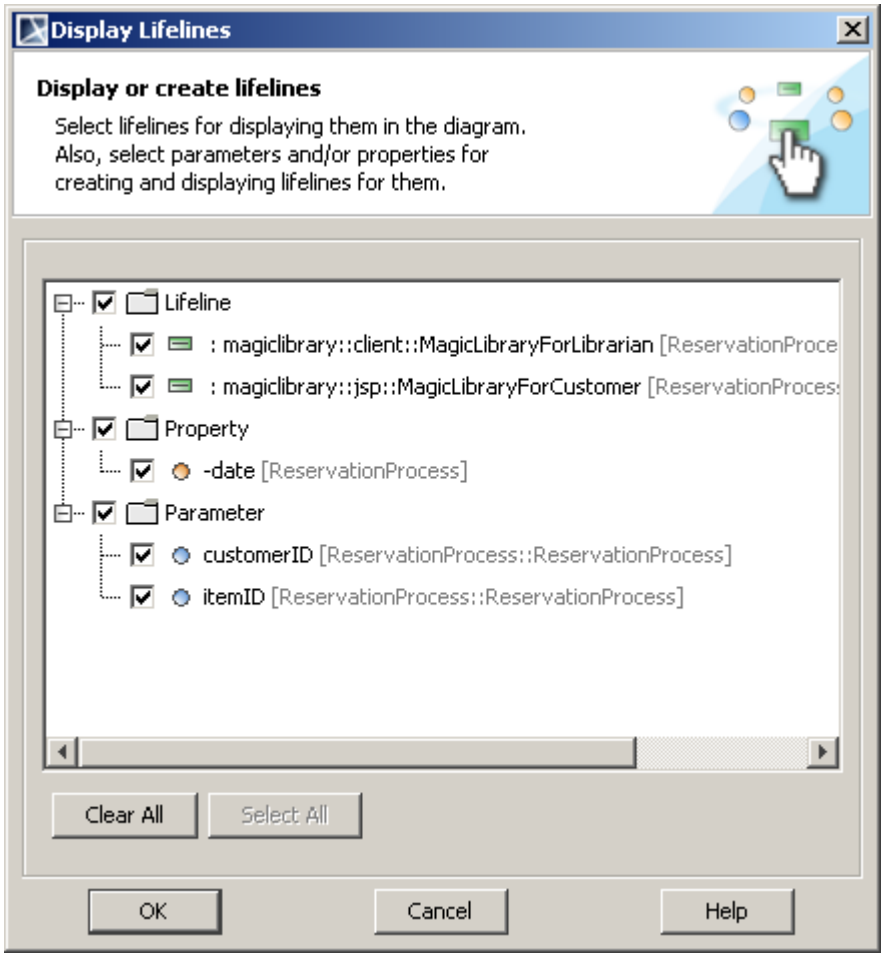


Figure 562 -- Display Lifelines dialog

Related sections

"Communication Diagram", on page 681
"Sequence Diagram", on page 683
"Message", on page 917

Working with Lifelines in Sequence Diagrams

When an object receives a message, an activity starts in that object. An activation (focus of control) shows the period during which an object is performing an action either directly or through a subordinate procedure. The activation bar is used to denote that activity.

To change the activation bar size

1. Click the desired activation bar on the Diagram pane.
2. Drag the activation bar to the desired direction.

NOTE

After resizing, the lines on the activation bar are thickened, but the size may not change automatically.

Copyright © 1998-2015 No Magic, Inc.

To destroy a sequence object - a large “X” at the end of its lifeline marks its destruction

- Select **Delete Mark** from the lifeline shortcut menu.
- Set a message as delete message.

To add a recursive message to a lifeline

- On the diagram pallet, click the **Recursive Message** button and then click the activation bar on which you want to draw the recursive message.

Related diagrams

"Sequence Diagram", on page 683

Link

Instance of an association is called a link.

A link is a tuple with one value for each end of the association, where each value is an instance of the type of the end.

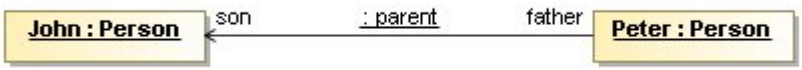


Figure 563 -- Role names displayed on link symbol

As of MagicDraw version 17.0.3, you can display on diagram

- Links created between instance specifications (the **Display Paths** command on the instance specification shortcut menu).
- Instance specifications at the other end of a link (the **Display Related Elements** command on the instance specification shortcut menu).

Links can be represented in the Relationship Map and Dependency Matrix.

Related diagrams

Object Diagram

Class Diagram

Related references

Working with Links

Active Validation Rules for Links

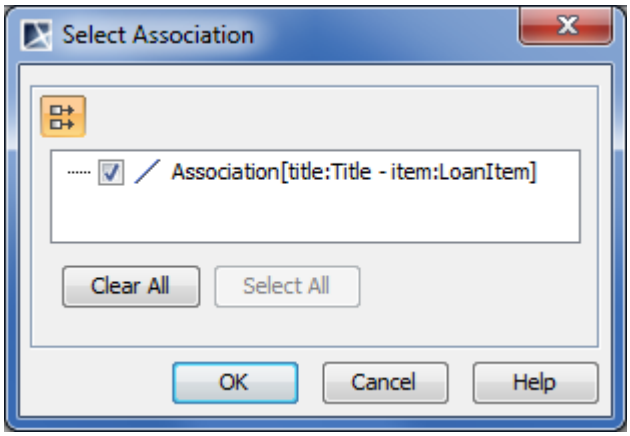
Working with Links

To create a link

1. Create an Object diagram.

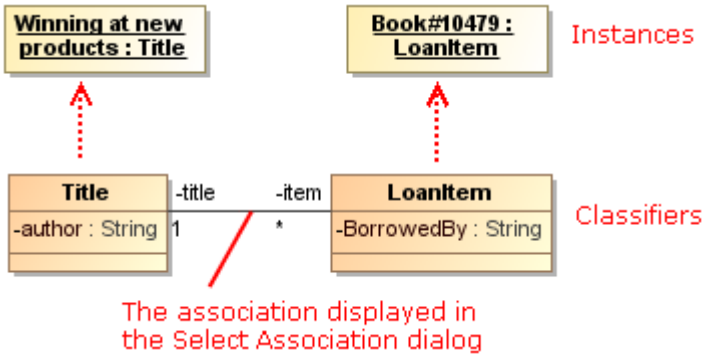
Copyright © 1998-2015 No Magic, Inc.

2. In the diagram draw a link between two instance shapes. The **Select Association** dialog opens.



NOTE

The **Select Association** dialog displays associations that are created between the classifiers of the link.



3. Select an association and click **OK**. The **Create Slots** dialog opens.

TIP!

For more information on how to create slots in links, see "To create slots on a link", on page 912.

4. Select slots and click **OK**.

To create slots on a link

1. In an Object diagram draw a link between two instances. The **Select Associations** dialog opens.

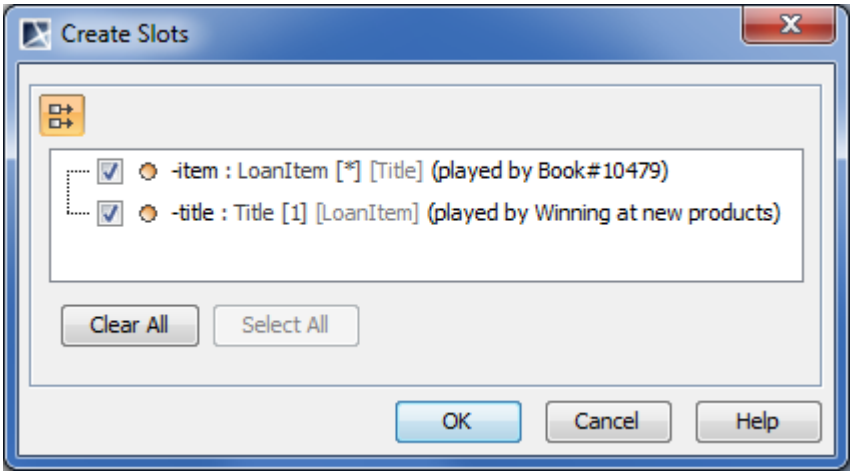
TIP!

For more information,

see "To create a link" on page 911.

Copyright © 1998-2015 No Magic, Inc.

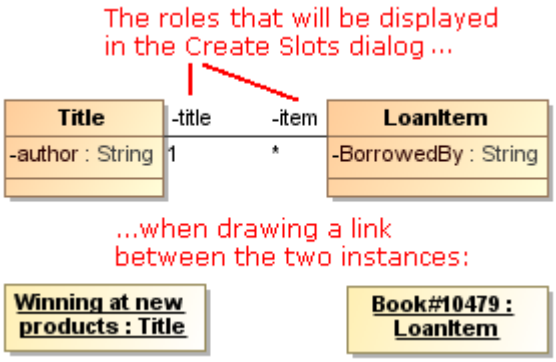
2. In the **Select Association** dialog, click **OK**. The **Create Slots** dialog opens.



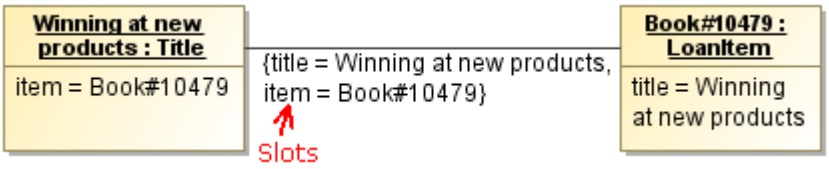
3. Select slots.

NOTE

The **Create Slots** dialog displays the roles of the association, that is created between the classifiers of the link.



4. Click **OK**.



NOTE

- The slots in the link are created automatically, only if the **Automatically Create Slots of Link** option is set to true. For more information, see "To turn on automatic creation of slots in links" on page 913.
- Slots on a link are created automatically, only when the link references the association. For more information, see "To create a link" on page 911.

To turn on automatic creation of slots in links

1. Open the **Environment Options** dialog.
2. In the **General** options group, under the **Editing** subgroup, set the **Automatically Create Slots in Link** option value to true.

To display slot values on a link, do one of the following

- Select a link on a diagram pane, then open its shortcut menu, and click **Suppress Slots**.

Copyright © 1998-2015 No Magic, Inc.

- In the link **Symbol Properties** dialog, click to clear the **Suppress Slots** check box.

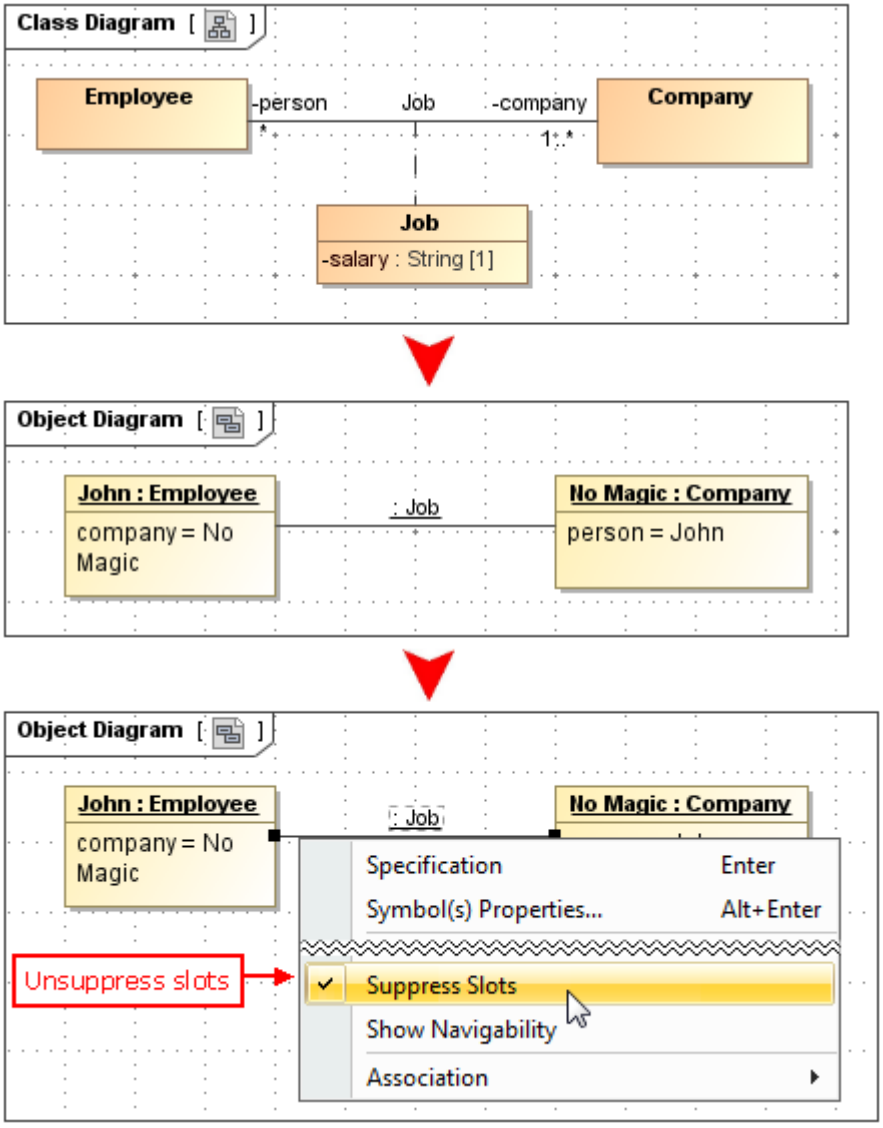


Figure 564 -- Example of showing slots on link. Unsuppressing slots

Copyright © 1998-2015 No Magic, Inc.

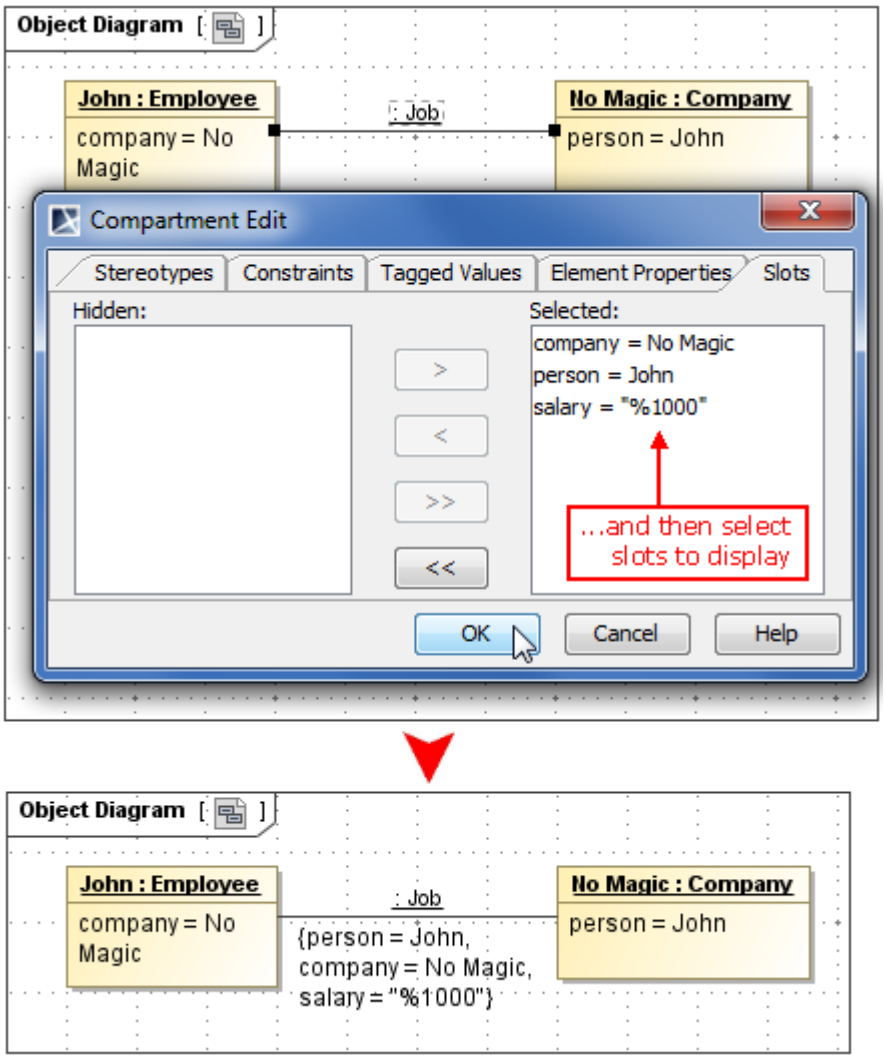


Figure 565 -- Example of showing slots on link. Selecting slots to display

NOTE

- The default value of the **Suppress Slots** option is true, that is, slots are not displayed on a link.
- Slots on a link are displayed in the following form:

{<*slot1 name*> = <*slot values*>,
<*slot2 name*> = <*slot values*>,
... }

TIP!

You can select to display or hide an individual slot in the **Compartment Edit** dialog (to open the dialog, select the link on a diagram pane and on its shortcut menu, click **Edit Compartment** > **Slots**).

To change a link end

- On a diagram, select an end of a link and move it to the other target. The link is connected to the other target.

Related concepts

Link

Copyright © 1998-2015 No Magic, Inc.

Related references

Active Validation Rules for Links

Object Diagram

Active Validation Rules for Links

Active Validation instantly checks the accuracy, completeness, and correctness of a model, displays errors in the model, and suggests solutions. There are two active validation rules for links:

- Instance Specification type mismatch
- Missing slots in links



For more information about the active validation, see "Active Validation" on page 632.

Instance Specification type mismatch

The *Instance Specification type mismatch* validation rule allows for updating slot values when referenced association end types are incompatible with connected instance specification types.

Missing slots in links

The *Missing slots on links* validation rule is useful then migrating projects from the MagicDraw 17.0.2 version or earlier. The validation rule detects links that have related associations, but do not have slots created in them. For each incorrect link the warning is displayed. To resolve the warning, there is the **Create slots for this link** resolution action. After choosing this action, slots are created in the link.



The validation rule is available only if the **Automatically Create Slots in Link** option is set to true in the **Environment Options** dialog.

Related concepts

Link

Related references

Active Validation

Manifestations

An artifact embodies or manifests a number of model elements. It owns the manifestations, each representing the utilization of a packageable element.

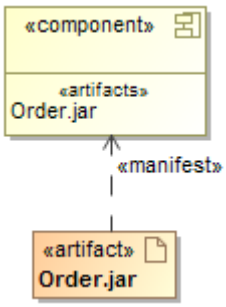
To create the manifestations, simply draw the Manifestation link from an artifact to a component.

To display the manifested artifacts on the component shape

- On the diagram, click the component shape, then, click the *Compartments* button. On the menu, click to select the **Artifacts** check box.

Copyright © 1998-2015 No Magic, Inc.

- On t he diagram, right-click the component shape and from the shortcut menu, select **Symbol Properties**. Then in the **Symbol Properties** dialog, click to clear the S**uppress Artifacts** check box.



Message

A Message defines a particular communication between lifelines of an Interaction. A communication can be, for example, raising a signal, invoking an Operation, creating or destroying an Instance. The Message specifies the kind of communication, as well as, the sender, and the receiver.

A message is represented by arrows between the lifelines. The style of the message line and arrowhead reflect the types of the message.



Figure 566 -- Example of Message in Sequence diagram



Figure 567 -- Example of Message in Communication diagram

You can see the description of a selected property in the description area of the Specification window. To see descriptions, be sure the Show Description option is turned on.

TIP!

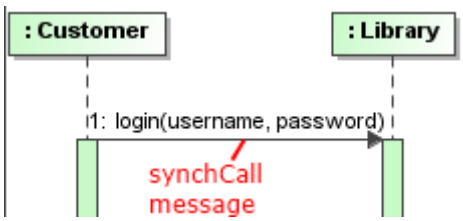
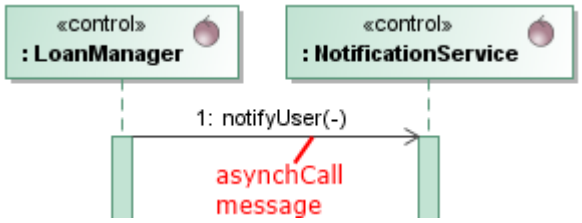
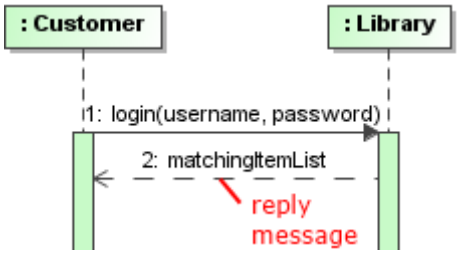
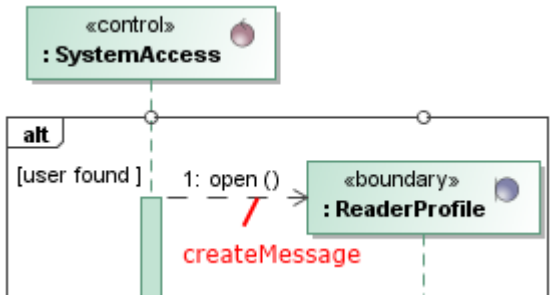
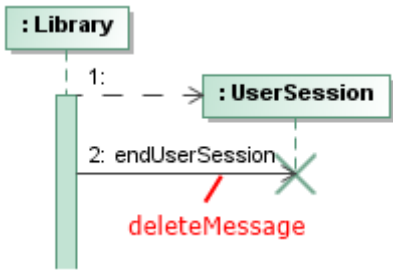
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Message kind

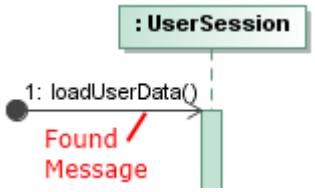
See the message kinds described in the following table.

Message kind	Message Sort	Description	Example
Complete			

Copyright © 1998-2015 No Magic, Inc.

Message kind	Message Sort	Description	Example
	synchCall	The message was generated by a synchronous call to an operation. Synchronous messages can overtake each other. When a message represents an operation, the arguments of the message must correspond to the parameters of the operation.	
	asynchCall	Asynchronous means that the caller continues immediately after the call. asynchCall messages do not have reply messages.	
	asynchSignal	The message was generated by an asynchronous send action. The argument of the message must correspond to the attributes of the signal.	The same as if the asynchCall message sample.
	reply	The message is a reply message to an operation call.	
	createMessage	The message designates the creation of another lifeline object. No other messages on a given lifeline in an interaction operand may appear above a lifeline.	
	deleteMessage	The message designates the termination of another lifeline	

Copyright © 1998-2015 No Magic, Inc.

Message kind	Message Sort	Description	Example
Lost	asynchCall	A lost message is a message where the sending event occurrence is known, but there is no receiving is no receiving event occurrence.	
Found	-	A found message is a message where the receiving event occurrence is known, but there is no (known) sending event occurrence.	

Message syntax

Message name is displayed on the message path on the diagram pane. Syntax for the message name is the following:

<messageident> ::= [<attribute> '='] <signal or operation name> ['(' [<argument> [',' <argument>] ')'] [':' <return value>]*

<argument> ::= ([<parameter name> '='] <argument-value>) / '-'

For example:

v=fo(P1="a", P2=True):12

TIP!

For more detailed description of the message name parsing rules, see "Message name parsing" on page 922.

Related diagrams

"Communication Diagram", on page 681
"Sequence Diagram", on page 683
Time Diagram

Related references

Common Actions with Messages
Messages in Sequence and Time Diagrams
Messages in Communication Diagrams

Common Actions with Messages

To set an action type for a message, do one of the following

- In the message Specification window, click the **Message Sort** property value cell and select the action type from the drop-down list.
- On the diagram pane, right-click the message and select a desired action type from the shortcut menu.

TIP!

- For more information about message action types, see "Message" on page 917.
- For more information about message action types in a Communication diagram see "Communication Diagram elements" on page 681.

Copyright © 1998-2015 No Magic, Inc.

To show / hide message numbers

1. Right-click the diagram pane to open its shortcut menu.
2. Then do one of the following:
   - In a Sequence or Time diagram, select / clear **Show Message Numbers**.
   - In a Communication diagram, select / clear **Numbering** > **Show Message Numbers**.

Assigning operations to messages

A model conventionally is created in the following order:

1. A class diagram with classes is created.
2. A sequence diagram uses these classes and their operations to represent the call order.

MagicDraw provides a faster way of assigning and creating operations than it is allowed in the traditional model creation process:

1. A sequence diagram is created to represent both classes and messages.
2. You can convert a message into a call message as well as create operations for classes in a single click. For more detailed description of the message name parsing rules, see "Message name parsing" on page 922.

You can assign operations only to call (synchronous and asynchronous) messages. Furthermore, only one operation can be assigned to a message.

To assign an operation to a message

Do one of the following:

- Assign an operation to a message in the message Specification window:
  1. Draw a call message between two lifelines or select an existing message on a diagram pane.
  2. Open the message Specification window (see the procedure "To open the element Specification window" on page 275).
  3. Click the **Signature (operation)** property value cell and then select an operation in the drop-down list.



- If the **Signature (operation)** drop-down list is empty, check if the lifeline, to which the current message is drawn, has a type assigned, and if this type has a specified operation.
- In the **Signature (operation)** drop-down list, both operations and signals of your project are listed. Make sure you have selected an operation. If you assign a signal to a call message, the message will automatically be converted to a send message.

- Type an operation name straight on the message on the diagram pane.



For more information about a message parsing, see "Message name parsing" on page 922.

- On the diagram, right-click the message and on the shortcut menu, click **Operation**. Then click to select an operation that you wish to apply on the element or click the **Create Operation** button.

If an operation you are assigning to a message has default parameter values, these default parameter values are assigned to message arguments. If types of the operation parameter and the message argument mismatch, however the default parameter value is assigned to the message argument.

Copyright © 1998-2015 No Magic, Inc.

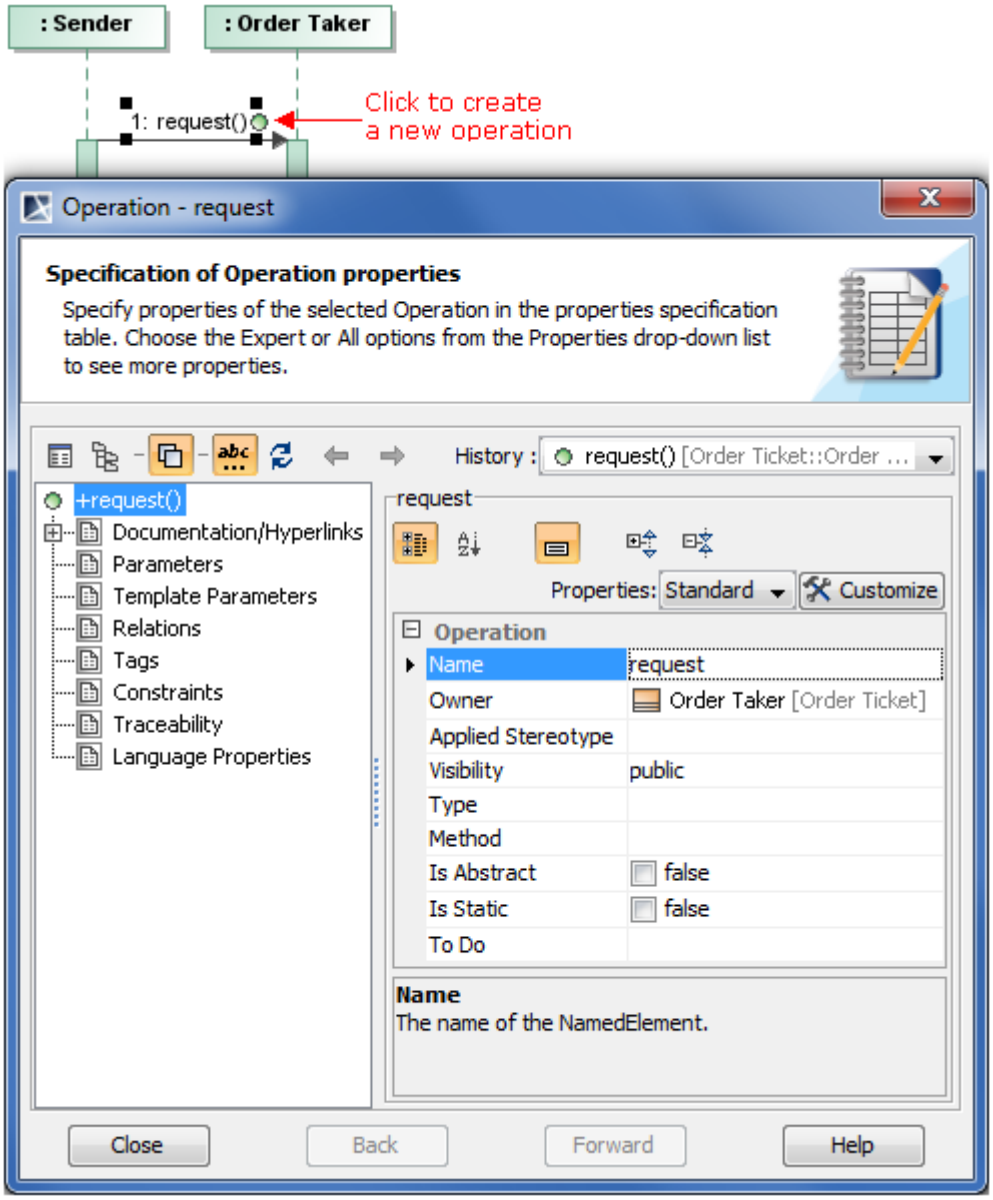
To create a new operation for a message

IMPORTANT! You can create a new operation only if the lifeline to which the message is drawn has a type assigned.

1. Draw a call message between two lifelines or select an existing message on a diagram pane.
2. Do one of the following:

- Click the **Create new operation according to this message** button in the shape of a small green circle at the end of the message name (see the preceding figure). An operation Specification window opens. Define the operation property values.

TIP! For more information about operations, see "Operation" on page 955.



Copyright © 1998-2015 No Magic, Inc.

- On the message shortcut menu, click **Operation** and then click the **Create Operation** button. Select the owner and click **OK**. Then in the operation Specification window, define the operation and click **Close**.

TIP! For more information about operations, see "Operation" on page 955.

- Type an operation name straight on a message on the diagram.

TIP! For more information about the message name parsing, see "Message name parsing" on page 922.

Message name parsing

Arguments, parameters, and return values that are specified on a message in a sequence diagram can be parsed and new operations can be created. Parsing and mapping message names to real model elements (operations, parameters, etc.) allows for easy transition from *prototype sequence diagrams* to *complete sequence diagrams* representing actual interactions between system parts.

Message name parsing

On a message name area on a diagram pane, you can type a name of a new operation together with arguments, parameters, and return value, and after a click on the button near the message name, the new operation will be created together its specifications.

Message syntax

The whole message name should be written according to message syntax rules.

TIP! For more information about message name syntax, see "Message" on page 917.

Message Parsing notification message

If any message name syntax errors are found, users are warned and suggested how to fix these mistakes. The Message Parsing notification message opens informing and pointing the concrete error.

TIP! For more information about the Message Parsing notification message, see "Notification message about incorrect syntax" on page 926.

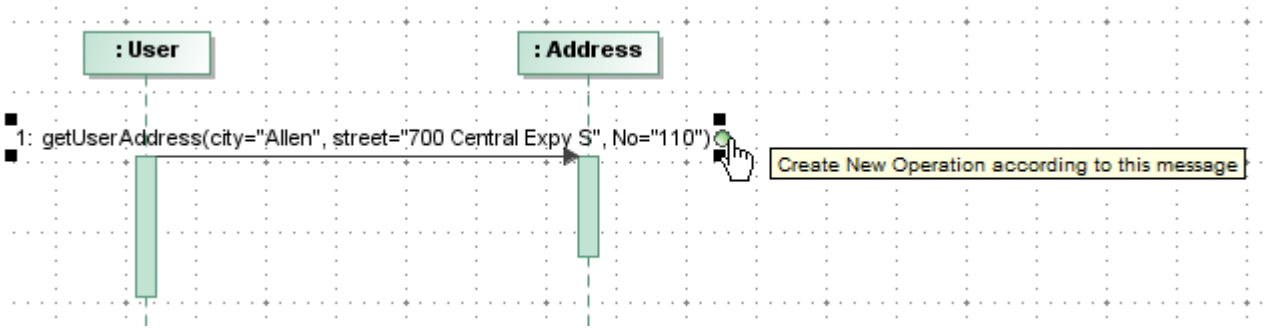


Figure 568 -- Example of message parsing

In the preceding example, you can see that on the message name is the following:

getUserAddress(city="Allen", street="700 Central Expy S", No="110"). After click on the Create New Operation according to this message button, the new *getUserAddress* operation will be created. The following

Copyright © 1998-2015 No Magic, Inc.

message arguments will be created - *Allen*, *700 Central Expy S*, and *110* together with the corresponding parameters - *city*, *street*, *No*.

To create a new operation from the diagram pane

1. Between two lifelines create a call message (asynchCall or synchCall).



The lifeline to which the call message is connected, must have a type specified. See an example in Figure 568 on page 922, there the *Address* type is assigned to the second lifeline.

2. On the diagram pane, on the message, type the message name according to the syntax rules. For more information about the message syntax, see "Message syntax" on page 919.
3. After you have typed the message name, exit the edit mode by clicking on the diagram pane.
4. Then again select the message label. The Create New Operation according to this message button is displayed next to the message name (see the example in Figure 568 on page 922).
5. Click the Create New Operation according to this message button. A new operation is created. and represented in the message Specification window, the **Signature (operation)** property cell.



- Note that a new operation is created only after click the Create New Operation according to this message button. If you will not click this button, message label will be treated as a message name.
- If the syntax is written incorrectly, the Message Parsing window opens. For more information, see "Notification message about incorrect syntax" on page 926.
- For more information on how to assign or create an operation to a message, see "To assign an operation to a message" on page 920, and "To create a new operation for a message" on page 921.

Parsing rules

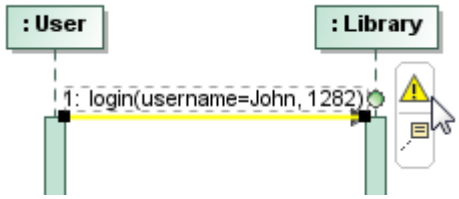
See message name samples and parsing results described in the following table.

Purpose	**Message name typed on the diagram pane**	**Action after** typing on the message label	**Result**
Create a new operation	*login ()*	Click the Create New Operation according to this message button.	The new operation is created: *login ()* 1. The operation is referenced in the message Specification window, the **Signature (operation)** property cell. 2. For the lifeline type, the *login* operation is created.
	NOTE: After click on the Create New Operation according to this message button, always a new operation is created. A new operation is created even if an operation with the same name already exist.		

Copyright © 1998-2015 No Magic, Inc.

Purpose	Message name typed on the diagram pane	Action after typing on the message label	Result
Create a new operation with an argument	*login (“John”)*	Click the Create New Operation according to this message button.	The new operation is created: *login (unnamed1=”John”)* 1. The operation is referenced in the message Specification window, the **Signature (operation)** property cell. 2. In the message Specification window > the **Arguments** property group, the following argument is created: *John*. 3. For the lifeline type, the *login* operation is created. 4. The *login* operation’s parameter is created and synchronized with the message argument. For more information about parameter and argument synchronization, see "Parameters Synchronization" on page 430.
	NOTE: Different kinds of arguments are supported. For the following arguments the following value specifications are created: *John* - Opaque Expression *12345* - Literal Integer *True* - Literal Boolean *“John”* - Literal String (always when quotation-marks are used) For more information about value specification, see "Assigning value specification as property value" on page 312.		
Create a new operation with a parameter and an argument	*login (username=”John”)*	Click the Create New Operation according to this message button.	The new operation is created: *login (username=”John”)* 1. The operation is referenced in the message Specification window, the **Signature (operation)** property cell. 2. In the message Specification window > the **Arguments** property group, the following argument is created: *John*. 3. For the lifeline type, the *login* operation is created. 4. The *login* operation’s parameter is created and synchronized with the message argument. For more information about parameter and argument synchronization, see "Parameters Synchronization" on page 430.

Copyright © 1998-2015 No Magic, Inc.

Purpose	**Message name typed on the diagram pane**	**Action after** typing on the message label	**Result**
Create a new operation with a return parameter	*login () : true*	Click the Create New Operation according to this message button.	The new operation is created: *login () : true* 1. The operation is listed in the message Specification window, the **Signature (operation)** property cell. 2. In the message Specification window > the **Arguments** property group the following argument is created: *true*. 3. For the classifier that is assigned to the lifeline, the *login* operation is created together with the parameter. 4. The *login* operation's parameter with *return* direction is created. The operation's parameter is synchronized with the message argument. For more information about parameter and argument synchronization, see "Parameters Synchronization" on page 430.
Create a new operation for the message together with a reply message	*status = login ()*	Click the Create New Operation according to this message button.	The new operation is created: *status = login ()* 1. The operation is listed in the message Specification window, the **Signature (operation)** property cell. 2. In the message Specification window > the **Reply message** property cell, the replay message is created. (The **Replay message** property cell is displayed in Expert mode only.) 3. The *status* argument is created in the replay message.
Create a new argument for the message **NOTE:** This is valid then the argument is added straight from the message label for the already existing operation	Before was: *login (username=John)* Add the argument 1282: *login (username=John, 1284)*	Exit the message label edit mode by clicking on the diagram pane.	A new argument is created and the message is highlighted with yellow color, that is, the validation warning appears informing that a parameter is missing and the message argument is not synchronized with the operation parameter.  To resolve the validation warning: 1. Click on the message. The message smart manipulator toolbar displays. 2. Click the button to synchronize the message argument with the operation parameter. For more information about parameter and argument synchronization, see "Parameters Synchronization" on page 430.

Copyright © 1998-2015 No Magic, Inc.

Purpose	**Message name typed on the diagram pane**	**Action after** typing on the message label	**Result**
Update the argument	Before was: *login (John)* Now change the argument to: *login (Joseph)*	Exit the message label edit mode by clicking on the diagram pane.	The argument is updated.



If you would like to update the operation name or parameter you can do these actions in the operation Specification window. These actions are not allowed straight from the message, in order not to corrupt the referenced operation.

Notification message about incorrect syntax

If the message syntax is incorrect you will receive the following notification message at the right-bottom corner of the MagicDraw window.

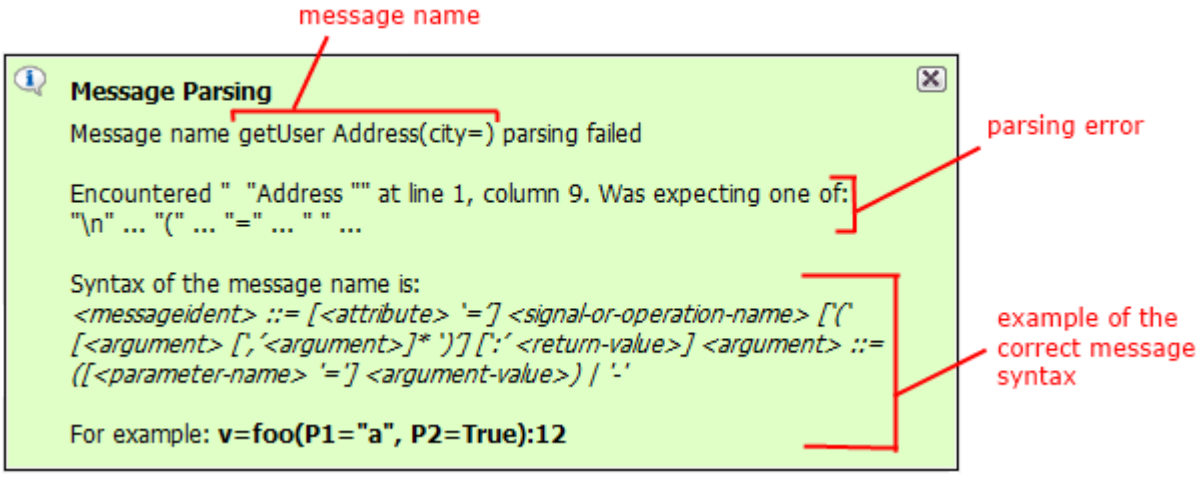


Figure 569 -- Example of Message Parsing notification message

In the Message Parsing notification message, in the first line you will see message name that is incorrect. In the example, the incorrect message name is *getUser Address(city=)*. Because of the incorrect message name, parsing cannot be executed and a new operation cannot be created.In the second paragraph of the Message Parsing notification message, you will see the explanation of what syntax is expected. Next, you can see an example of the correct message syntax.

Assigning signals to messages

You can assign signal only to send or call (synchronous or asynchronous) messages. Furthermore, only one signal can be assigned to a message.



If a signal is assigned to a call message, the message is automatically converted to a send message.

To assign a signal to a message using a drag-and-drop

1. Select a signal in the Containment tree.
2. Drag the selected signal to a desired message.

Copyright © 1998-2015 No Magic, Inc.

To assign a signal to a message via the message Specification window

1. Draw a send message between two lifelines or select an existing one on a diagram pane.
2. Open the message Specification window (see the procedure "To open the element Specification window" on page 275).
3. Click the **Signature (signal)** property value cell and then select a signal in the drop-down list.

Creating signal receptions for messages

Assigning a signal reception to a message is very similar to the procedure of assigning an operation to a message.



There are two conditions that must be satisfied when creating a new signal reception. They are as follows:

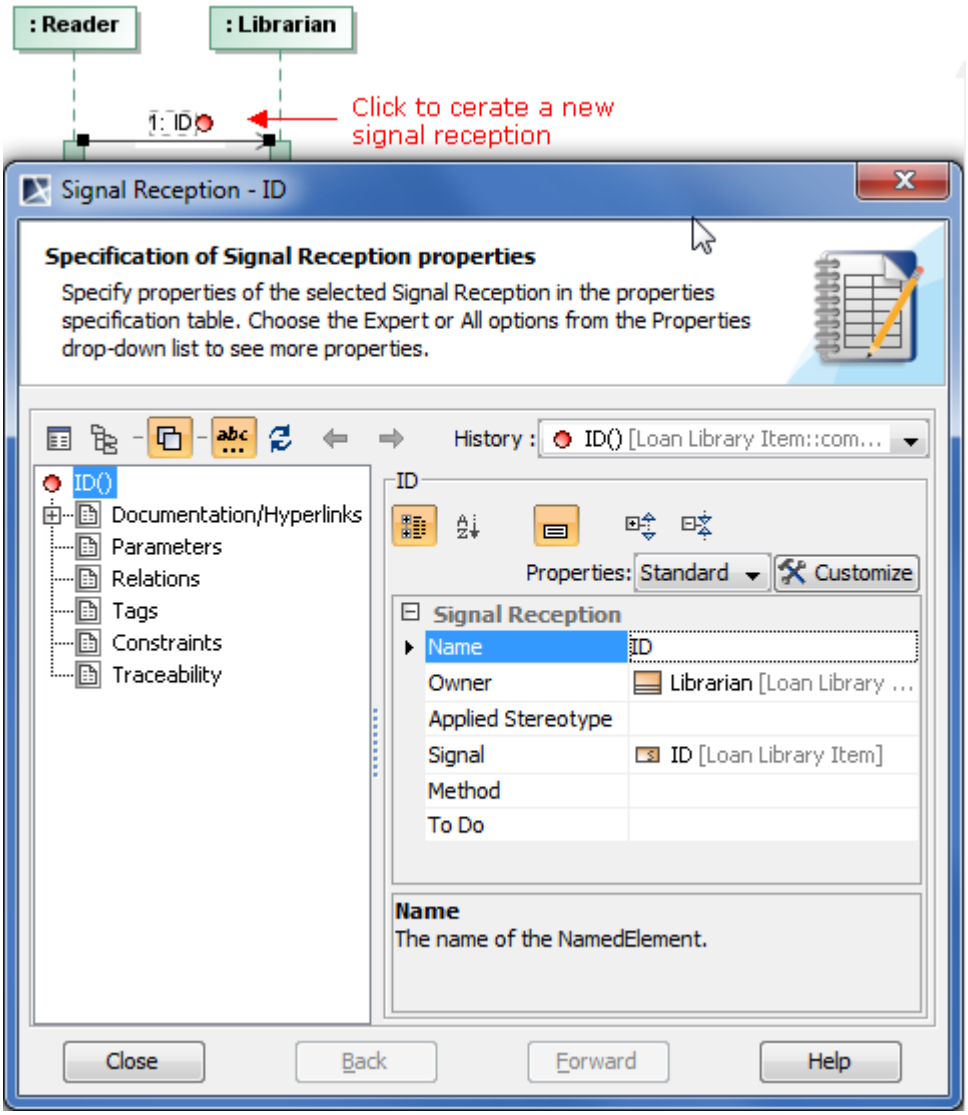
- At least one signal must exist in your project.
- A possible signal reception receiver (an activation to which the message points) must have a type assigned.

To create a new signal reception for the message

1. Draw a send message between the lifelines or select an existing one on a diagram pane.
2. Assign a signal to the message (see the procedure "To assign a signal to a message using a drag-and-drop" on page 926).
3. Do one of the following:
    - Click the button in the shape of a small red circle at the end of the message name (see the following figure). Define signal reception property values in the signal

Copyright © 1998-2015 No Magic, Inc.

reception Specification window. The default name of the signal reception is the name of the message for which the signal reception is being created.



- From the message shortcut menu, select the **Create New Signal Reception** (the command with the icon representing a small red circle). Define signal reception property values in the signal reception Specification window. The default name of the signal reception is the name of the message for which the signal reception is being created.

Messages in Sequence and Time Diagrams

Messages allow for displaying an interaction between objects. A message is labeled with either the message name or the assigned operation (signal) name and its arguments.

Related diagrams

Time Diagram

"Sequence Diagram", on page 683

Related references

"Message", on page 917

Copyright © 1998-2015 No Magic, Inc.

Drawing message from/to Lifeline to/from diagram frame

Outgoing and incoming messages communicate with the outside environment by connecting to the border gates or in other words to the diagram frame.

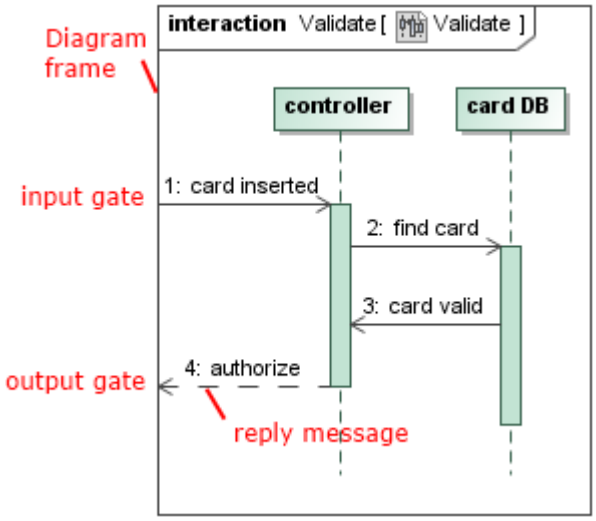


Figure 570 -- Example of messages connected to diagram frame

In the preceding example, you can see the *card inserted* message that is drawn from the diagram frame border to the *controller* lifeline. The *authorize* message is created from the *controller* lifeline to the diagram frame.



For a message to be correct, its receive end should be known. However, we never know who will connect to the gate from the outside. That's why we are using the reply message when creating message from the lifeline to the diagram frame (see *authorize* message).

Creating nested activation

Nested activations allow you to model:

- A parallel execution of operations that belong to a single class.
- Callback messages.

The nested activation can be created between at least two messages that point to the same activation.

Nested activations can be created for the following message sorts:

1. Synchronous Call Message (synchCall)
2. Asynchronous Call Message (asynchCall)
3. Asynchronous Signal Message (asynchSignal)

To create a nested activation



Be sure, you have at least two messages pointing to the same activation in your model.

1. Select any subsequent message.
2. Do one of the following:
   - From the message shortcut menu, select **Create Nested Activation**.
   - On the message smart manipulator toolbar, click the Create Nested Activation button.

Copyright © 1998-2015 No Magic, Inc.

The message will be connected to the nested activation.

To merge a nested activation with a parent activation

1. Select a message that has a nested activation.
2. Do one of the following:
   - From the message shortcut menu, select **Reduce Nesting Level**.
   - On the message smart manipulator toolbar, click the Reduce Nested Activation button.

The message will be connected to the parent activation.

Nested activations can be used in the following cases:

- To model parallel executions for a non-active lifeline
- To model parallel executions for an active lifeline
- To model a callback message

To model parallel executions for a non-active lifeline

NOTE

A non-active lifeline is the one that has a non-active class as a type assigned. The non-active class is the one whose Is Active property is set to false. This property is available in the Expert mode.

This is the case of creating a simple nested activation, described in the procedure "To create a nested activation" on page 929.

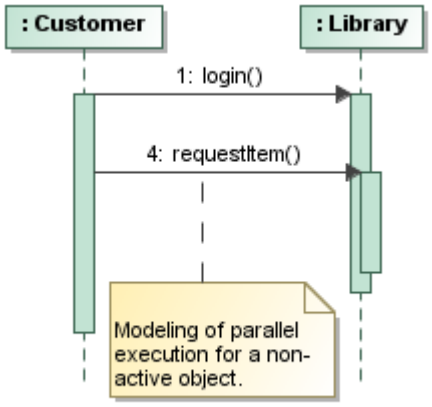


Figure 571 -- Parallel executions for non-active object

To model parallel executions for an active lifeline

NOTE

An active lifeline is the one that has an active class as a type assigned. The active class is the one whose Is Active property is set to true. This property is available in the Expert mode.

1. Select a lifeline and from its shortcut menu, select **Show Entire Activation**. All activations of the selected lifeline becomes all-in-one.
2. Create an outgoing message or select an existing message pointed to active object. Be sure this is not the first one for the outgoing activation.

Copyright © 1998-2015 No Magic, Inc.

3. From the message shortcut menu, select **Create Nested Activation**.

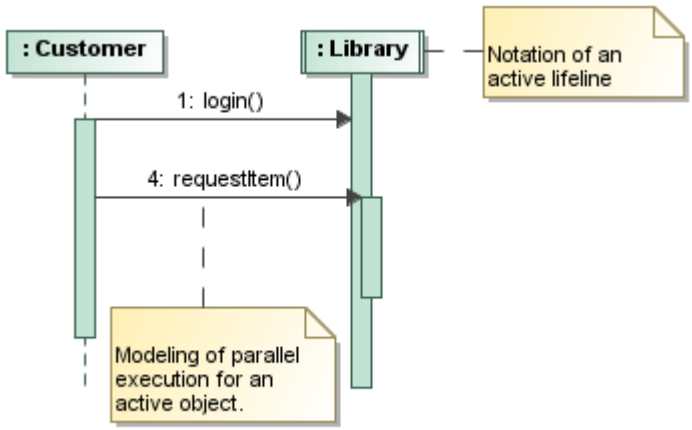


Figure 572 -- Parallel executions for active object

To model a callback message

1. Create or select an incoming message that is modeled as a callback message.
2. From the message shortcut menu, select **Create Nested Activation**.

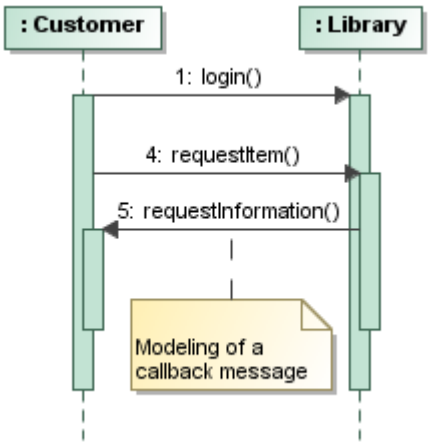


Figure 573 -- Callback messages

Messages in Communication Diagrams

Messages in communication diagrams can only be depicted on connectors. Therefore, you should draw a connector first and then assign a message to it.

To assign a message to a connector

1. On the diagram pallet, click the button corresponding a desired message type.

Copyright © 1998-2015 No Magic, Inc.

2. Click a desired connector on the diagram pane. A message arrow will be placed on the selected connector.

NOTE

A message flow has two directions: right and left. Choose one of them by clicking the associated button on the diagram pallet.



To set the advanced numbering of messages

1. From the diagram shortcut menu, select **Numbering**.
2. Select the **Use Advanced Numbering**.

To remove the advanced numbering of messages

1. From the diagram shortcut menu, select **Numbering**.
2. Clear the **Use Advanced Numbering** selection.

To change a current message numbering

1. From the diagram shortcut menu, select **Numbering** > **Change Numbering**.
2. Increase, decrease, and / or change the level of numbering in the **Change Communication Numbering** dialog.

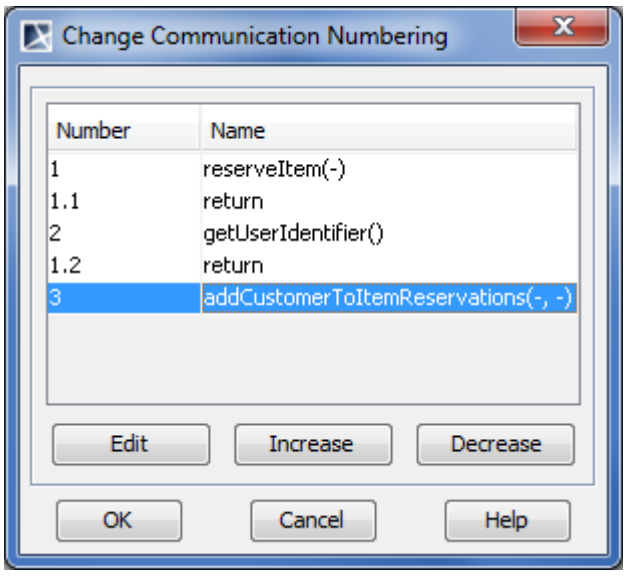


Figure 574 -- Change Communication Numbering dialog

Column	Description
Number	Message number.
Name	Message name.

Copyright © 1998-2015 No Magic, Inc.

Button	Description
Edit	Opens the **Type Number** dialog. Type the number of the message. **NOTE:** If the **Edit** button is inactive, remove the automatic advanced numbering of messages (see the procedure "To remove the advanced numbering of messages" on page 932).
Increase	Increases the selected number by one.
Decrease	Decreases the selected number by one.

Activators and predecessors

The predecessor denotes the set of messages. The completion of these messages enables the execution of the current message. The meaning of the predecessor is that the execution of a message is not enabled until all of the communications of which the sequence numbers appeared in the list have occurred. Therefore, the list of predecessors represents a synchronization of threads. The message corresponding to the numerically preceding sequence number is an implicit predecessor and does not need to be explicitly listed.

All of the sequence numbers with the same prefix form a sequence. The numerical predecessor is the one in which the final term is one less. That is, number 6.4 is the predecessor of 6.5, where the number “6” is an activator (see example in the following figure).

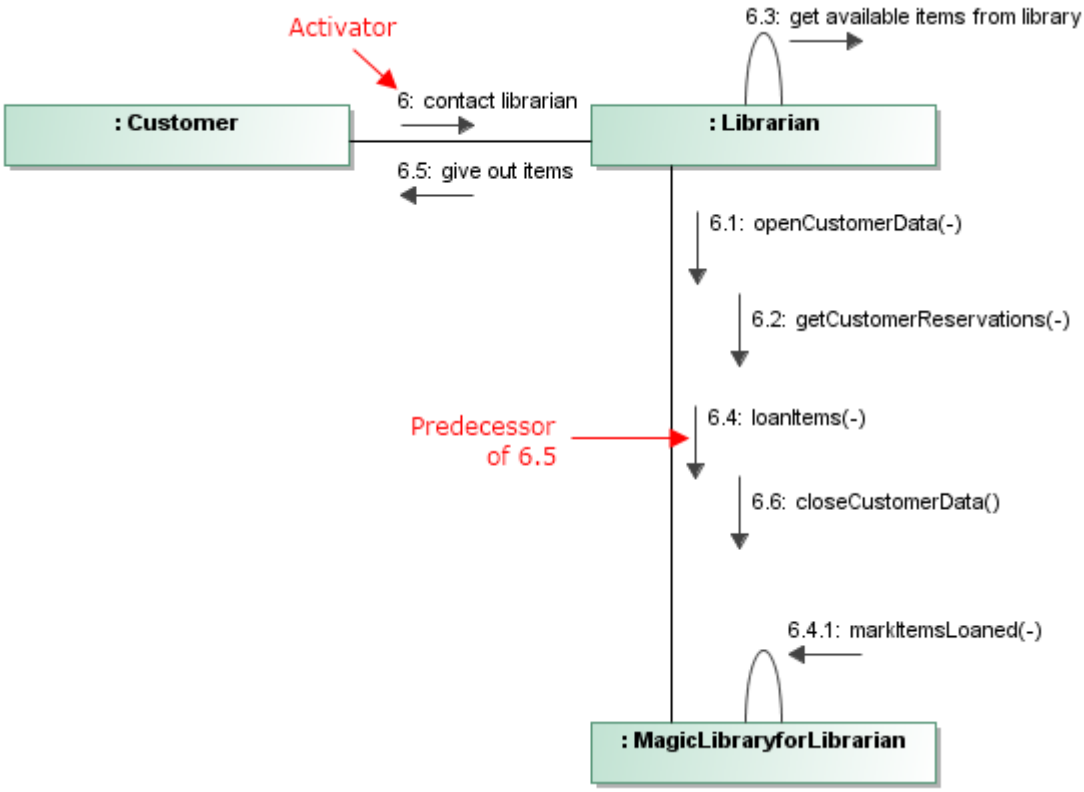


Figure 575 -- Message numbering with activators and predecessors

To show predecessors beside the message number

NOTE

Make sure the **Use Advanced Numbering** is selected (see the procedure "To set the advanced numbering of messages" on page 932).

Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

- From the message shortcut menu, select **Show Predecessors**.
- In the message **Symbol Properties** dialog, set **Show Predecessors** to *true*.

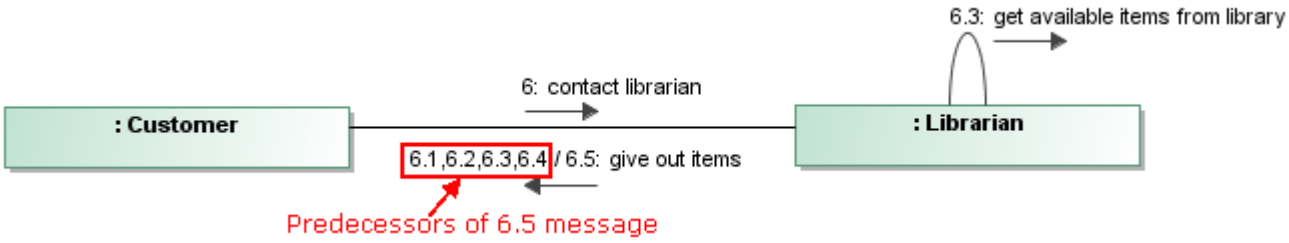


Figure 576 -- Predecessors displayed beside message number

To change an activator number of messages

NOTE

Make sure the **Use Advanced Numbering** is selected (see the procedure "To set the advanced numbering of messages" on page 932).

1. From the message shortcut menu, select **Activator**.

Copyright © 1998-2015 No Magic, Inc.

2. Select the activator that you want to assign to the selected message. Numbering of the selected message and subsequent messages decreases by one level. The first level number is the number of an activator message (see the following figure).

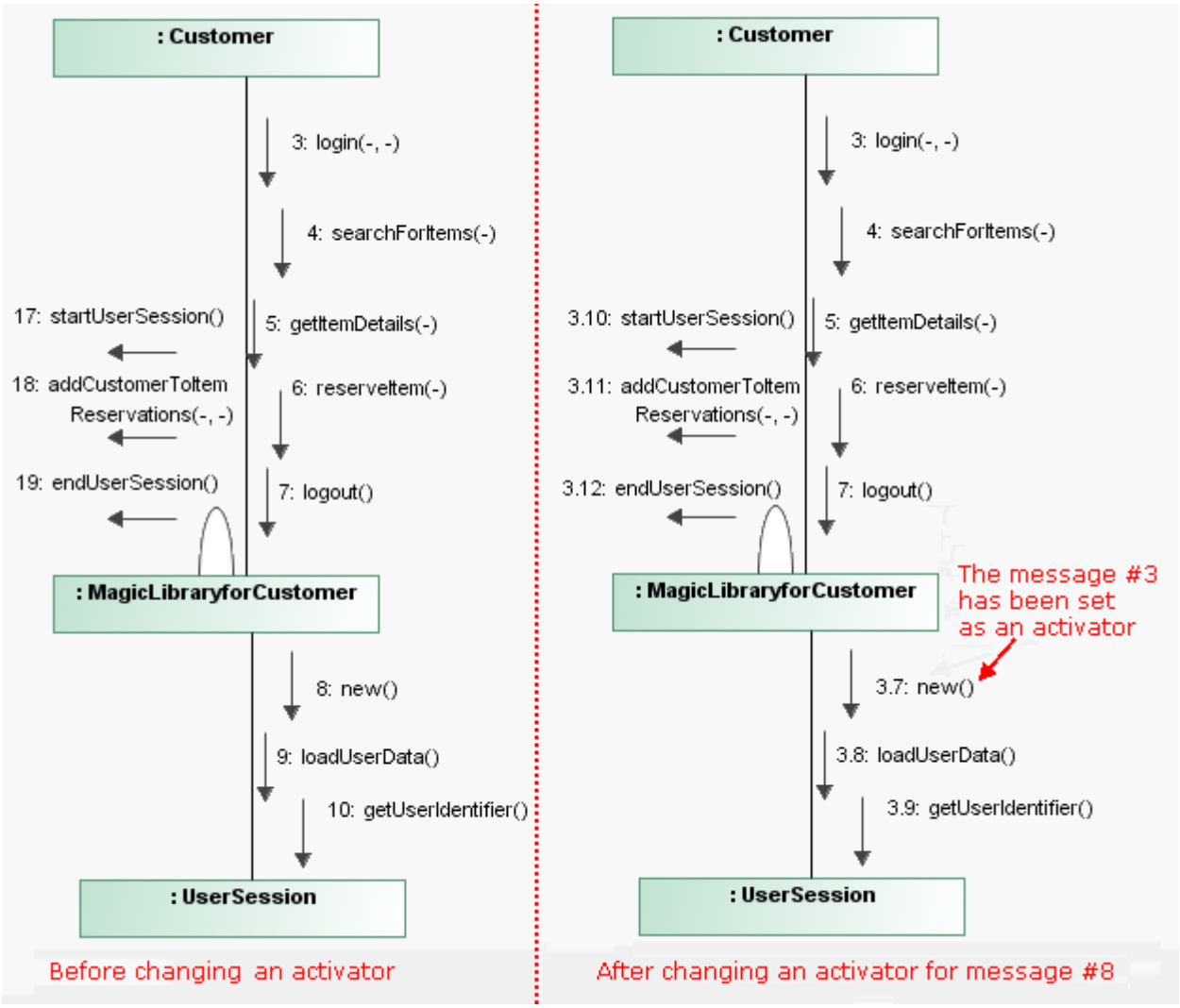


Figure 577 -- Changing activator

IMPORTANT!

If you change the activator number to a predecessor message, this number will change for subsequent predecessors.

Model

A model contains a (hierarchical) set of elements that together describe the physical system being modeled. It can also contain a set of elements that represents the environment of the system, typically Actors together with their interrelationships, such as Associations and Dependencies.

A model is presented as a package with a small triangle in the upper right corner of the large rectangle. The triangle can be shown in the tab.

Copyright © 1998-2015 No Magic, Inc.

The model is defined as a package, that is, it has package properties in the **Model** Specification window. For a detailed description of packages, see Section "Package" on page 957.

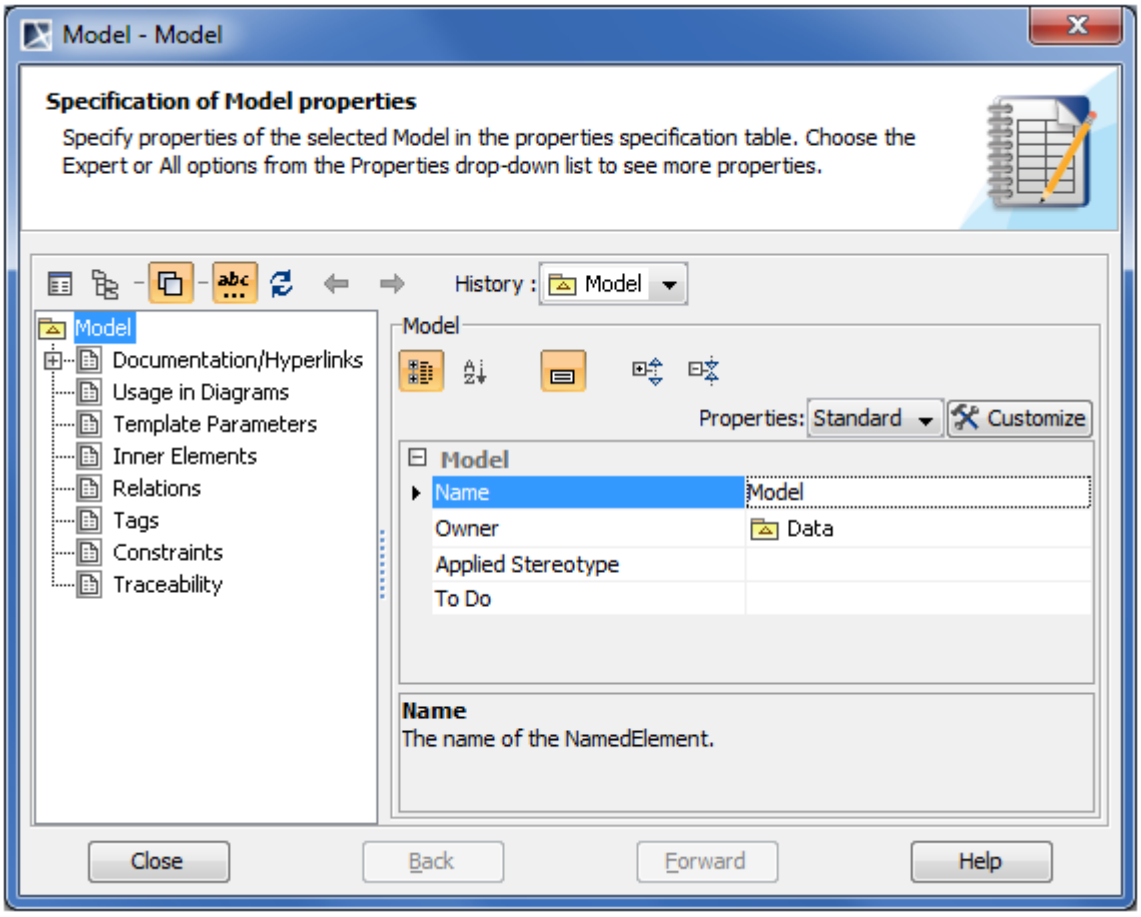


Figure 578 -- Model Specification window

TIP!

For information about the specification properties not covered in this section, see "Specification Window" on page 273.

Node

TIP!

For information about the specification properties not covered in this section, see "Specification Window" on page 273.

Any computer or device that is relevant to the implemented system can be shown as a node. The node is drawn as a three-dimensional cube with a name inside it. Devices in a system are typically represented with a stereotype that specifies the device type. The nodes can be represented as types and as instances.

It is shown as a figure that looks like a 3-dimensional view of a cube.

You can specify node properties in the node Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

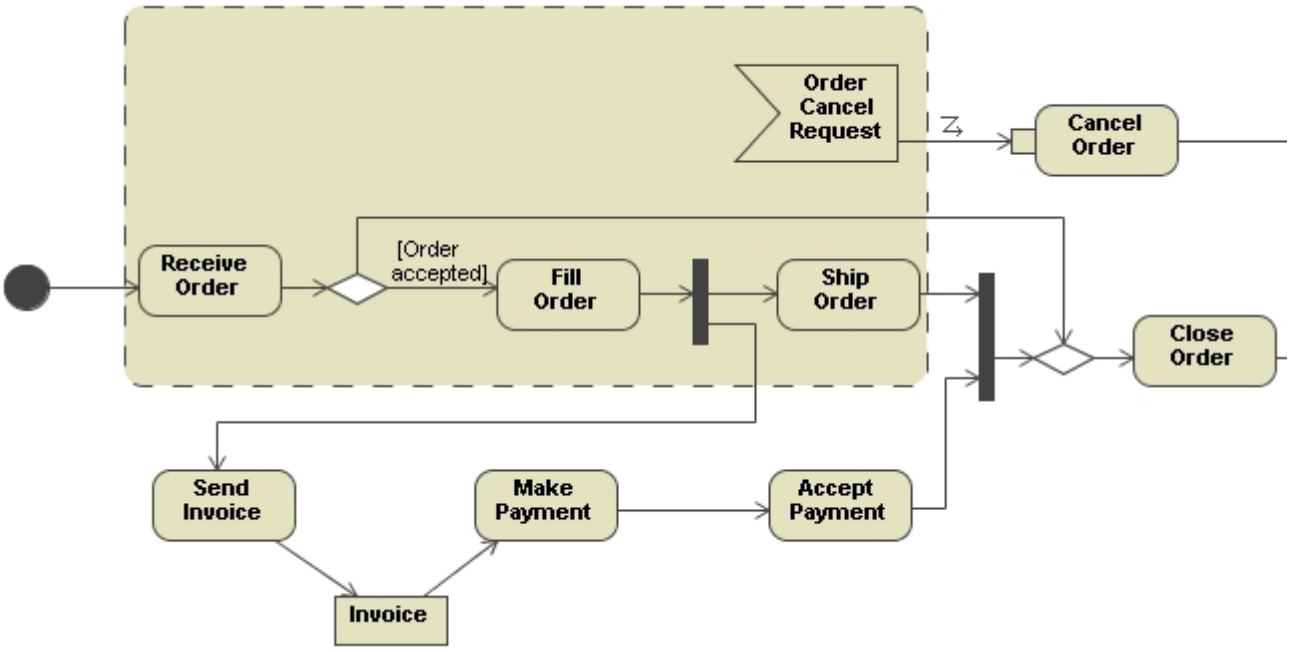
Copyright © 1998-2015 No Magic, Inc.

TIP!

- For more information about working with symbols, see "Diagramming" page 198.
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Structured activity node

A structured activity node is an executable activity node that can have an expansion into the subordinate nodes. It represents a structured portion of the activity that is not shared with any other structured node, except for nesting.



Copyright © 1998-2015 No Magic, Inc.

Activity parameter node

It is an object node for inputs and outputs to activities. The activity parameters are object nodes at the beginning and end of the flows, to accept inputs to an activity and provide outputs from it.

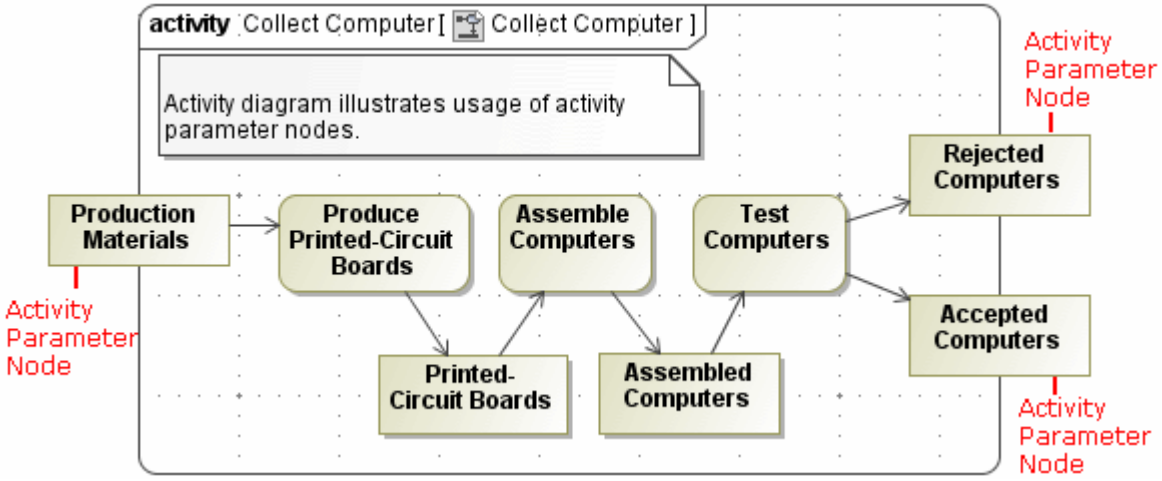


Figure 579 -- Example of Activity Parameter Nodes in Activity diagram

To create an activity parameter node on an activity diagram frame

TIP!

You can draw an object flow straight to or from an activity diagram frame. As a result an activity parameter node will be created and object flow will be connected to it.

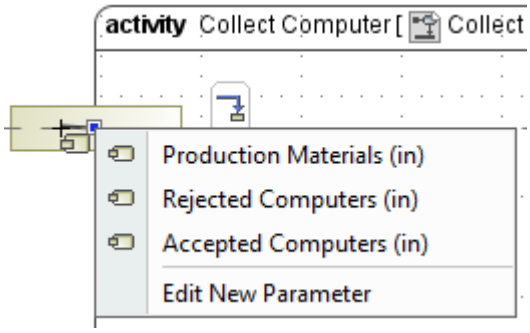
1. In an activity diagram pallet, click the *Object Flow* button.
2. Click the activity diagram frame. The activity parameter node is created and the menu opens.
3. Select the **Edit New Parameter** command to specify the parameter. The **Parameter** dialog opens.
4. Specify the parameter and click **Close**.

Copyright © 1998-2015 No Magic, Inc.

5. Click on the diagram to create a call behavior action via the object flow relation.

TIP!

- The parameter for the activity parameter node is created automatically.
- The parameter name, type and multiplicity are synchronized with the pin or object node, from which / to which the object flow is created. The direction of the parameter is set according to the object flow direction.
- When creating an activity parameter node on a diagram frame, after click on the diagram frame, the menu opens. In the menu there are listed the following activity parameter nodes:
    - that are not displayed on the diagram frame
    - and owned by the same activity as the activity diagram
    - and has the assigned parameters (that are owned by the same activity too).



- If a parameter with *inout* direction is selected for the activity parameter node creation, then two activity parameter nodes are created on the diagram frame.

Copyright © 1998-2015 No Magic, Inc.

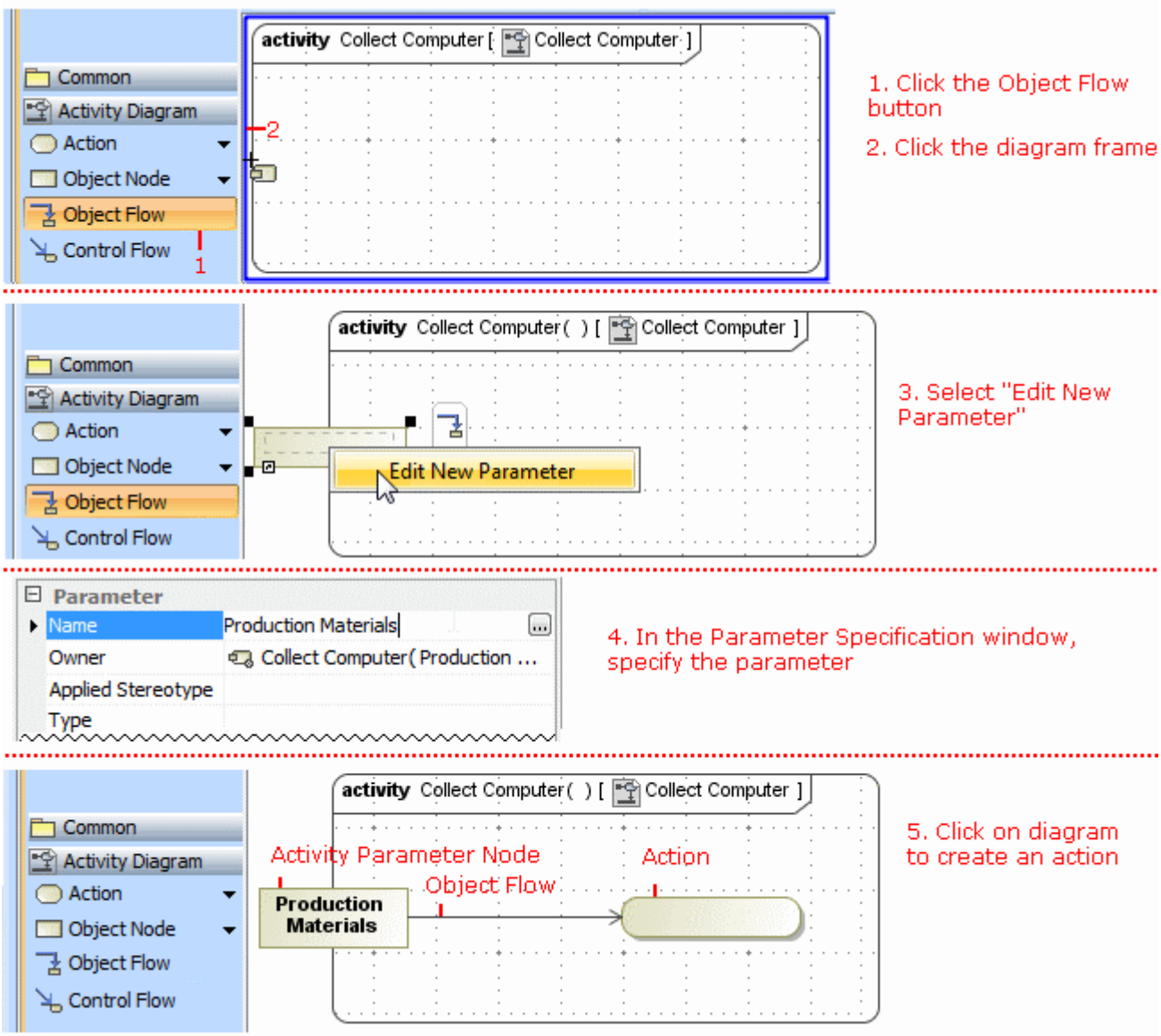


Figure 580 -- Example of Activity Parameter Node creation on Activity diagram frame

Copyright © 1998-2015 No Magic, Inc.

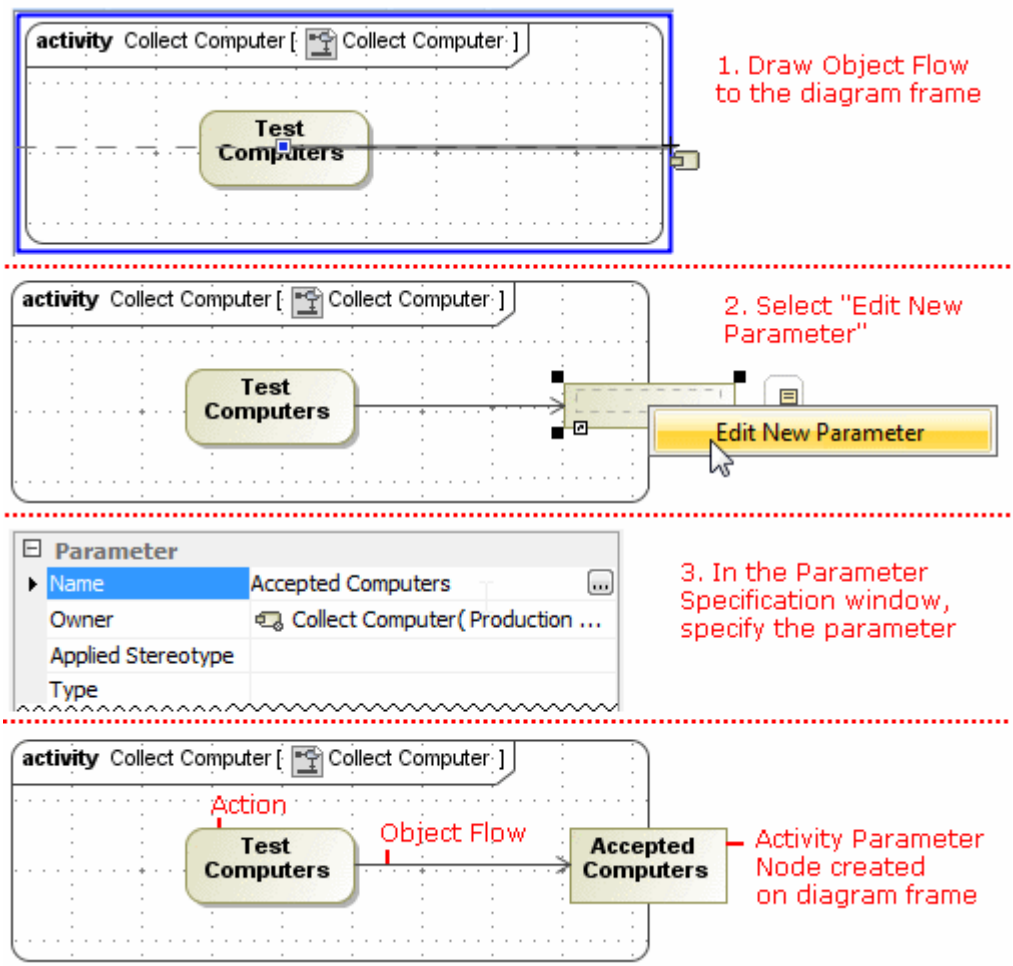


Figure 581 -- Example of drawing Object Flow to diagram frame

To display or hide a direction on an activity parameter node

1. Right-click the activity parameter node on diagram and then select **Symbol Properties**.
2. In the **Symbol Properties** dialog, click to select or clear the **Show Direction** check box.

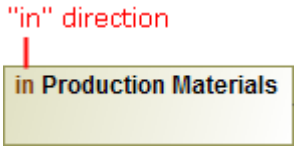


Figure 582 -- Example of the parameter direction displayed on Activity Parameter Node

NOTE

If value of the **Show Parameter Direction** symbol property is true, but value of the **Name Display Mode** is **Show Name**, then direction is not be displayed on the activity parameter node shape.

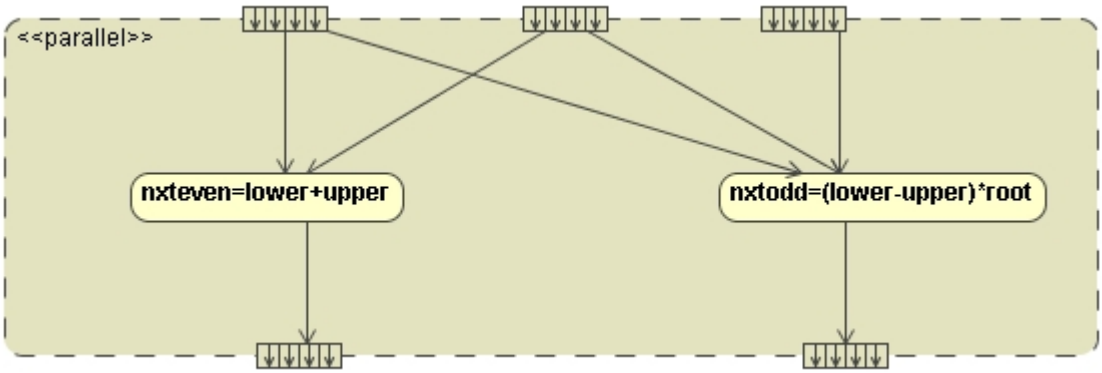
TIP!

- You can see the direction of the parameter displayed in the Model Browser, before the parameter name.
- If an activity parameter node is associated with the parameter of '*inout*' direction, then direction to the activity parameter node is set according to the connected flows. That is if outgoing flow is connected to the newly created activity parameter node, then this activity parameter node is set as input node and the 'in' parameter direction is used.

Copyright © 1998-2015 No Magic, Inc.

Expansion Region and Expansion Nodes

The Expansion Region and Expansion Nodes can be drawn in the activity diagram (the Input and Output Expansion Nodes may be found in the diagram toolbar Object Node button group):

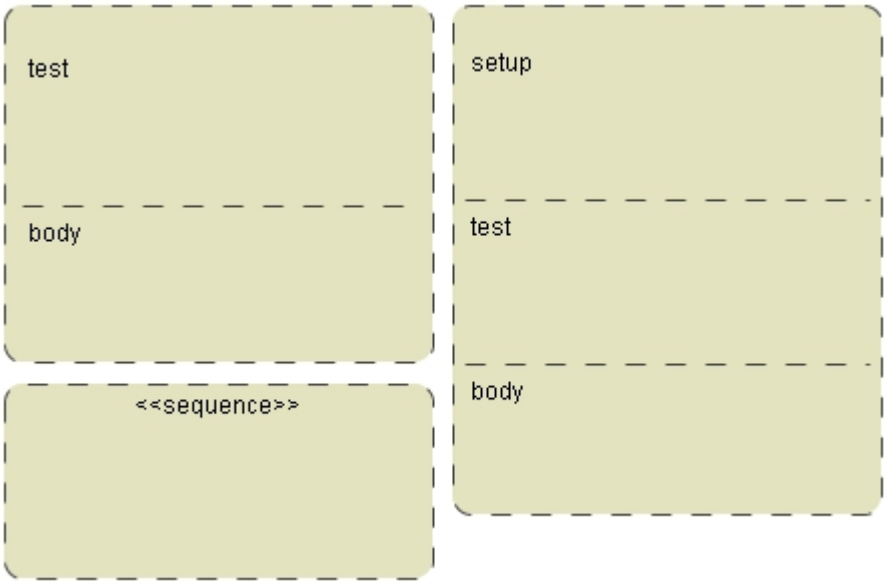


If, Loop and Sequence Conditional Nodes

A conditional node is a structured activity node that represents an exclusive choice among some number of alternatives.

A sequence node is a structured activity node that executes its actions in order.

A loop node is a structured activity node that represents a loop with the setup, test, and body sections.



Copyright © 1998-2015 No Magic, Inc.

Note

A note is a graphical symbol containing textual information. It is used to add any explanatory information needed for your element or diagram. A note is usually connected to the element symbol using an anchor line.

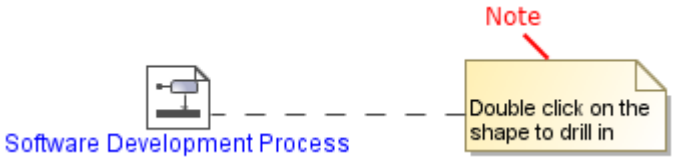


Figure 583 -- Example of note

In the note, you can also display and edit the element's to which the note is anchored documentation, constraint, tagged value, and element properties.

A text in the note can be displayed as a plain text or as an HTML text.

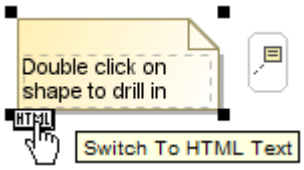
You can format the note symbol properties in the **Symbol Properties** dialog.



For more information about symbol representation properties, see "Formatting Symbols" on page 317.

To switch the note text to HTML or a Plain text, do one of the following

- On the diagram pane, select the note symbol and click the button that appears on the lower-left corner of the shape:
  - The **HTML** button - to switch the text in the note to the HTML text.
  - The **Plain** button - to switch the text in the note to the plain text.



You can edit the HTML text in the note using the HTML editor.



For more information about working with HTML editor, see "HTML Editor" on page 422.

To display an anchored element documentation on a note

1. Anchor a note to an element symbol, if the note is not anchored (see the procedure "To draw an anchor from a symbol" on page 838), or select the existing anchored note.
2. Open that note's shortcut menu.
3. Select **Text Display Mode** > **Show Documentation**. The anchored element documentation is displayed on the note's symbol.



You can edit element documentation directly on the note symbol.

To display element properties, constraints, and/or tagged values on a note symbol

1. Anchor a note to an element symbol, if the note is not anchored (see the procedure "To draw an anchor from a symbol" on page 838), or select the existing anchored note.
2. Click the note on the diagram.

Copyright © 1998-2015 No Magic, Inc.

3. Click the *Compartments* smart manipulator.
4. On the menu, click to select the following check boxes:
   - **Element Properties**
   - **Constraints**
   - **Tagged Values**

Element properties, constraints, and / or tagged values are displayed on the note symbol in the separate compartments.

You can choose which element properties, constraints, or tagged values to display on the note symbol by editing an appropriate compartment.

To edit a compartment on a note symbol

NOTE

To see compartment editing results, make sure element properties, constraints, and/or tagged values are displayed on the note symbol. For displaying instruction, see the procedure "To display element properties, constraints, and/or tagged values on a note symbol" on page 943.

1. Anchor a note to an element symbol, if the note is not anchored (see the procedure "To draw an anchor from a symbol" on page 838), or select the existing anchored note.
2. To open the **Compartment Edit** dialog, do one of the following:
   - On the diagram, right-click the note and click **Edit Compartments**.
   - On the diagram, click the note and click the *Edit Element Properties* smart manipulator.

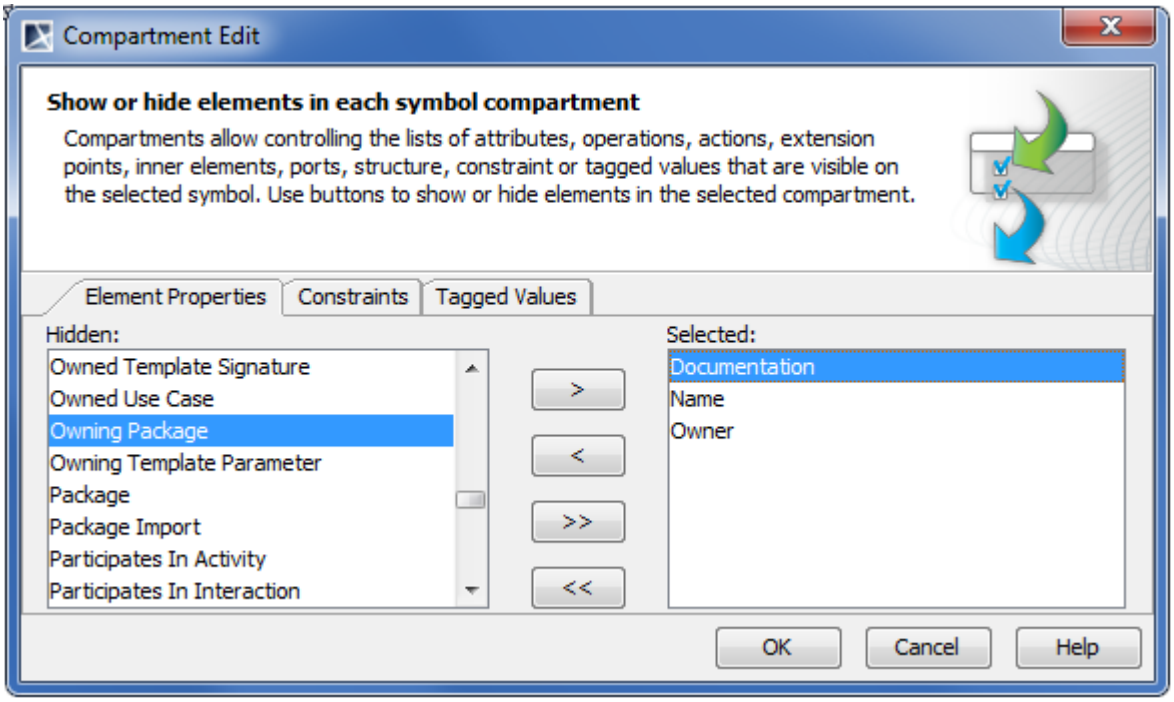


Figure 584 -- Compartment Edit dialog

3. In the **Hidden** list, select one or more items you want to display on the note's compartment.
4. Using buttons **>**, **<**, **>>**, or **<<** move selected items to or remove from the **Selected** list. Only items from the selected list will be displayed in the compartment on the note's symbol.

TIP!

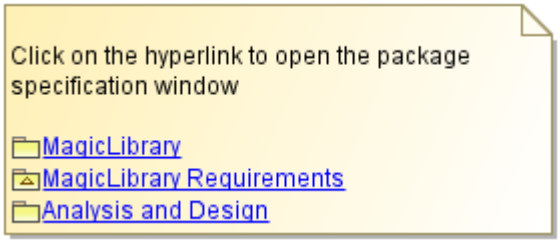
You can open other tab and organize what items to display in the compartment.

5. Click **OK** after you have done.

Copyright © 1998-2015 No Magic, Inc.

To add a hyperlink to an element on a note symbol

1. Select one or more elements in the Model Browser, or Specification window.
2. Drag the selected elements on the note symbol on the diagram pane. The hyperlinks to specifications of dragged elements are created on the note. Click the link on the note to open the Specification window of the selected element.



NOTE

After the hyperlink to the element on the note is added, the text format in the note changes to HTML.

Related elements

Anchor

Related procedures

Formatting Symbols
HTML Editor
Smart Manipulation

Object Flow

TIP!

You can view object flows that are connected via pins in the action Specification window, as well as the **Display Related Elements** and **Display Paths** commands are available from the action shortcut menu.

To select output/input pin automatically when drawing an Object Flow

When drawing an object flow, MagicDraw automatically selects output or input pins, creates a new under certain conditions or you can select a pin from the list. For example, a list of the available to select input pins opens when drawing object flow to the action that has more than one hidden input pin.

Copyright © 1998-2015 No Magic, Inc.

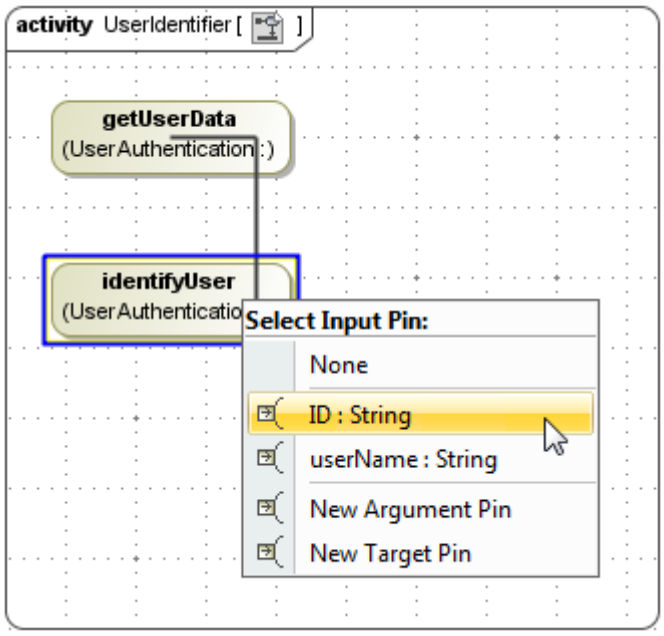
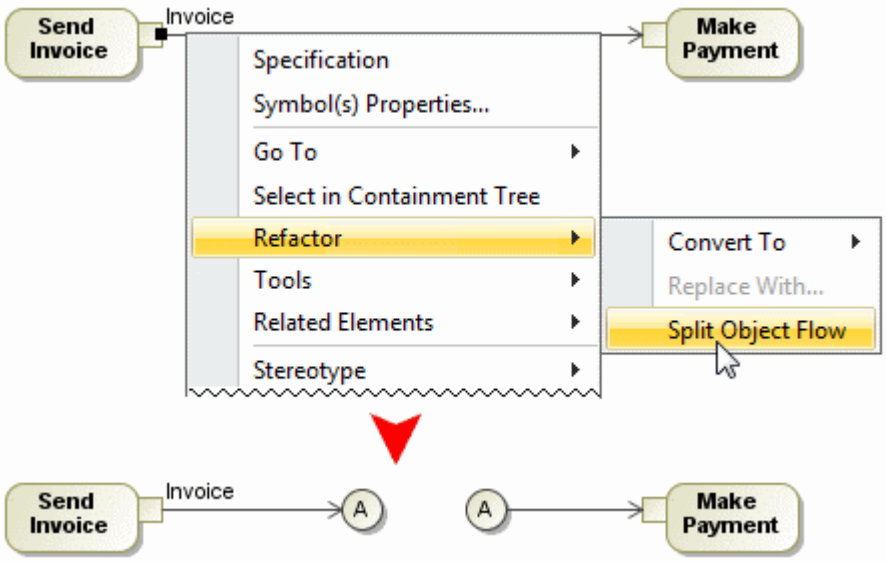


Figure 585 -- Example of existing input pins

To split Object Flows

- From an object flow shortcut menu on a diagram, select **Refactor** and then the **Split Object Flow** command.



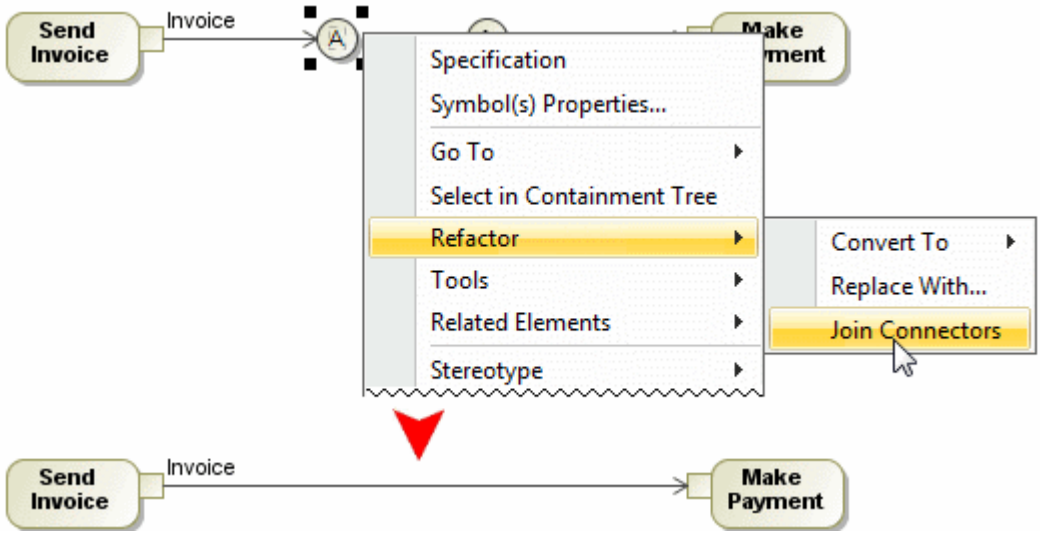
TIP!

- The object flow splits into the two connectors that are marked with *A* letter. Double-click the connector to select the other connector on the diagram pane.
- Split of object flows helps to navigate in a complicated diagram.
- For more information about Refactoring, see "Refactoring" on page 359.

Copyright © 1998-2015 No Magic, Inc.

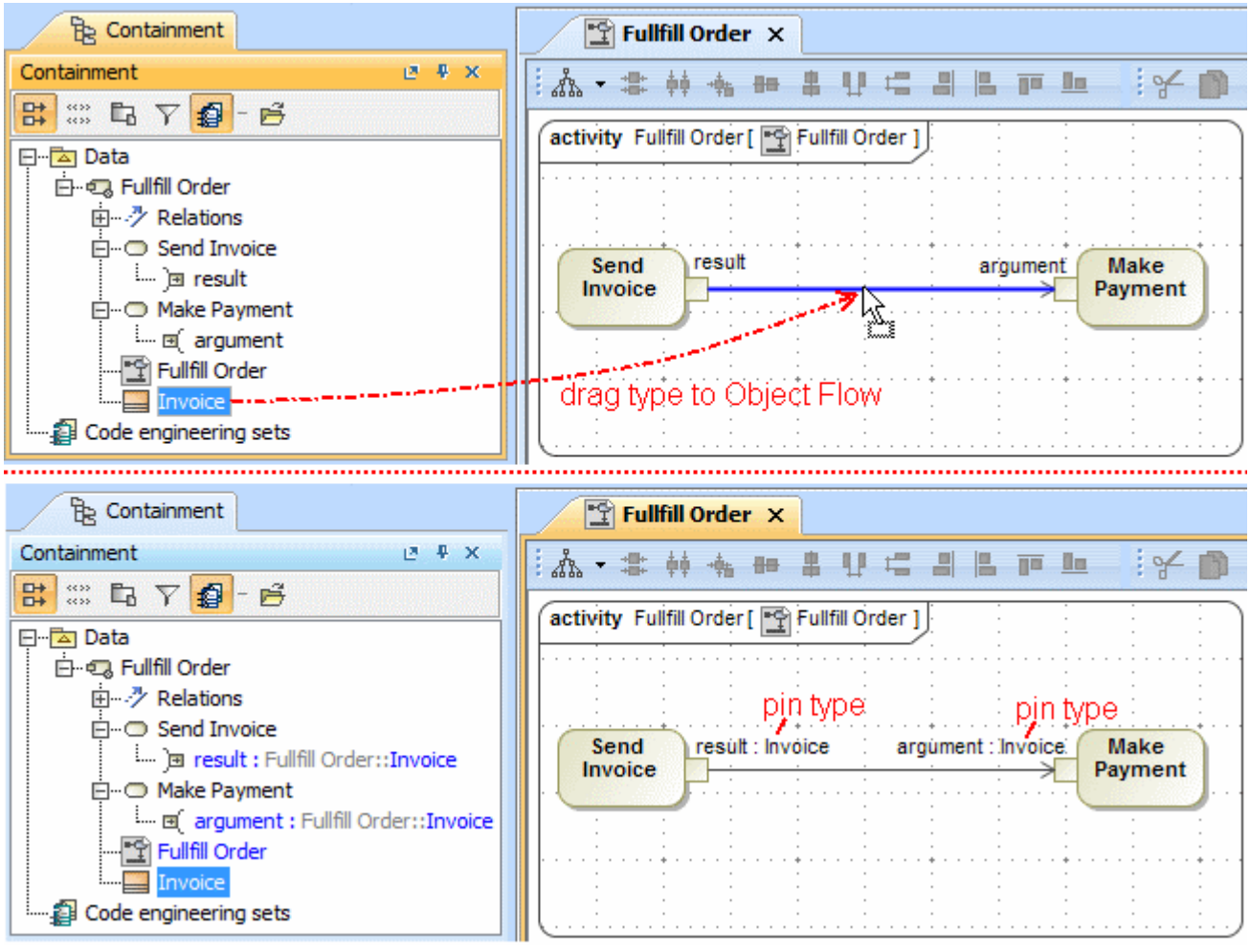
To join Object Flow connectors

- From a connector shortcut menu, select the **Refactor** command and then **Join Connectors**.



To set type of Pins by using a drag-and-drop operation

1. In the Model Browser, select a type of pins.
2. Drag the type to an object flow on a diagram.



Copyright © 1998-2015 No Magic, Inc.

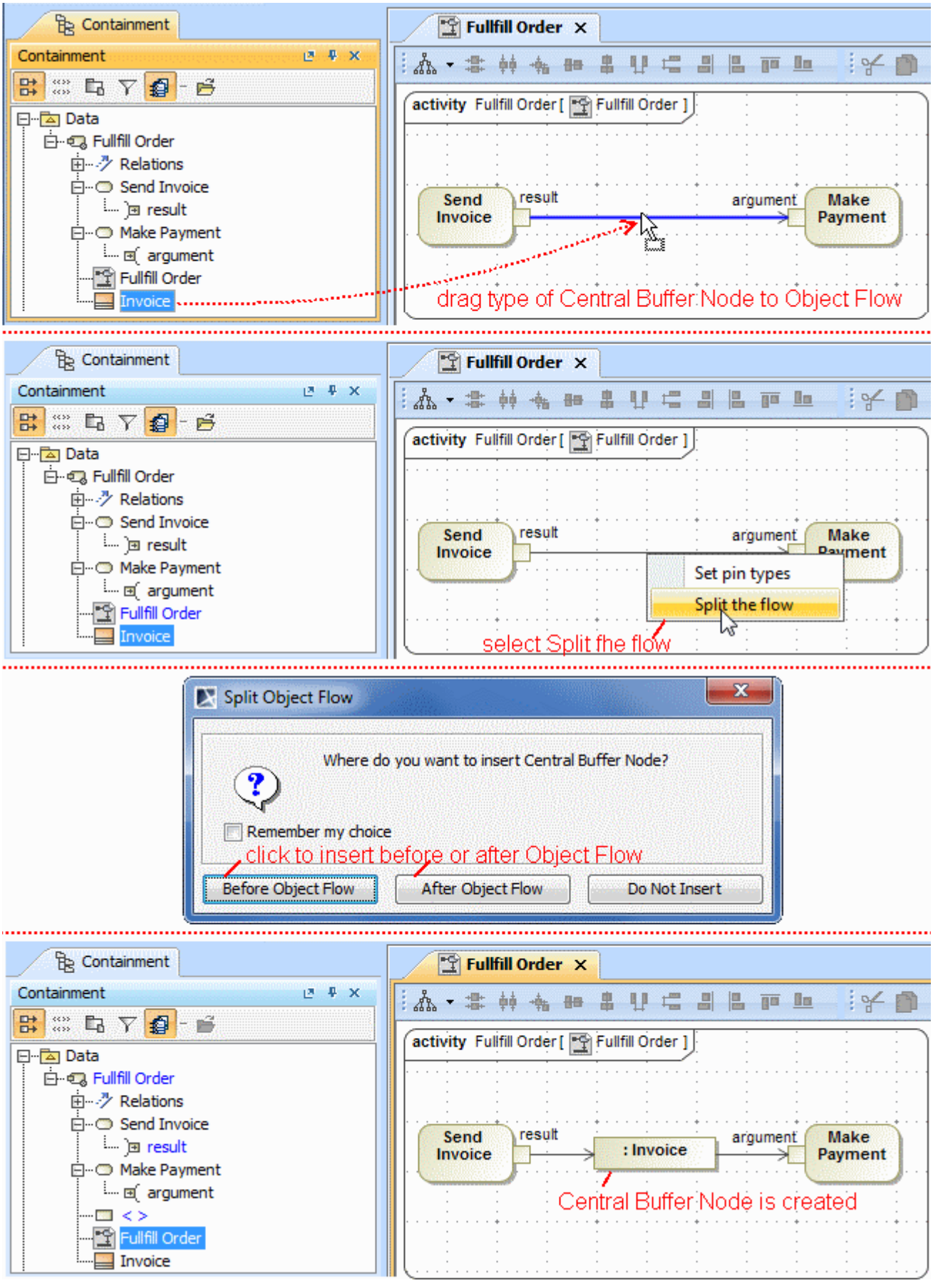


- If type of pins is set already, after drag-and-drop type will be changed to the new one.
- To display a type of a pin on a diagram, from the pin shape shortcut menu, select **Symbol Properties** and then select the **Show Type** check box.

To create Central Buffer Nodes on Object Flows by using a drag-and-drop operation

1. In the Model Browser select a type of a central buffer node, and drag it to an object flow path on a diagram.
2. Wait for a few seconds and drop it.
3. Select the **Split the flow** command.
4. The **Split Object Flow** message opens. Select to insert the central buffer node before or after the object flow.

Copyright © 1998-2015 No Magic, Inc.



Copyright © 1998-2015 No Magic, Inc.

Related diagrams

Activity Diagram

Related procedures

Formatting Symbols

Related dialogs

Specification Window

Object Node

An object node is an activity node that indicates an instance of a particular classifier, possibly in a particular state, may be available at a particular point in the activity. It can be used in a variety of ways, depending on where the objects are flowing from and to, as described in the semantics section.



For more information about working with symbols, see "Diagramming" page 198.

You can specify object node properties in the central buffer node Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



As UML2 does not support generic Object Node, MagicDraw creates the particular Central Buffer Node element.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To set a classifier to an object node

1. Double-click the object node or select **Specification** from the shape shortcut menu. The **Central Buffer Node** Specification window opens.
2. Select a classifier you wish to assign to an object node from the **Type** drop-down list.

To assign a state or final state to an object node

1. Click the **"..."** button in the **Central Buffer Node** Specification window, **In State** field. The **Select Elements** dialog opens.
2. Select a state from the existing model elements, or click **Create**. The **State** Specification window opens. Specify a new state, which will be assigned to an object node.

To convert an object node to a pin

1. Select an object node on the diagram pane.
2. From the object node shortcut menu, select the **Refactor** command, then **Convert To**.

Copyright © 1998-2015 No Magic, Inc.

3. In the opened list, select to convert the object node to the input pin or output.

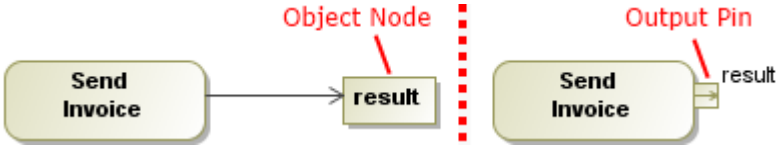


Figure 586 -- Example of Object Node (on the left) conversion to Output Pin (on the right)

Related diagrams

Activity Diagram

Related procedures

Formatting Symbols

Related dialogs

Specification Window

Opaque Behavior

A behavior with implementation-specific semantics. The opaque behavior is introduced for implementation-specific behavior or for use as a place-holder before one of the other behaviors is chosen.

To create an opaque behavior

1. In the Model Browser, right-click a package.
2. From the package shortcut menu, select **Create Element** > **Opaque Behavior**.
3. Type a name for the new opaque behavior.
4. In the Specification window of the opaque behavior, click the cell of the **Body and Language** property value, and then click ... . The **Body and Language** dialog opens.
5. From the **Language** drop-down list, select a language.
6. In the **Body** area, define the expression.

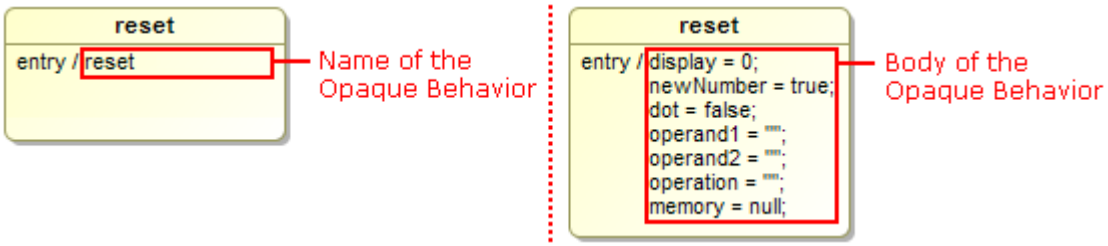


In case you selected the StructuredExpression language, find more about building structured expressions in "Specifying Criteria for Querying Model" on page 551.

7. Close the dialog.
8. Close the Specification window.

NEW! To change an opaque behavior display mode on state or transition shapes

1. Right-click the State or Transition shape on the diagram, and then click **Symbol Properties**.
2. Click the **Opaque Behavior Display Mode** value cell to open the list of available modes, and then click **Body** or **Name**.



Copyright © 1998-2015 No Magic, Inc.

To create a symbol of an opaque behavior

- Drag the selected opaque behavior from the Model Browser to the diagram pane.

Creating executable opaque behaviors

Language of an executable opaque behavior can be OCL 2.0, binary, BeanShell, Groovy, JRuby, JavaScript, Jython, or StructuredExpression. Also, it must have the proper number of parameters with proper types.

How many parameters can an opaque behavior have?

The number of parameters an opaque behavior can have depends on the selected language of the opaque behavior body:

- An OCL 2.0 expression must have a single parameter.
- A binary expression must declare the exact number of parameters has the Java class, to which the expression body of the opaque behavior references.
- Other script expressions, such as JavaScript or Groovy, can have as many parameters as you need.
- Structured expression can have as many parameters as you need.

How to create a parameter for the opaque behavior?

To create a parameter for the opaque behavior

1. Open the Specification window of the opaque behavior.
2. Click **Parameters** on the left and the click the **Create** button on the **Parameters** panel. The Specification window of the new parameters opens.
3. Type a name of the new parameter.
4. Specify the multiplicity of the parameter: either select a value from the drop-down list or type a new one.



In this case, only the upper bound of the multiplicity is important.
The lower bound is not important, so for example, the multiplicity [0..1] has the same meaning as the multiplicity [1].

5. Specify the type of the parameter: either select a value from the drop-down list or create a new one by typing directly in the cell.



Possible types:

- Built-in UML primitives (Integer, Real, String, and so forth)
- UML element types
- Java classes (java.io.File, java.util.Properties, and so forth)

6. If the multiplicity upper bound is 1, you may skip this step. Otherwise, specify the following:
   - If the arguments are unique.
   - If the arguments are ordered.



If you do not see the **Is Unique** and **Is Ordered** properties, select **All** from the **Properties** drop-down list in the upper right of the Specification window.

7. Close the Specification window.

The new parameter is created.

Copyright © 1998-2015 No Magic, Inc.

How to access the arguments and other values from script body of the opaque behavior?



The following instructions applies to BeanShell, Groovy, JRuby, JavaScript, and Jython scripts only.

To access an argument from a script body, you should refer to the corresponding parameter name.

The script body can access the following values:

- Arguments passed to opaque behavior as parameters.
- Globally defined values:
  - *project* (current project)
  - *application*

How many statements can a script have?



The following instructions applies to BeanShell, Groovy, JRuby, JavaScript, and Jython scripts only.

The script can have multiple statements. In this case the result of the entire script is the result of the last statement.

What MagicDraw functionality can a script use?



The following instructions applies to BeanShell, Groovy, JRuby, JavaScript, and Jython scripts only.

The script can call MagicDraw Open API.



For the list of available MagicDraw Open API methods, refer to *javadoc.zip*, which can be found in *<MagicDraw installation directory>\openapi\docs*.

For the instructions how to use these methods, see MagicDraw OpenAPI UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

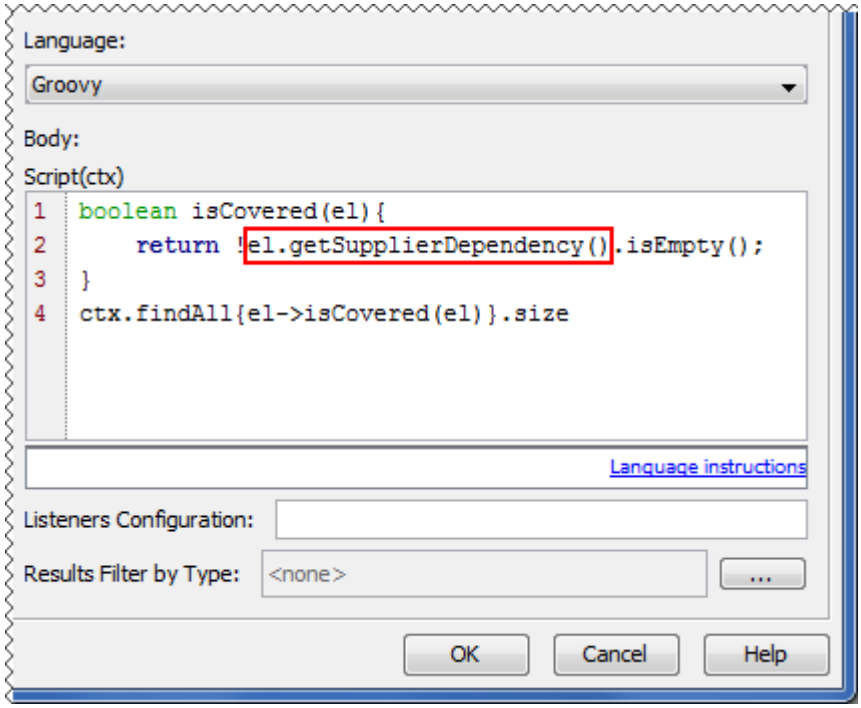


Figure 587 -- Using UML metamodel Open API to navigate between UML elements

More complex model access operations are available in *ModelHelper* and *StereotypesHelper*.

TIP!

Use *import* statements to shorten java class names as shown in the following figure.

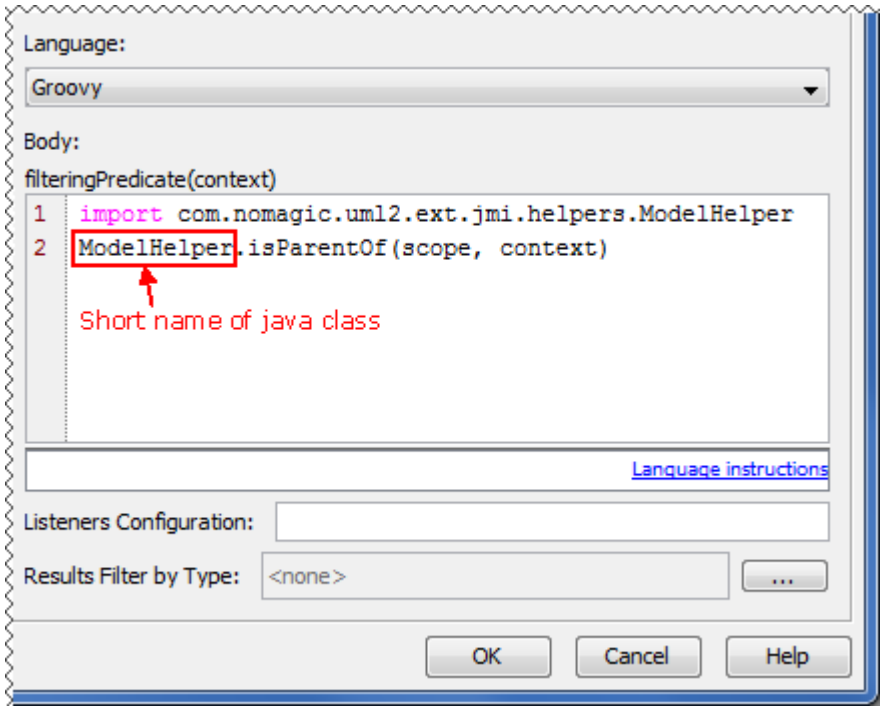


Figure 588 -- Using ModelHelper to define filtering condition script

Copyright © 1998-2015 No Magic, Inc.

Operation

Entries in the operation compartment are strings that show the operations defined on classes as well as those that are supplied by the classes. An operation is a service that can be requested to perform by an instance of the class. It has a name and a list of arguments.

Usually class attributes are accessed through the operations. The operations are used to perform specific actions, such as system calls, utility functions, and queries. The operation signature provides all information needed to use that operation.

You can specify operation properties in the operation Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To create a new operation

Do one of the following:

- Double-click the selected class or select **Specification** from the class shortcut menu. The **Class** Specification window opens.
- Click the **Operations** property group and then click the **Create** button. The **Operation** Specification window opens. Define a new operation and click **OK**.
- In the Containment tree, right-click a class and then from the shortcut menu, select **Create Element** > **Operation**.
- On the diagram, click the class shape, press the Ctrl+Alt+O shortcut keys and type the operation name.
- On the diagram, click the class shape and then click the *Create Operation* button.
- On the diagram, click the class shape, then click the *Create Element* button. From the menu, select the **Operation** command.

To change an operation name

1. Click the operation in the selected class on the diagram pane or in the Browser tree.
2. Type a new name.

- Change an operation name in the **Operation** Specification window.

To define the type of an operation

- In the **Operation** Specification window, the **Type** drop-down list, select the operation type.
- Type a colon ":" and the name of the operation type just after the operation name on the diagram pane. If you specify a nonexistent type of an operation, a new class is created.

To quickly create a behavior or a diagram for an operation

- Right-click the operation, point to **NEW! Create Method** > **Behavior** or **Diagram**, and then click the behavior or diagram you want to create.

To edit / add an operation parameter

1. Open the **Operation** Specification window.

Copyright © 1998-2015 No Magic, Inc.

2. Click the **Parameters** tab.
3. Double click on the existing parameter name in the expanded tree or click the **Create** button. The **Parameter** Specification window opens.

- Type a parameter text (in parenthesis) directly on a diagram.
- Select an operation in the Browser tree, select **New** from its shortcut menu and select **Parameter**. The **Parameter** Specification window opens.

The Parameter Specification window

The **Parameter** Specification window defines an operation argument.

You can specify parameter properties in the parameter Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To add additional information about the return type of an operation

1. Open the **Operation Specification** window.
2. In a **Properties** list, select **Expert** to switch the Expert mode. More properties for the operation appears.
3. In the **Type Modifier** list, click one of the following signs:
   - & - one class has a reference to other model element.
   - * - one class has a pointer to other model element.
   - [] - one class has an array of other model elements.

To define an operation as abstract, static, or query

1. Open the **Operation Specification** window.
2. In a **Properties** list, select **Expert** to switch the Expert mode. More properties for the operation appears.
3. In the **General** tab, select the **Is Abstract**, **Is Static**, and/or **Is Query** check box(es). The following table describes operation types.

Name	Description
Is Abstract	The operation does not have an implementation, and one must be supplied by a descendant.
Is Static	This operation scope means that the values returned by the parameter have no duplicates.
Is Query	The operation does not change the state of the system.

To set the operation visibility

1. Open the **Operation Specification** window.
2. In the **Visibility** list, select **Public**, **Package**, **Private**, or **Protected**. The following table describes visibility types.

Visibility name	Function
Public '+'	The operation can be accessed by any other object from the outside.

Copyright © 1998-2015 No Magic, Inc.

Visibility name	Function
Package '~'	The operation can be accessed by an element from the same package.
Private '-'	The operation can be accessed only from that class.
Protected '#'	The operation can be accessed from the inside of that class and the classes derived from that class.



The operation visibility is shown in the operation signature.

To set an operation Concurrency: sequential, guarded or concurrent

Name	Function
Sequential	The callers must coordinate, so that only one call to an Instance (on any sequential Operation) is made at a time. If simultaneous calls occur, then the semantics and the integrity of the system cannot be guaranteed.
Guarded	Multiple calls from concurrent threads may occur simultaneously to one Instance (on any guarded Operation), but only one is allowed to commence. The others are blocked until the performance of the first Operation is complete. It is the responsibility of the system designer to ensure that deadlocks do not occur due to simultaneous blocks. The Guarded Operations must perform correctly (or block themselves) in case a simultaneous sequential Operation or guarded semantics cannot be claimed.
Concurrent	Multiple calls from concurrent threads can occur simultaneously to one Instance (on any concurrent Operation). All of them can proceed concurrently with correct semantics. The Concurrent Operations must perform correctly in case a simultaneous sequential or the guarded Operation, or concurrent semantics cannot be claimed.

1. Open the **Operation** Specification window.
2. Select the **Expert** mode from the **Properties** field. More properties for the operation show.
3. Select the concurrency type in the **Concurrency** drop-down list.

Package

A package groups classes and other model elements together. All types of UML model elements can be organized into packages. Each diagram must be owned by one package and the packages themselves can be nested within other packages. Subsystems and models are special kinds of packages.

The packages can have dependency, generalization, realize, containment, and association relationships. These relationships are usually derived from the relationships between the classes inside those packages.

Copyright © 1998-2015 No Magic, Inc.

Working with packages

You can specify package properties in the package Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about working with symbols, see "Diagramming" on page 198.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To add inner elements to the selected package

1. Open the **Package** Specification window.
2. Click the **Inner Elements** tab.
3. Click the **Create** button, and then select an element you wish to add.
4. The selected element Specification window will open. If you selected a diagram, the **Diagram** Specification window will open.
5. Define properties you need and click **Back** to return to **Package** Specification window.

To change the package header (name, stereotypes, tagged values, and constraints) position

- From the package shortcut menu, select **Header Position:**
  - Select **Top** to place a package header at the top of a package shape.
  - Select **In Tab** to place a package header in a package tab.
- Open the **Symbol Properties** dialog:
  - Set the **Header Position** property to **Top** to place a package header at the top of a package shape.
  - Set the **Header Position** property to **In Tab** to place a package header in a package tab.

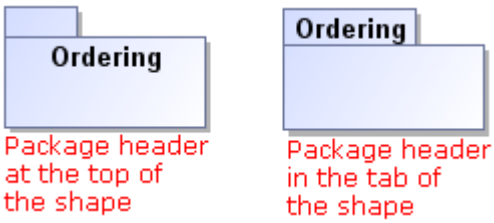


Figure 589 -- Position of a package header

To show the list of elements assigned to a package on the package shape

1. Open the package shortcut menu.

Copyright © 1998-2015 No Magic, Inc.

2. Select **Show Inner Elements List**.

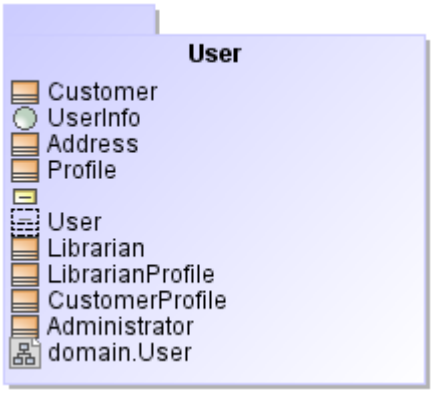


Figure 590 -- Package inner elements displayed in a package shape

To display inner elements of the package in a diagram:

1. Right-click a package in a diagram to open the package shortcut menu.
2. Select **Related Elements** > **Display Inner Elements**. The **Select Inner Elements** dialog will open).

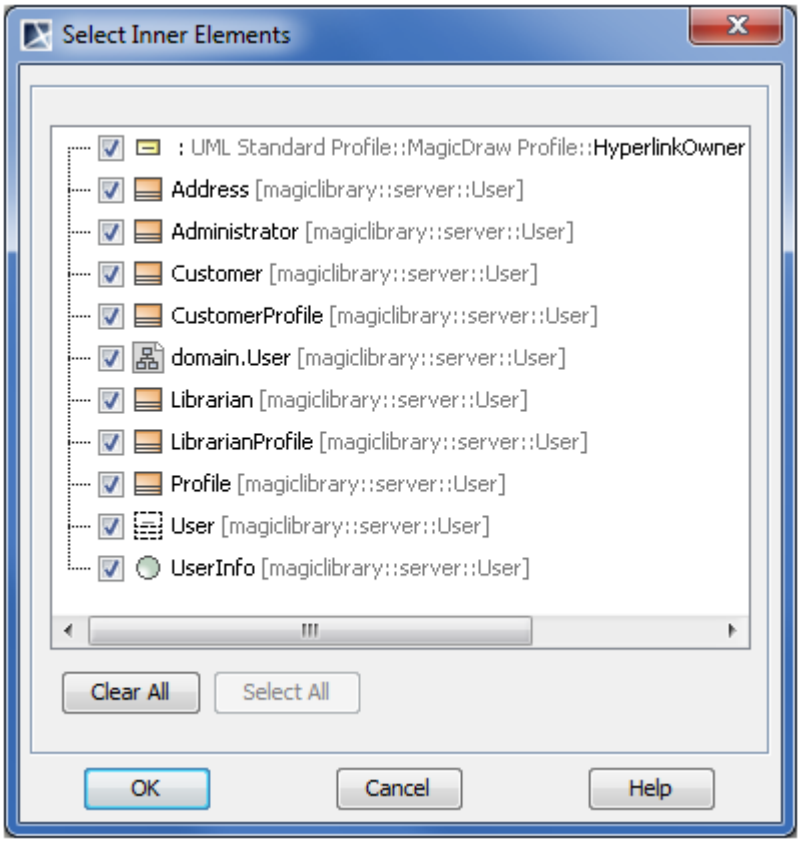


Figure 591 -- The Select Inner Elements dialog

Copyright © 1998-2015 No Magic, Inc.

3. Select the elements to be displayed and click **OK**. The selected elements will be displayed in the diagram's package.

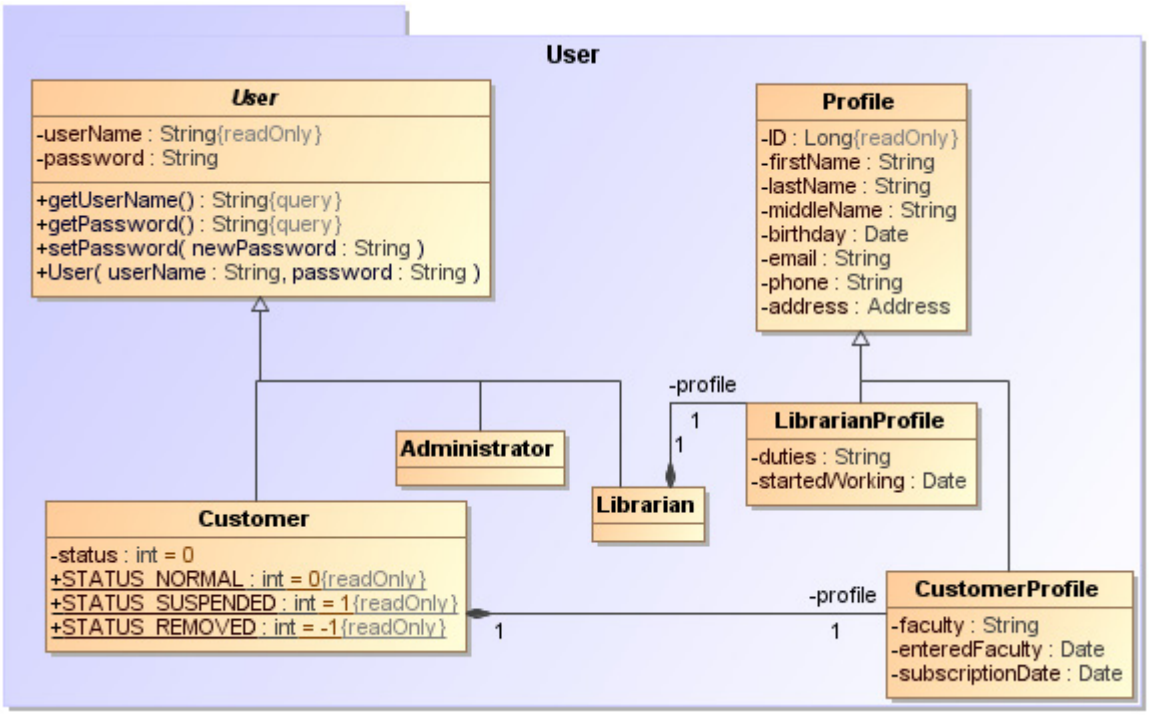


Figure 592 -- Package with inner elements displayed

NOTE

The rake icon is displayed on the shape if the package refers to another package diagrams. The rake icon isn't shown by default therefore you need to set the **Show Rake Icon** property value to *true* in the **Property** dialog. For more information see "NEW! Displaying rake icon on symbol" on page 322.

Part

To create a part

Do one of the following:

- In the Composite Structure diagram pallet, click the Part button and then click on the diagram.
- Drag a part type (for example, a class) to the composite structure diagram and drop it.

To create a new composite structure diagram for the type of the part

Do one of the following:

- Click the part on a diagram, then on the smart manipulator toolbar, click the *New Composite Structure Diagram* button. The new diagram opens in the same diagram tab.

TIP!

- A new diagram is created under the part type.
- If type was not assigned for the selected part then a new type is created automatically.
- Name of the part type is used for created diagram.

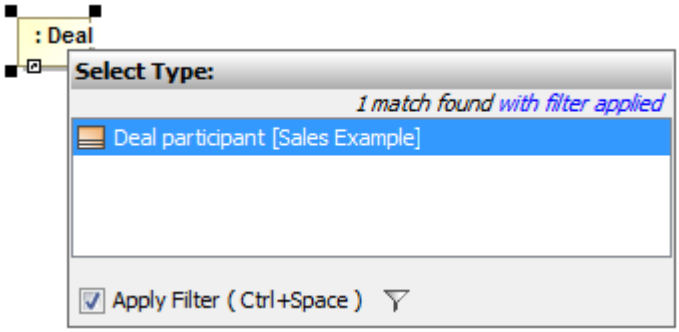
Copyright © 1998-2015 No Magic, Inc.

- From the part shape's shortcut menu, select **Create Diagram** and then diagram from the list. This accelerates the creation of composite structure diagram (the **Create Diagram** menu item is available only if the type is specified for the part).

To define a type of a part

Do one of the following:

- In the Composite Structure diagram pallet, click the Part button and then click on the diagram. Then, in the **Select Type** dialog, type the first letter of the part type and then select the part type from the list. Press **Enter**.



NOTE

The **Select Type** dialog is not displayed if the *Type Selection* Mode is disabled. For more information, see "Mapping ports on a part type change" on page 721.

- Select a part shape on the diagram pane and press Ctrl+T on Windows OS and Linux OS or Cmd+T on OS X. Then, in the **Select Type** dialog, select a type and press **Enter**.
- Select a part shape on the diagram pane and on the smart manipulator toolbar, click the *Specify Type* button. Then, in the **Select Type** dialog, select a type and press **Enter**.

NOTE

If a type is specified already for the selected part, then the *Change Type* button is displayed in the part's smart manipulator toolbar.

- Select a part shape on the diagram and then type the name of the type. The type will be created with the specified name.

NOTE

This step is not valid, if the *Type Selection* Mode is disabled. In this case to specify a type on a part shape, type a colon and then the name. For example, "*: Deal participant*". For more information, see "Mapping ports on a part type change" on page 721.

- In the part Specification window, do one of the following:
  - Click the **Type** property and then select a type of a port in the list.
  - Click the **Type** property, then click the button and select a type in the element Selection dialog.

TIP!

When creating a port on a part, if a part does not have a specified type, a new type for the part is created automatically.

NOTE

The rake icon is displayed on the shape if the part refers to another composite structure diagram. The rake icon isn't shown by default therefore you need to set the **Show Rake Icon** property value to *true* in the **Property** dialog. For more information see "NEW! Displaying rake icon on symbol" on page 322.

Related diagrams

Composite Structure Diagram

Copyright © 1998-2015 No Magic, Inc.

Pin

A pin represents a connection point for input and output values of an action. There is one input pin for each input parameter and one output pin for each output parameter.

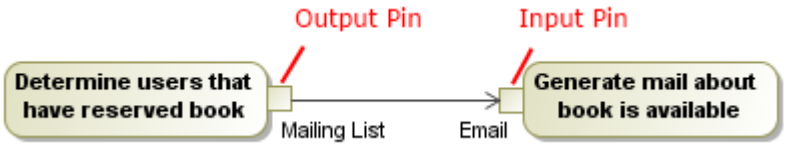


Figure 593 -- Example of output and input pins

Using MagicDraw you can create the following types of pins:

- Input Pin. An input pin is a pin that holds input values to be consumed by an action.
- Output Pin
- Value Pin
- Action Input Pin

TIP!

- When a new action is created that must have mandatory pins, pins are automatically added to that action on the diagram pane as well as in the Model Browser.
- As of MagicDraw 17.0.4 version, the *Pins Displaying* validation suite has been introduced. It includes the *Action Containing Hidden Pins* validation rule that detects all hidden pins for the concrete action symbol. For more information about validation, see "Validation" on page 612.
- When deleting the last pin symbol from the diagram, the pin element is deleted from the project. This is valid when the **Display All Pins in Diagrams** option value is *true*. For more information, see "To display all pins on diagrams" on page 964.
- For a send signal action a pin is created, but the symbol of the pin is not represented on the diagram pane.
- You can navigate to the associated parameter from the pin shortcut menu. Just right-click the pin, click **Go To** and then select a parameter.

You can format the pin symbol properties in the **Symbol Properties** dialog.

You can specify pin properties in the pin Specification window. In the specification window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about symbol representation properties, see "Formatting Symbols" on page 317.
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To assign pins for an action, do one of the following

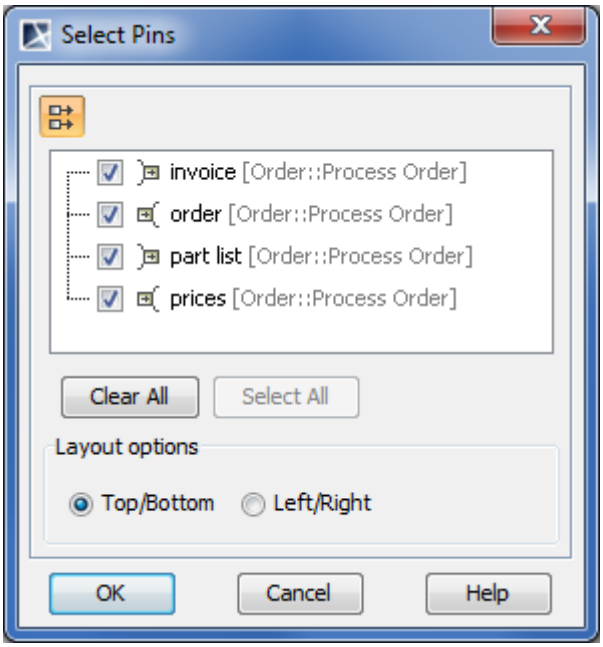
- Create a pin directly on the diagram pane:
  1. Draw an action symbol on the activity diagram pane.
  2. In the activity diagram pallet, select a pin you want to create. Click the **Input Pin** button or click the little arrow next to the **Input Pin** button to expand the menu and to select Output Pin, Value Pin, or Action Input Pin.
  3. On the diagram pane, click the action. The pin for this action is created.
- Create a pin in the action Specification window:

Copyright © 1998-2015 No Magic, Inc.

1. Open the action Specification window.
2. Select the **Pins** property group.
3. To create an input pin, in the **Input Pin** category click the **Argument** property specification cell. Or to create an output pin, in the **Output Pin** category click the **Result** property specification cell.
4. Click the Add button. The list of available to create pins opens.
5. Select a pin you want to create. The pin Specification window opens.
6. Type the pin name.
7. Click **Close** to close the Specification window or click **Back** to return to the action Specification window.

To display Pins on shapes

1. On the diagram pane, select a shape whereon you want to display pins.
2. Open its shortcut menu and click **Related Elements** > **Display Pins**. The **Select Pins** dialog opens.



3. In the dialog, select pins you want to display on the shape.
4. In the **Layout options** area, select whether on **Top/Bottom** or **Left/Right** shape borders pins should be displayed.

NOTE

If the **Top/Bottom** layout option is selected, an input pin resides on the top of the action border, an output pin on the bottom of the border. If the **Left/Right** layout option is selected, an input pin resides on the left of the action border, an output pin on the right of the border.

5. Click **OK** when you are done.

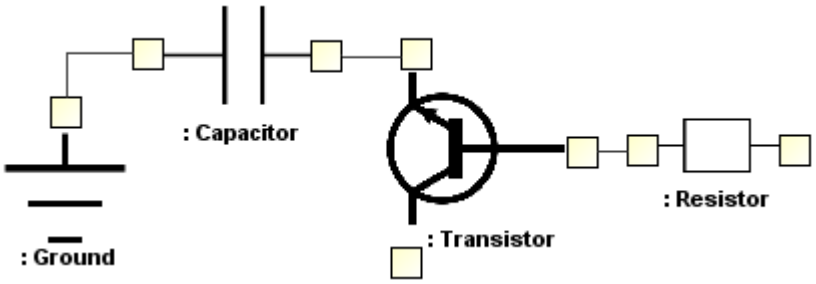
NOTE

To change pin displaying options, delete pin symbols from the shape and repeat the preceding procedure.

Copyright © 1998-2015 No Magic, Inc.

TIP!

If you have an element displayed as image, you can connect a pin to that element with no gaps.



To display all pins on diagrams

1. In the **Project Options** dialog, click the **General project options** group.
2. Under the **Diagramming** category, click to select the **Display All Pins in Diagrams** check box.

To show the pin's type

Do one of the following:

- **NEW!** Right-click the pin's shape and on the shortcut menu click **Show Type**.
- Open the **Symbol Properties** dialog of the pin's shape and set the **Show Type** property to *true*.

To convert an input pin to an output pin or conversely

1. Select a pin on the diagram or in the Model Browser.
2. From the element shortcut menu, select the **Refactor** command and then **Convert To**.
3. From the opened list, select to convert the pin to the input pin, output pin, action input pin, or value pin.

To convert a pin to an object node

1. Draw an input or output pin, on an action:
2. On the diagram pane, from the input or output pin shortcut menu, select the **Refactor** command and then **Convert to Object**. The pin is converted to an object node.



Figure 594 -- Example of Output Pin (on the left) conversion to Object Node (on the right)

TIP!

For more information about the converting functionality, see "Converting Elements" on page 360.

To navigate from a pin to the associated parameter

1. Select a pin on a diagram.
2. Right-click the pin, click **Go To** and then select a parameter.

Related diagrams

Activity Diagram

Copyright © 1998-2015 No Magic, Inc.

Related procedures

Formatting Symbols

Related dialogs

Specification Window

Port

A port is a property of a classifier that specifies a distinct interaction point between that classifier and its environment, or between the (behavior of the) classifier and its internal parts. Ports are connected to the properties of the classifier by connectors through which requests can be made to invoke the behavioral features of the classifier.

A Port can specify the services a classifier provides (offers) to its environment as well as the services that a classifier expects (requires) from its environment. It has the ability to specify that any requests arriving at this port are handled.

The Class model element and Component model elements can have any number of Ports.

You can specify port properties in the port Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

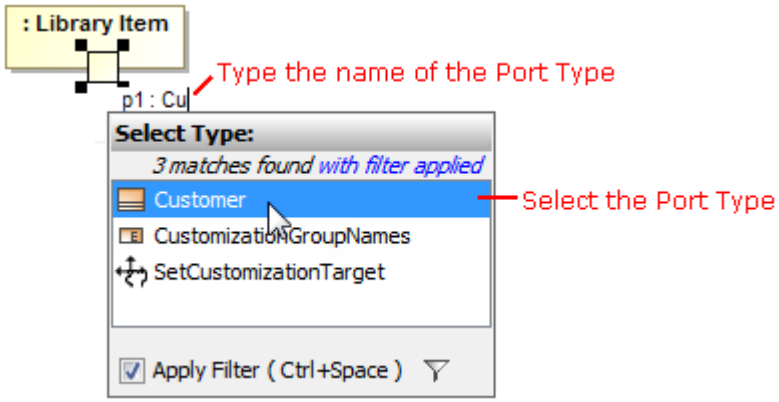


- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To create a port

Do one of the following:

- Create a port by using diagram pallet:
  1. On the Class, Component, or Composite Structure diagram pane draw a shape on which you want to create a port.
  2. On the diagram pallet, click the *Port* button.
  3. On the diagram pane, click the shape on which you want to add a port. The **Select Type** dialog appears.





The **Select Type** dialog is not displayed if the *Type Selection* Mode is disabled.

Copyright © 1998-2015 No Magic, Inc.

4. Type the first letter of the port type and select the port type from the list. Then press **Enter**. The port is created.

NOTE The port with required interface resides on top of the shape border, and the port with provided interface resides on bottom of the shape border.

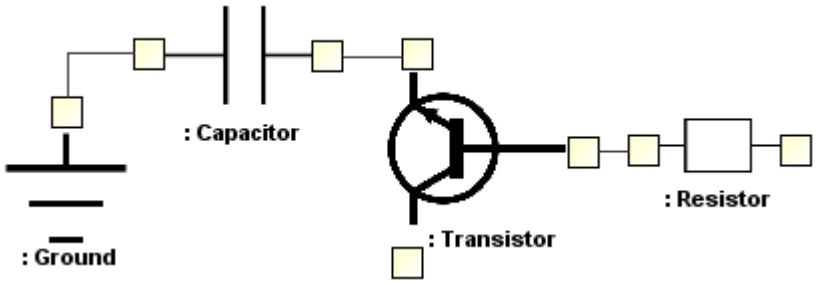
- Create a port by using smart manipulator toolbar:
  1. In the Class, Component, or Composite Structure diagram draw a shape on which you want to create a port.
  2. Select the shape and then click the *Port* button on the smart manipulator toolbar. The **Select Type** dialog appears.

NOTE The **Select Type** dialog is not displayed if the *Type Selection* Mode is disabled.

  3. Type the first letter of the port type and select the port type from the list. Then, press **Enter**. The port is created.
- Create a port in the element Specification window:
  1. Open the element Specification window for which you want to create a port.
  2. Select the **Ports** property group.
  3. Click the **Create** button. The **Port** Specification window opens.
  4. Specify the port properties and click **Close**. The port is created and assigned to the element.

TIP!

- For more information on ports creation on parts when drawing a connector, see "To select a port automatically when drawing a connector from a port to a part" on page 864.
- If you have an element displayed as image, you can connect a port to that element with no gaps:



To define a port's type

Do one of the following:

- Select a port shape on the diagram and press Ctrl+T on Windows OS and Linux OS or Cmd+T on OS X. Then, type the first letter of the port type and select the type from the list.
- Select a port shape on the diagram pane and click the *Specify Type* button on the smart manipulator toolbar. Then, type the first letter of the port type and select the type from the list.

NOTE If a type is set for the selected port then the *Change Type* button is in the port's smart manipulator toolbar.

- In the port Specification window, do one of the following:
  - Click the **Type** property and then select a type of a port.
  - Click the **Type** property, then click the button and select a type in the element Selection dialog.

Copyright © 1998-2015 No Magic, Inc.

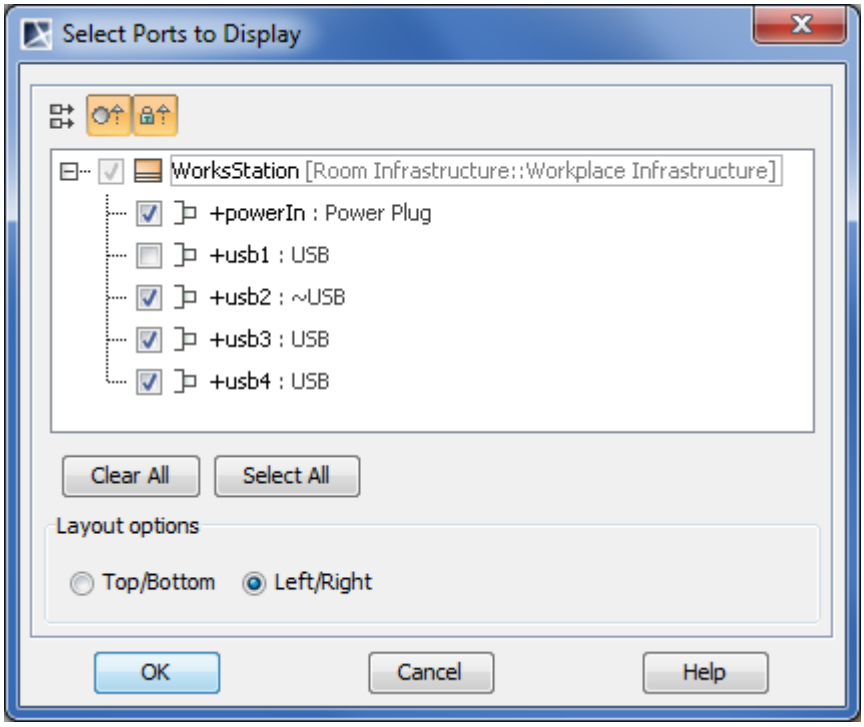
To show the port's type

Do one of the following:

- **NEW!** Right-click the port's shape and on the shortcut menu click **Show Type**.
- Open the **Symbol Properties** dialog of the port's shape and set the **Show Type** property to *true*.

To display ports on shapes

1. On the diagram pane, select a shape whereon you want to display ports.
2. Open its shortcut menu and click **Related Elements** > **Display Ports**. The **Select Ports to Display** dialog opens.



3. In the dialog, select ports you want to display on the shape.
4. In the **Layout options** area, select whether on **Top/Bottom** or **Left/Right** shape borders ports should be displayed.



If the **Left/Right** layout option is selected, a port with a required interface is displayed on the left of the shape border and a port with a provided interface is displayed on the right of the shape border.

If the **Top/Bottom** layout option is selected, on the top of the shape border a port with a required interface is displayed, and on the bottom of the shape border is displayed a port with a provided interface.

Copyright © 1998-2015 No Magic, Inc.

5. Click **OK** when you are done.

NOTE

- To change port displaying options, delete port symbols from the shape and repeat the preceding procedure.
- On the diagram pane, near the port symbol, the port type is displayed if port has referenced type, but do not have name.
- On the diagram pane, near the port symbol, the port type is not displayed if type is provided or required interface and the interface is displayed on the diagram.
- When a part symbol is created (also when dropping classifiers on composite structure diagrams), all its ports are displayed on the diagram.

To create quickly a port on a composite structure diagram frame

TIP!

You can draw a connector straight to or from a composite structure diagram frame. As a result a port will be created and the connector will be connected to it.

1. In a composite structure diagram, click a part or a port shape.
2. On the shape's smart manipulator toolbar, click the *Connector* button.
3. Click the composite structure diagram frame. The port is created and the *Select Port* menu appears.

NOTE

The menu is displayed only if the composite structure diagram has hidden ports.

4. Click one of the following commands:
   - A hidden port or a nested port of the composite structure diagram. The chosen port will be displayed on the diagram frame.
   - **New Port** - click to create a port on a diagram frame, with the same name, type and multiplicity, as is set on the source part or port.
   - **Nested Port** - click to create a new nested port.

NOTE

The **Nested Port** command exists only when creating a connector from the nested port to the diagram frame.

TIP!

- You can also create quickly a port on the composite structure diagram frame when drawing a connector straight from a composite structure diagram frame.
- If a connector is created from a port with the defined type, then only compatible hidden ports are listed in the *Select Port* menu.
- If a connector is created from a port that has interfaces, then ports that have compatible interfaces are listed in the *Select Port* menu.

Copyright © 1998-2015 No Magic, Inc.

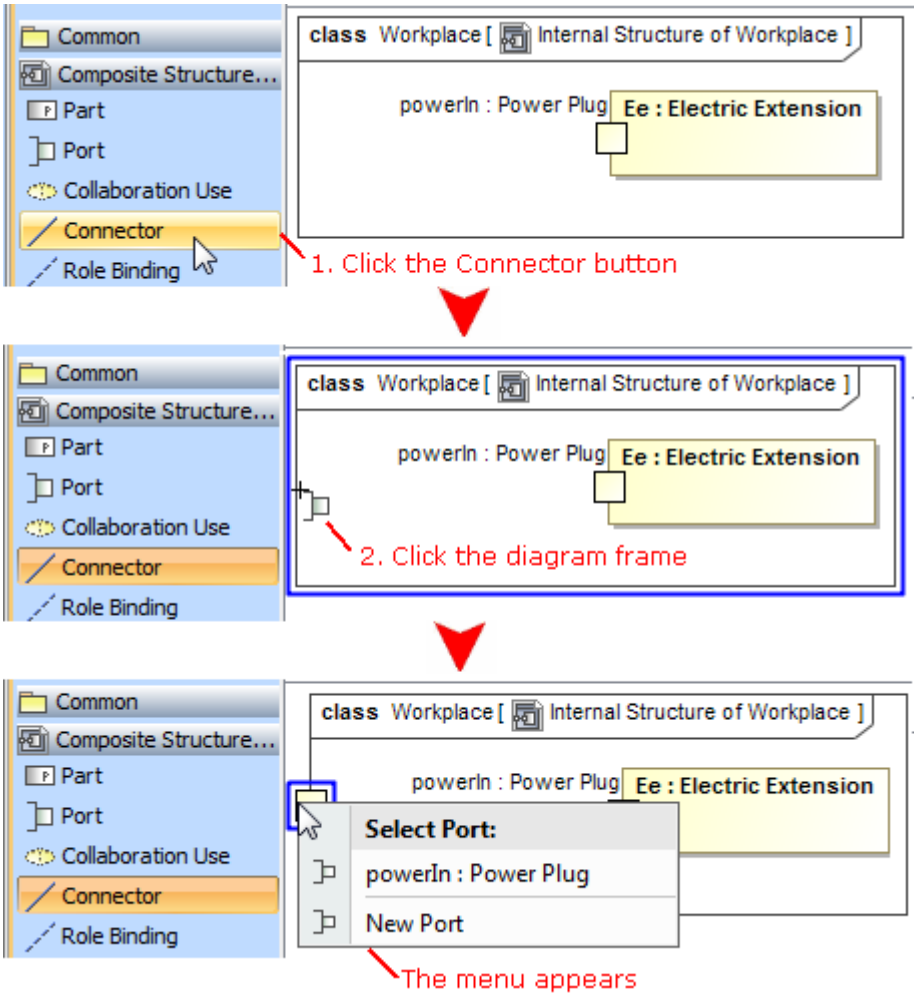


Figure 595 -- Example of Port creation on Composite Structure diagram frame

Copyright © 1998-2015 No Magic, Inc.

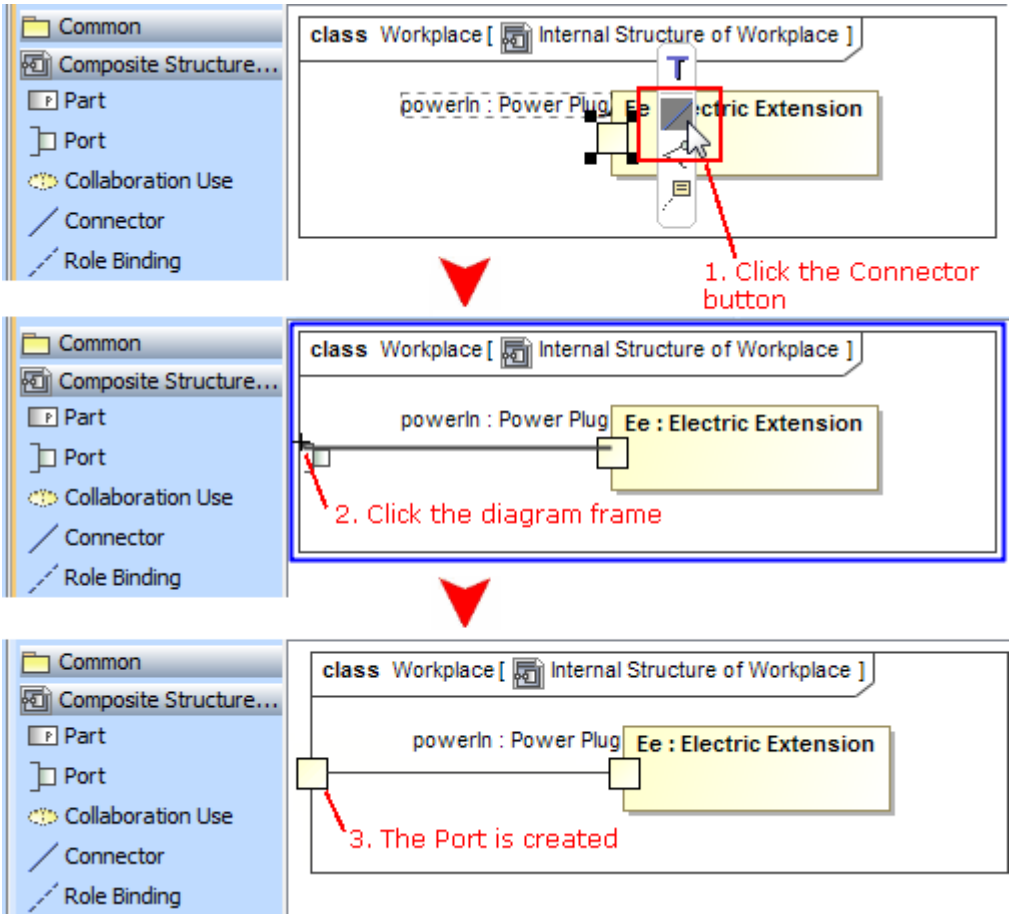


Figure 596 -- Example of drawing Connector to Composite Structure diagram frame

To display ports on shape and to customize ports list

1. On the diagram, click a symbol, then click the *Compartments* button. On the menu, click to select the **Ports** check box. The ports compartment is displayed on the shape.
2. On the diagram, click the shape, then click the *Compartments* button. On the menu, click the **Edit Compartments** command.
3. In the **Compartment Edit** dialog, click the **Ports** tab.
4. Move ports for *Hidden* to the *Selected* list or vice versa.

To change labels position on port

1. Select a port on a diagram.
2. Right-click the port, click **Symbol Properties** and then change the **Position of Labels** property value.

Copyright © 1998-2015 No Magic, Inc.

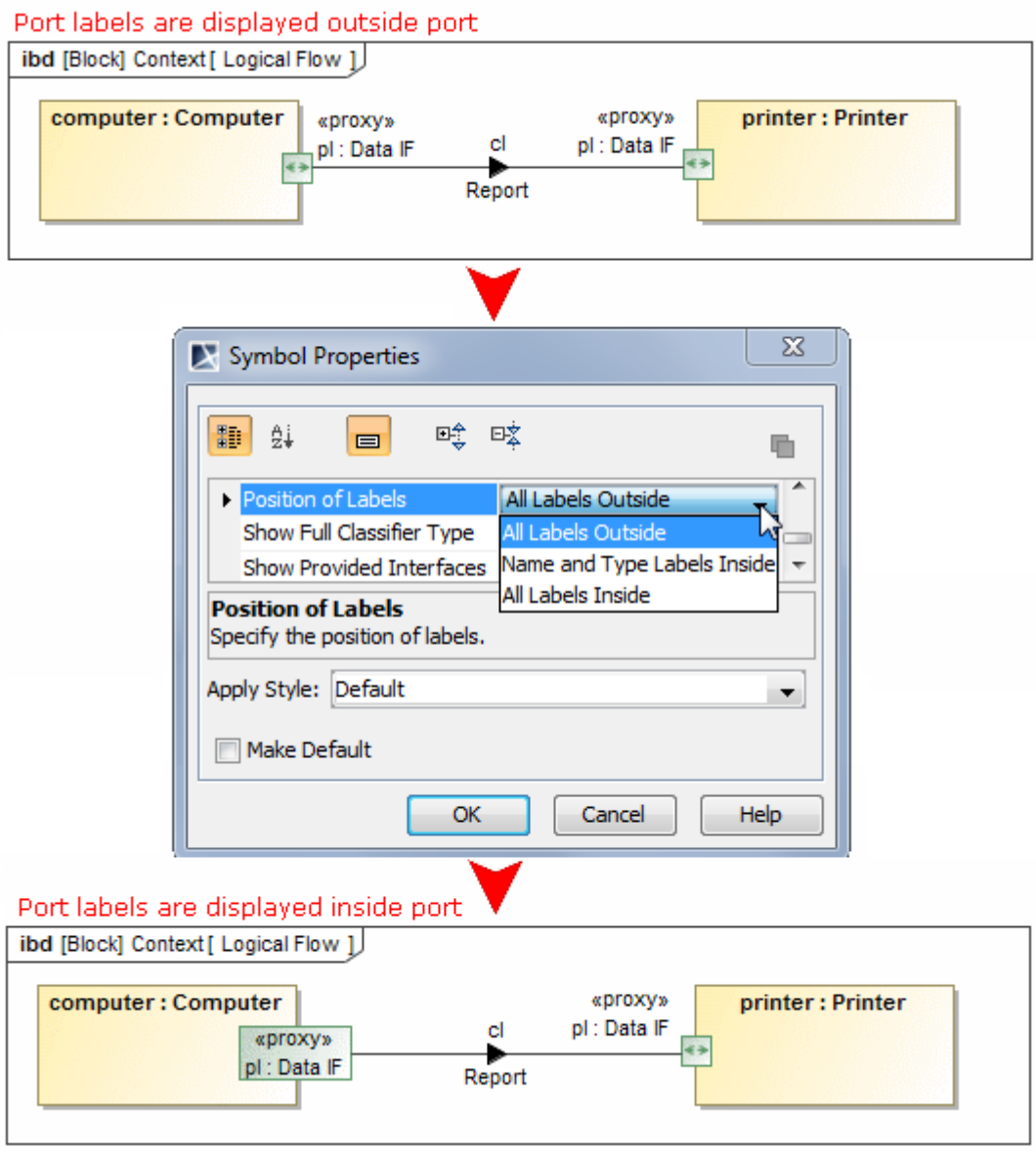


Figure 597 -- Example of how to change labels position on port

To specify the Provided/Required Interfaces for a Port even if the Port type is not specified

1. Open the **Port** Specification window.
2. Click the **Provided/Required Interfaces** property group.
3. Click the **Add** button and then select to add provided or required interface.
4. If the port type is not defined, the **Select Port Type** dialog appears. Select one of the following options and then click **OK**:
    - **Set provided interface as port type** (available on provided interface creation only). The provided interface will be specified as the port type.
    - **Create "dummy" port type automatically**. Create a “dummy” port type and relations between the type and interface.
    - **Select or create a port type manually**. The **Select Port Type** dialog will open, to allow you to select or create a Port.

Copyright © 1998-2015 No Magic, Inc.

5. In the element Selection dialog, select or create an interface and click **OK**. The provided or required interface is created.



Draw a realize relationship from a port to an interface to create a provided interface. Draw a usage relationship from a port to an interface to create a required interface.

To convert a Port to an Association

- Right-click a port(s), and on the shortcut menu, click **Refactor**, **Convert To**, **Association**.
- On the diagram, on the shape, in the ports compartment, select a port, and drag-and-drop it anywhere on the diagram.



You can convert a port to an association, only if the port type is specified and if a port is not a member end of some association.

NEW! To display a Port name in a signature on a shape

1. Right-click the shape on the diagram, and then click **Symbol Properties**.
2. In the **Symbol Properties** dialog, click to select the **Show Port** property.

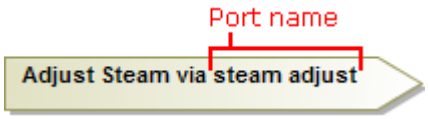


If you cannot find the **Show Port** property in the **Symbol Properties** dialog, change the properties display mode to *Expert* or *All*. For more information about properties display mode, refer to "Symbol Properties dialog" on page 317.

You can display the Port name on the following shapes:

- Call Operation Action, Call Behavior Action, Send Signal Action, Send Object Action, Broadcast Signal Action, and Start Object Behavior Action.

For example: *via <Port name>*.



- Transition, Protocol Transition, State, and Accept Event Action.

For example: «*from*» *<Port name>.*

Related diagrams

Class Diagram
Component Diagram
Composite Structure Diagram

Profile

A profile is a kind of a package that extends a reference metamodel. The primary extension construct is a stereotype. Stereotypes are defined as a part of profiles.

A profile introduces several constraints or restrictions to ordinary metamodeling. Constraints and restrictions are realized using metaclasses defined in the package. It is a restricted form of a metamodel that always must be related to a reference metamodel, such as UML, as it is described below. It cannot be used without its reference metamodel, and it defines a limited capability to extend metaclasses of the reference metamodel. The extensions are defined as stereotypes that apply to the existing metaclasses.

Copyright © 1998-2015 No Magic, Inc.

Each profile contains a set of stereotypes. Profiles are defined as separate modules. Profiles are loaded on demand, that is, when you start or open your project only profiles used in that project are loaded.

Profiles are defined using the UML extensibility mechanisms that allow modelers to customize UML for specific domains, for example, for software development processes. In MagicDraw, the mechanism of the profile is similar to the functionality of modules.

MagicDraw comes with a number of predefined profiles: UML Standard Profile, DDL, EDOC, and other. All MagicDraw profiles are stored in *<MagicDraw installation directory>\profiles*.

Profile properties

The profile is defined as a package, that is, it has package properties. For the detailed description of packages, see "Package" on page 957.

You can specify profile properties in the profile Specification window. You can also find descriptions of each property in this window. Descriptions are provided in the description area below the property list.



- For more information about using the Specification window, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Creating profiles

You can create a profile in one of the following ways:

- Using the shortcut menu of a package, model, or other profile.
- Using the profile diagram pallet.
- Using the package diagram pallet.
- Using the class diagram pallet.

To create a profile using the package / model / profile shortcut menu

1. In the Containment tree, select a package, a model, or another profile, wherein you want to create the new profile.
2. From the shortcut menu of the selected package, model, or profile, select **Create Element** > **Profile**.

To create a profile using the profile diagram pallet

1. Either create a new profile diagram or open an existing one.
2. On the diagram pallet, click the **Profile** button.
3. Click a free space of the diagram pane.

To create a profile using the package diagram pallet

1. Either create a new package diagram or open an existing one.
2. On the diagram pallet, click the **Package** arrow to see more buttons.
3. Click the **Profile** button.
4. Click a free space of the diagram pane.

To create a profile using the class diagram pallet

1. Either create a new class diagram or open an existing one.

Copyright © 1998-2015 No Magic, Inc.

2. On the class diagram pallet, do one of the following:
   - Expand the **Profile Diagram** button group and click the **Profile** button.
   - Expand the **Package Diagram** button group, click the **Package** arrow to see more buttons, and then click the **Profile** button.
3. Click a free space of the diagram pane.

Pseudo State

The Pseudo state is typically used to connect multiple transitions into more complex state transitions paths. For example, by combining a transition entering a fork pseudo state with a set of transitions exiting the fork pseudo state, we get a compound transition that leads to a set of orthogonal target states.

You can specify pseudo state properties in the pseudo state Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about working with symbols, see "Diagramming" on page 198.
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Initial

Every object belongs to a particular state as soon as it is created. So, it is useful to explicitly show that particular state. A solid filled circle represents the initial state of an object. There can only be one initial state for an object. The initial state denotes the starting place for a transition, the target of which is a composite state.

Final state

The final state symbol (a circle surrounding a smaller solid circle) is used to represent the object destruction. The final state is optional in the diagram because there is a system that runs without interruption after the start of the activities. Also, there can be several final states in the same state diagram, denoting that the life of the object can finish depending on several conditions.

Terminate

Entering a terminate pseudo state implies that the execution of the state machine by means of its context object is terminated. The state machine does not exit any states nor does it perform any exit actions other than those associated with the transition leading to the terminate pseudo state.

Entry Point

An entry point connection point reference as the target of a transition implies that the target of the transition is the entry point pseudo state as defined in the submachine of the submachine state. As a result, the regions of the submachine state machine are entered at the corresponding entry point pseudo states.

Copyright © 1998-2015 No Magic, Inc.

Exit Point

An exit point connection point reference as the source of a transition implies that the source of the transition is the exit point pseudo state as defined in the submachine of the submachine state that has the exit point connection point defined. When a region of the submachine state machine has reached the corresponding exit points, the submachine state exits at this exit point.

Deep History

The Deep History represents the most recent active configuration of the composite state that directly contains this pseudo state; e.g. the state configuration that was active when the composite state was last exited. A composite state can have at most one deep history vertex.

Shallow History

The Shallow History represents the most recent active substate of its containing state (but not the substates of that substate). A composite state can have at most one shallow history vertex. A transition coming to the shallow history vertex is equivalent to a transition coming to the most recent active substate of a state.

Junction

The junction vertices are semantic-free vertices that are used to chain multiple transitions together. They are used to construct the compound transition paths between states. For example, a junction can be used to combine multiple incoming transitions into a single outgoing transition representing a shared transition path (this is known as merge). Conversely, it can be used to split an incoming transition into multiple outgoing transition segments with different guard conditions.

Choice

The choice vertices, when reached, result in the dynamic evaluation of the guards or the triggers of its outgoing transitions. This realizes a dynamic conditional branch. It allows splitting of transitions into multiple outgoing paths such that the decision on which path to take may be a function of the results of prior actions performed in the same run-to-completion step.

Fork and Join

The fork vertices are used to split an incoming transition into two or more transitions terminating on the orthogonal target vertices (i.e., vertices in different regions of composite state). The segments going out of a fork vertex must not have guards or triggers.

The join vertices are used to merge several transitions emanating from the source vertices in different orthogonal regions. The transitions entering a join vertex cannot have guards or triggers.

Realization

The realization is a specialized abstraction relationship between two sets of model elements, one represents a specification (the supplier) and the other represents an implementation of the latter (the client). The realization can be used to model stepwise refinement, optimizations, transformations, templates, model synthesis, framework composition, etc.

Copyright © 1998-2015 No Magic, Inc.

The realization relationship is drawn as a dashed line with a solid triangular arrowhead (a “dashed generalization symbol”). The client (the one at the tail of the arrow) supports at least all of the operations defined in the supplier (the one at the arrowhead), but not necessarily the data structure of the supplier (attributes and associations).



For more information about working with symbols, see "Diagramming" on page 198.

The realization paths can be grouped in a tree. This feature makes the appearance of the diagram more structural and understandable.



In MagicDraw, you will find three kinds of a realization relationship:

- **Interface Realization**. A dashed line with a solid triangular arrowhead. An Interface Realization is a specialized Realization relationship between a Classifier and an Interface. This relationship signifies that the realizing classifier conforms to the contract specified by the Interface.
- **Realization**. A solid line that represents a relationship between a classifier and an interface.
- **Substitution.** A dashed line with an arrowhead and «*substitute*» stereotype. A substitution is a relationship between two classifiers. It signifies that the substituting classifier complies with the contract specified by the contract classifier. This implies that instances of the substituting classifier are runtime substitutable where instances of the contract classifier are expected.

To create a realization tree if a class or an interface already has a number of realization paths attached to it

- Select the **Refactor** > **Make Sub Tree** command from the class or the interface shortcut menu.

To remove a realization from the tree or to ungroup a tree

- Select the realization and select the **Remove From Tree** command from the path shortcut menu.
- Select a tree head and select the **Ungroup Tree** command from the tree shortcut menu.

To specify the selected realization path in the Specification window

- Double-click the path.
- Select **Specification** from the path shortcut menu.
- Select the path and press ENTER.

Realization and its kinds Specification windows

You can specify realization properties in the realization Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

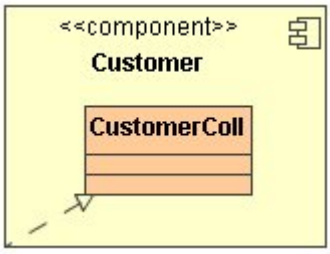
Creating the realizing classifiers

The realizing classifiers are a set of Realizations owned by the Component. The Realizations reference the Classifiers of which the Component is an abstraction (i.e., that realize its behavior).

To create a Realization relationship between a component and a classifier:

Copyright © 1998-2015 No Magic, Inc.

1. Drag the classifier shape to the component shape.
2. Select a classifier or component and select **Related Elements** from its shortcut menu, then select the **Display Paths** command. The realization relationship will be displayed on the diagram pane.



Reception

Signal receptions can be specified for classes or interfaces.

You can specify reception properties in the reception Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To create a new reception

Do one of the following:

- In the **Class** Specification window, click the **Signal Receptions** property group and then click the **Create** button. Select the signal and click **OK**. The **Signal Reception** Specification window opens.
- In the Containment tree, right-click a class and from the shortcut menu, select **Create Element** and then **Signal Reception**. Select the signal and click **OK**. Type the name for the new signal reception.
- On the diagram, click the class, and then, click the *Create Element* smart manipulator. At the bottom of the menu, click the arrow to expand the menu and then click the **Signal Reception** command. Select the signal and click **OK**.
- On the diagram, click the class, press the Ctrl+Alt+R shortcut keys. Then, select the signal and click **OK**.
- On the diagram, click the class shape, and then, click the *Create Signal Reception* smart manipulator. Note that the signal reception compartment is suppressed by default.
- Drag the signal from the Containment tree or from the diagram to the class shape on the diagram.

To set the signal for a signal reception

Do one of the following:

- Create a signal reception. The **Select Signal** dialog opens. Select an existing or create a new signal for the signal reception.

Copyright © 1998-2015 No Magic, Inc.

- In the **Signal Reception** Specification window, in the **Signal** drop down menu, select a signal. You can also click the "..." button. In the **Select Element** dialog, select a signal or click the **Create** button to create a new one.

To display signal reception on shape and to customize displayed list

1. On the diagram, click a symbol, then click the *Compartments* button. On the menu, click to select the **Signal Receptions** check box. The signal receptions compartment is displayed on the shape.
2. On the diagram, click a shape, then again click the *Compartments* button. On the menu, click the **Edit Compartments** command.
3. In the **Compartment Edit** dialog, click the **Signal Receptions** tab.
4. Move signal receptions for *Hidden* to the *Selected* list or vice versa.

To change the order of the signal receptions

- On the diagram, right-click the shape and then from the shortcut menu, select the **Symbol Properties** command. In the **Symbol Properties** dialog, under the Signal Receptions group, click to change the **Signal Receptions Sort Mode** property.

NEW! Report Template

This is a custom element used to store a report template file (.mrzip) attached to the project.

In contrast to the report templates stored in the local report templates folder (*<MagicDraw configuration files directory>\data\reports*), a report template attached to the project as this specific type of model element is not lost after opening the project on another computer. It is safely transfered within the project (either local or server).



For more information about configuration files location on different operating systems, refer to "MagicDraw Configuration Files Location" on page 61.

To attach a report template to the project

1. From the main menu, select **Tools** > **Report Wizard**. The **Report Wizard** dialog opens.
2. Select the report template you want to attach to the project and click the **Attach** button.

Copyright © 1998-2015 No Magic, Inc.

The selected report template is duplicated, and the duplicate is attached to the open project – you can see it in the Containment tree directly under the root package. As the following figure shows, in the **Report Wizard** dialog it as marked with the "[Attached to Project]" tag.

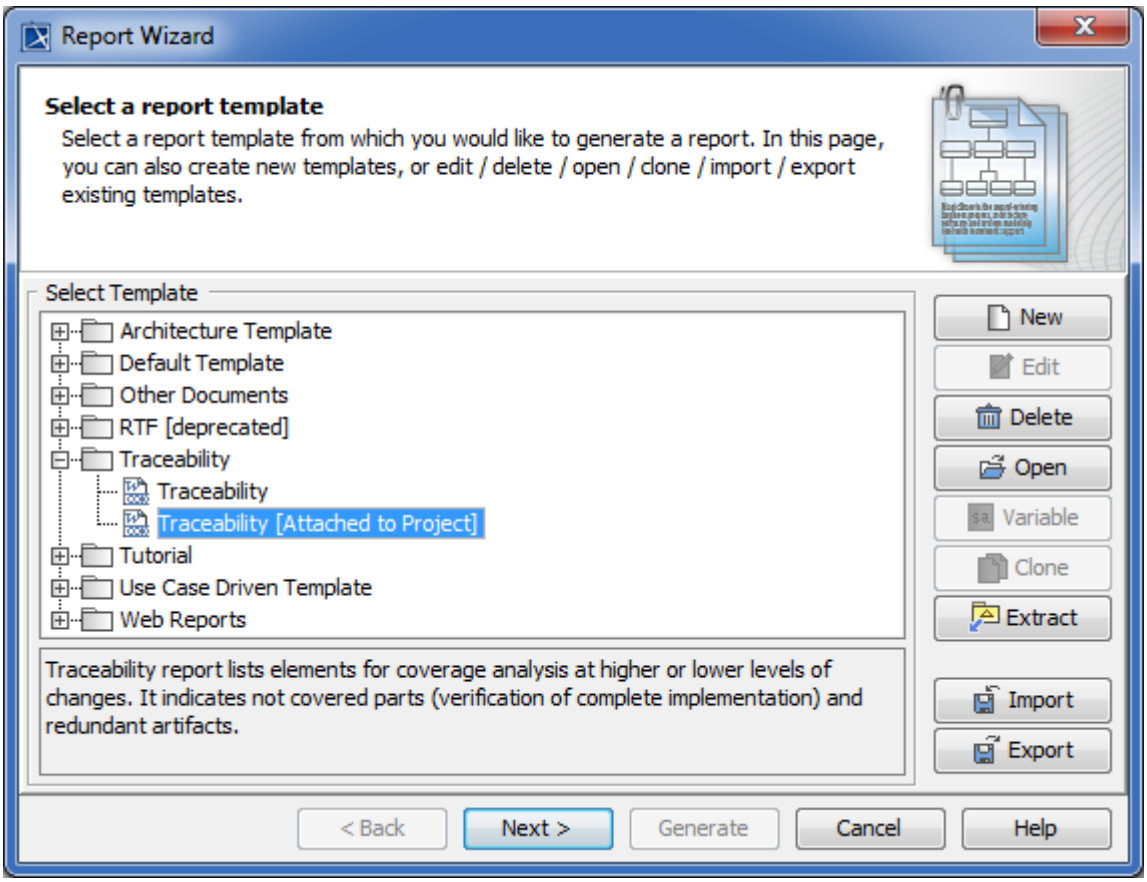


Figure 598 -- Attached to project report template in Report Wizard dialog

You can start generating a report from the attached report template in one of the following ways:

- Right-click the attached report template in the Containment tree and from the shortcut menu select **Generate Report**. The **Report Wizard** dialog opens with the attached report template already selected. Proceed to the subsequent steps of the wizard.
- Right-click the namespace (for example, a package) in the Containment tree and from the shortcut menu select **Generate Report**. Then select the appropriate report templates category and click the attached report template.

  TIP! The "[Attached]" tag helps to identify attached to project report templates on this menu (see the following figure).

  Proceed to the subsequent steps of the wizard.
- From the main menu, select **Tools** > **Report Wizard**. In the **Report Wizard** dialog, select the report template and proceed to the subsequent steps of the wizard.

Copyright © 1998-2015 No Magic, Inc.

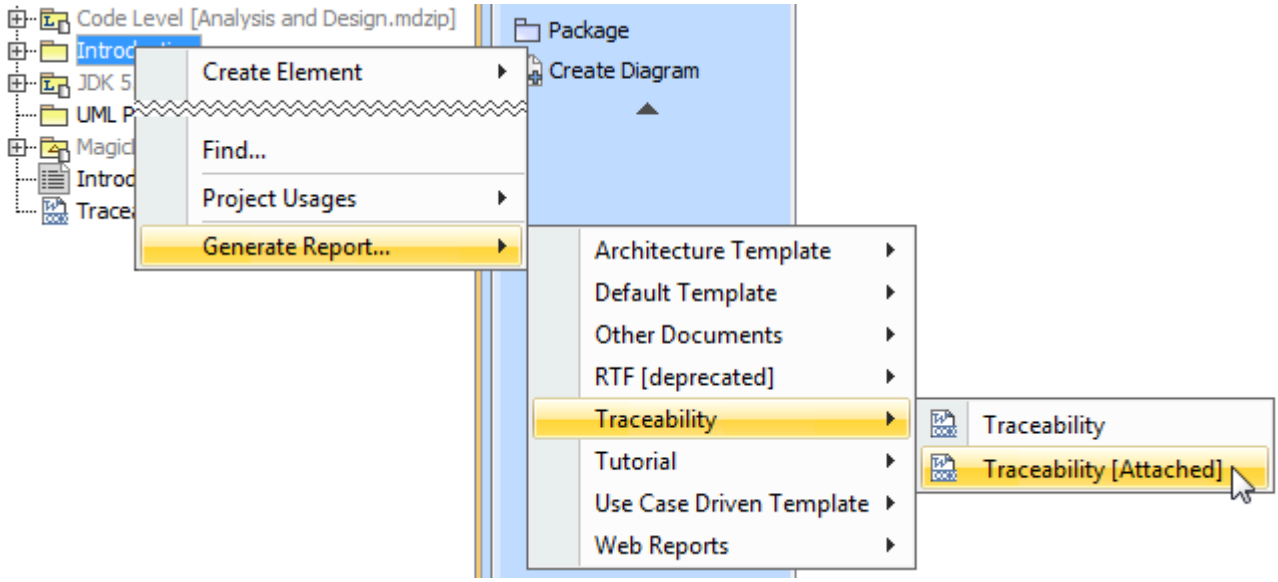


Figure 599 -- Selecting attached to project report template for report generation

If you no longer need the attached report template, you can simply delete it from the model. Just select it in the Containment tree and press Delete.

If you need to modify the attached report template, do the following:

1. Extract it to the local report templates folder.
2. Open the extracted template for edit and make changes in it.
3. After you finish the editing, re-attach the updated report template to your project.

To extract a report template to the local report templates folder

1. From the main menu, select **Tools** > **Report Wizard**. The **Report Wizard** dialog opens.
2. Select the report template you want to extract to the local report templates folder and click the **Extract** button.
3. If the message asking to confirm the update of already existing report template appears, click **Yes**.

The report template appears in the local report templates folder and can be opened for edit. It also remains attached to the open project.

Related external resources

MagicDraw ReportWizard UserGuide.pdf

Send Signal Action

The Send Signal Action is an action that creates a signal instance from its inputs and transmits it to the target object, where it can cause the start of the state machine transition or the execution of an activity. The argument values are available to the execution of associated behaviors. The requester continues the execution immediately. Any reply message is ignored and is not transmitted to the requester.

Copyright © 1998-2015 No Magic, Inc.

You can specify send signal action properties in the send signal action Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about working with symbols, see "Diagramming" on page 198.
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

State

A state is a condition during the lifetime of an object or an interaction during which the object meets certain conditions, performs an action, or waits for an event. The state is defined by the concepts of duration and stability. An object can not be in an unknown or undefined state. A state may have two compartments to provide more information about that state:

- The first compartment is the name compartment, it contains the state name, for example: running, going up.
- The second compartment is the activity compartment, it contains the events and actions of the state.

You can specify state properties in the state Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about working with symbols, see "Diagramming" on page 198.
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To create a region

Do one of the following:

- On the Containment tree, right-click the state, click **Create Element,** and then click **Region**.
- On the diagram, right-click the state and then click the **Add New Region** command.
- On the diagram, select the state symbol and click the *Create Region* smart manipulator. For more information about smart manipulators, see "Smart Manipulation" on page 233.
- In the state Specification window, click the **Inner Elements** property group, then click **Create** and from the menu, select **Region**. Specify the region and click **Close** or **Back**.

To remove the region

Do one of the following:

- On the diagram, right-click the state and on the shortcut menu, click **Remove Region**.
- In the state Specification window, click the **Inner Elements** property group. Then click the region and click the **Delete** button.

To insert a state to the region quickly

- In the Containment tree or on the diagram, click the state, and drag it to the state's region.

Copyright © 1998-2015 No Magic, Inc.

To display or hide the region name

1. On the diagram, right-click the state and from the shortcut menu, select **Symbol Properties**.
2. In the **Symbol Properties** dialog, in the right-top corner, click to expand the menu, and select the **Expert** mode.
3. Select or clear the **Show Region Name** check box.

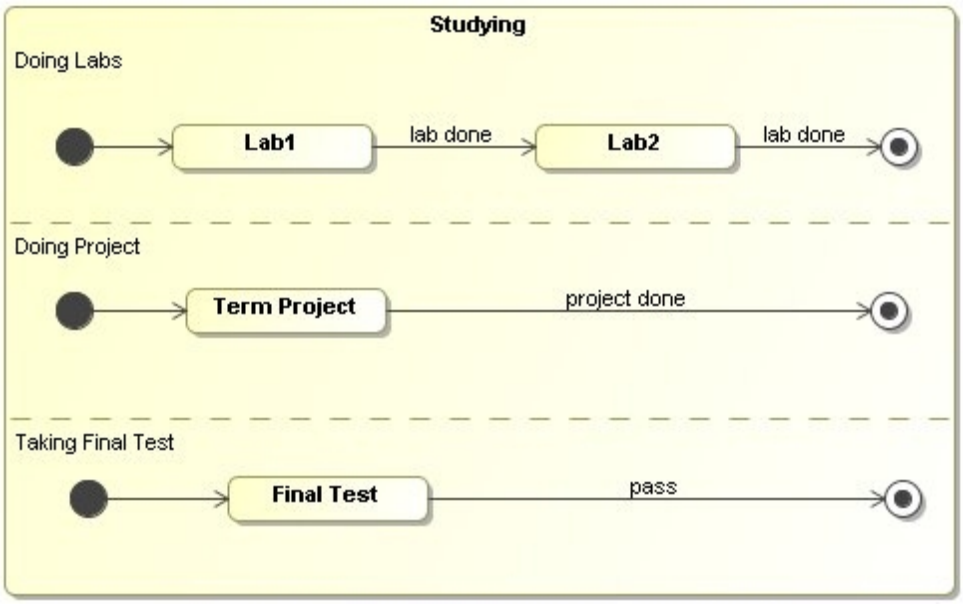


Figure 600 -- Displaying region name on the State symbol

Changing State to Composite/submachine/orthogonal State

You can change your current state to a simple state, composite state, orthogonal state, and submachine state.

To change the state to the composite state

- Add one region to the state. For more information, see "To create a region" on page 981.

To change the state to the orthogonal state

- Add two regions to the state. For more information, see "To create a region" on page 981.

To change the state to the submachine state

1. On the diagram, right-click the state, then on the shortcut menu, click **Submachine**.
2. In the dialog, click to select a submachine that you wish to apply on the state or click the **Create** button, to create the submachine.

NOTE To change the state to the submachine state, the state has to be without regions.

Composite State

A composite state either contains one region or is decomposed into two or more orthogonal regions. Each region has a set of mutually exclusive disjoint subvertices and a set of transitions. A given state may only be decomposed in one of these two ways.

Copyright © 1998-2015 No Magic, Inc.

Any state enclosed within a region of the composite state is called a substate of that composite state. It is called a direct substate when it is not contained by any other state; otherwise it is referred to as an indirect substate.

Each region of the composite state may have an initial pseudostate and a final state. A transition to the enclosing state represents a transition to the initial pseudostate in each region.

You can specify composite state properties in the state Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about working with symbols, see "Diagramming" on page 198.
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.
- For more information on working with states, see "State" on page 981.

Submachine

A submachine state specifies the insertion of the specification of a submachine state machine. The state machine that contains the submachine state is called the containing state machine. The same state machine can be a submachine more than once in the context of a single containing state machine.

The submachine state is semantically equivalent to a composite state. The regions of the submachine state machine are the regions of the composite state. The entry, exit, behavior actions, and internal transitions, are defined as part of the state. The submachine state is a decomposition mechanism that allows factoring of the common behaviors and their reuse.

You can specify submachine properties in the state Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about working with symbols, see "Diagramming" page 198.
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.
- For more information on working with states, see "State" on page 981.

Adding connection point reference

The connection point reference represent an entry to or exit from the submachine state. It can be used as the source or target of a transition.

To draw the connection point reference on the submachine state

1. In the state diagram toolbar, click the **Connection Point Reference** button. Click on the diagram pane on the submachine state shape. The **Select Entry/Exit Point** dialog opens.
2. Select *entry point* to define the entry into the submachine state or *exit point* to define the exit from the submachine state. The Connection Point Reference is drawn on the submachine state with a defined entry or exit point.

To see the assigned entry/exit point, open the **Connection Point Reference** dialog. The **Entry** or **Exit** properties will display the defined entries.

Copyright © 1998-2015 No Magic, Inc.

To assign the entry/exit points to the Connection Point Reference

Select the Connection Point Reference on the diagram pane. Open the shortcut menu and select the **Select Entry/Exit Point** command.



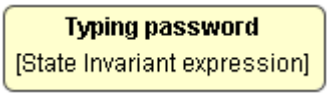
In the **Select Entry/Exit Point** dialog, only these entry and exit points are listed, which are created at the same State Machine as the submachine state. If there are no entry/exit points at the same state machine, the **Select Entry/Exit Point** dialog is not opened when drawing the Connection Point Reference.

Defining State Invariant

To define a state condition

- Double-click the state to open the **State** Specification window and in the **State Invariant** field, type a condition and submit changes.
- Near the **State Invariant** field, click the **...** button. The **Edit State Invariant** dialog opens. Type the condition and close the dialog by submitting changes.

The State Invariant value is displayed on the diagram pane on the state shape in brackets:



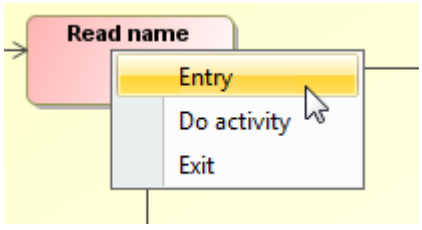
Assigning behavior to state

You can define a behavior to be executed correspondingly to the listed events while being in the state whenever the state is entered and exited.

To assign a behavior to a state

Do one of the following:

- Use a drag-and-drop operation:
  1. In the Model Browser, select a behavior type element.
  2. Drag it to the selected state on the diagram. The list of activities appears.



  3. Click a desired activity. The behavior is assigned to the selected state's activity.
- Use the transition's Specification window:
  1. Open the state's Specification window.

Copyright © 1998-2015 No Magic, Inc.

2. In the **Entry, Do Activity**, or **Exit** category, click the **Behavior Type** property value cell. The list of available behavior types opens. Select the behavior type from the list.

NOTE

If you do not see the **Behavior Type** property, click the **+** button near the **Entry**, **Do Activity**, or **Exit** category to expand its content.

3. According to the selected behavior type, the **Behavior Element** value is defined automatically. Additional properties **Name** and **Owned Diagram** appears. Specify these properties if it is needed.
4. Click the **Close** button after you have defined desired properties.

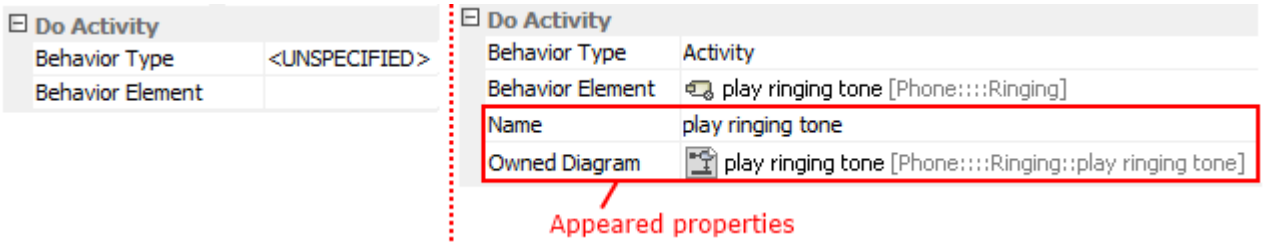


Figure 601 -- Do Activity group before assigning behavior type (on the left) and after assigning behavior type (on the right)

If a behavior of an Activity, Interaction, Protocol State Machine, or State Machine type is assigned to an Entry, Exit, or Do activity, a diagram for that behavior is created automatically.

TIP!

- For more information, see "Behavior" on page 851.
- For more information on how to change the opaque behavior display mode, refer to "NEW! To change an opaque behavior display mode on state or transition shapes" on page 951.

Stereotype

A stereotype defines how an existing metaclass may be extended. It enables the use of platform or domain specific terminology or notation in place of, or in addition to, the ones used for the extended metaclass.

Just like a class, a stereotype may have properties, which may be referred to as tag definitions. When a stereotype is applied to a model element, the values of the properties may be referred to as tagged values.

Any model element from the reference metamodel (any UML model element) can be extended by a stereotype. For example in UML, States, Transitions, Activities, Use cases, Components, Attributes, Dependencies, etc. can all be extended with the stereotypes.

The stereotypes are created as separate model elements and can be drawn in almost all MagicDraw diagrams.

Stereotype notation

Stereotype notation in diagrams use the guillemets « » instead of symbols «» (see the following figure).

However, when editing elements in a diagram, you can still enter the stereotype names between the «» symbols.

Copyright © 1998-2015 No Magic, Inc.



Figure 602 -- Stereotype Notation

To create a new stereotype

To create a new stereotype it is recommended to first create a profile (see the Section above).

- Use the **Stereotype** button on the Profile diagram pallet.
- From the Profile or Package shortcut menu in the Model Browser, select **Create Element**, and then select **Stereotype**.

To change the stereotype's metaclass

1. Do one of the following:
   - **NEW!** On a diagram pane, select the stereotype, then on its smart manipulator toolbar, click the **Metaclass** button.
   - In the stereotype's Specification window, click the **Metaclass** value cell, and then click the **Edit** button.
2. In the **Select Metaclass** list, click to select the metaclass you want to apply.
3. Click **Apply** when you have finished.

To create a stereotype with an image

1. Open the stereotype Specification window.
2. Click the **Icon** "**...**" button and from the **Open** dialog, select an image you want to place for the stereotype. Click the **Open** button.

TIP!

For more information about displaying stereotype icon on shape, see "Displaying icon or image" on page 319.

To display the stereotype icon instead of the shape on the diagram pane

1. Assign the stereotype with the icon to the element.
2. Select the shape on the diagram and then click the *Compartments* smart manipulator.
3. Click to expand the menu and then click **Suppress All**.

To view assigned image properties

1. In the stereotype Specification window, right-click the **Icon** property or the image assigned to the property to open the shortcut menu.
2. On the shortcut menu, click **Open Specification**. The **Image** specification window opens wherein you can see and specify image properties.

To apply a stereotype

Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

- Open the corresponding element Specification window. Click the "..." button in the general pane, next to the **Applied Stereotype** property. Select one or more available stereotypes from the open list and click **Apply**.
- On the element shortcut menu, click **Stereotype** and then click to select a stereotype that you wish to apply on the element. Click **Apply** when you are done.
- On the diagram pane in the element name area, type two open angle brackets «, type the stereotype name and type two close angle brackets ». Then you can type the element name itself.



If you want to name element Books and assign «table» stereotype, in the element name area type the following: «table» Books. The name completion for the stereotypes works in the name editing mode, press Ctrl+Spacebar or Ctrl+Backspace to get a list of possible to apply stereotypes.

NEW! To apply a stereotype without using the mouse

1. Right-click the element on which you need to apply the stereotype. The shortcut menu opens.
2. Select **Stereotype**. The **Select Stereotype** dialog opens.
3. Use the Up and Down Arrow keys to walk through the list.
4. Press Ctrl+Spacebar to select the desired item.
5. Press Enter to apply the stereotype.

Stereotypes can be ordered. Symbol style of the first in the list stereotype will be applied to the element.

To order stereotypes

1. On the symbol shortcut menu, click the **Stereotype** command. The **Select Stereotype** dialog opens.
2. Click the **Order** button. The **Order Stereotypes** dialog opens.
3. Click **Up** or **Down** buttons to order stereotypes.

Saving of stereotype information in XMI

Ability to choose options where to save applied stereotype information in XMI file. Information can be saved at the end of the file or inside the element information.

By default stereotype information is stored at the end of XMI file. To store information inside element, open **Environment Options** dialog (choose **Options** - **Environment** command from main menu), and in **General** - **Save/Load** section select property **Save** stereotype information within element.

Stereotype properties

You can specify stereotype properties in the stereotype Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Copyright © 1998-2015 No Magic, Inc.

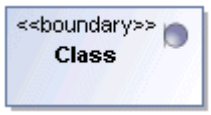
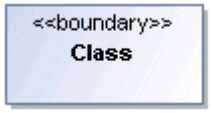
Changing the stereotype display mode

You can change the stereotype name and its icon visibility on an element shape.

To change the stereotype display mode on the element shape

1. Create a stereotype and assign an icon to the stereotype.
2. Assign the stereotype to an element.
3. Draw the shape of the element on the diagram.
4. Click the shape, and then, click ⊟ buttons to suppress all the compartments.
5. Click the shape and then click the ⊡ *Compartments* button. From the menu, select the **Stereotypes** and then click the **Shape Image** check box.

The stereotype display modes are described in the following table.

Show Stereotypes Property Value	Shape	Icon of the stereotype on the shape	Name of the stereotype	Image of the stereotype instead of the element shape
Text and icon		displayed	displayed	-
Text		not displayed	displayed	-
Icon		displayed	not displayed	-
Shape Image and Text		-	displayed	displayed*
Shape Image		-	not displayed	displayed*
Do Not Display		not displayed	not displayed	-

* - To display the image of a stereotype instead of the element shape all element compartments should be suppressed.



Shape Image and Text and **Shape Image** properties are not included in the Relationships, Roles and Diagram Frame element property list.

Parent topic

Stereotype

Copyright © 1998-2015 No Magic, Inc.

Related diagram

Profile Diagram

Related procedure

Editing Property Values

Related dialog

Specification Window

Related references

See an UML extensions sample project in *<MagicDraw installation directory>\Samples\Product Features\UML Extensions.mdzip*.

For a detailed description of UML Profiling and DSL in MagicDraw, see the MagicDraw UMLProfiling&DSL UserGuide.pdf.

Subsystem

A subsystem is treated as an abstract single unit. It groups model elements by representing the behavioral unit in a physical system.

To draw a subsystem

- On the Use Case diagram pallet, click the Subsystem button, and then click the diagram pane.

Swimlane

Actions and subactivities can be organized into swimlanes in the activity diagrams. The swimlanes are used to organize responsibility for actions and subactivities according to the class. They often correspond to the organizational units in a business model.

The swimlanes limit and provide a view on the behaviors invoked in the activities. They consist of one or more partitions. They can be vertical and horizontal.

An activity diagram can be divided visually into “swimlanes”, each separated from the neighboring swimlanes by vertical or horizontal solid lines on both sides. Each swimlane represents a responsibility for part of the overall activity, and may eventually be implemented by one or more objects. The relative ordering of the swimlanes has no semantic significance, but can indicate some affinity. Each action is assigned to one swimlane. Transitions can cross lanes. There is no significance to the routing of a transition path.

You can specify swimlane properties in the swimlane Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



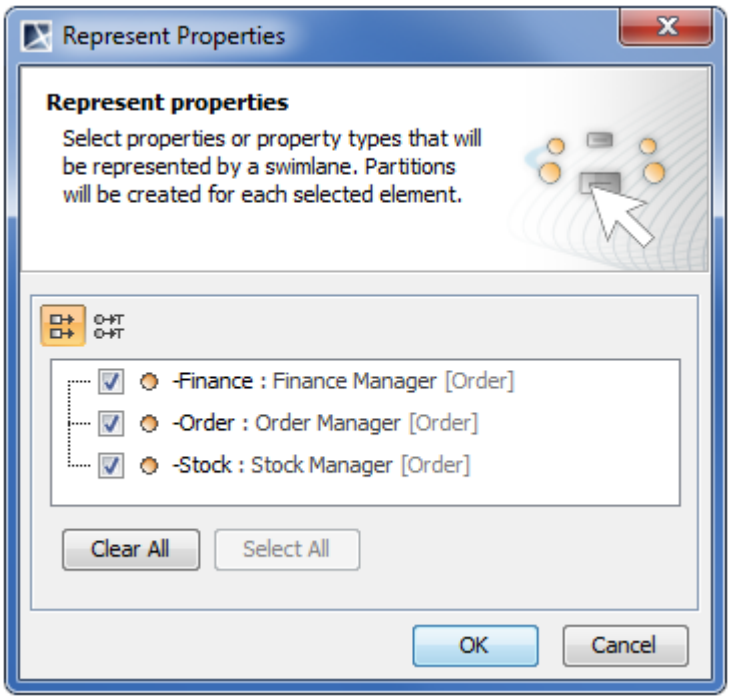
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To create a swimlane

Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

- On the activity diagram pallet, click the *Vertical Swimlanes* button and then click the diagram. The **Represent Properties** dialog appears. Select properties and click **OK**.



NOTE

The **Represent Properties** dialog appears, if:

1. the activity diagram is created under the classifier
2. the classifier has properties

- From the Containment tree, drag one or more property, actor, class, or instance specification elements to the diagram. The swimlane is created and the representative element is set.

To set representative elements

Do one of the following:

- Open the Specification window of the swimlane and click the cell of the **Represents** property value. Then click and in the open dialog, select the element. Click **OK** when you are done.
- From the Containment tree, drag the representative element to the partition on the diagram.

To add an additional partition

Do one of the following:

- On the diagram, right-click the swimlane, point to **Insert Swimlane** and then select **Insert Vertical Swimlane** or **Insert Horizontal Swimlane**.
- From the Containment tree, drag one or more property, actor, class or instance specification elements to the swimlane on the diagram.

To draw multidimensional swimlanes

1. Draw a vertical swimlane.
2. From the swimlane shortcut menu, select the **Insert Horizontal Swimlane** command.

Copyright © 1998-2015 No Magic, Inc.

3. Insert as many horizontal and vertical swimlanes as you need.

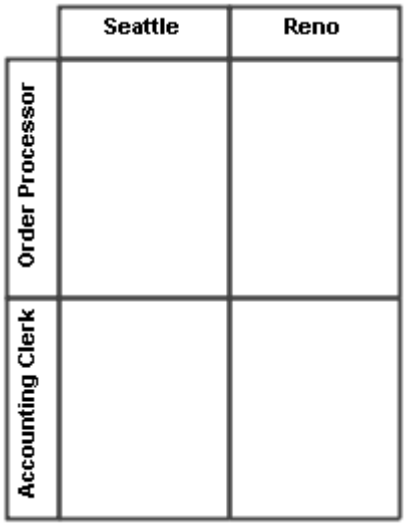


Figure 603 -- Example of multidimensional swimlane

To add model elements to a swimlane

If a swimlane is already drawn in the activity diagram, drawing an action (or any other element) will highlight the swimlane in blue. This means that the action shape will depend on the swimlane symbol.



If the model elements depend on a swimlane symbol, they will be deleted if the swimlane symbol is deleted.

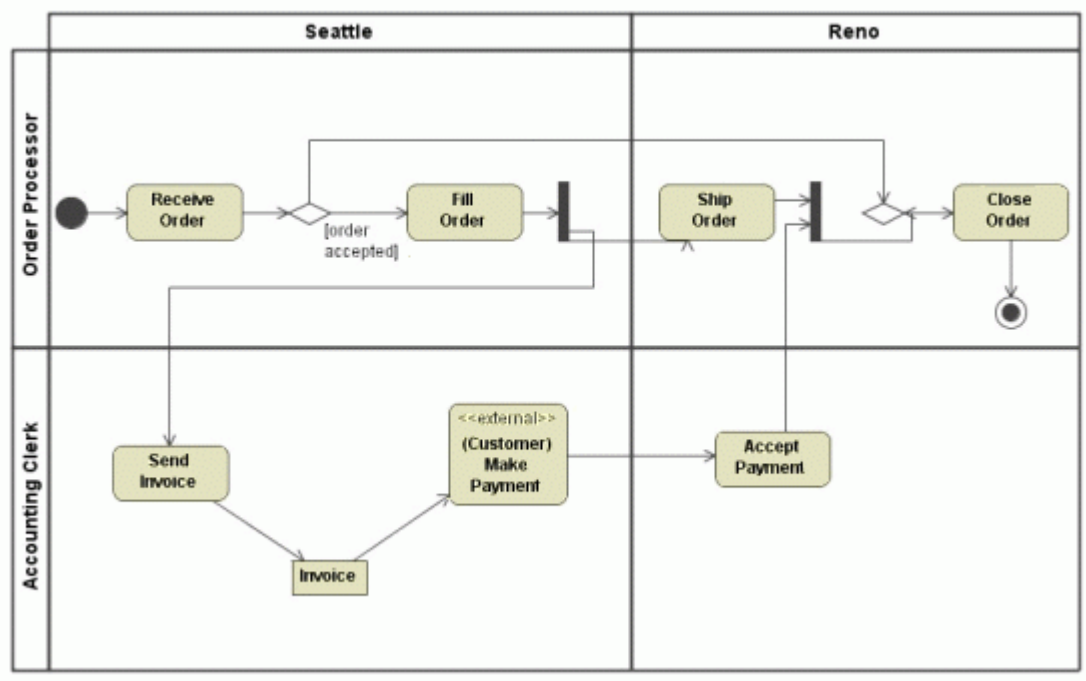


Figure 604 -- Example of multidimensional swimlane

Related diagrams

Activity Diagram

Copyright © 1998-2015 No Magic, Inc.

Related procedures

Formatting Symbols

Related references

Specification Window

Tag

Just like a class, a stereotype can have properties, which can be referred to as tag definitions. When a stereotype is applied to a model element, the values of the properties may be referred to as tagged values.

The tag definitions are used to define new meta attributes of the extended metaclass, they are used as regular class attributes.



For more information about working with class attributes, see "To create a symbol of an opaque behavior" on page 952.

An actual instance of the tag definition is a tagged value (tag).

A tag holds extra information like:

- additional information that does not come with UML, for example Precondition for Use Cases.
- management data about the state and progress of the project such as author, status, and tested.
- language specific data for code generation tools.

A tagged value consists of two parts: name and value (example: Author = Joe).

To create a new tag definition

1. Create a new stereotype.
2. Open the stereotype Specification window.
3. Click the **Tag Definitions** tab and use the **Create** button to add a new tag definition for stereotype. Select the type of this property. It can be a standard UML data type or another user defined Stereotype. Regular classes should not be used as types of tag definition.
4. In the stereotype Specification window, click **Close** to save changes.



For more information about stereotype properties, see "Stereotype properties" on page 987.

Editing tagged value

To create a new tagged value

- In the element Specification window, **Tags** group, select an available tag definition and click the **Create Value** button.

Copyright © 1998-2015 No Magic, Inc.

- Set the tag value using drag and drop. Now you can drag an element from the Containment tree and drop in the Specification window tag value area.

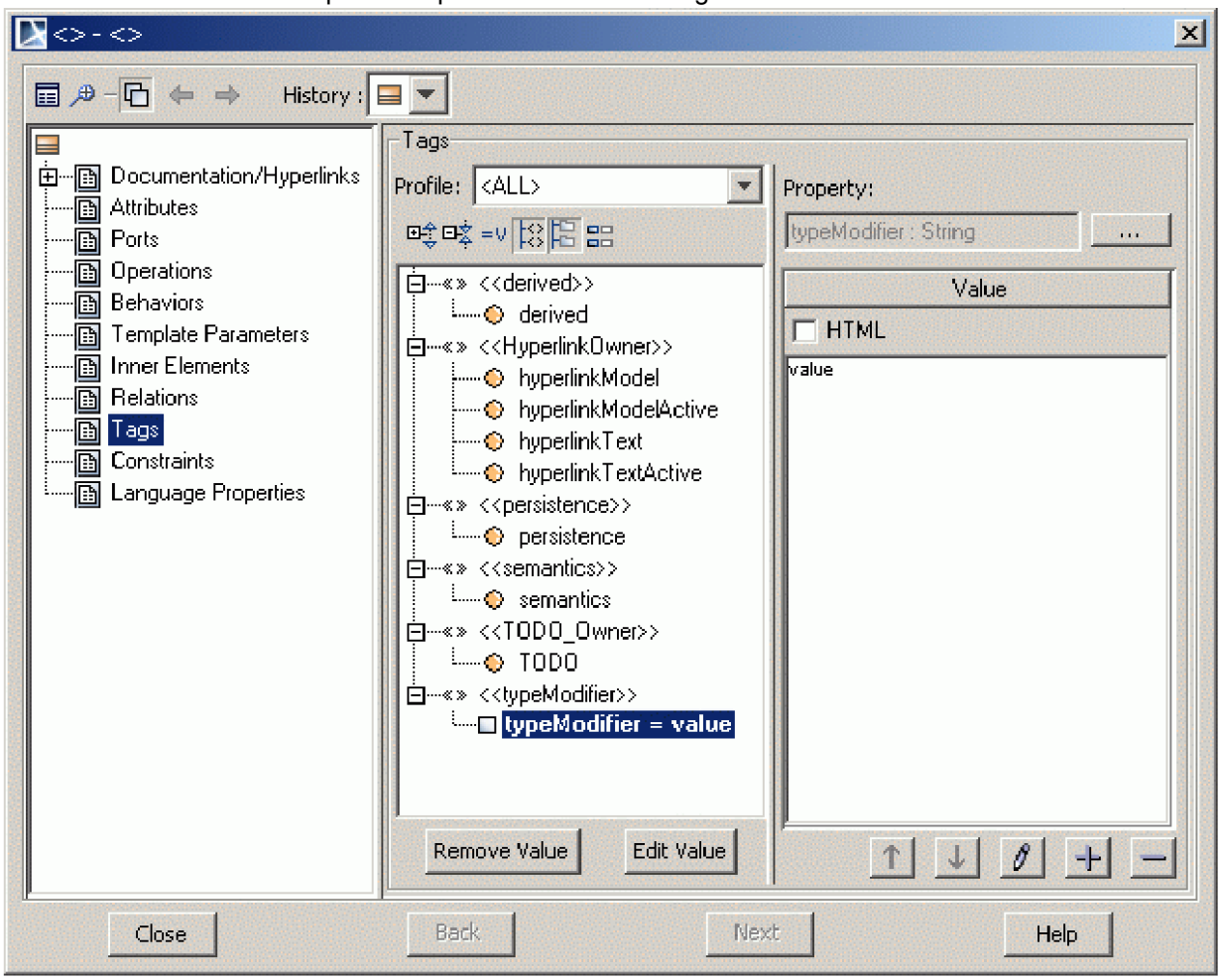


Figure 605 -- Tags dialog

Button	Description
Up	Moves the created value to an upper position in the list.
Down	Moves the created value to a lower position in the list.
Edit	Opens an editable window for value.
Add	Adds a new tagged value to the list.
Remove	Removes a tagged value from the list.

Copyright © 1998-2015 No Magic, Inc.

To create default tag values

1. In the stereotype Specification window, expand the **Tag Definitions** branch and select the tag definition.
2. Assign a value for the **Default Value** property.
3. Create an element and assign a stereotype for this element. The element will have tags with assigned default values.

To set a default tag value to tag with empty value

MagicDraw allows for setting default tag values to tag with empty value. This functionality is needed when a stereotype is already assigned to an element and a new mandatory tag definition with a default value is created for the stereotype. After creating such a tag definition, the model elements that have the modified stereotype applied will have the newly created tags unset.

To set default values instead of empty values:

1. From the **Tools** main menu, select the **Set Empty Tags to Defaults** command. The **Select Package** dialog opens.
2. Select the scope of elements to which you want to set the default tag values.



If you want to assign default values to all project tags, in the *All data tree* select the root package *Data* and click the **Add All** button. The root package *Data* is added to the *Selected objects* list. Click OK.

3. The **Question** dialog opens informing you that this action will set the mandatory tags without values to defaults.

If you do not want to see this message again next time, clear the **Show this message next time** check box.

4. Click **Yes**. Now, the element with stereotype tags has been assigned default values.

The following conditions are required to set the default tag values:

1. The element should have an assigned stereotype with a specified default value property.
2. The tag definition, to which you want to assign a default value, should contain no value.



The default tag values are not set when the stereotype property (tag definition) multiplicity is equal to 0, 0..1, 0..*, *.

Parent Topic

Tag

Related topics

Stereotype

Attribute

Template

Template is used to define concrete types for a bunch of elements without concrete types. In other words, you can create a template and then you can use this template as many times as you need. Moreover, when you have elements of the same structure (or with the same parameters) but with the different values, the template is used for easier defining parameters for each element.

Templates parametrization can be applied to classifiers, packages, and operations. Most often template parametrization is used for classes and interfaces.

Copyright © 1998-2015 No Magic, Inc.

Templates are often used in code engineering (Java, C#, and other programming languages).



For more information about how to use templates in code engineering, see MagicDraw CodeEngineering UserGuide.pdf.

The template consist of the following parts:

- A parameterized element. It is an element with defined template parameters. The parameterized element can also be called as a template element.
- An actual element. It is an element for which a template is applied, that is an element with actual values. The actual element can also be called an templateable element.
- A template binding relationship. It is a relationship from the actual element to the parameterized element.
- A template parameter substitution. It is used to define actual parameters of the actual element that substitutes formal template parameters. Template parameter substitutions are created on the template binding relationship.

Case study: an example of the template and its usage in the following figure.

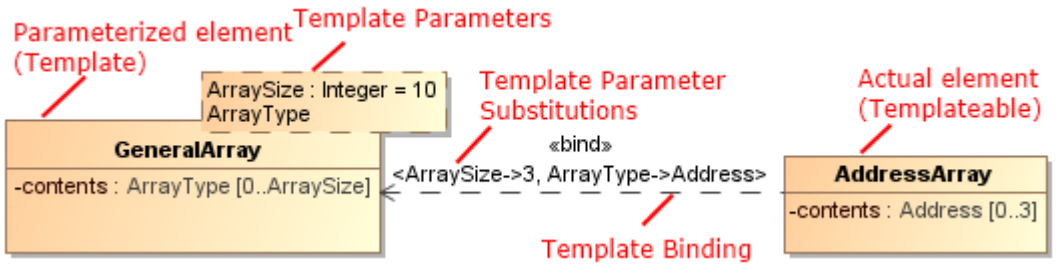


Figure 606 -- Example of template

In this example, you can see the *GeneralArray* template class, which has the *ArraySize* and *ArrayType* template parameters that are represented in the dashed rectangle on the upper-right corner of the class. These parameters represent the size and type of the general array.The attribute *GeneralArray::contents : ArrayType [0..ArraySize]* represents the *ArrayType* and the ArraySize template parameters. The *ArraySize* template parameter is of the *Integer* type and its default value is 10. The *ArrayType* template parameter is of the *Class* type (it is not represented on the shape) and its default value is not specified. The default value of the template parameter is used if no actual value is supplied for the parameter in a binding. Typically, the parameter types are classifiers, but they can also be integers or other types.

The *GeneralArray* class and its template parameters creates a template of the general array with undefined types. Such template can be used in other concrete arrays as many times as you need.

Next let's see, how to use the template and how to create an actual element.

Let's say, we would like to create an array of addresses. So, we create the *AddressArray* class. Alone (without template) *AddressArray* class would have undefined types. A template binding is created from the *AdressArray* class to the *GeneralArray* class. Now the template binding connection points to the *GeneralArray* template class.

The template binding specifies template parameter substitutions - the actual values of parameters for the address array. According to the example, we create the following actual values of the *AdressArray* class - the array size having the value *3* and the array of the *Address* type.

Copyright © 1998-2015 No Magic, Inc.

The output of these definitions is the *AddressArray* class having the attribute *AddressArray::contents : Address [0..3]*.



Attributes for the actual element classes should be created manualy. In the example, the *AdressArray::contents : Address [0..3]* attribute is not created automatically - you have to create this attribute manually. Is your optional solution to create such attributes in your model or not.

Step by step instructions on how to model a template in the MagicDraw are described in "Working With Templates" on page 996.

You can format template parameter symbol properties in the **Symbol Properties** dialog.

You can specify template parameter properties in the Template Parameter Specification window and template binding properties in the Template Binding Specification window. In the specification window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.



- For more information about symbol representation properties, see "Formatting Symbols" on page 317.
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Related diagram

Class Diagram

Related procedures

Formatting Symbols
Working With Templates
Editing Property Values

Related references

Specification Window
Selecting an Element

Related resources

MagicDraw CodeEngineering UserGuide.pdf

Working With Templates

In the section, we will show how to create a template and how to use it.

Follow these instructions from the first step consistently. Because, for example, if you do not define the template parameter, you will not be able to draw a template binding relationship. You can omit optional procedures only.

Precondition. Let's assume, you already have an element to which you want to apply a template. This element is an actual element. The actual element can be a classifier, package, or operation.

To create and apply a template an actual element, see the following procedures:

- To create a template
- To apply a template to an actual element

In the following procedure, we will show you how to create a template with parameters and how to define these parameters.

Copyright © 1998-2015 No Magic, Inc.

To create a template

1. Create or select a template element for which you will create template parameters.
2. Open the element Specification window.
3. Change the property display mode to **Expert** or **All** if it is not changed already. The **Template Parameters** property group appears in the property group list.



The **Template Parameters** property group is displayed in the **Expert** mode for the following elements:

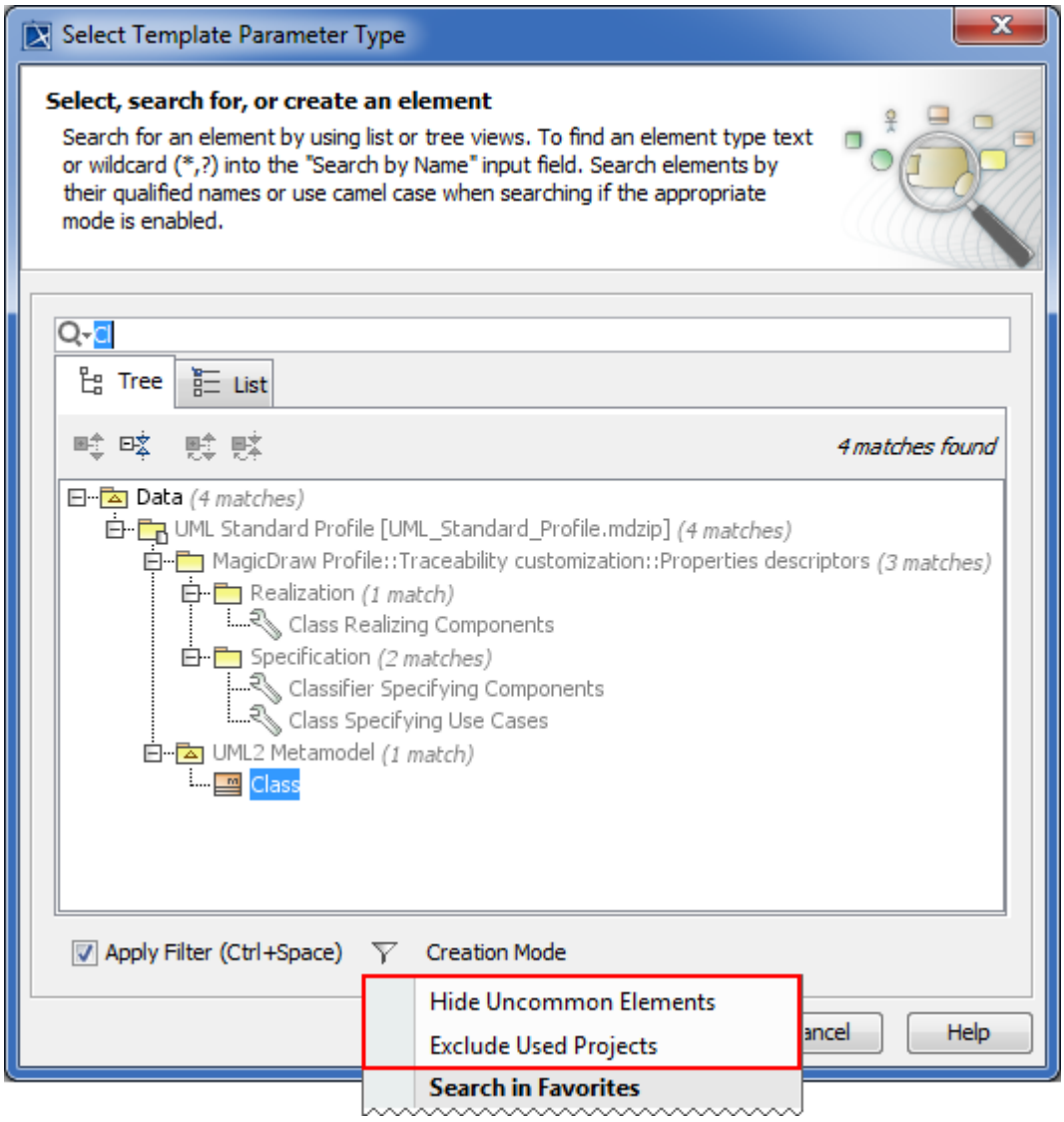
- Class
- Interface
- Package

For the rest of elements, the **Template Parameters** property group is displayed in the **All** mode.



For more information about changing mode in the element Specification window, see "Property Group Toolbar" on page 278.

4. Click the **Template Parameters** property group.
5. To add a new template parameter, click the **Create** button. The **Select Template Parameter Type** dialog opens.



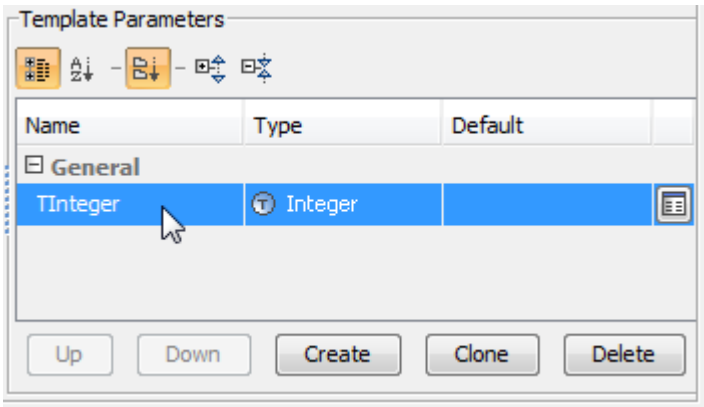
Copyright © 1998-2015 No Magic, Inc.

6. In the model tree, select an existing or create a new type of the template parameter.

NOTE

- If you do not want to specify the concrete template parameter type, you can select the *Class* type (the *Class* from the UML metamodel). When the *Class* type is specified, type is not displayed on the diagram pane but only the name of the template parameter is represented.
- If you need to select one of the UML metamodel elements, such as *Class*, be sure the **Hide Uncommon Elements** and **Exclude Used Projects** commands are both unchecked (as in the preceding figure). For more information, see "Searching for Elements in Element Selection Dialog" on page 355.

7. Click **OK**. The template parameter is created. Name of the template parameter is defined automatically by compounding of the *T* letter and the name of the template parameter type - *T<type name>*. For example, *TInteger*. You can change the name of the template parameter if you need it.



8. Assign the default value to the created template parameter. This step is optional. For information about assigning values in the Specification window, see "Editing Property Values" on page 298.

NOTE

To assign a particular value of the value specification, you need to specify a Value Specification first and then type its particular value in the **Default** value cell. For more information about Value Specification, see "Value Specification" on page 1027.

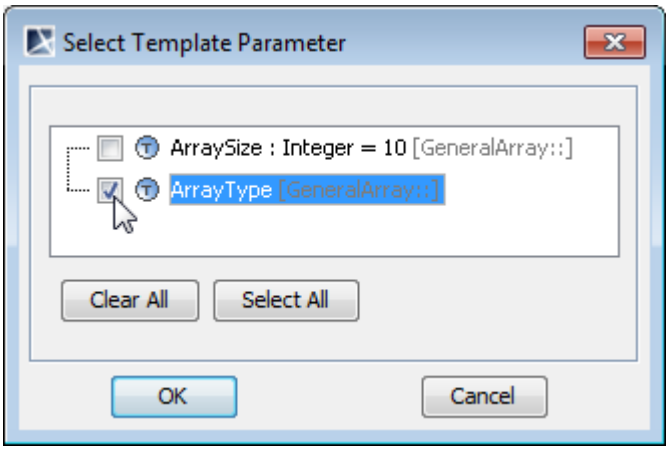
After the template is created, you can apply it to the actual element. The following procedure describes how to apply the template to the actual element.

To apply a template to an actual element

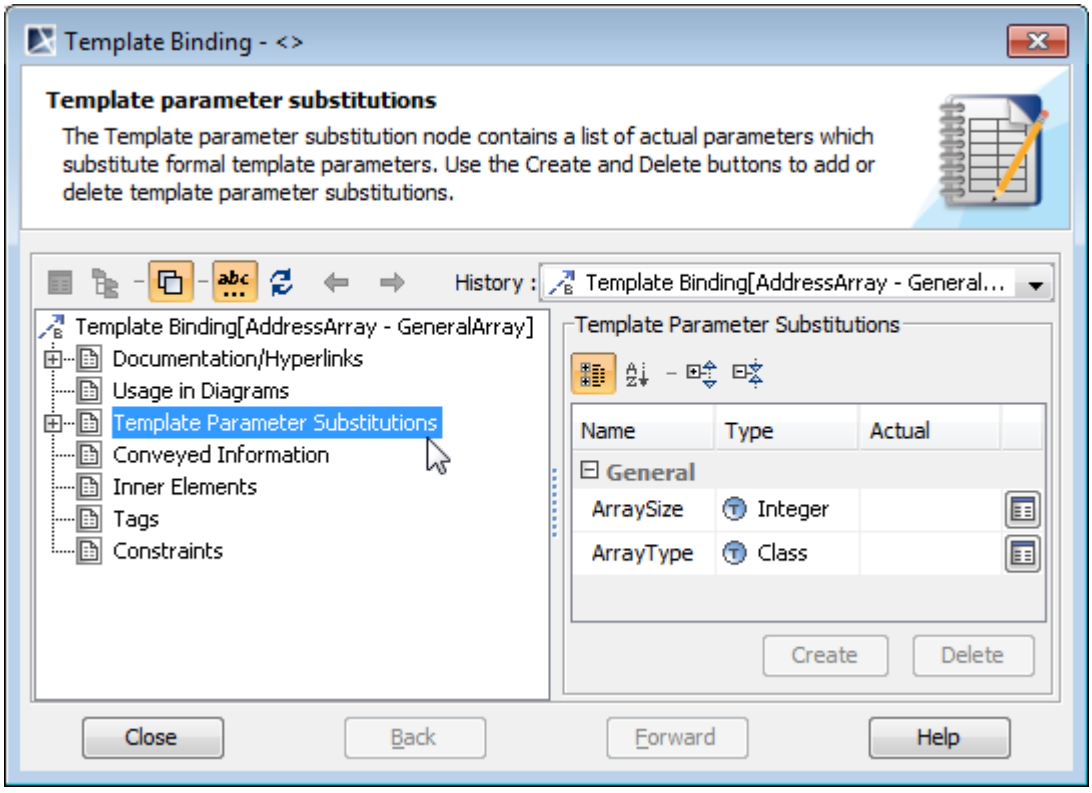
1. Bind the actual element to the template element - in the diagram pallet, click the **Template Binding** button or click the **B** shortcut key and draw a template binding relationship from the actual element to the template element.
2. Define a template binding substitution. Open the **Template Binding** Specification window and click the **Template Binding Substitutions** property group.

Copyright © 1998-2015 No Magic, Inc.

3. To add a new template binding substitution, click the **Create** button. The **Select Template Parameter** dialog opens wherein the template parameters are listed of the element to which the template binding relationship is connected.



4. Select one or more template parameters you need to add as template parameter substitutions. Click **OK** to get back to the **Template Binding** Specification window.



5. Assign the actual value for each template binding substitution. This step is optional. For information about assigning values in the Specification window, see "Editing Property Values" on page 298.

NOTE

To assign a particular value of the value specification, you need to specify a Value Specification first and then type its particular value in the **Actual** value cell. For more information about Value Specification, see "Value Specification" on page 1027.

Related concepts

Template

Related diagram

Class Diagram

Copyright © 1998-2015 No Magic, Inc.

Related procedures

Formatting Symbols

Editing Property Values

Related references

Specification Window

Selecting an Element

Related resources

MagicDraw CodeEngineering UserGuide.pdf

Transition

A transition is a directed relationship between a source vertex and a target vertex. It can be a part of a compound transition, which takes the state machine from one state configuration to another, representing the complete response of the state machine to an occurrence of an event of a particular type. The transition relationship is created between two states. The transition specifies event occurrences and guard conditions. When these events occur and conditions are satisfied with the object in the source, state will perform the specified effect and will enter the target state. So, in the transition, the main subjects are as follows:

- Event trigger and its parameter.
- Guard condition.
- Effect.

On a transition path the event trigger, guard condition, and effect has the following syntax:

<*event name*> (<*parameter*>) [<*guard condition*>] / <*effect name*>

EXAMPLE

An example of the transition.

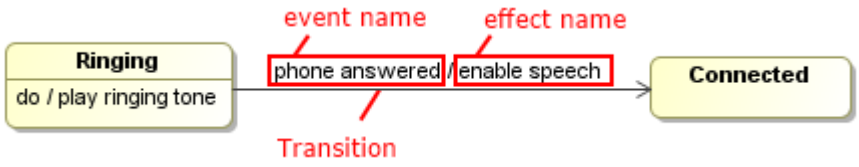


Figure 607 -- Example of transition

In the preceding example, the transition relationship is created from the *Ringing* state to the *Connected* state. The transition has the specified event - *phone answered*, and the specified effect - *enable* speech. When the *phone answered* event occurs, the *Ringing* state performs the speech effect, and then the *Connected* state is entered.

The event on the transition is specified as event of the concrete type. The event trigger is the event whose reception in the source state makes the transition eligible to fire. That is, the trigger is specified by the event. Event types are as follows:

- Any Receive Event
- Call Event
- Change Event
- Signal Event
- Time Event

Copyright © 1998-2015 No Magic, Inc.

TIP!

- For more information about event types, see "Event" on page 879.
- You can assign an event type in the **Transition** Specification window, or there is a quick way to assign an event type straight on the transition from the diagram pane. For more information, see "Assigning event type" on page 1001.
- The effect on the transition is specified as one of the behaviors types. For more information, see "Assigning Behavior Type" on page 1003.
- You can format the transition symbol properties in the **Symbol Properties** dialog. For more information about symbol representation properties, see "Formatting Symbols" on page 317.

You can specify transition properties in the **Transition** Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Related diagrams

State Machine Diagram

Protocol State Machine Diagram

Related procedures

Formatting Symbols

Assigning event type

Related references

Specification Window

Related resources

Example of the State Machine diagram in *<MagicDraw installation directory>\samples\diagrams\state machine diagram.mdzip*.

Example of the Activity diagram in *<MagicDraw installation directory>\samples\diagrams\activity diagram.mdzip*.

Assigning event type

You need to assign an event to the transition in order to specify an occurrence of an event which can trigger a state transition.

You can assign an event type to a transition in the following ways:

- Using the transition's Specification window.
- By typing an appropriate command straight on the selected transition path.

To assign an event type in the **Transition** Specification window

1. Open the transition's Specification window.
2. In the **Trigger** category, click the **Event Type** property value cell. The list of available event types opens.

NOTE

If you do not see the **Event Type** property, click the **+** button near the **Trigger** category to expand its content.

Copyright © 1998-2015 No Magic, Inc.

3. Select the event type from the list.
4. According to the selected event type, **Event Element** and **Trigger** property values are defined and additional corresponding properties appears in the **Trigger** category (see the following figure).
5. Click the **Close** button after you have defined desired properties.

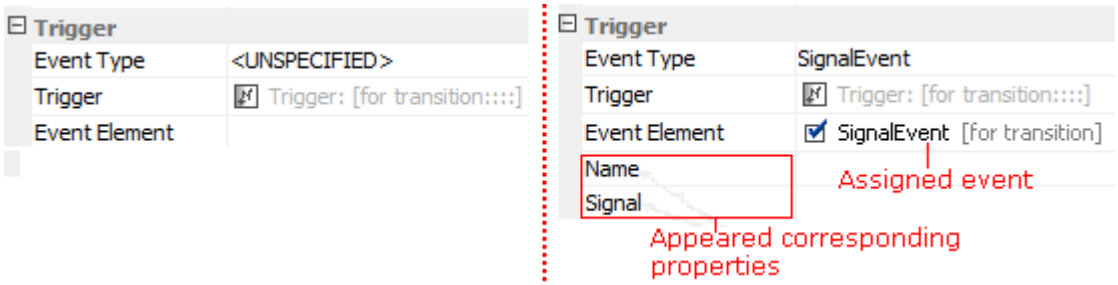


Figure 608 -- Trigger category before assigning event type (on the left) and after assigning event type (on the right)

To assign an event type straight on a transition path

1. Select a transition path on the diagram pane.
2. Type the command of the appropriate event type (see the following table for command syntax). The event type is assigned to the transition. In the following figure, see an example of the signal event type assignment.

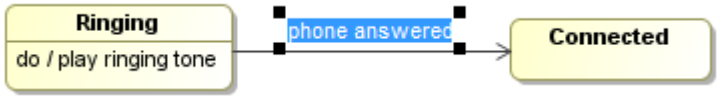


Figure 609 -- Example of signal event, that was specified typing text straight on transition

See the list with the event type and its command syntax in the following table.

Name	Command syntax
Any Receive Event	*all*
Call Event	<*operation name* ()> **NOTE:** Before assigning operation straight on the transition path the Call Event together with this operation should be already created in the same State Machine as the transition.
Change Event	*when* (<*expression*>)
Signal Event	<*signal name*>
Time Event	• *after* (<*time*>) - for an event occurring at a relative time • *at* (<*time*>) - for an event occurring at an absolute time



The event type assigning by typing the command straight on the diagram pane is also available for a transition to self in the State Machine diagram, as well as for a protocol transition and protocol transition to self in the Protocol State Machine diagram.

Related concepts

Transition

Event

Related references

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Assigning Behavior Type

A behavior type is a behavior that is performed when the transition fires.



For more information about behaviors, see "Behavior" on page 851.

To assign a behavior type

Do one of the following:

- Use a drag-and-drop operation:
  1. In the Model Browser, select a behavior type element.
  2. Drag it to the selected transition on the diagram. The behavior is assigned to the transition.
- Use the transition's Specification window:
  1. Open the transition's Specification window.
  2. In the **Effect** category, click the **Behavior Type** property value cell. The list of available behaviors opens. Select the behavior from the list.



If you do not see the **Behavior Type** property, click the **+** button near the **Effect** category to expand its content.

  3. According to the selected behavior, additional corresponding properties will appear in the **Effect** category. Specify desired properties.
  4. Click the **Close** button after you have defined desired properties.

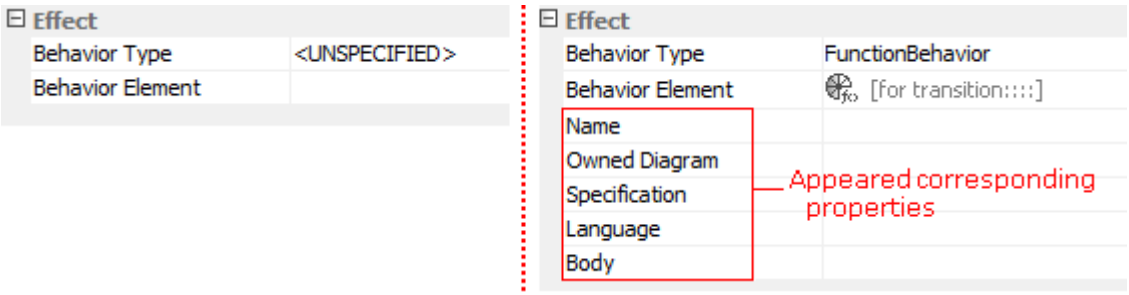


Figure 610 -- Effect group before assigning behavior type (on the left) and after assigning behavior type (on the right)



- To quickly trace the effect of the transition, from the transition shortcut menu, select **Go To** and then the effect element.
- If a behavior of an activity, interaction, protocol state machine, or state machine type is assigned to a transition, a diagram for that behavior is created automatically. For more information, refer to "Behavior" on page 851.
- For more information on how to change the opaque behavior display mode, refer to "NEW! To change an opaque behavior display mode on state or transition shapes" on page 951.

Related concepts

Transition

Event

Behavior

Related references

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Use Case

Use cases are a means for specifying required usages of a system. Typically, they are used to capture the requirements of a system, that is, what a system is supposed to do.

See an example of the use cases in the following figure.



Figure 611 -- Example of use cases

In the preceding example, you can see the fragment of the Library system. The *Register Return* use case has the association relationship with the *Librarian* actor.

The behavior of a use case can be specified by a activity, state machine, or by sequence. It can also be described by a text description - a use case scenario that can be depicted in the activity diagram.

For more details about modeling with use cases, see the following sections:

- Use Case Relations. You can learn about relations that can be used among use cases in this section.
- Use Case Description Profile. The purpose and usage of profile is described in this section.
- Use Case Scenario. This section presents the use case scenario functionality.

You can format a use case symbol properties in the **Symbol Properties** dialog.

TIP!

For more information about symbol representation properties, see "Formatting Symbols" on page 317.

You can specify a use case in the use case Specification window. In the same window, you can find the description of each use case property. Descriptions are presented in the description area of the Specification window.

NOTE

The rake icon is displayed on the shape if the use case can be realized by other behavior diagrams such as use case, activity, state machine and sequence. The rake icon isn't shown by default therefore you need to set the **Show Rake Icon** value to *true* in the **Property** dialog. For more information see "NEW! Displaying rake icon on symbol" on page 322.

Related concepts

Use Case Relations
Use Case Description Profile
Use Case Scenario

Related diagrams

Use Case Diagram

Related procedures

Formatting Symbols
Editing Property Values

Related references

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Related resources

Example of Use Case diagram in *<MagicDraw installation directory>\samples\diagrams\use case diagram.mdzip*.

Use Case Relations

A use case can have an association relation with an actor. The specified actor initiates the associated use case.

Use cases can be related to other use cases by the following relationships:

- Generalization
- Include
- Extend

Related concepts

Use Case

Actor

Related procedures

Adding extension points

Related resources

Example of Use Case diagram in *<MagicDraw installation directory>\samples\diagrams\use case diagram.mdzip*.

Adding extension points

An extension point in the use case represents the location at which the extension can be inserted. In the use case diagram, the extension point is used together with the extend relationship.

To add an extension point to the use case

Do one of the following:

- Draw an extend relationship from the extending use case to the extended use case. The following actions depends on whether or not the extended use case has an extension point:
  - If there is no an extension point, the question about adding a new extension point appears. Click **Yes** to add a new extension point and type the name of the extension point. Click **No** to cancel creating an extension point.
  - If the use case already has one or more extension points, the **Extension Points** dialog opens wherein existing extension points are listed. In the dialog, click **Create** to create a new extension point. The **Extension Point** Specification window opens. Specify extension point properties and click **Close** when you are finished.
- In the **Use Case** Specification window, click the **Extension Points** property group and then click the **Create** button. Then, in the **Extension Point** Specification window, specify the extension point and click **Close**.
- In the Containment tree, right-click the enumeration, then from the shortcut menu, select **Create Element** and then **Extension Point**.
- On the diagram, click the enumeration shape, press the Ctrl+Alt+E shortcut keys and type the extension point's name.
- On the diagram, click the enumeration shape, and then, click the *New Extension Point* button.

Copyright © 1998-2015 No Magic, Inc.

Related concepts

Use Case
Use Case Relations
Use Case Description Profile
Use Case Scenario

Related diagrams

Use Case Diagram

Related procedures

Formatting Symbols

Related references

Specification Window

Use Case Description Profile

MagicDraw provides the use case description extensions - additional use case properties. Properties are as follows: Use Case ID, Author, Date, Use Case Complexity, Pre Condition, and others.

To extend the use case with additional properties you have to load the use case description profile. By default, in a new project, the User Case Description Profile is not loaded. You have to load this profile manualy.

To load the use case description profile

1. Open the **Use Case** Specification window.

Copyright © 1998-2015 No Magic, Inc.

2. Click the **Load Profile** button. The Use Case Description Profile is loaded and additional properties are added to the use case.

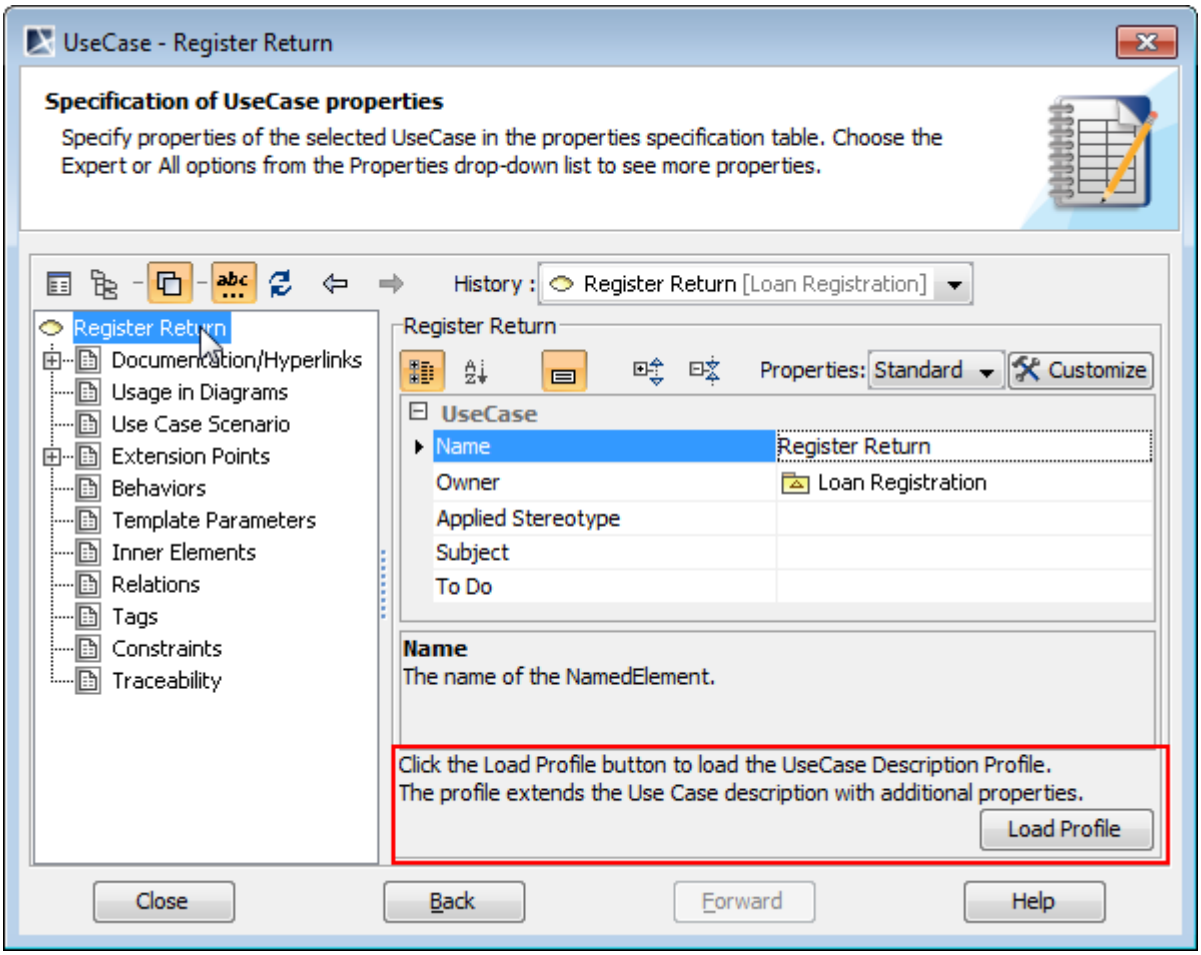


Figure 612 -- Load Profile button in Use Case Specification window

After the use case description profile is loaded, the following additional properties appears in the use case Specification window:

- The **Use Case ID** property in the UseCase general property group. This property is designed to number use cases to the particular sequence.

  TIP! For more information, see "Generic Numbering Mechanism" on page 394.

- The **Use Case Description** property group. In this property group, you can specify properties describing the particular use case, such as an Author, Date, Goal, and other.
- The **Use Case Scenario Obsolete** property group. Use cases numbering values used in projects created with earlier MagicDraw versions, are stored in this property group after importing the project to MagicDraw 17.0.2 or later. By default, this property group is displayed in the Expert mode.

  TIP! For more information, see "The use case scenario created with the MagicDraw 17.0.1 or earlier" on page 1026.

Copyright © 1998-2015 No Magic, Inc.

The following figure illustrates an example of the **Use Case** Specification window after the use case description profile is loaded.

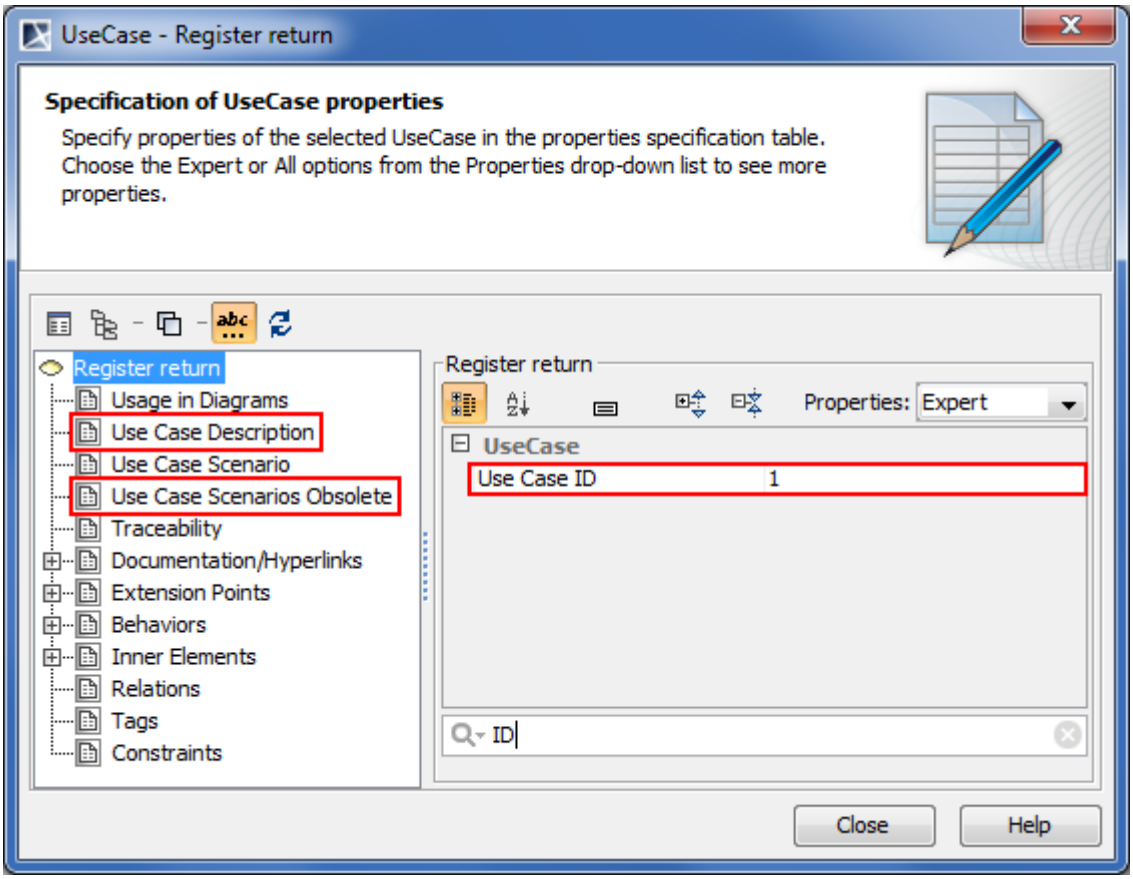


Figure 613 -- Use Case Specification window with Use Case Description Profile loaded

You can specify the use case extension properties in the use case Specification window. In the same window, you can find the description of each property. Descriptions are presented in the description area of the Specification window.

TIP!

- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

Related concepts

Use Case
Use Case Relations
Use Case Scenario

Related procedures

Formatting Symbols

Related references

Specification Window

Related resources

“Working with UML Profiles” in MagicDraw UMLProfiling&DSL UserGuide.pdf

Copyright © 1998-2015 No Magic, Inc.

Use Case Scenario

NOTE

This functionality is available in Standard, Professional, Architect, and Enterprise editions.

With the help of the use case scenario editor, you can do the following:

- Create, review, and edit steps of the use case scenario by using the convenient textual values editor.
- Automatically create an activity diagram representing the textual use case scenario flow. This gives you the possibility to have the textual numbered action flow and its graphical representation.
- Create the use case scenario from the activity diagram for the particular use case and then automatically represent this action flow as textual information in the use case scenario.

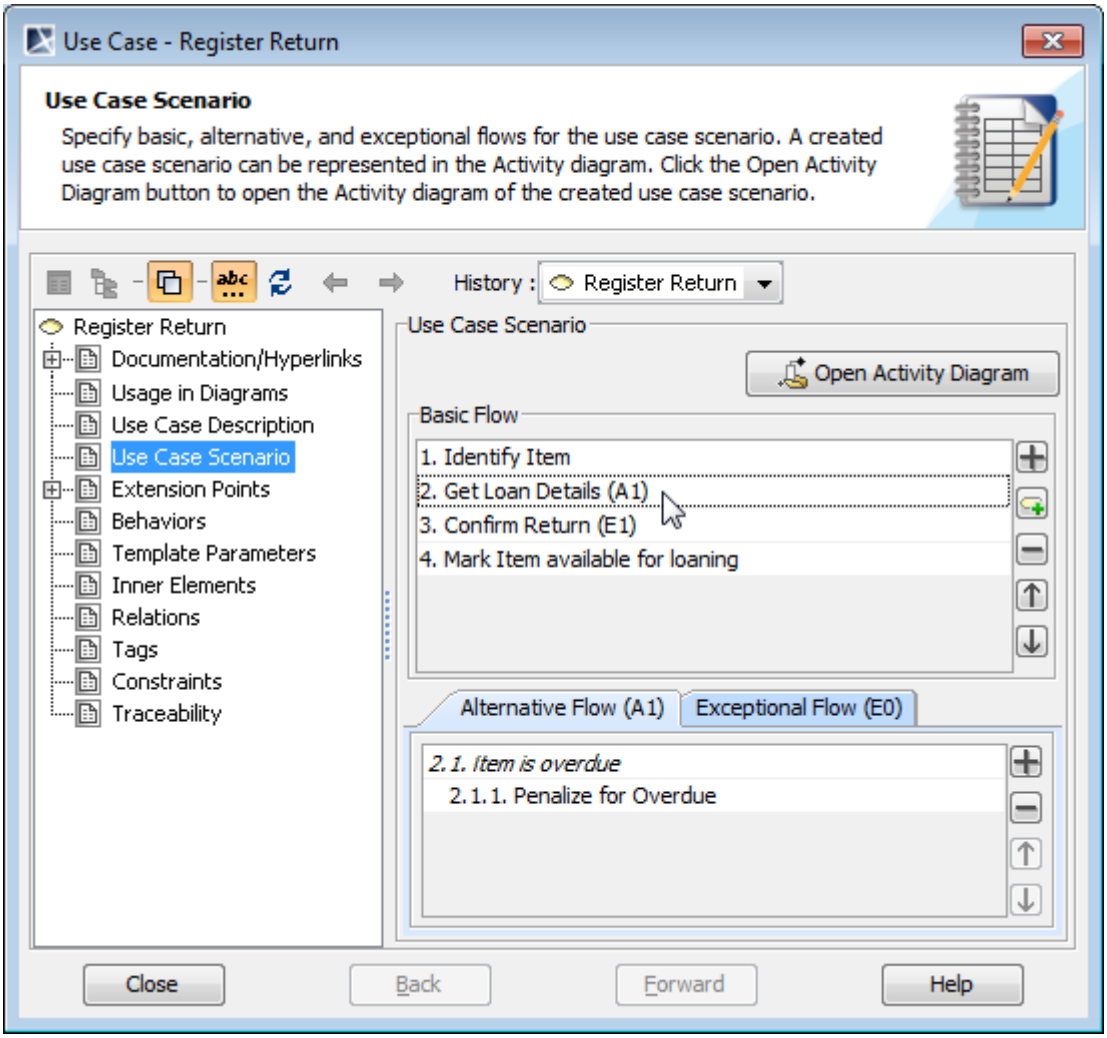


Figure 614 -- Example of use case scenario

Copyright © 1998-2015 No Magic, Inc.

The following figure depicts the same scenario represented in the activity diagram.

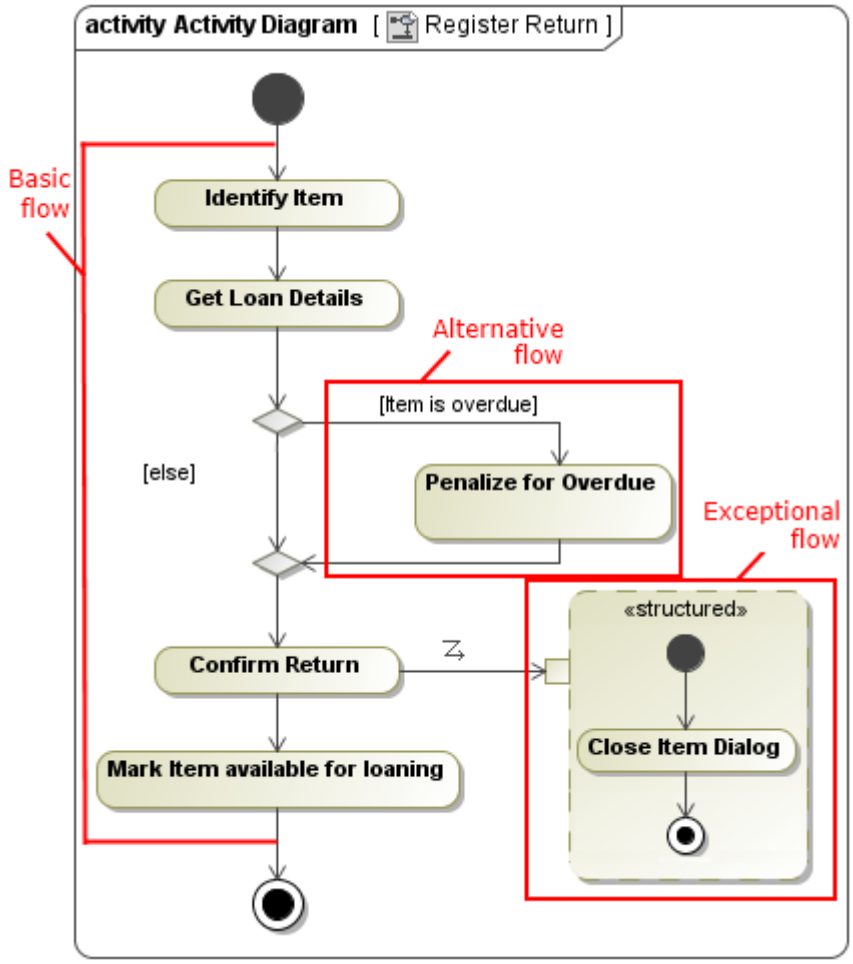


Figure 615 -- Example of activity diagram with basic, alternative, and exceptional flows

A use case scenario can have defined basic, alternative, and exceptional flows.

Basic flow

A basic flow represents the sequence of basic steps or actions of the use case scenario. You can also add steps from included or extended use cases to the flow. Each basic flow step can have one or more alternative and exceptional paths. Preceding figures (Figure 614 and Figure 615) depict the basic flow consisting of four steps. Letters and numbers in the brackets next to the basic flow step indicates how many alternative conditions and exceptional types the particular step has. Letter A indicates alternatives flows and letter E indicates exceptional flows.

In an activity diagram, the basic flow is the main path down from the initial node to the final node.



For more information about creating basic flow steps, refer to the procedure "To create a basic flow" on page 1013.

Alternative flow

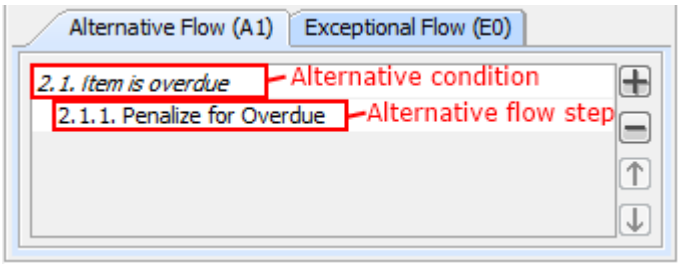
An alternative flow is an alternative path of the basic flow. You can define an alternative flow from the particular step of the basic flow. The alternative flow is an alternative solution that is performed after the defined condition is satisfied. The alternative flow contains steps that are executed if condition(s) occurs.

Copyright © 1998-2015 No Magic, Inc.

So, in the use case scenario, the alternative flow is specified by the two parameters - by the alternative condition and the alternative flow steps.

EXAMPLE

The following figure shows the content of the **Alternative Flow** tab of the Use Case Scenario in the use case Specification window.

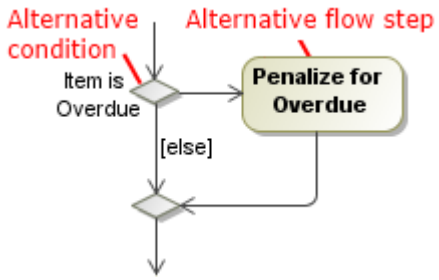


Here, the alternative flow is created for the second step of the basic flow — 2. *Get Loan Details*. The alternative flow has the condition named *2.1 Item is overdue*. The condition contains one alternative flow step named 2.1.1 *Penalize for overdue*.

In the activity diagram, the alternative flow is created between decision and merge nodes. The alternative condition is represented as decision node - that is, the name of the alternative condition in the activity diagram is the name of the decision node. In the activity diagram, the alternative flow step is are represented as call behavior action.

EXAMPLE

See the alternative flow representation in the activity diagram in the following figure.



In the activity diagram, you can read the alternative flow as follows: if the item is overdue, then - penalize for overdue; else, continue to the basic flow.

TIP!

- For more information about the use case scenario and the activity diagram mapping, see "A use case scenario and an activity diagram mapping schema" on page 1020.
- For more information about creating an alternative flow, refer to the procedure "To create an alternative flow" on page 1016.

Exceptional flow

An exceptional flow is an exceptional path from the particular basic flow step, or it can be a quick solution for exit. The exceptional flow contains steps that are executed if something goes wrong, such as an input from the actor that the system cannot handle. An example can be if in the opened dialog user clicks the *Cancel* button.

In the use case scenario, the exceptional flow is specified by two parameters - by the exception type and the exceptional flow steps.

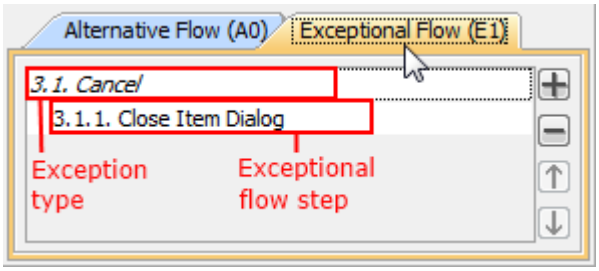
The type of the exception object indicates the nature of the exception. As a type of the exception can be used any class.

Copyright © 1998-2015 No Magic, Inc.

The exceptional flow steps are performed as actions when the execution occurs. The exceptional flow steps are specified for the concrete type. Normally, there is one exceptional flow (or in other words one exception type) assigned for one basic flow step. There can be any number of exceptional flow steps specified for one exception type.

EXAMPLE

The following figure shows the content of the **Exceptional Flow** tab of the Use Case Scenario in the use case Specification window.



Here, the exception flow is created for the third step of the basic flow — 3. *Confirm Return*. The exceptional flow has the type named *3.1 Cancel*. That is the *Cancel* class is assigned as the exception type.The type contains one exceptional flow step named 3.1.1 *Close Item Dialog*.

In the activity diagram, the exceptional flow is connected using the exception handler relationship. The exception handler has the type specified - the same as is the exception type, in our sample the *Cancel* class. Exceptional flow steps in activity diagram are represented as call behavior actions.

TIP!

- For more information about the use case scenario and the activity diagram mapping, see "A use case scenario and an activity diagram mapping schema" on page 1020.
- For more information about creating an alternative flow, refer to the procedure "To create an alternative flow" on page 1016.

Related concepts

Use Case

Include

Extend

Related procedures

Creating and editing use case scenarios

Representing use case scenarios in activity diagrams

The use case scenario created with the MagicDraw 17.0.1 or earlier

Related references

Specification Window

Related resources

“Working with use case scenario in MagicDraw Open API” in MagicDraw OpenAPI UserGuide.pdf.

”Generating a report for the use case scenario” in MagicDraw ReportWizard UserGuide.pdf.

Creating and editing use case scenarios

You can create a use case scenario in the Use Case Specification window using the **Use Case Scenario** property group wherein you can specify the basic, alternative, and exceptional flows, as well as to open the activity diagram. The following procedures describe how:

- To edit a use case scenario
- To create a basic flow

Copyright © 1998-2015 No Magic, Inc.

- To add a basic flow step from other use cases
- To create an alternative flow
- To create an exceptional flow

NOTE

- You cannot create or edit the use case scenario for use cases having the read-only accessibility.
- If you are working with a server project, make sure you have the permission to edit the project and lock elements before editing them.For more information about locking elements, see "Locking model elements and diagrams for editing" on page 1055.

TIP!

For more information on how to represent the created use case scenario in the activity diagram, see "Representing use case scenarios in activity diagrams" on page 1024.

To edit a use case scenario

1. Select a use case and open its Specification window.
2. Click the **Use Case Scenario** property group.

To create a basic flow

1. Open the use case scenario for editing (see the preceding procedure).
2. In the **Basic Flow** area, click the Add new step button or press Alt+ENTER and type a name of the basic flow step. Repeat this action for each basic flow step you need to create.

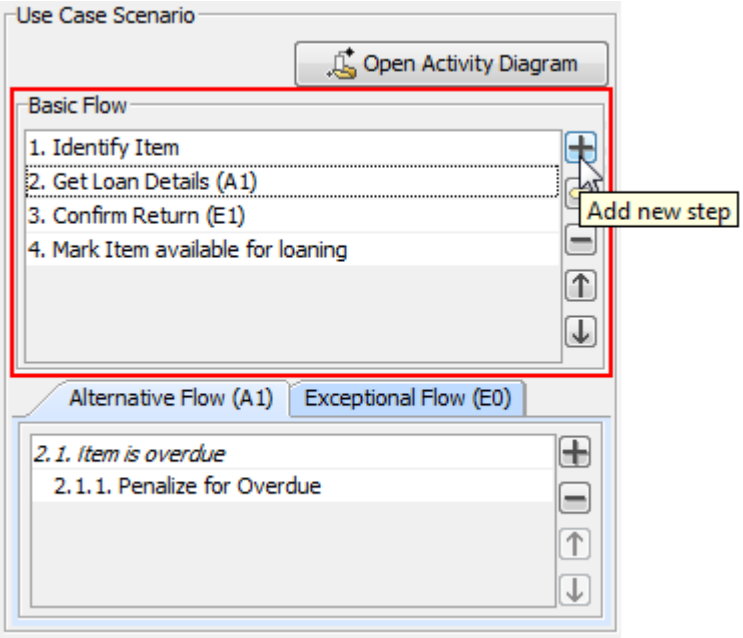


Figure 616 -- Adding new step of basic flow.

Steps of the basic flow are numbered automatically. Use Up and Down buttons to change the order of steps. In the activity diagram, steps are represented according to the ordering in the use case scenario.

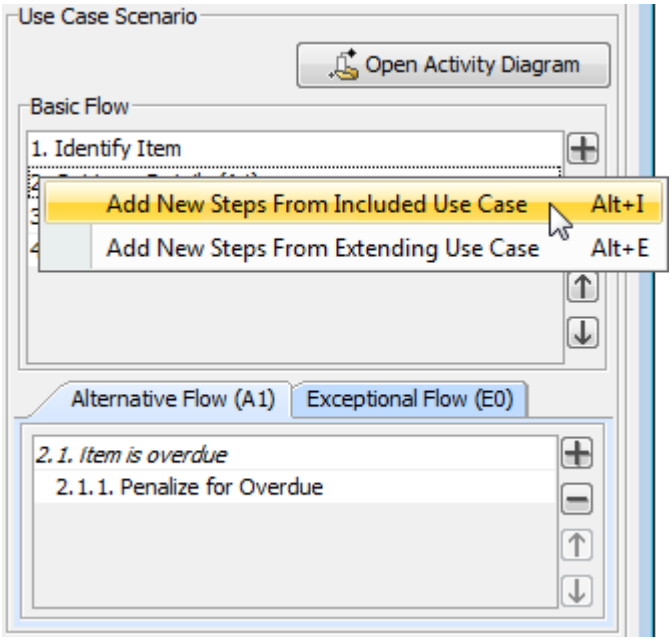
Copyright © 1998-2015 No Magic, Inc.

To add a basic flow step from other use cases

NOTE

Make sure the selected use case has included or extending use cases.

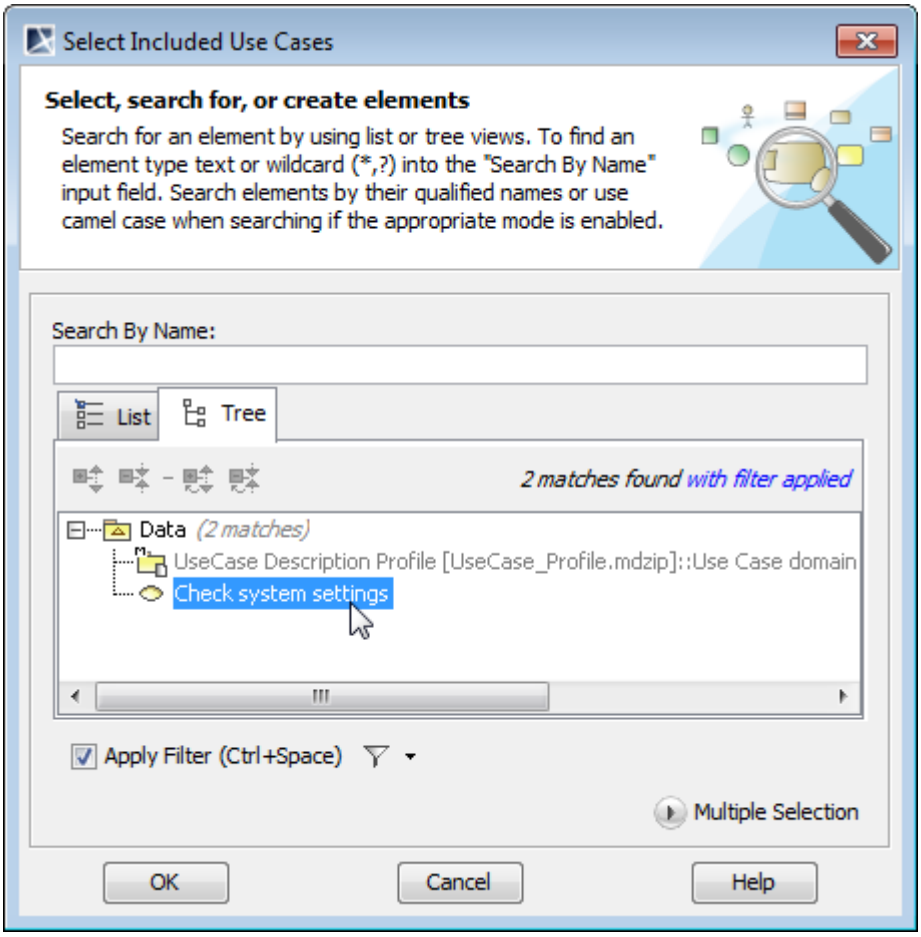
1. Open a use case scenario for editing. See the procedure "To edit a use case scenario" on page 1013.
2. Select a basic flow step for which you want to add steps from other use case.
3. Click the Add steps from other use cases button. The shortcut menu opens.



4. Do one of the following:
   - Click **Add New Steps From Included Use Case** or press ALT+I to add steps from the included use case.
   - Click **Add New Steps From Extending Use Case** or press ALT+E to add steps from the extending use case.

Copyright © 1998-2015 No Magic, Inc.

Depending on your choice, the **Select Included Use Case** or **Select Extending Use Case** dialog opens.



5. In the opened dialog, select the use case(s) from which you want to add steps to the selected step of the basic flow. Click **OK** after you have selected a use case.

IMPORTANT!

- Steps from included use cases are added to the basic flow as substeps of the newly created basic flow step.
- Steps from extending use cases having no extension points use cases are added to the basic flow as substeps of the selected basic flow step.
- Steps from extending use cases having extension points are added as steps of an alternative flow to the selected basic flow step. The extension point is considered the alternative condition.

See the examples in the following figures.

Copyright © 1998-2015 No Magic, Inc.

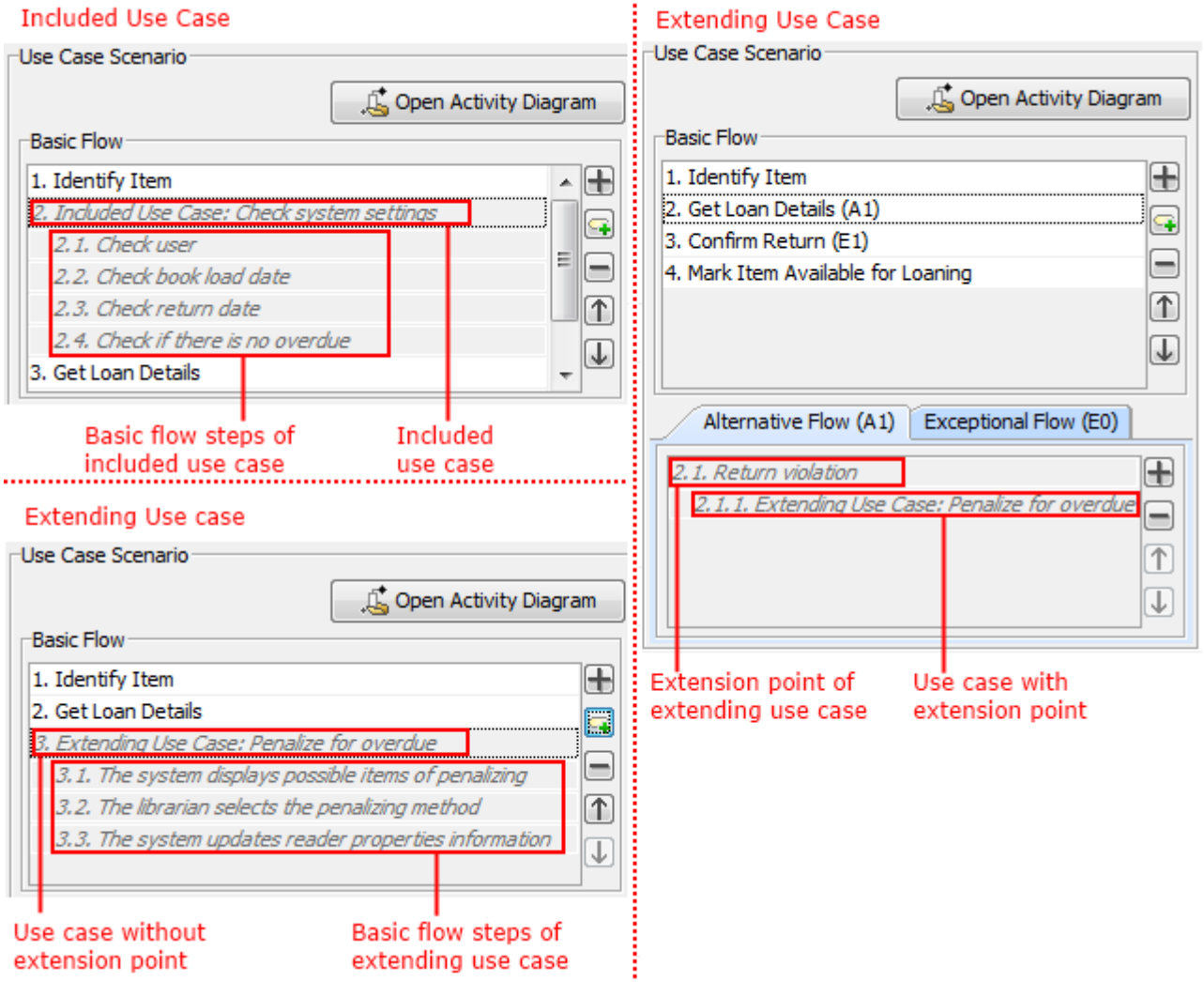


Figure 617 -- Steps added from included (in upper-left) and from extending use cases (in bottom-left and right)

IMPORTANT!

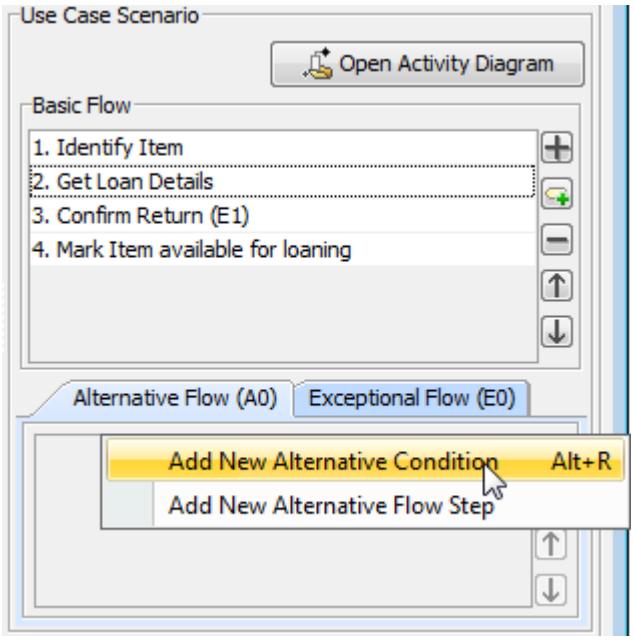
Steps added from included or extending use cases are read-only, and you are not allowed to modify them in the current scenario. You can modify added steps only in use cases from which these steps are added.

To create an alternative flow

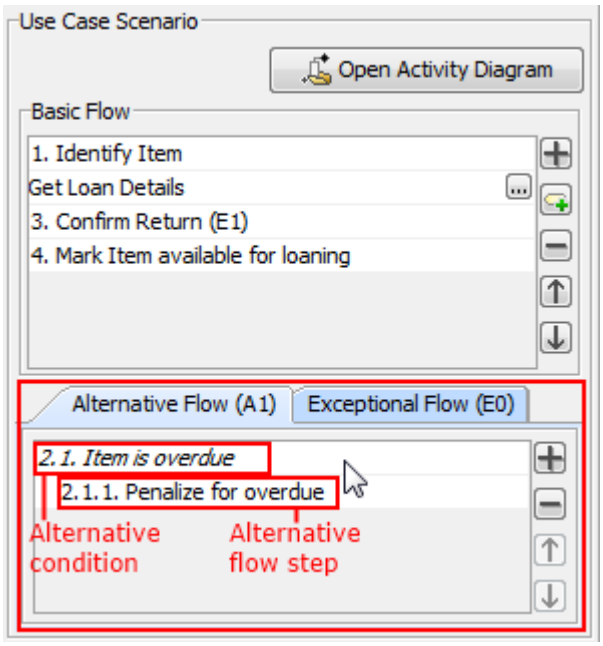
1. Open the use case scenario for editing. See the procedure "To edit a use case scenario" on page 1013.
2. Select the basic flow step for which you want to create an alternative flow.

Copyright © 1998-2015 No Magic, Inc.

3. In the **Alternative Flow** tab, click the Add new button. The shortcut menu opens.



4. Do one of the following:
   - Select the **Add New Alternative Condition** command or press ALT+R to add a new alternative condition. A new alternative condition is created together with the alternative flow step. Type the name of the alternative condition. The alternative condition for the basic flow is created. See an example in the following figure.
   - Select the **Add New Alternative Flow Step** command to add a new alternative flow step. If in the alternative flow there was an alternative condition created already, a new alternative flow step is created for the selected alternative condition. If in the alternative flow there was no an alternative condition, a new alternative flow step is created together with a new alternative condition. Now you can type the name of the new alternative flow step.

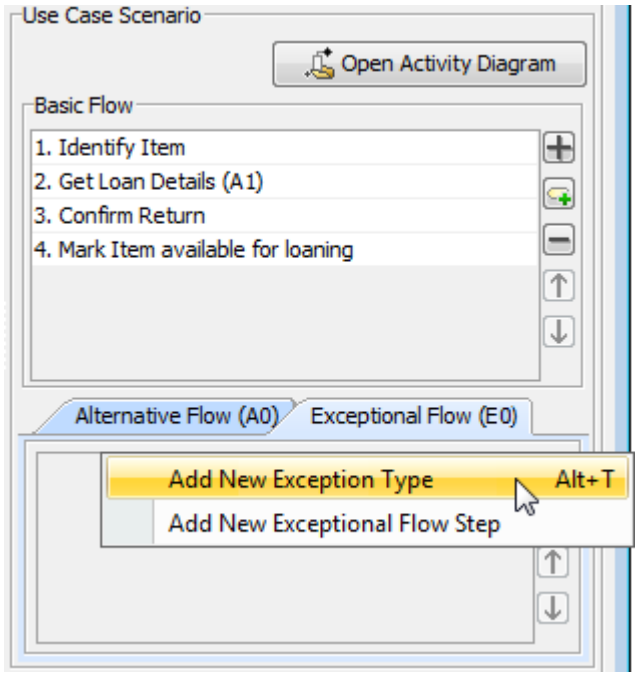


Conditions and steps of the alternative flow are numbered automatically. Use Up and Down buttons to change the order of conditions in the alternative flow or the order of steps in the particular condition. In the activity diagram, steps are represented according to the ordering in the use case scenario.

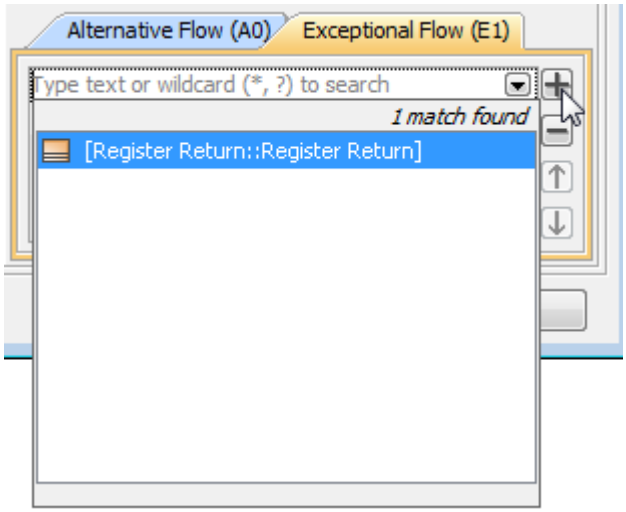
Copyright © 1998-2015 No Magic, Inc.

To create an exceptional flow

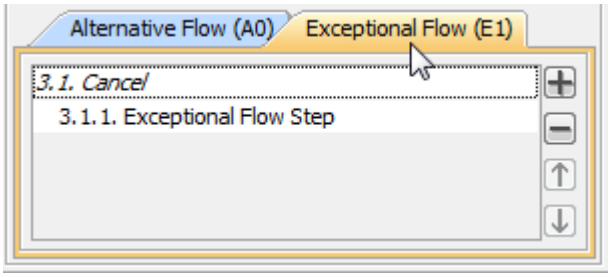
1. Open the use case scenario for editing. See the procedure "To edit a use case scenario" on page 1013.
2. In the use case scenario, select the basic flow step for which you want to create an exceptional flow.
3. In the **Exceptional Flow** tab, click the Add new button. The shortcut menu opens.



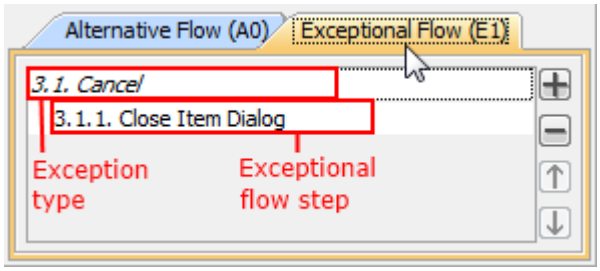
4. Do one of the following:
   - Select the **Add New Exception Type** command or press ALT+T. The drop down list opens. See an example in following figure. Type the name of a new exception type. The exception type is created, that is, a new class is created. For example, type *Cancel*. The created exception type contains not named exceptional flow step. See an example in the next figure.



Copyright © 1998-2015 No Magic, Inc.



- Select the **Add New Exceptional Flow Step** command. If in the exceptional flow there was an exceptional type created already, a new exceptional flow step is created for the selected exception type. If in the alternative flow there was no an exception type, a new exceptional flow step is created together with a new exception type. Now you can type the name of the new exceptional flow step. See an example in the following figure.



Types and steps of the exceptional flow are numbered automatically. Use Up and Down buttons to change the order of types in the exceptional flow or the order of steps in the particular type. In the activity diagram, steps are represented according to the ordering in the use case scenario

Related concepts

Use Case Scenario

Related procedures

Representing use case scenarios in activity diagrams

A use case scenario and an activity diagram mapping schema

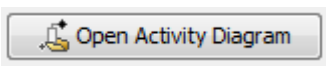
Related references

Specification Window

Representing use case scenarios in activity diagrams

After you have created a use case scenario, you can represent this scenario in an activity diagram. See the following procedure describing how to represented the use case scenario in the activity diagram and also how to turn on or off the automatic layout in the activity diagram when representing the use case scenario.

To represent a use case scenario in an activity diagram

1. Open the use case scenario for editing. See the procedure "To edit a use case scenario" on page 1013.
2. Click the **Open Activity Diagram** 

button. The activity diagram opens with the use case scenario represented on it.

Copyright © 1998-2015 No Magic, Inc.

You need to reopen the activity diagram every time after you made modifications to the use case scenario in the Specification window. Modifications to the activity diagram whereon the use case scenario is represented appears in the textual use case scenario automatically.



For use cases having the read-only accessibility, the use case scenario cannot be represented on the activity diagram.

In the activity diagram, all symbols are layed out automatically every time the diagram is opened.



To leave symbols in the same place while opening the activity diagram after modifying the use case scenario in the Specification window, change the **Layout use case scenario activity diagram** option value to *false*. You can find this option in the **Project Options** dialog, the **General project options** group.

Related concepts

Use Case Scenario

Related procedures

Creating and editing use case scenarios

Related dialogs

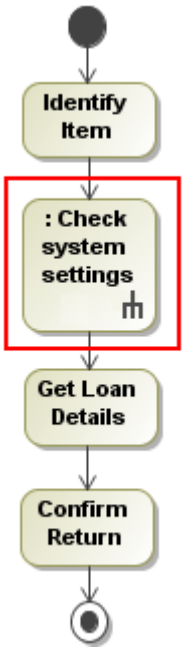
Specification Window

A use case scenario and an activity diagram mapping schema

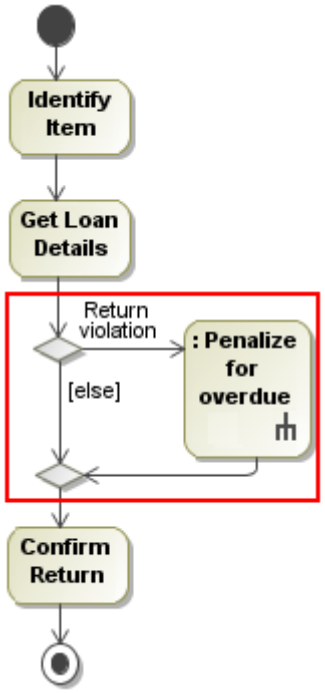
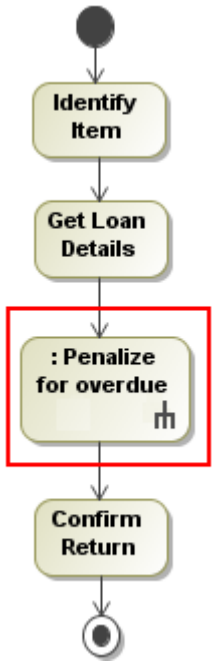
To understand the use case scenario representation in the activity diagram, see the following table.

Use Case Scenario	Mapping in Activity diagram	Example
One (the first) Basic flow step	1. The activity with the activity diagram is created under the owning use case just after the first basic flow step is created. 2. The name of the created activity and activity diagram are the same as the owning use case name. 3. In the activity diagram, the call behavior action is created for the basic flow step. 4. The initial node is created before the call behavior action. 5. The final node is created after the call behavior action. 6. The control flow relationships are created from the initial node to the call behavior action and from the call behavior action to the final node.	

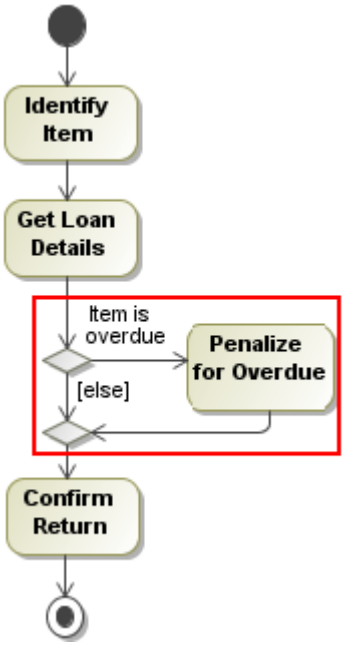
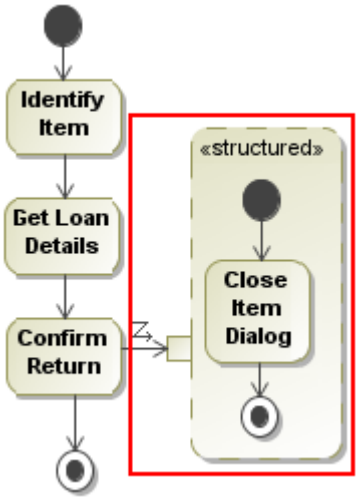
Copyright © 1998-2015 No Magic, Inc.

Use Case Scenario	Mapping in Activity diagram	Example
Basic flow steps	1. The call behavior action is created for each basic flow step. 2. The initial node is created before the first call behavior action. 3. The final node is created after the last call behavior action. 4. The control flow relationships are created from the initial node to the first call behavior action, between each call behavior action, and from the last behavior action to the final node.	
Included use case	1. The call behavior action is created for the basic flow step of the included use case. 2. The call behavior action is inserted to the activity basic flow according to the order as it was inserted in the basic flow. 3. The call behavior action is connected with the control flow relationships. The call behavior action is not named. 4. The call behavior action has the following behavior defined - the activity. This activity name corresponds the name of the included use case. 5. The activity (the behavior or the call behavior action) is owned by the included use case. 6. If the included use case has its own use case scenario, this scenario is represented in the activity - the activity diagram is created inside the activity and flows are represented. **IMPORTANT!** To represent the included use case from the activity diagram to the use case scenario, you have to follow all the rules, and in addition in your project you have to connect the including use case with the included use case with the included relationship.	

Copyright © 1998-2015 No Magic, Inc.

Use Case Scenario	Mapping in Activity diagram	Example
Extending use case (with the extension point)	1. The extending use case with the extension point is added as an alternative flow - the decision and merge nodes are created. 2. The decision node name corresponds the name of the alternative condition. 3. The call behavior action is created for the alternative flow step of the extending use case. The call behavior action is not named. 4. The call behavior action has the behavior defined - the activity which name corresponds the name of the extending use case. 5. The activity (the behavior or the call behavior action) is owned by the extending use case. 6. If the extending use case has its own use case scenario, this scenario is represented in the activity diagram - the activity diagram is created inside the activity under the extending use case. 7. The *[else]* guard property is defined for the control flow, that is created from the decision node to the merge node. **IMPORTANT!** To represent the extending use case from the activity diagram to the use case scenario, you have to follow all these rules described above, and in addition in your project you have to connect the extended use case with the extending use case with the extend relationship.	
Extending use case (without any extension point)	1. The call behavior action is created for the basic flow step of the use case scenario. 2. The call behavior action is connected with the control flow relationships according to the extending use case order in the basic flow. 3. The call behavior action is not named. 4. The call behavior action has the behavior defined - the activity which name corresponds the name of the extending use case. 5. The activity (the behavior or the call behavior action) is owned by the extending use case. 6. If the extending use case has its own use case scenario, this scenario is represented in the activity - the activity diagram is created inside the activity under the extending use case. **IMPORTANT!** To represent the extending use case from the activity diagram to the use case scenario, you have to follow all these rules described above and in addition in your project you have to connect the extended use case with the extending use case with the extend relationship.	

Copyright © 1998-2015 No Magic, Inc.

Use Case Scenario	Mapping in Activity diagram	Example
Alternative flow: alternative condition, alternative flow step	1. The alternative flow is interrupted in the basic flow by using the decision and merge nodes. 2. The decision and merge nodes are created after the call behavior action and the latter in the use case scenario represents the basic flow step of this alternative flow. In other words, in an activity diagram the elements of the alternative flow are created after the basic flow step to which this alternative flow belongs. 3. The decision node name corresponds the name of the alternative condition. 4. The call behavior action is created for each alternative flow step. 5. All these elements are connected with the control flow relationships. 6. The *[else]* guard property is defined for the control flow, that is created from the decision node to the merge node.	
Exceptional flow: exception type, exceptional flow step	1. The structured activity node with the input pin is created. 2. The call behavior action for which the exceptional flow was created is linked with the input pin by using the exception handler relationship. 3. The class element is created under the activity. The class corresponds the exception type in the use case scenario, that is, the class name corresponds the exception type name in the use case scenario. 4. The class is assigned to the input pin as a type property. 5. In the structured activity node the call behavior action is created for each exceptional flow step. 6. The call behavior actions inside the structured activity node are connected with the control flow relationships. 7. In the structured activity node the initial node is created before the first call behavior action, and the final node is created after the last call behavior action. 8. In the structured activity node the initial node is connected with the first call behavior action by using the control flow relationship and the last call behavior action is connected with the final node by using the control flow relationship.	

Related concepts

Use Case Scenario

Related procedures

Creating and editing use case scenarios

Representing use case scenarios in activity diagrams

Creating use case scenario from activity diagrams

Related dialogs

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Creating use case scenario from activity diagrams

An activity diagram owned by a use case can be transformed to the use case scenario in the textual format that is displayed in the use case Specification window. Changes done in the scenario on the activity diagram are synchronized with the same use case scenario represented in the use case Specification window. To transform the scenario from the diagram to the text, the scenario steps should be modeled under strict rules in the activity diagram. For more information about the use case scenario and activity diagram mapping schema, see "A use case scenario and an activity diagram mapping schema" on page 1020.

To create a use case scenario from an activity diagram

1. Under the use case, create an activity diagram. Do one of the following:
   - In the use case Specification window, click the **Inner Elements** property group. In the **Inner Elements** specification pane, click the **Create** button and select **Create Diagram** > **Activity Diagram** in the opened list. Name the activity diagram the same as the use case and click **Close**. The activity together with the activity diagram is created in the Model Browser.
   - Select the use case in the Model Browser. From its shortcut menu, select **Create Diagram** > **Activity Diagram**. Name the activity diagram the same as the use case.The activity diagram together with the activity is created.

   The activity diagram name should match the use case name. This is a mandatory requirement in order to represent the scenario from the activity diagram in the use case Specification window.
2. In the activity diagram, create the scenario flow for the use case. Elements should be created under the strict mapping rules in the activity diagram. For more information about mapping rules, see "A use case scenario and an activity diagram mapping schema" on page 1020.
3. After the activity diagram is created, open the Specification window of the use case owning the the created diagram and click the **Use Case Scenario** property group. The created scenario is represented in a textual form in the **Use Case Scenario** pane.

Copyright © 1998-2015 No Magic, Inc.

If a scenario is modeled incorrectly in the activity diagram, you will not be able to transform it to a textual form. In the Specification window, you will get a warning that use case scenario cannot be read with the list of possible reasons why this scenario cannot be read.

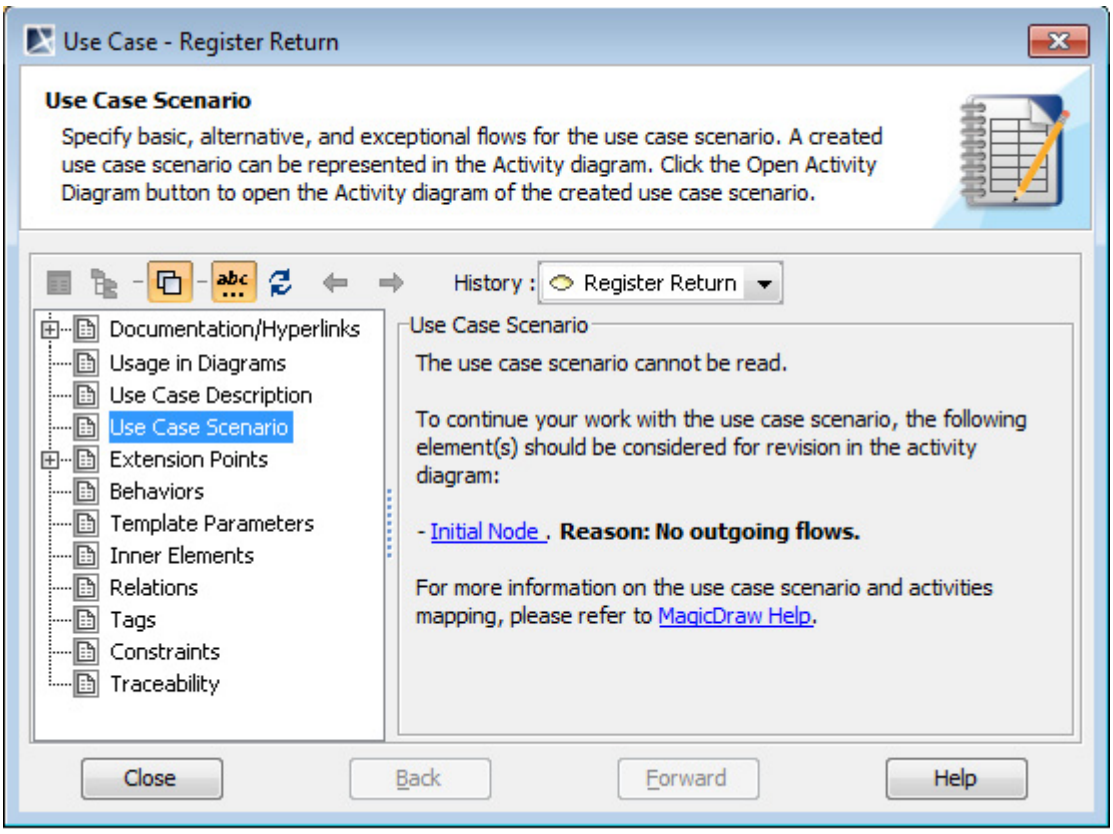


Figure 618 -- Example of not readable Use Case Scenario

In this example, you can see that there is listed the initial node element and the reason of incorrect modeling — *No outgoing flows*. According to the use case scenario and activity diagram mapping rules, it means that you have to create a control flow relationship from the initial node to the first call behavior action (the call behavior action that represents the first basic flow in the use case scenario). After you will solve this incorrect modeling in the activity diagram, you will be able to continue work with the use case scenario and the activity diagram.

Related concepts

Use Case Scenario

Related procedures

Creating and editing use case scenarios

Representing use case scenarios in activity diagrams

Related references

Specification Window

Copyright © 1998-2015 No Magic, Inc.

The use case scenario created with the MagicDraw 17.0.1 or earlier

As of MagicDraw version 17.0.2, use case scenarios of projects created with earlier MagicDraw versions are stored in the **Use Case Scenario Obsolete** property group, in the use case Specification window.

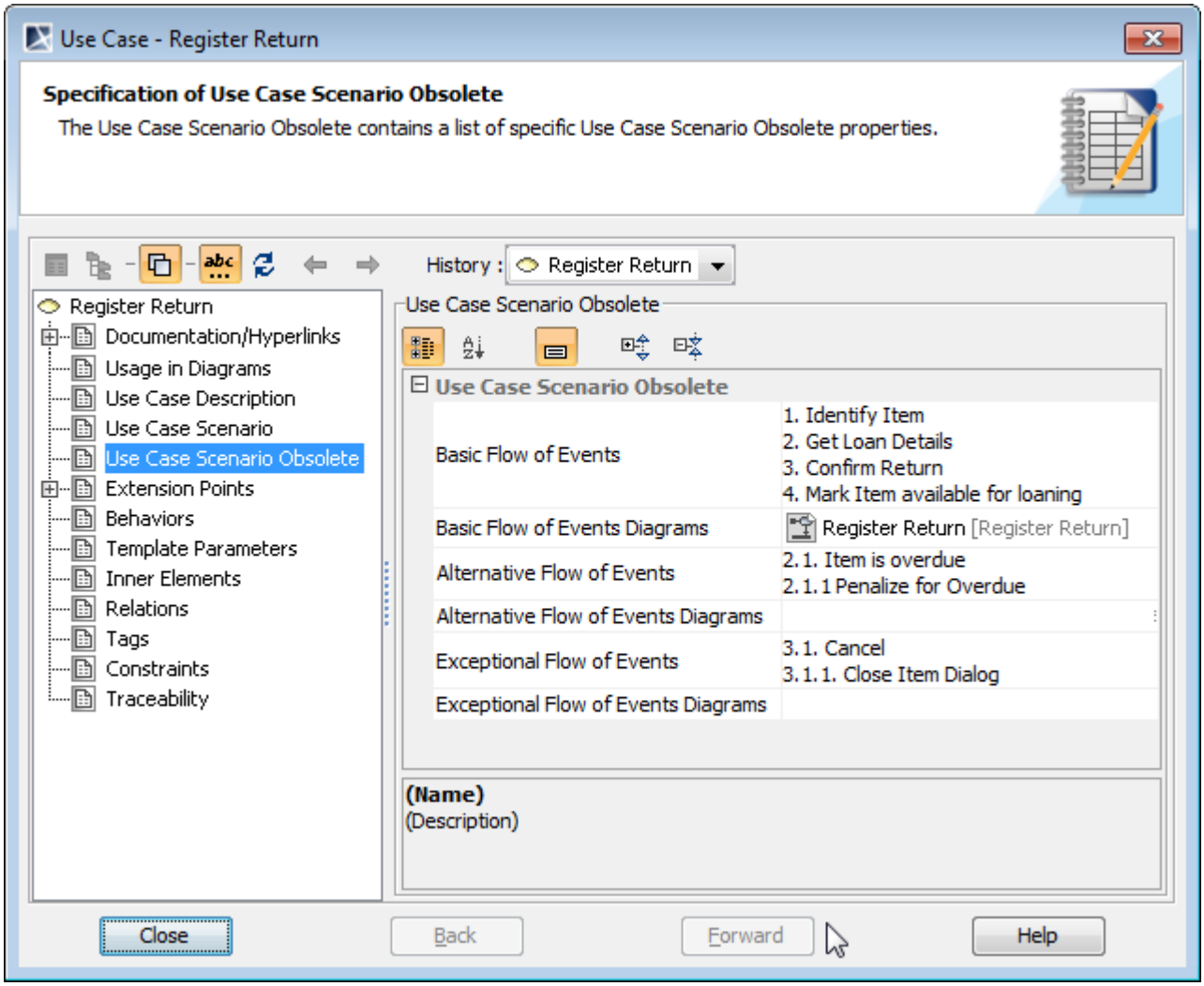


Figure 619 -- Example of use case scenario obsolete

IMPORTANT!

- To see the **Use Case Scenario Obsolete** property group, please, change the property display mode to **Expert**. For more information about property display mode, see "Property Group Toolbar" on page 278.
- The **Use Case Scenario Obsolete** property group is displayed only if the Use Case Description Profile is loaded. For more information about the use case description profile, see "Use Case Description Profile" on page 1006.

Related concepts

Use Case Scenario

Related procedures

Creating and editing use case scenarios

Related references

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Value Specification

You can create a standalone value specification in a model using the appropriate element shortcut menu.

You can use any of the following ways to create a value specification:

- Via the element's shortcut menu.
- Via the element's Specification window.
- By defining a default value to an element.

You can also change an assigned value specification after it has been created.

To create a value specification from the element's shortcut menu

1. Select an appropriate element in the Model Browser.

Copyright © 1998-2015 No Magic, Inc.

2. Right-click the element. On the shortcut menu point to **Create Element** > **Value Specification** and then select a suitable value specification.

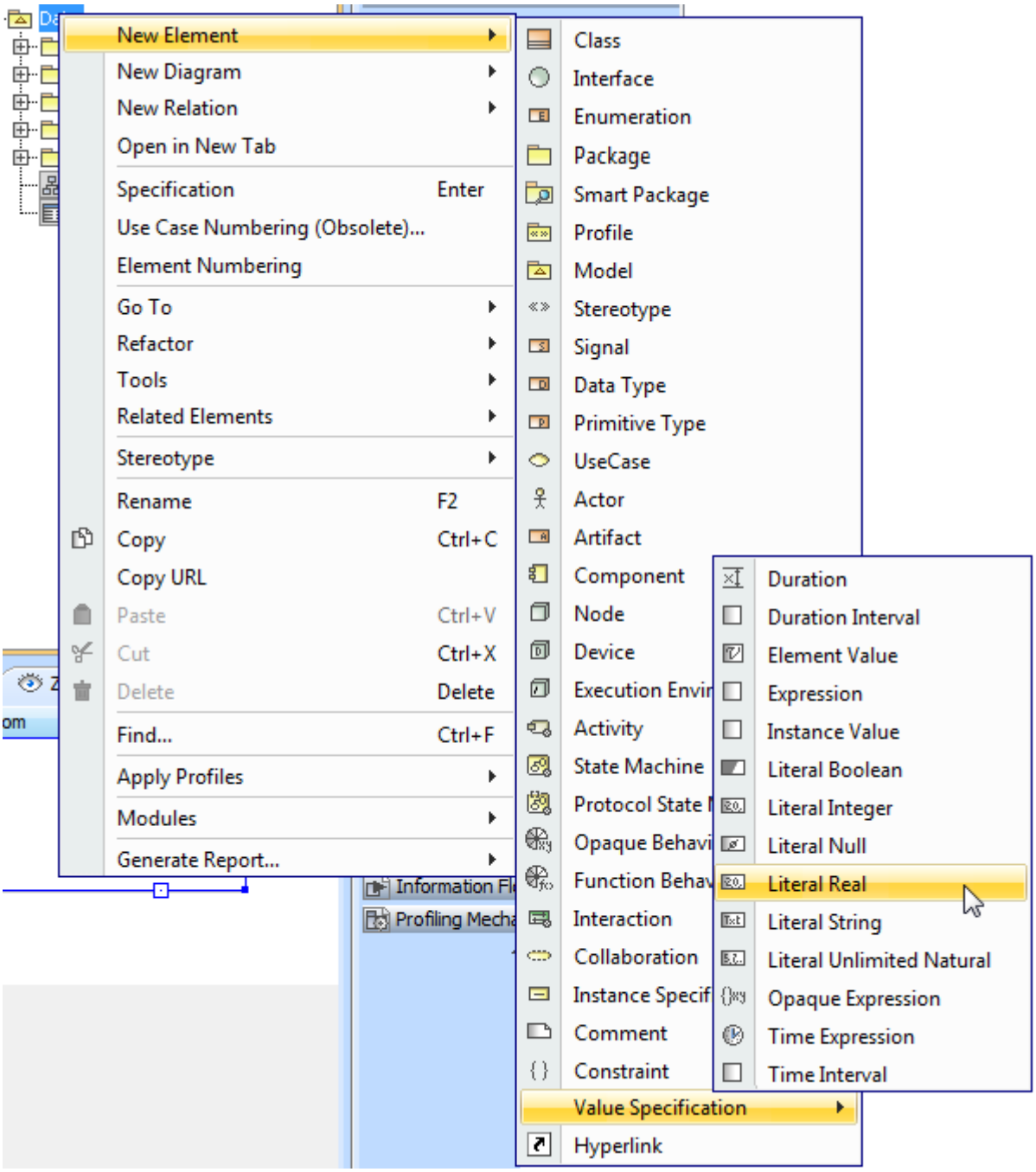


Figure 620 -- Creating value specification from element's shortcut menu

To assign a value specification in the element's Specification window

1. Open the Specification window of the selected element.
2. Click an appropriate property value cell.
3. Click the Show Shortcut Menu button (the black arrow).
4. On the shortcut menu, click **Value Specification** and then select a value specification.

To create a value specification automatically

1. Assign a default value to a property for which you want to create a value specification.

Copyright © 1998-2015 No Magic, Inc.

2. The value specification of the corresponding type will be assigned automatically according to the assigned default value in your model.

To change an assigned value specification

1. Open the Specification window of the selected element.
2. Click an appropriate property value cell.
3. Click the Show Shortcut Menu button (the black arrow).
4. From the shortcut menu, select **Value Specification** > **Delete** <value specification>.
5. Assign a new value specification.

Auxiliary Diagram Symbols

In this section, there are listed auxiliary diagram symbols, such as, simple text box, image shape, the whole diagram overview, and others. These symbols can be used in all MagicDraw diagrams.

You can create auxiliary diagram symbols from the **Common** category in the diagram pallet.

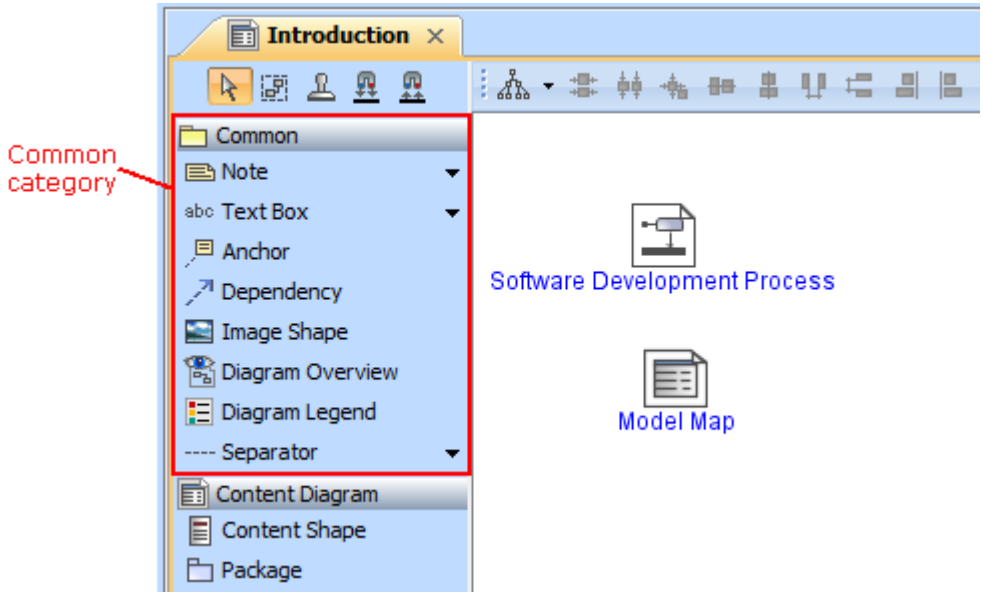


Figure 621 -- Diagram pallet, Common category

Refer to the following symbol descriptions:

- Image Shape
- Diagram Overview
- Separator
- Rectangular Shape
- Text Box
- Diagram Legend

Copyright © 1998-2015 No Magic, Inc.

Image Shape

The image shape provides a simple and quick way to insert a picture into a diagram. This can be logo, graph, table, or other images. The preferred shape size after the insertion is the actual image size. See an example of image shape in the following figure.

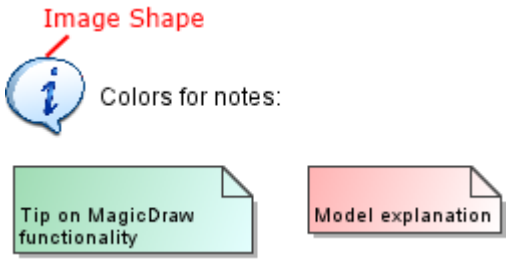


Figure 622 -- Example of Image Shape

You can format the image shape representation properties in the **Symbol Properties** dialog.

TIP!

For more information about symbol representation properties, see "Formatting Symbols" on page 317.

To insert an image on the diagram pane

1. On the diagram pallet, the **Common** category, click the **Image Shape** button or press the I key.
2. Click on the diagram pane. The **Open** dialog opens.
3. Select an image in the *.gif, *.jpg, *.jpeg, *.svg, or *.png format and click **Open**.

Related procedures

Formatting Symbols

Diagram Overview

NOTE

The diagram overview functionality is available in Standard, Professional, Architect, and Enterprise editions.

A diagram overview shape provides ability to overview other diagrams on a diagram pane. Moreover, using diagram overview shape you can overview dependency matrices, tables, and relation maps on a diagram pane

Copyright © 1998-2015 No Magic, Inc.

.

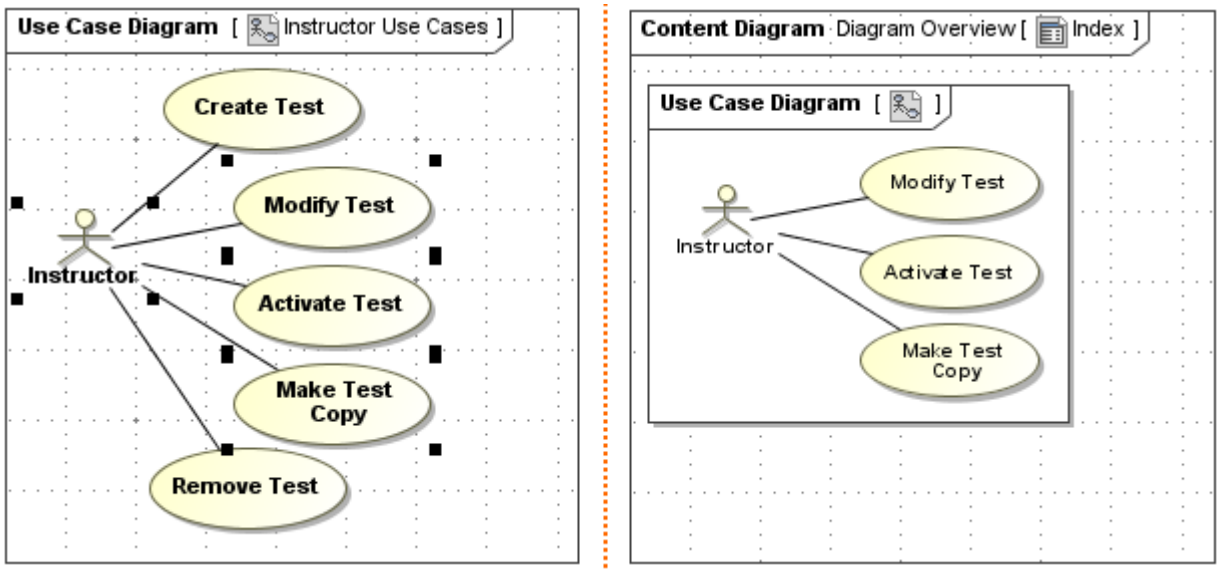


Figure 623 -- Example of diagram and its diagram overview shape

TIP! For more information about the diagram overview shape, see "Overviewing Other Diagrams" on page 210.

You can format the diagram overview representation properties in the **Symbol Properties** dialog.

TIP! For more information about symbol representation properties, see "Formatting Symbols" on page 317.

Double click on the diagram overview shape, to open the particular diagram. Specification window. In the same window, you can find the description of each diagram property. Descriptions are presented in the description area of the Specification window.

TIP!
- For more information about the Specification window usage, see "Specification Window" on page 273.
- For more information about specifying property values, see "Editing Property Values" on page 298.

To create a diagram overview shape from the diagram pallet

1. On the diagram pallet, the **Common** category, click the  **Diagram Overview** button. The **Select Diagram** dialog opens (to learn more about using this dialog, please see "Selecting an Element" on page 352).
2. Select a diagram for which you want to create a diagram overview shape and click **OK**.
3. Click on the diagram pane. The diagram overview shape is created and it shows the full content of the corresponding diagram.

Related procedures

Overviewing Other Diagrams

Formatting Symbols

Related dialogs

Specification Window

Copyright © 1998-2015 No Magic, Inc.

Related references

A diagram overview sample project in *<MagicDraw installation directory>\samples\product features\diagram overview.mdzip*.

Separator

You can use a horizontal or vertical separator to rule off different parts of a diagram. The Rotate button allows for changing the separator from the horizontal to vertical position, and vice versa. See the separator examples in the following figure.

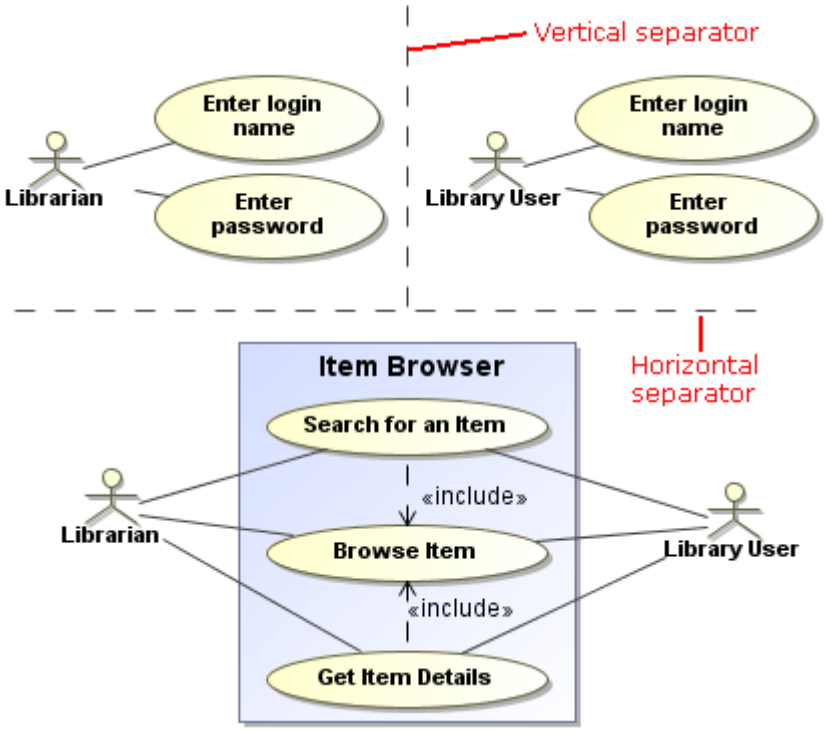


Figure 624 -- Using vertical and horizontal separators

You can format the separator symbol properties in the **Symbol Properties** dialog.

TIP! For more information about symbol representation properties, see "Formatting Symbols" on page 317.

A text next to the separator can be displayed as a plain text or as an HTML text. You can edit the HTML text using the HTML editor.

TIP! For more information about working with HTML editor, see "HTML Editor" on page 422.

To change the separator's position

1. Select the separator.
2. Click the *Rotate* smart manipulator.

To set the text position on a separator

1. From the separator shortcut menu, select **Symbol Properties**.
2. In the **Symbol Properties** dialog, change the **Text Horizontal Position** property to one of the following:
   - **Left** (default)

Copyright © 1998-2015 No Magic, Inc.

- **Center**
- **Right**

To set a separator line style

1. From the separator shortcut menu, select **Line style.**
2. Select the style you need
   - **Dashed** (default)
   - **Solid**
   - **Dotted**

Related procedures

Formatting Symbols

HTML Editor

Rectangular Shape

You can use a rectangular shape to delineate different parts of a diagram. See an example of the rectangular shape in the following figure.

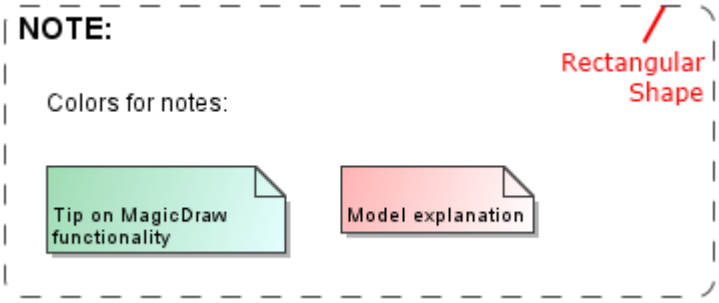


Figure 625 -- Example of Rectangular Shape

You can format the rectangular shape representation properties in the **Symbol Properties** dialog.

TIP!

For more information about symbol representation properties, see "Formatting Symbols" on page 317.

A text next to the separator can be displayed as a plain text or as an HTML text. You can edit the HTML text using the HTML editor.

TIP!

For more information about working with HTML editor, see "HTML Editor" on page 422.

To set a rectangular shape line style

1. From the rectangular shape shortcut menu, select **Symbol Properties**.
2. In the **Symbol Properties** dialog, change the **Line Style** property to one of the following: **Dashed** (default), **Solid**, or **Dotted**.

Related procedures

Formatting Symbols

HTML Editor

Copyright © 1998-2015 No Magic, Inc.

Text Box

A text box is a box wherein you can type any text.

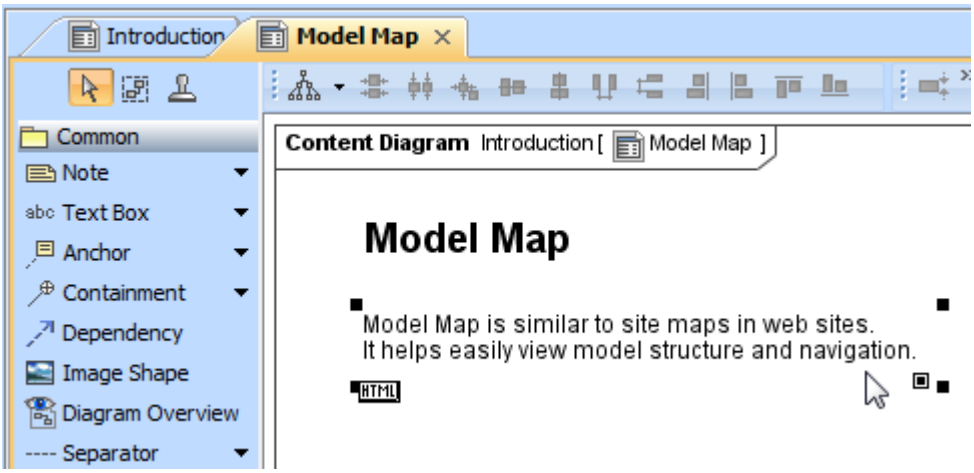


Figure 626 -- Example of text box

A text in the text box can be displayed as a plain text or as an HTML text.

You can format the text box symbol properties in the **Symbol Properties** dialog.

TIP! For more information about symbol representation properties, see "Formatting Symbols" on page 317.

To switch the text of the text box to HTML or a Plain text

Do one of the following:

- On the diagram pane, select the text box and click the button that appears on the lower-left corner of the shape:
  - The **HTML** button - to switch the text to the HTML text.
  - The **Plain** button - to switch the text to the plain text.



- Open the selected text's shortcut menu and select or click to clear the **HTML Text** command.

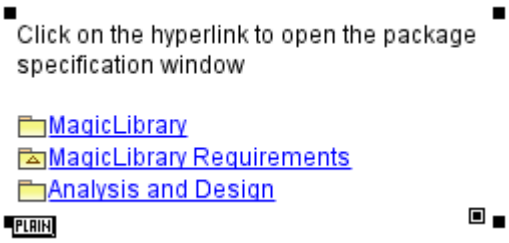
You can edit the HTML text using the HTML editor.

TIP! For more information about working with HTML editor, see "HTML Editor" on page 422.

To add a hyperlink to an element on a text box

1. Select one or more elements in the Model Browser, or Specification window.
2. Drag the selected elements on the text box on the diagram pane. The hyperlinks to specifications of dragged elements are created on the text box. Click the link on the text box to open the Specification window of the selected element.

Copyright © 1998-2015 No Magic, Inc.



NOTE After the hyperlink to the element on the text box is added, the text format in the text box changes to HTML.

Related procedures

Formatting Symbols
HTML Editor

Diagram Legend

The Diagram Legend is designed to define different styles for diagram symbols. It allows for grouping symbols visually on a diagram according to purposes or other defined criteria. Using this feature, you can define as many legend items as you need and apply them to symbols and paths on a diagram. In the project, the diagram legend is concerned as a model element. It can be edited in typical ways as any other element in the project.

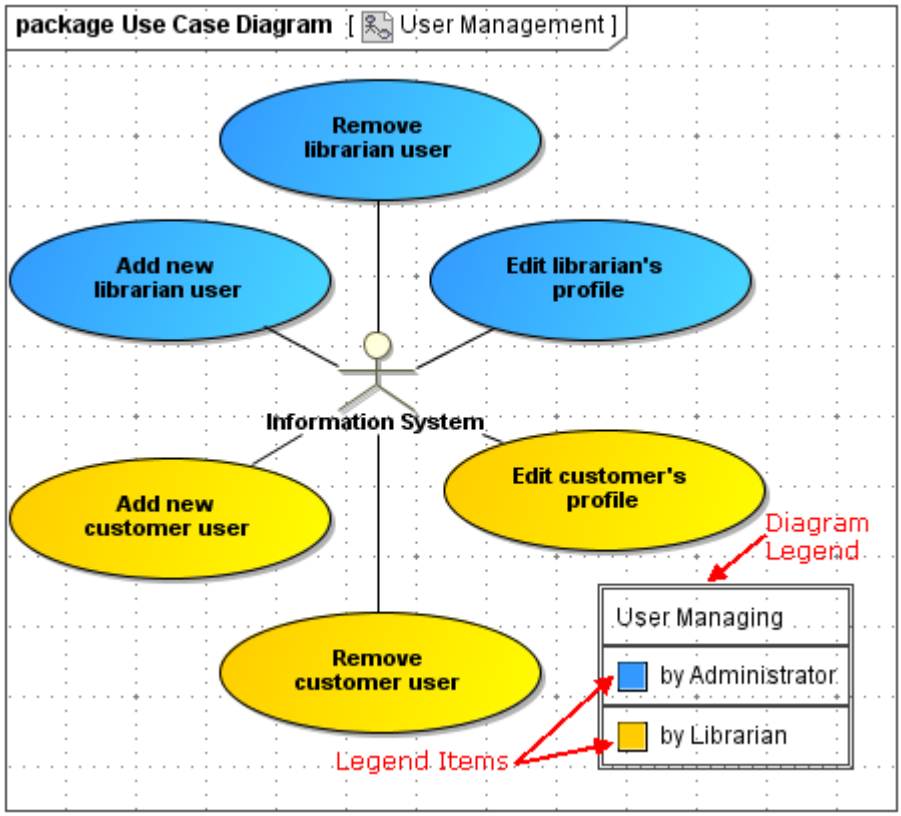


Figure 627 -- Diagram example with applied diagram legend

To create a diagram legend

1. On the diagram pallet **Common** category, click **Diagram Legend**. The **Select/Create Diagram Legend** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

2. In the dialog, select a new owner for the diagram legend and click the **Create** button.



Make sure, the creation mode is turned on.

3. In the opened **Diagram Legend** Specification window, type the diagram legend name and define **Legend Items**. You can define as many legend items as you need. Each legend item property value is a separate model element and is created as an inner element of the diagram legend.



For more information about defining inner elements as property values, see "Creating inner element as property value" on page 311.

4. After the diagram legend and its items have been defined, close the Specification window.
5. Click **OK**.
6. Click the diagram to paste the created legend on it.

For each legend item, you can define which item properties should be applied for the selected symbol. It means, you can specify all legend item properties and select which of properties to apply for the selected element. This possibility is useful when you want to apply the particular legend item except several properties which, for example, should be left default. To do this, you do not need to create a new legend item, but just select which legend item property values to apply.The currently existing property values applied for the element symbol (for example, it can be a color or font style of a symbol) will be left instead of those properties you did not select.

To choose which legend item properties to apply

1. Create a diagram legend and its items or select an existing one.
2. In the legend's Specification window or in the Model Browser, select the item you want to edit and open its Specification window.
3. Turn on the Expert properties display mode. In the **Properties** list on the Legend Item property toolbar, select **Expert**. The **Apply Property** column appears in the properties Specification pane.

Copyright © 1998-2015 No Magic, Inc.

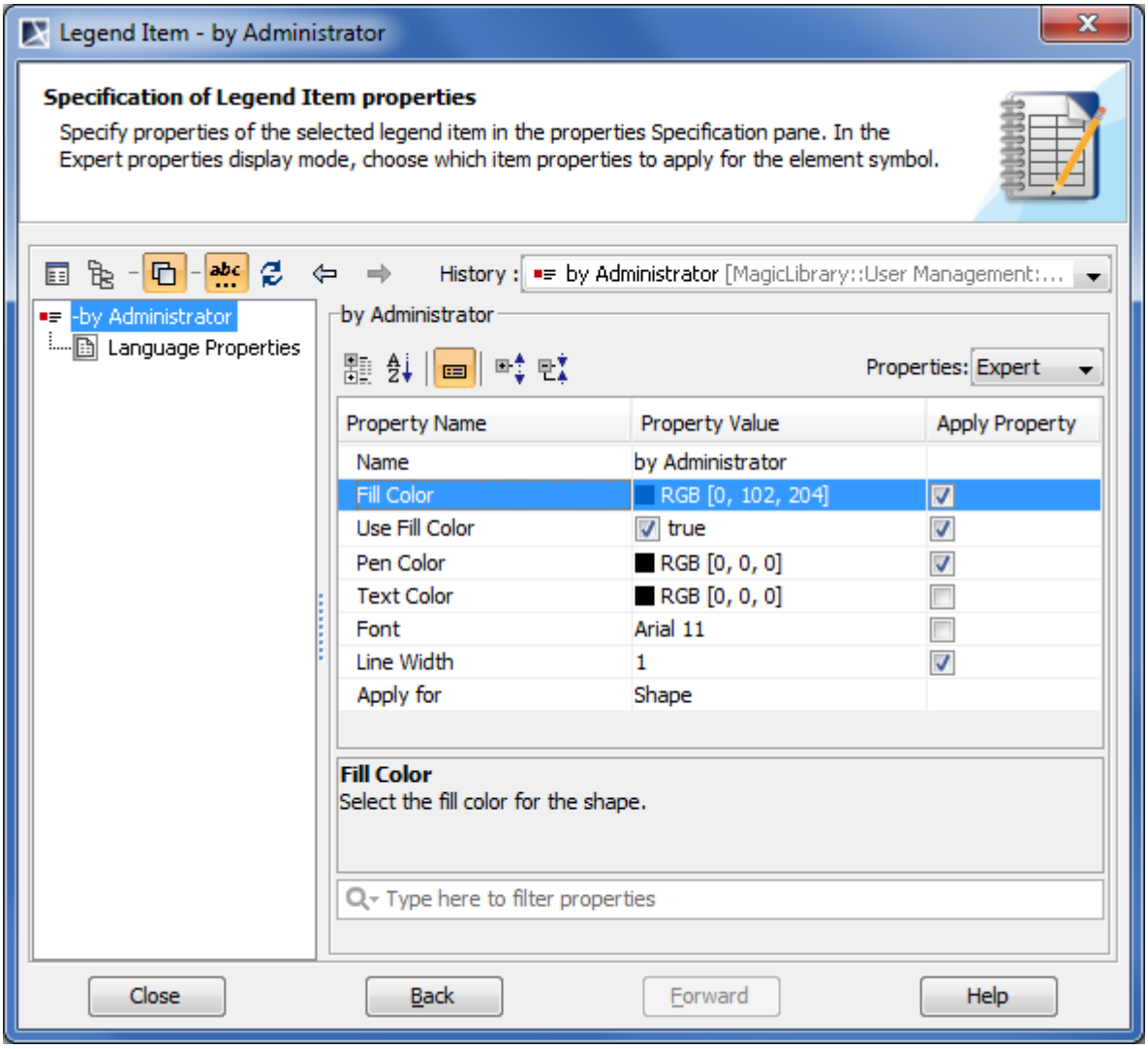


Figure 628 -- Example of Legend Item Specification window

4. In the **Apply Properties** column, select properties you want to apply for the element symbol.
5. Click **Close** when you are done.

After the diagram legend and its items are defined, you can apply the selected legend item for one or more selected element symbols on a diagram.

To apply a legend item

1. On a diagram, select one or more elements for which you want to apply the legend item.
2. On its shortcut menu, click **Use Diagram Legend**, then click the name of a diagram legend you want to use, and then click the legend item name you want to apply.

NOTE

To restore the default style of a selected element symbol, on the selected element shortcut menu, click **Use Diagram Legend** > **Default style**.

Related procedures

Editing Property Values

Formatting Symbols

Copyright © 1998-2015 No Magic, Inc.

Related references

Specification Window

Selecting an Element

Copyright © 1998-2015 No Magic, Inc.

12WORKING WITH SERVER PROJECTS

Using servers you are able to work with projects collaboratively. Depending on a role in a team or company, users can do various actions with server projects and use projects according their needs. As depicted in the following figure, a server stores projects, projects history, users with their permissions.

You can easily create such permissions where stakeholders can access server projects for reviewing or other purpose, but they cannot modify any project. Engineers, like developers, quality assurance engineers, analysts, and other, can access, modify, create a new or remove any existing project, manage project changes and historical versions. Users with administrator rights have full access to manage projects and project user permissions, as well as perform all actions as engineers.

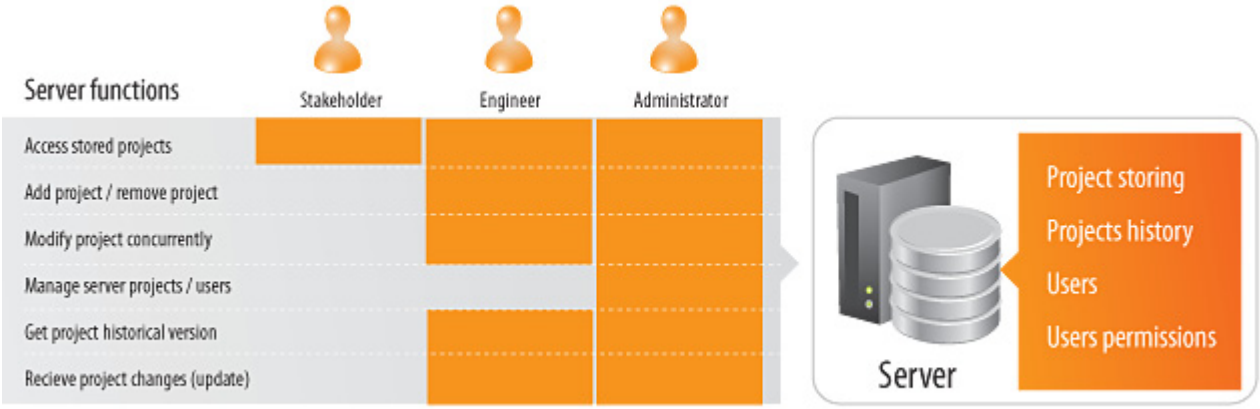


Figure 629 -- Server functions

Having a server you can assign as many team members as needed to work simultaneously on the same project using multiple workstations. The server project is stored on the server and can be accessed by team members through the network.

As long as you set up the permission for the users and project, we suggest working with it in the following order:

1. Log on to the server.
2. Open a server project.
3. Lock elements you want to edit to prevent other users could not change project elements you are working with.
4. Edit these elements or create new ones.
5. Unlock locked elements after editing.
6. Commit the project to the server that other users could see changes.

No Magic offers MagicDraw Teamwork Server for a collaborative work.

Read the following sections to learn about using MagicDraw in the collaborative environment.



For the MagicDraw Teamwork Server administration instructions, see MagicDraw Teamwork UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

Working in MagicDraw Teamwork Server

With MagicDraw Teamwork Server you can assign as many developers as needed to work simultaneously on the same project using multiple workstations. The resulting project is saved on the server for sharing with other MagicDraw applications. Users with administrator rights can create new users by creating a name and assigning various permissions to work on projects. The permissions assigned will determine whether the new user can update, commit, edit, create, and delete model elements, diagrams, and projects.

To enable Teamwork support, you should install and run MagicDraw Teamwork Server. Each MagicDraw application acts as a client of Teamwork Server.

At www.nomagic.com/support/demos, under the section **Project management an collaboration**, you will find the short videos demonstrating how to work with Teamwork Server.



The Teamwork Server functionality is available with MagicDraw Standard, Professional, Architect, and Enterprise editions only.

Teamwork Concepts

Get acquainted with basic Teamwork concepts before getting started to work in the collaborative environment.

Create server project

Create a new project that is stored in Teamwork Server and is available for other users with the appropriate permissions. For more information, see "Creating server projects" on page 1045.

Update project

Get the latest server project you are working on changes from the server. For more information, see "Updating server projects" on page 1071.

Lock elements for edit

Lock project elements to modify them. While an element is locked, no one user in the project except you cannot modify it. For more information, see "Locking model elements and diagrams for editing" on page 1055.

Unlock elements

Unlock model elements and/or diagrams, so that other users could have an ability to modify them. For more information, see "Unlocking model elements and diagrams after editing" on page 1065.

Commit project

Commit the project to the server to save your changes. After committing, a new project version will be created in the server. Additionally, it is also possible to unlock all elements locked by you or add a version comment while committing the project. For more information, see "Committing to Teamwork Server" on page 1069.

Update lock information

Refresh the list of elements that are locked by other users.

Author

A user who has committed a particular project version.

Version

A unique number assigned to the version of the project. Project version numbers begin at zero (for the initial version) and increase with every new project version.

Copyright © 1998-2015 No Magic, Inc.

Comment

An optional description about changes in the committed project version.

Version Tag

Information about the status of a project (approved, initially tested, and so on), or other important information.

Administrator Login

The default Administrator's account in Teamwork Server:

Login name: Administrator

Password: Administrator



For more information, see “Managing Teamwork” in MagicDraw Teamwork UserGuide.pdf.



To prevent illegal access, it is advisable to change the default Administrator's password.

Teamwork Server Administrator's Console

A remote connection for Teamwork Server status observation and administrative control. The server holds information about active users and loaded projects. The Administrator can shutdown or restart the server, change its properties, and view log files (including debug information) for the server and separate projects.

Repository

A storage place for projects and their versions that is managed by the MagicDraw Teamwork Server.

Project category

A concept which enables visual grouping of projects in the Teamwork Server repository.

Native user

A user whose account data is stored locally in the native Teamwork Server repository.

External user

A user whose account data (all except the login name) is stored in an external database, such as Subversion, ClearCase, or LDAP.

Server Module[12]

A server project containing one or more shared packages. Modules are created with a purpose to reuse them or to decompose projects into parts.

Working with Projects in Teamwork Server



Working in Teamwork

You must have certain permissions to work with server projects. The following are the most important steps when working with server projects:

12. Starting from version 18.1, referred as “used project” in MagicDraw UI. This section (including subsections) mentions the old keyword, which will be replaced in the documentation of the next MagicDraw version.

Copyright © 1998-2015 No Magic, Inc.

1. Create a server project or add a non-server project to the Teamwork Server. Open the server project.
2. Lock the elements you want to edit and edit them in the open project.
3. Select the model elements to unlock and commit the project.
4. Create categories for visual grouping of server projects.

If you want to lock an element for editing, the application will check for the permissions you have and the availability of the element.

Before committing elements, you can unlock the elements you have previously locked. Committing to Teamwork Server will create a new project version.

This chapter introduces the main procedures that can be useful while working with Teamwork Server:

- Starting collaboration session
- Creating and managing categories
- Creating server projects
- Opening recently used server projects
- Adding projects to server
- Creating server projects in IDE integration
- Migrating server projects to MagicDraw Teamwork Server 17.0.1 or later
- Locking model elements and diagrams for editing
- Locking project structure
- Unlocking model elements and diagrams after editing
- Committing to Teamwork Server
- Saving server projects locally
- Converting server projects to local
- Updating server projects
- Versioning
- Visibility of version tags
- Open Server Project dialog
- Code engineering in server projects
- Project Branching in Teamwork

Starting collaboration session

To log in to Teamwork Server

1. From the **Collaborate** menu, select **Login**. The **Login** dialog opens (see the following figure).
2. Enter your login name, password, Teamwork Server name, and the port number.
3. If you want the client application to remember your credentials in order to login next time to the server automatically upon starting the client application, select the **Auto login to server** check box.
4. If you need to connect to the server using the SSL connection select the **Use Secured Connection (SSL)** check box.



For more information about the configuration of SSL connection, see “Secured Connection tab” in MagicDraw Teamwork UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

IMPORTANT!

If the SSL connection is established in the server side, you should also use the SSL connection in the client side when connecting to the server.

5. Click **OK**.

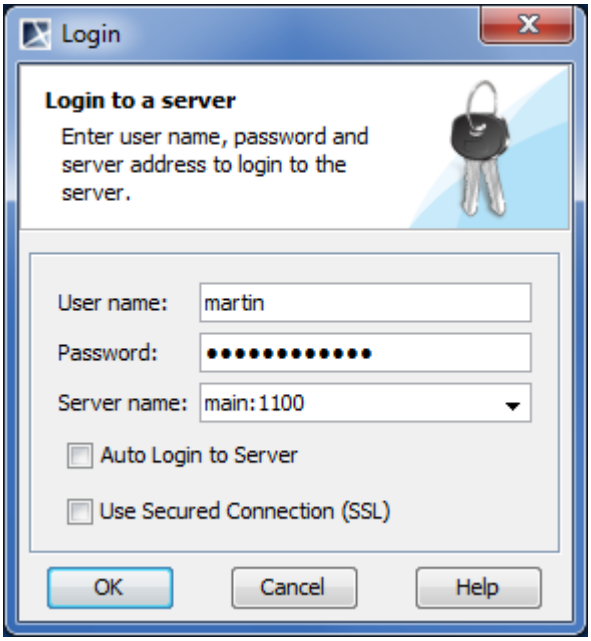


Figure 630 -- Login dialog

To change the password for the specific user

- From the **Collaborate** menu, select **Change Password**.

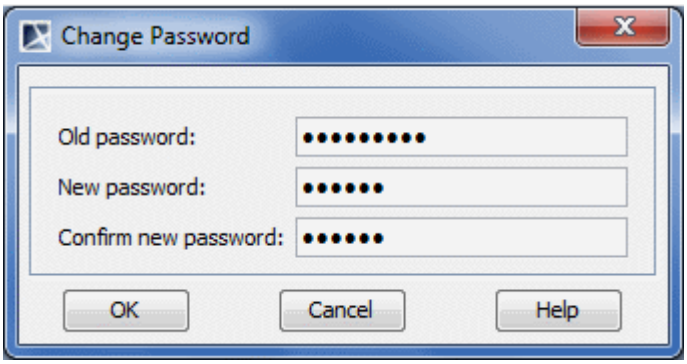


Figure 631 -- Change Password dialog

Element name	Function
Old password	Type the current password.
New password	Type the new password.
Confirm new password	Retype the new password to confirm it.

NOTE

Passwords cannot be changed if the Teamwork Server is configured with the SVN repository and pass-through authentication. In this case, use the appropriate native tools to change the password.

Copyright © 1998-2015 No Magic, Inc.

To log out from the Teamwork Server

- From the **Collaborate** menu, select **Logout**.

Creating and managing categories

Teamwork Server supports visual grouping of projects. You can build a structure of the whole project repository by creating categories and grouping server projects into these categories.

You can create an unlimited number of your own categories, rename or remove them, or move projects to them.

Server projects that do not belong to any created category will be automatically moved to the system category named *Uncategorized*.

To create a new category

1. From the **Collaborate** menu, choose **Projects**. The **Manage Projects** dialogs opens.
2. Do either:
   - On the toolbar of the open dialog, click the Create Category button. The **Create Category** dialog opens.
   - In the open dialog, right-click the project list area and from the shortcut menu, select **Create Category**.
3. Enter the new category name.
4. Click **OK**.

To rename a category



You cannot rename the category named *Uncategorized*.

1. From the **Collaborate** menu, choose **Projects**. The **Manage Projects** dialogs opens.
2. In the open dialog, select a category or a project that belongs to the category you want to rename and then do either:
   - Click the Rename Category button.
   - Right-click the selection and from the shortcut menu, select **Rename Category**.

   The **Rename Category** dialog opens.
3. Enter a new category name.
4. Click **OK**.

To move a project to another category



A project can belong to only one category at the same time.

1. From the **Collaborate** menu, choose **Projects**. The **Manage Projects** dialogs opens.
2. In the open dialog, select the project you want to move and do either:
   - Click the Move to Category button.
   - Right-click the project and from the shortcut menu, select **Move to Category**.

   The **Move to Category** dialog opens.
3. Select a category.
4. Click **OK**.

Copyright © 1998-2015 No Magic, Inc.

To remove the category



- You cannot remove the system category named *Uncategorized*.
- Projects from the removed category will be automatically moved to the system category.

1. From the **Collaborate** menu, choose **Projects**. The **Manage Projects** dialogs opens.
2. In the open dialog, select the category or the project that belongs to the category you want to remove and then do either:
   - Click the **Remove Category** button.
   - Right-click the selection and from the shortcut menu, select **Remove Category**.
3. Confirm your decision.

Related references

Manage Projects dialog

Creating server projects

This section provides the step-by-step instructions on how to create a new server project.

To create a new server project

1. From the **Collaborate** menu, choose **Projects**.
2. Click the **Add** button. The **Add Server Project** dialog opens.
3. Type a new project name.
4. Either select an existing category or type a new category name.
5. Click **OK**.

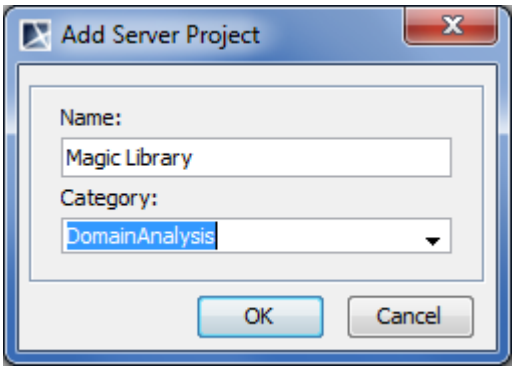


Figure 632 -- Example of Add Server Project dialog

Related references

Manage Projects dialog

Opening recently used server projects

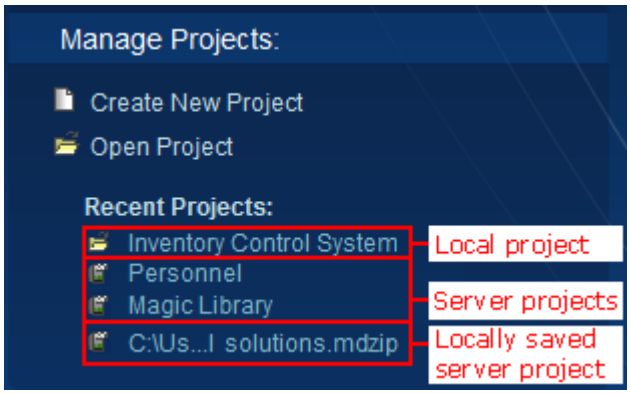
In order to enhance the usability of server projects, MagicDraw introduces a possibility to quickly open server projects, that were recently used. The latest project version will be opened. You can also define the list size of the recently used projects.

To open a recently used server project

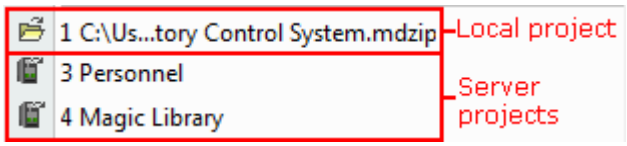
- Do one of the following:
  - From the **Collaboration** menu, select **Recent Teamwork Projects** and then select the projects you want to open.

Copyright © 1998-2015 No Magic, Inc.

- On the Welcome screen, under **Recent Projects**, click the project name.



- From the File menu, select the project name.



To manage the list of recent server projects

1. On the main menu, click **Options** and then select **Environment**. The **Environment Options** dialog opens.
2. In the **General** option list > **General** options group, set the **Recent Teamwork Projects List Size** option to the wanted number of projects.

Adding projects to server

You can add any local project to Teamwork Server with MagicDraw version 7.0 and later.

This section provides step-by-step instructions on how to add a project to Teamwork Server.



Add Project to Teamwork

To add a project to server



Project will be added to Teamwork Server together with all used local modules.

1. Log in to Teamwork Server.
2. Open the project you want to add to Teamwork Server.

Copyright © 1998-2015 No Magic, Inc.

3. From the **Collaborate** menu, choose **Add Project to Server**. The **Add Project to the Server** dialog opens.

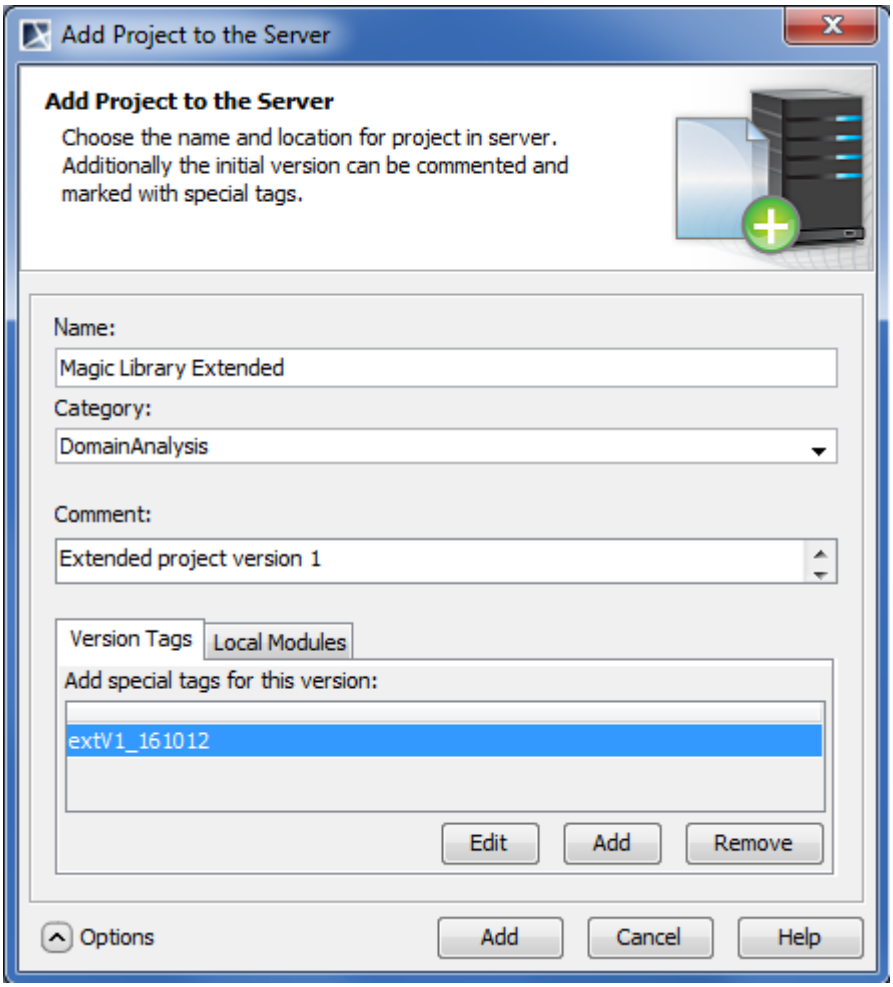


Figure 633 -- Example of Version Tags tab on Add Project to the Server dialog

4. Enter different name of the project for the Teamwork Server, if there is such need.
5. Either select an existing category or type a new category name.
6. Type your comments about the project.
7. In the **Version Tags** tab, add a tag, if needed.
8. The **Local Modules** tab lists all local modules used in the project (you can change module names on Server). From the **Action** drop-down list, select what do you want to do with local modules when adding a project to Teamwork Server:
    - Add to Server adds the whole selected module to Teamwork Server. Select a module name on the server from the Name on Server drop-down list or type a new one.
    - Leave Local Module leaves the selected module on a local workstation.

      IMPORTANT! Standard modules provided by MagicDraw can take only this action.
    - Use Server Module uses a module or its branch from Teamwork Server. You can select the whole module (trunk) or a branch. For example, which is given in Figure 634 on page 1048, the *Production* branch of the *Magic Library* and *Users* modules is selected.

Copyright © 1998-2015 No Magic, Inc.

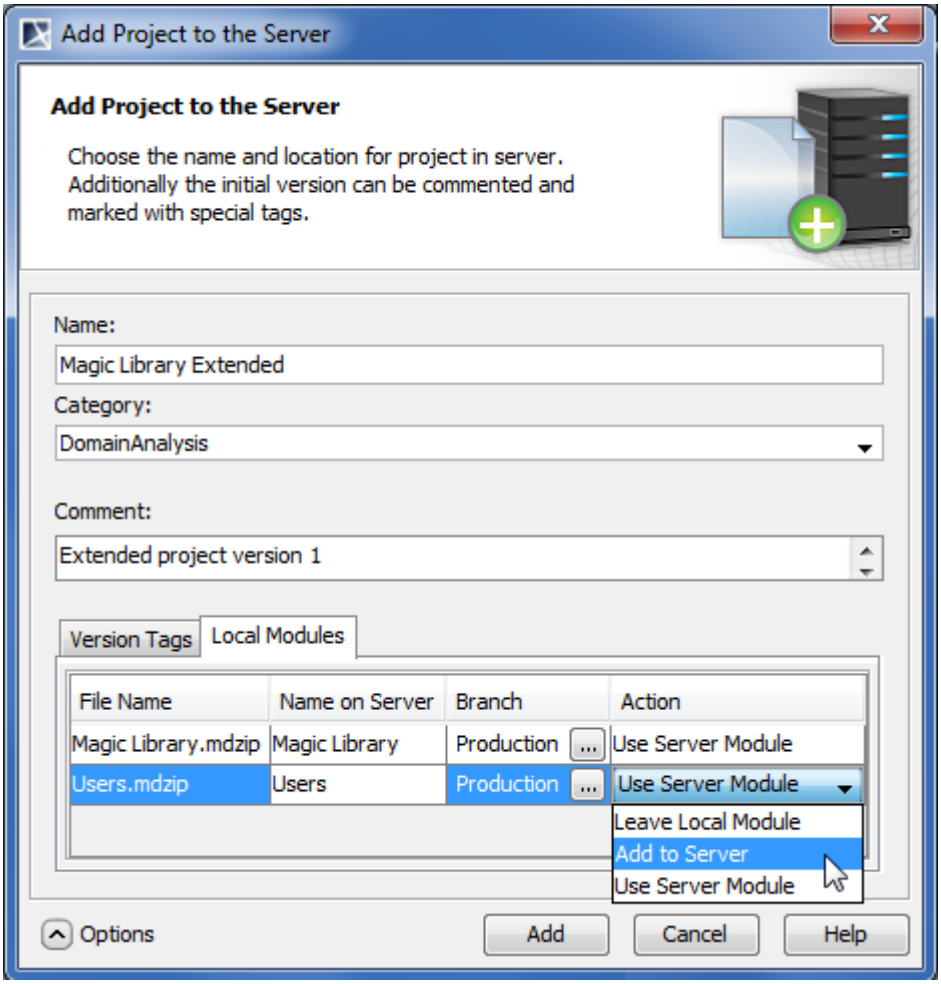


Figure 634 -- Example of Local Modules tab on Add Project to the Server dialog

9. Click the **Add** button.

Copyright © 1998-2015 No Magic, Inc.

Creating server projects in IDE integration

1. Start a new MagicDraw project in the IDE integration you are working on. The **New Project Wizard** dialog opens.

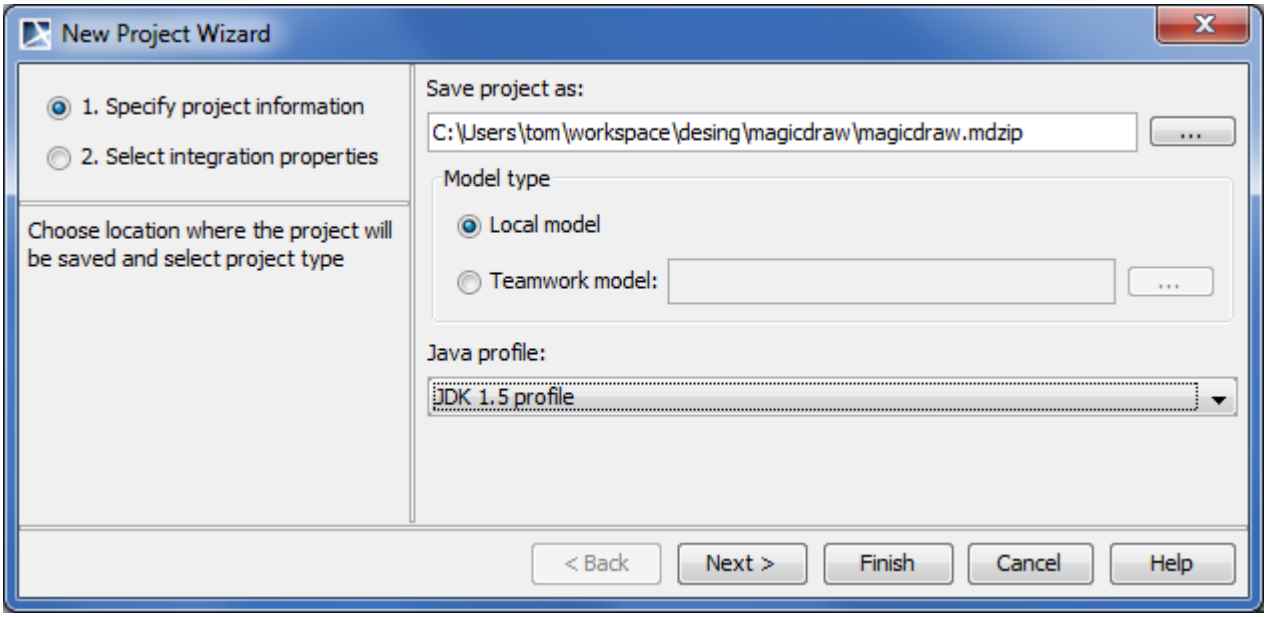


Figure 635 -- New Project Wizard dialog

2. Select **Teamwork model** and click the **...** button to login to Teamwork Server.
3. After log in, select an existing project or add a new one. The model from IDE source will be updated in the selected project. Click **Next.**
4. Specify integration properties and click **Finish**. The server project opens in MagicDraw.

Migrating server projects to MagicDraw Teamwork Server 17.0.1 or later

As of MagicDraw version 17.0.1, the project file inner structure has been changed. Now the latest project file has additional auxiliary information about a project structure. After MagicDraw Teamwork Server has been upgraded to version 17.0.1, you should upgrade your projects used in earlier server versions to the latest file format. After that, you will have full project management features available. To migrate a project to the latest format, projects and used modules must be re-saved (re-committed) in the server.

For project migration, you can use the automatic server project migration feature or migrate each server project manually.



While the project is upgrading, other users are prevented from any modifications in that project. It is highly recommended to migrate all projects by the same person.



You can migrate server projects to 17.0.5 or later version without upgrading the server.

Starting with version 17.0.4, upgrading Teamwork Server is not required to access server projects with the upgraded version of the client application.

For the instructions how to enable connections to the server from later client versions, see Activating Teamwork Server license after purchase at nomagic.com.

Copyright © 1998-2015 No Magic, Inc.

Automatic server project migration

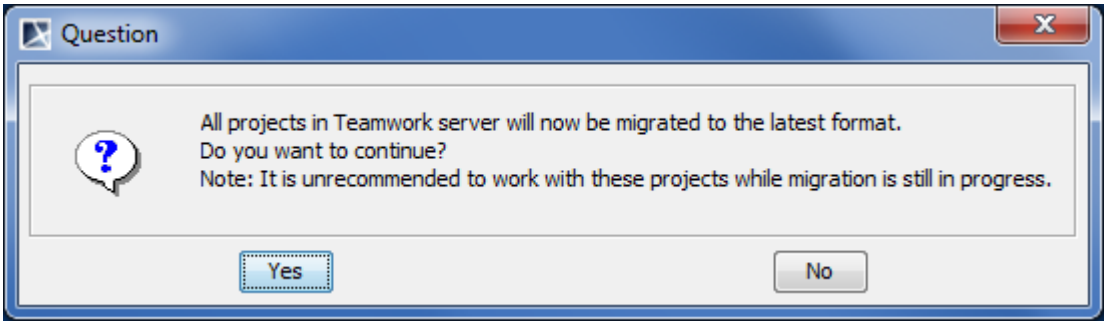
The automatic server project migration feature allows for migrating all server projects at one time.



You must have the Administer project permission for migrating projects. For more information about the permissions, see “User permissions” in MagicDraw Teamwork UserGuide.pdf.

To migrate all server projects automatically

1. From the **Collaborate** menu, select **Migrate Project to Version 17.0.1**.
2. The question message appears.



3. Do one of the following:
   - Click **Yes** to migrate all server projects.
   - Click **No** to cancel this action.

After the project migration is completed, you will get a message informing about successful project migration or containing the list of projects that were not migrated. You need to migrate those projects manually.

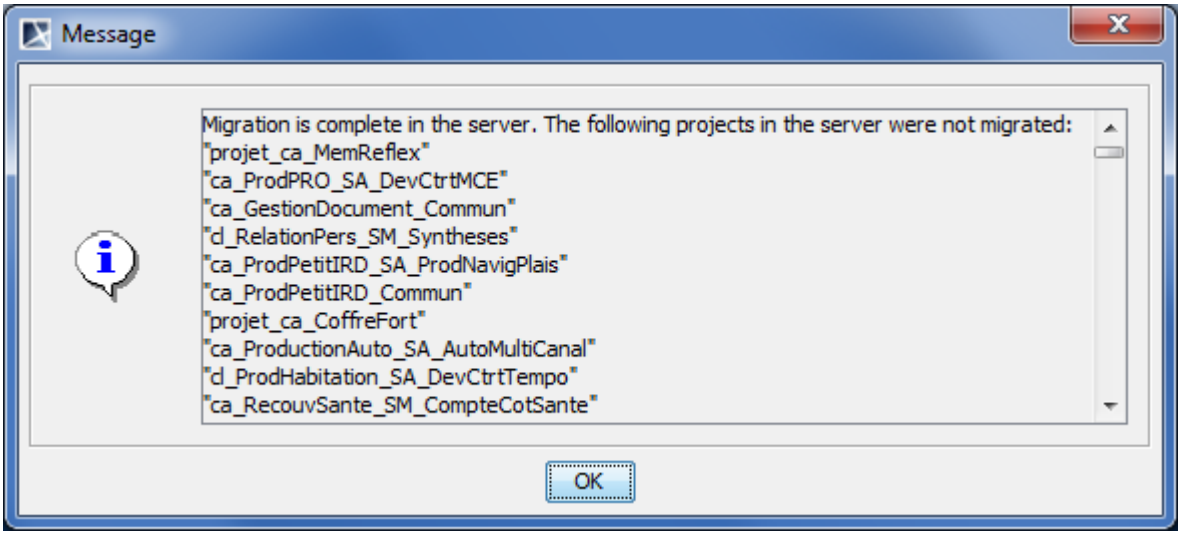


Figure 636 -- Example of message appearing after migrating server projects

Copyright © 1998-2015 No Magic, Inc.

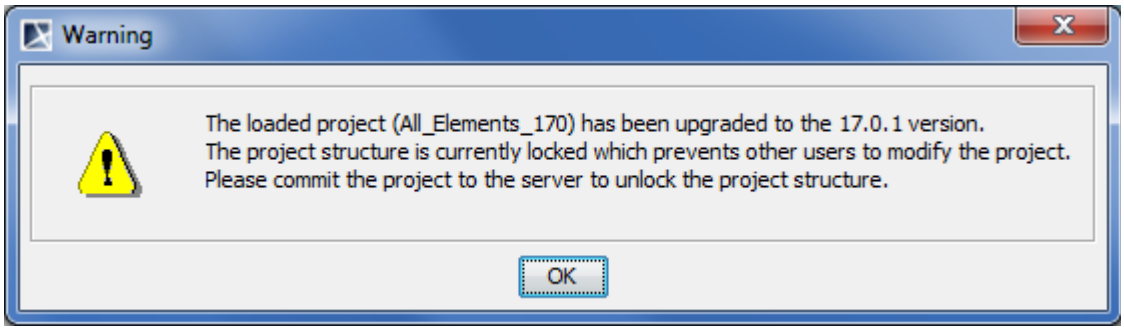
Manual server project migration

To migrate server projects manually



You must have the Administer project permission for migrating projects. For more information about the permissions, see “User permissions” in MagicDraw Teamwork UserGuide.pdf.

1. Open a server project in Teamwork Server. The message notifying that project has been upgraded and requires to be committed to a server of a new version appears. See the following example of the message.



2. Click **OK** and commit the project to the server.



For more information about committing a project, see section "Committing to Teamwork Server" on page 1069.

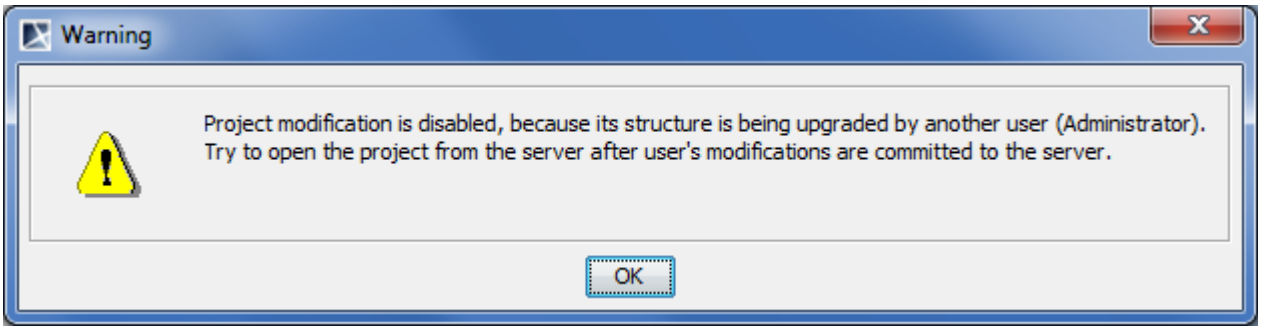
Perform these actions for all projects and modules you want to migrate to Teamwork Server version 17.0.1 or later.

Troubleshooting

In particular cases, server projects are not upgraded to latest file format. It may happen in the following cases:

- When the entire project or some modules are not committed to the server.
- When the project structure remains locked after the project has been committed to the server.

In these cases, you get the following message while loading the project:



You are not allowed to make any modifications in such opened project as projects are opened in the read-only mode.

To solve this problem, the following actions should be performed:

1. Close the project.
2. If any user is currently upgrading the project, he or she should commit this project to the server after the project has been upgraded.

Copyright © 1998-2015 No Magic, Inc.

After the project has been committed, the user should check if the project structure is unlocked. If the project structure is locked, the user who locked the project or an administrator should unlock the project manually in the **Project Properties** dialog.

TIP!

For more information about project structure locking, see section "Locking project structure" on page 1060.

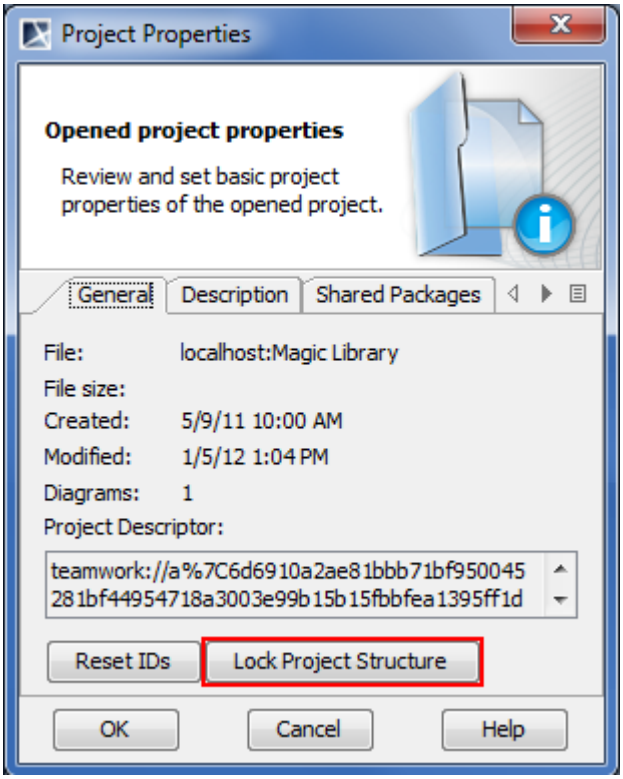


Figure 637 -- View of Project Properties dialog when project structure is unlocked

3. Open the project. If the project was upgraded and committed properly, the project structure is unlocked, and project is available for modifications.

Related procedures

Locking project structure

Committing to Teamwork Server

Related references

Commit Project to the Server dialog

Working with Project Properties Dialog

Locking Model Elements and Diagrams

In the project, you can lock or unlock particular items. The following table describes the connection between locked items and editable data.

Locked Item	Editable Data
Element	Element specification properties (in the Specification window).

Copyright © 1998-2015 No Magic, Inc.

Locked Item	Editable Data
Diagram	• Diagram specification properties (in the Specification window). • Diagram properties (in the Diagram Properties dialog). • Symbol layout on the diagram pane. • Symbol properties (in the Symbol Properties dialog).
Diagram with elements	• Diagram specification properties (in the Specification window). • Element specification properties (in the Specification window and on the diagram pane). • Diagram properties (in the Symbol Properties dialog). • Symbol properties (in the Symbol Properties dialog). • Symbol layout on the diagram pane.



Locking is allowed, if selected elements, symbols, or diagrams are not locked by the other user.

Related concepts

Representation of locked elements
Locking commands
Unlocking Model Elements and Diagrams

Related procedures

Unlocking Model Elements and Diagrams

Related references

Specification Window
Specifying Diagram Style Properties
Formatting Symbols

Representation of locked elements

In the **Containment** tree, names of elements are presented in two colors. They are as follows:

- Locked elements are presented in black, meaning they are editable.
- Unlocked elements are presented in grey, meaning they are not editable.

Copyright © 1998-2015 No Magic, Inc.

The following figure illustrates the **Containment** tree with locked and unlocked elements.

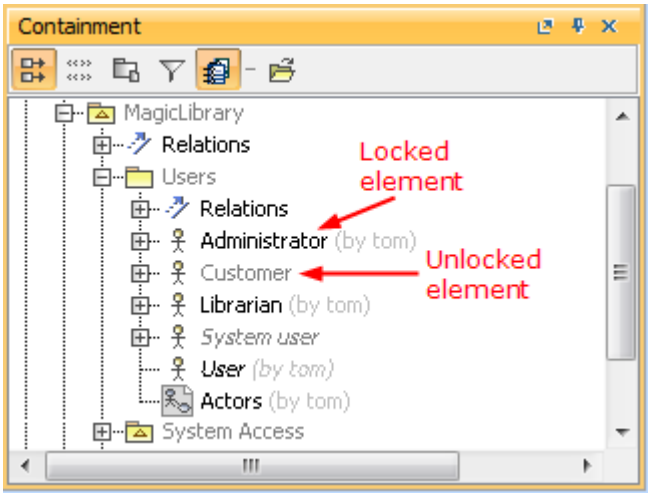


Figure 638 -- Locked and unlocked elements in Containment tree

Locking commands

Availability of locking commands depends on the item selected for locking. Commands are described in the following table:

Command in shortcut menu	What is locked	Real-life examples
Lock Element for Editing in Diagram **NOTE:** The command is available only on symbol's shortcut menu.	• The selected element in the Model Browser. • The diagram pane whereon the selected symbol is. • The element represented by the selected symbol. • The diagram in the Model Browser.	You can use this command in the following situations: • When you want to modify only the particular element in the diagram. • When you have already locked the element, but it is impossible to modify it in the diagram. For example, you have locked the element in the browser but modification is forbidden in the diagram.
Lock Element for Edit	The selected element in the Model Browser.	You can use this command, if you want to modify the specification of the element in the Specification window.
Lock Elements for Edit Recursively	The selected element and its owning elements in the Model Browser.	You can use this command if you want to modify inner elements that are contained by the selected element. For example, a project has a package Analysis which contains two more packages - Design and Implementation. Each of the package contains elements. Lock the Analysis package recursively to edit all elements in the package.

Copyright © 1998-2015 No Magic, Inc.

Command in shortcut menu	What is locked	Real-life examples
Lock Diagram Content for Edit **NOTE:** The command is available only on symbol's or diagram's shortcut menu.	• Symbols on a diagram pane of a selected diagram. • The diagram pane of the selected diagram. • Elements of the selected diagram in the Model Browser. • The selected diagram in the Model Browser.	You can use this command when you are working with the diagram and elements that are represented on this diagram. You will be able to modify each element specification on the diagram and the diagram itself.
Lock Diagram for Edit **NOTE:** The command is available only on symbol's or diagram's shortcut menu.	• Symbols on the diagram pane of the selected diagram. • The diagram pane of the selected diagram. • The selected diagram in the Model Browser.	You can use this command if you want to change diagram properties only. You will be able to modify every symbol representation on the diagram and change the symbols layout.

Locking model elements and diagrams for editing

Before the project editing process, you need to lock the model elements so other users cannot edit them. Elements and diagrams can only be locked by one and only one user at a time.

You may choose whether to update your model when locking elements or diagrams or not.

To enable the model update when locking

1. On the main menu, click **Options** > **Environment**. The **Environment Options** dialog opens.
2. In the options group list, select **Collaboration**.
3. In the options group select the **Update Model When Locking** option and do one of the following:
   - Set to *false* to lock elements or diagrams without updating the model.
   - Set to *true* to update model when locking selected elements or diagrams.

If you have slow network and do not remove a lot of elements from the model, it is recommended not to update your model on each element or diagram locking.



In any case, be rational in your decision. When elements or diagrams are locked without updating the model, there is a risk to edit the element or diagram that does not exist in the model any more.

To lock elements and / or diagrams

1. Do one of the following:
   - Select an element in the Model Browser.
   - Select a diagram in the Model Browser.
   - Select a symbol on a diagram pane.
   - Right-click the diagram pane.
2. On the selected item's shortcut menu, point to **Lock** or **Lock Diagram**, and then click a wanted locking command.



For more information about locking commands, see "Locking commands" on page 1054.

Copyright © 1998-2015 No Magic, Inc.

NOTE The list of commands may vary depending on the item you have selected.

To lock several elements

1. Do one of the following:
   - Select several elements in the Model Browser.
   - Select several symbols on the diagram pane.
2. On the selected elements shortcut menu, click **Lock for Edit**. This allows for editing locked element specification properties in the Specification window.

To lock an element in the Specification window

1. Open the Specification window of the selected element or symbol.
2. Click the Lock element for Edit button (see the highlighted button in the following figure). This allows for editing locked element specification properties in the Specification window.

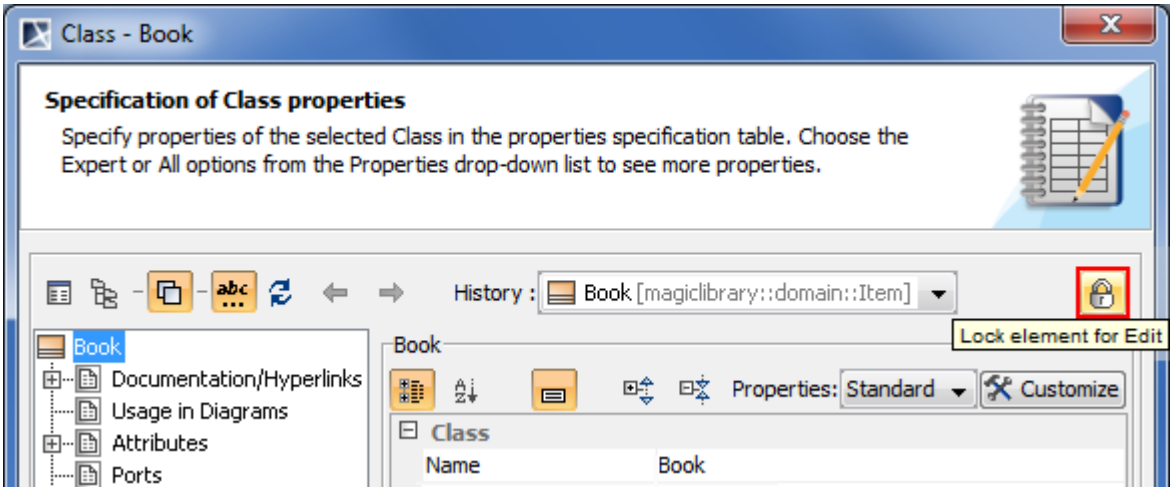


Figure 639 -- Lock element for Edit button (fragment of Specification window)

To lock an element assigned as a property value

1. Open the Specification window of the selected element or symbol.
2. In the element property list, select the element assigned as a the property value.
3. On the selected value shortcut menu, point to **Lock**, and then click one of available commands:
   - **Lock Element for Edit**.
   - **Lock Elements for Edit Recursively**. This command is available if the selected value is an owner for some elements.

TIP! For more information about locking commands, see "Locking commands" on page 1054.

Copyright © 1998-2015 No Magic, Inc.

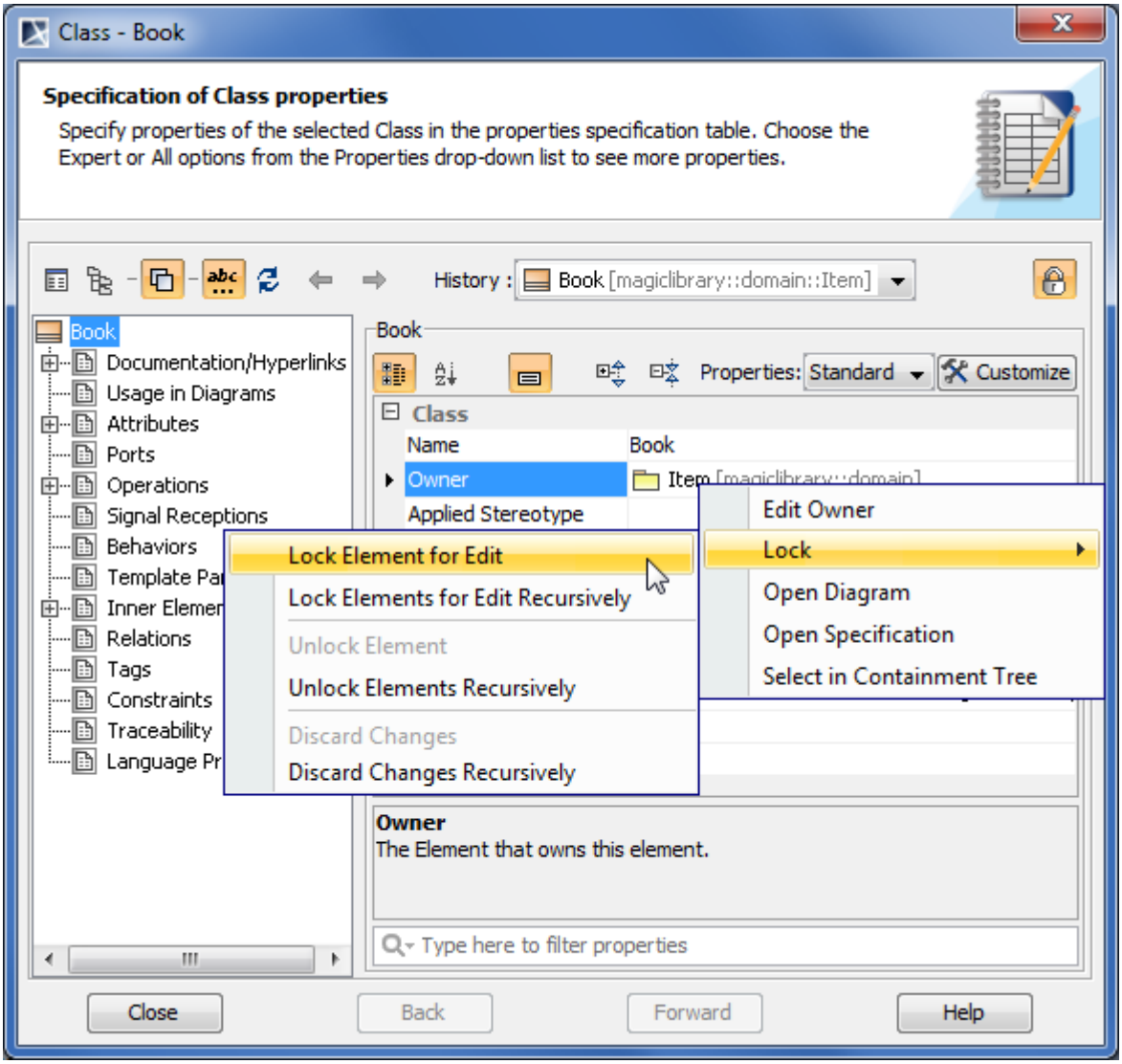


Figure 640 -- Locking property value in Specification window

To lock symbol styles from the Lock View tab

1. In the Model Browser's **Lock View** tab, expand the **Project Options** package.
2. From the **Symbol Styles** shortcut menu, select **Lock Symbol Styles**. All symbol styles are locked and can be modified.

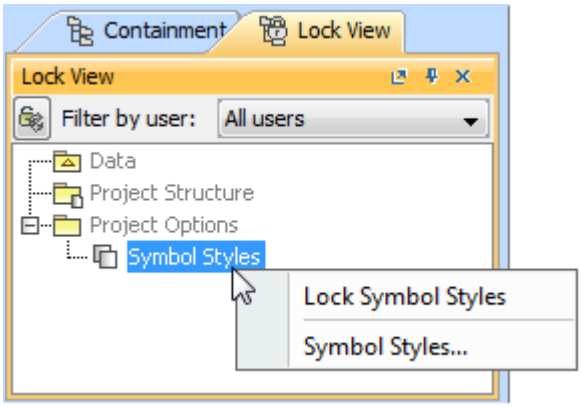


Figure 641 -- Lock View tab

Copyright © 1998-2015 No Magic, Inc.

To lock symbol styles from the Project Options dialog

1. Do one of the following:
   - On the **Options** menu, click **Project**.
   - In the Model browser's **Lock View** tab, select **Project Options** and from the shortcut menu, select **Project Options**.
2. In the opened Project Options dialog, click the **Lock Symbol Styles** button. All symbol styles are locked and can be modified.

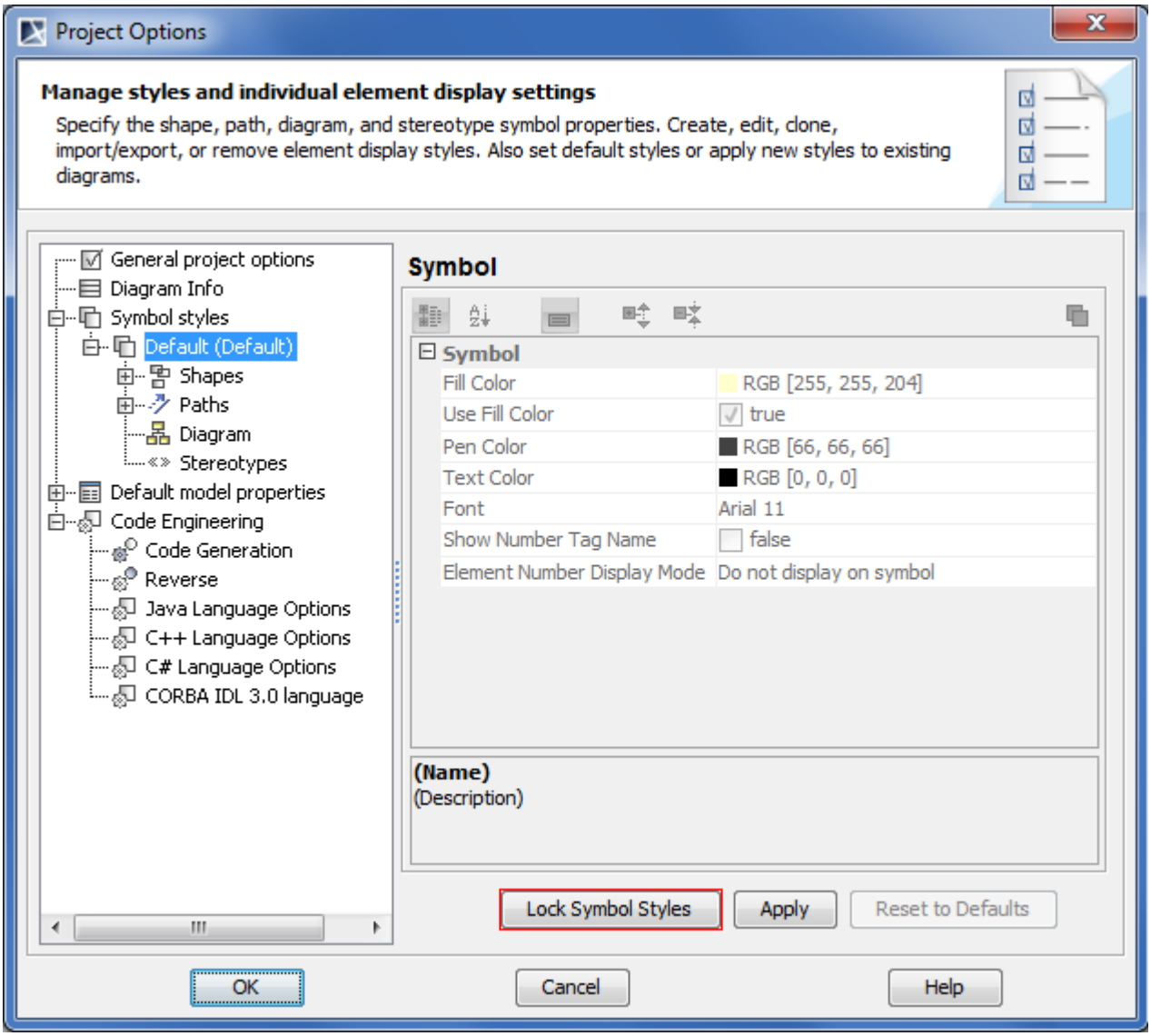


Figure 642 -- Project Options dialog. Lock symbol styles

NOTE

If symbol styles in the Project Options dialog are not locked, the **Make Default** check box in the **Symbol Properties** dialog and Set Selected Symbol Styles as Default button in the diagram toolbar are inactive (see the following figures).

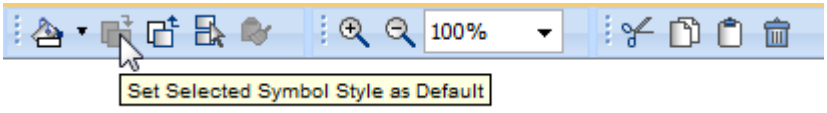


Figure 643 -- Set Selected Symbol Styles as Default button in diagram toolbar

Copyright © 1998-2015 No Magic, Inc.

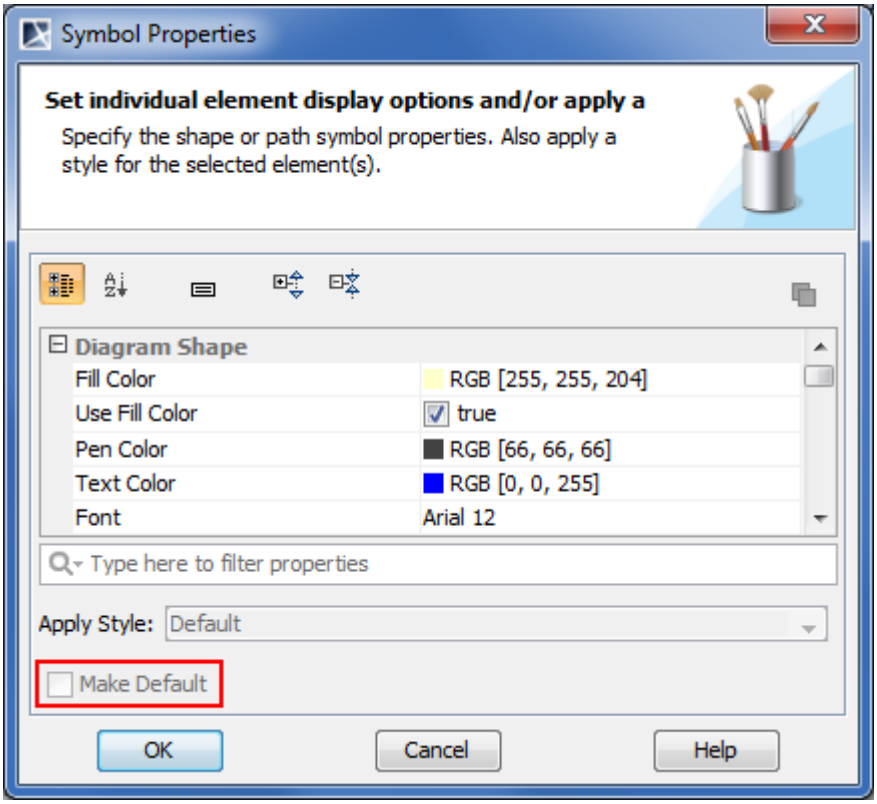


Figure 644 -- Make Default check box in Symbol Properties dialog

Related references

Lock View tab

Locking for reverse engineering

To reverse a code to the model that was already reversed, you need to lock elements that store code engineering information.

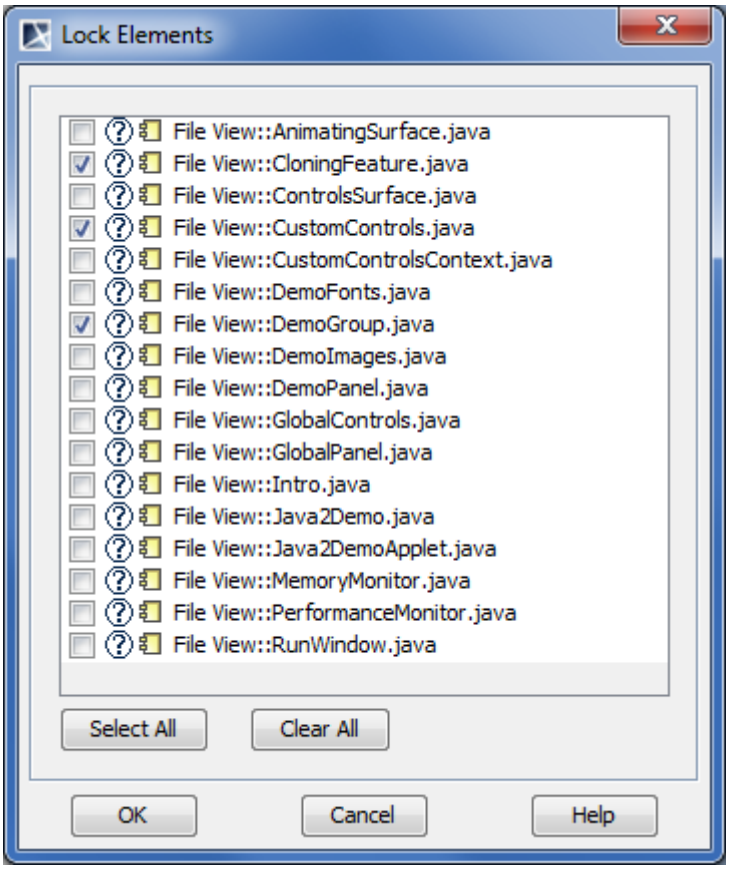
To lock elements for reverse engineering

1. In the **Code engineering sets** branch, select a code engineering set or code engineering element you want to reverse.
2. On the selected items shortcut menu, click **Reverse**. The **Reverse Options** dialog opens.
3. In the **Reverse Options** dialog, set required options and click **OK**.

TIP! For more information about **Reverse Options** dialog, see “Reverse” in MagicDraw CodeEngineering UserGuide.pdf.

Copyright © 1998-2015 No Magic, Inc.

4. The **Lock Elements** dialog opens. Select elements you want to lock and click **OK**.



IMPORTANT! Make sure you have selected all files you are reversing.

5. Depending on options you have selected in the **Reverse Options** dialog, you may need to perform some more steps. Please follow instructions provided on your screen.

Locking project structure

Starting with version 17.0.1, MagicDraw introduces an additional model integrity protection for a collaborative environment. All project structure modifications can only be done by a single person at the same time. To prevent simultaneous structure modification, project structure locking is realized in MagicDraw.

If you are in a collaborative environment and working simultaneously on the same project, the following operations cannot be performed in parallel:

- Modifying shared packages. Adding and removing shared packages are now controlled by the project structure lock.
- Importing new project features. Some plugins may install additional features, such as specific options or code engineering abilities. An opened project structure is locked automatically, and this project cannot be opened by other users.
- Upgrading project versions. The project structure is locked when the project is migrated from an earlier version to the later version (for example, from version 17.0 to 17.0.1). While one user is upgrading project version, the project is in read-only mode for other users until the first user commits this project.

The project structure is automatically locked to ensure the correct model integrity when you perform one of these operations. While the project structure is locked other users may continue getting error messages which may prevent to work with the model. The project structure is unlocked automatically when the project is committed. Alternatively, you can unlock the structure manually. Use the following procedure.

Copyright © 1998-2015 No Magic, Inc.

To lock a project structure from the Lock View tab

1. In the Model Browser's **Lock View** tab, select the **Project Structure** package.



Fro more information about the Lock View tab, see "Lock View tab" on page 89

2. From the shortcut menu, select the **Lock Project Structure** command.

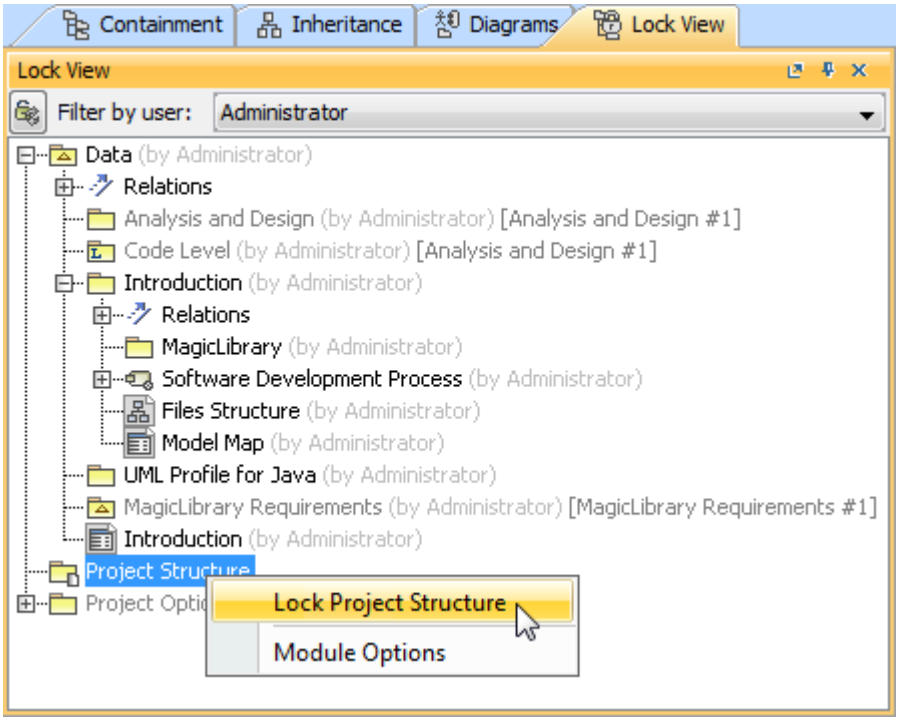


Figure 645 -- Lock View tab. Lock Project Structure command

To lock the project structure from the Project Properties dialog

1. From the **File** menu, select **Project Properties**. The **Project Properties** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

2. In the **General** tab, click the **Lock Project Structure** button.

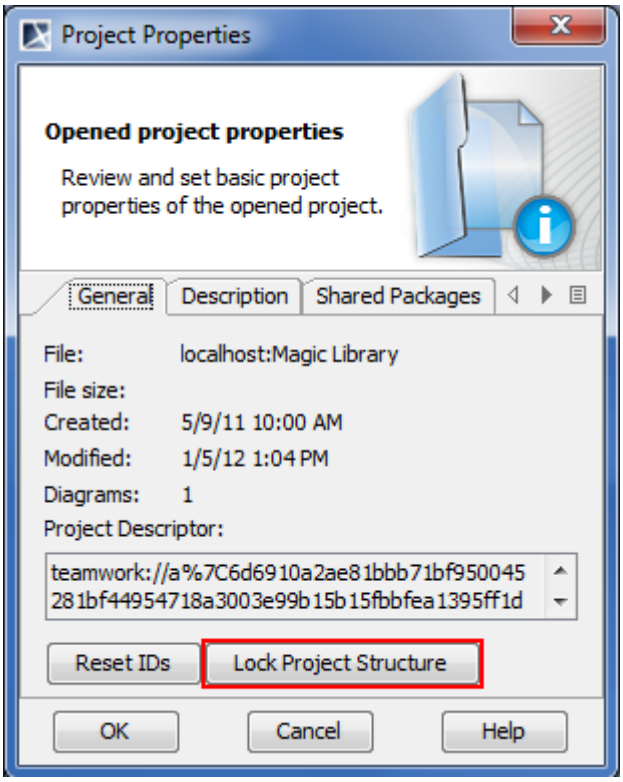


Figure 646 -- Project Properties. Lock Project Structure button

To unlock the project structure manually

1. Do one of the following:

Copyright © 1998-2015 No Magic, Inc.

- From the **File** menu, select **Project Properties** and in the opened **Project Properties** dialog, click the **Unlock** button.

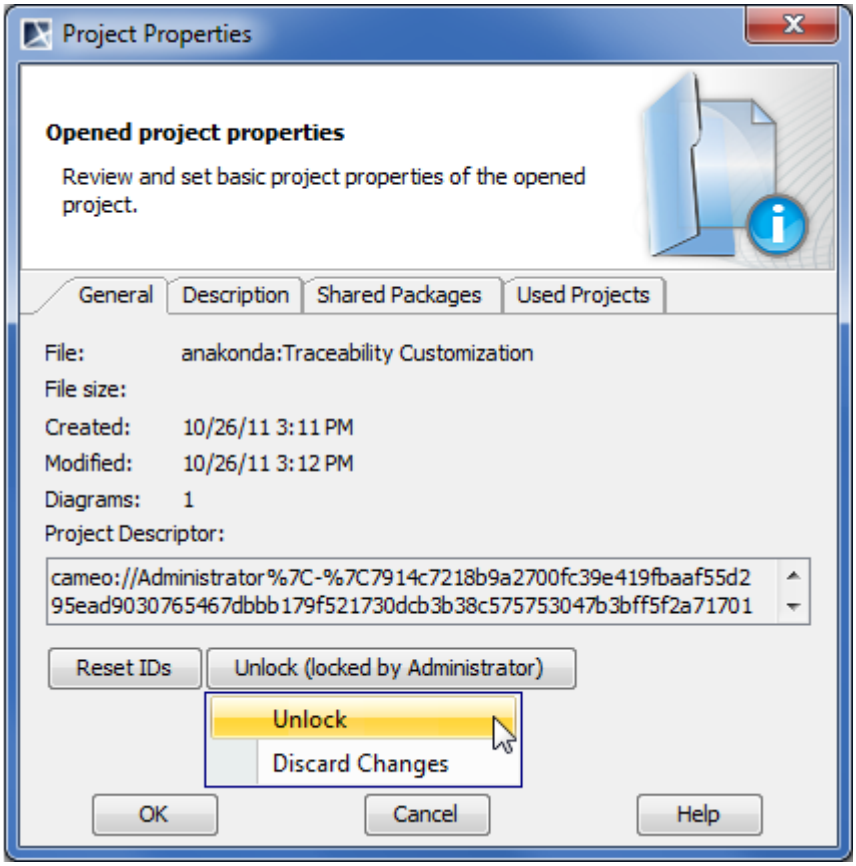


Figure 647 -- Project Properties dialog

- In the Model Browser's **Lock View** tab, select the **Project Structure** package and on its shortcut menu, click the **Unlock** button.

2. Select one of the following commands:
   - **Unlock** - to unlock the project structure.
   - **Discard Changes** - to unlock the project structure discarding changes made after the last commitment.



If the project structure is not locked, but you still get errors concerning project locks, it may be that structure of used modules in a project are being locked because of modules are also being upgraded. Please check if structures of all modules are unlocked and modules are not of earlier formats.

To check and upgrade a project modules format

1. From the **Options** menu, select **Modules**. The **Modules** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

2. Select a module. If the module is not upgraded to the latest format, you get a warning at the bottom of the dialog.

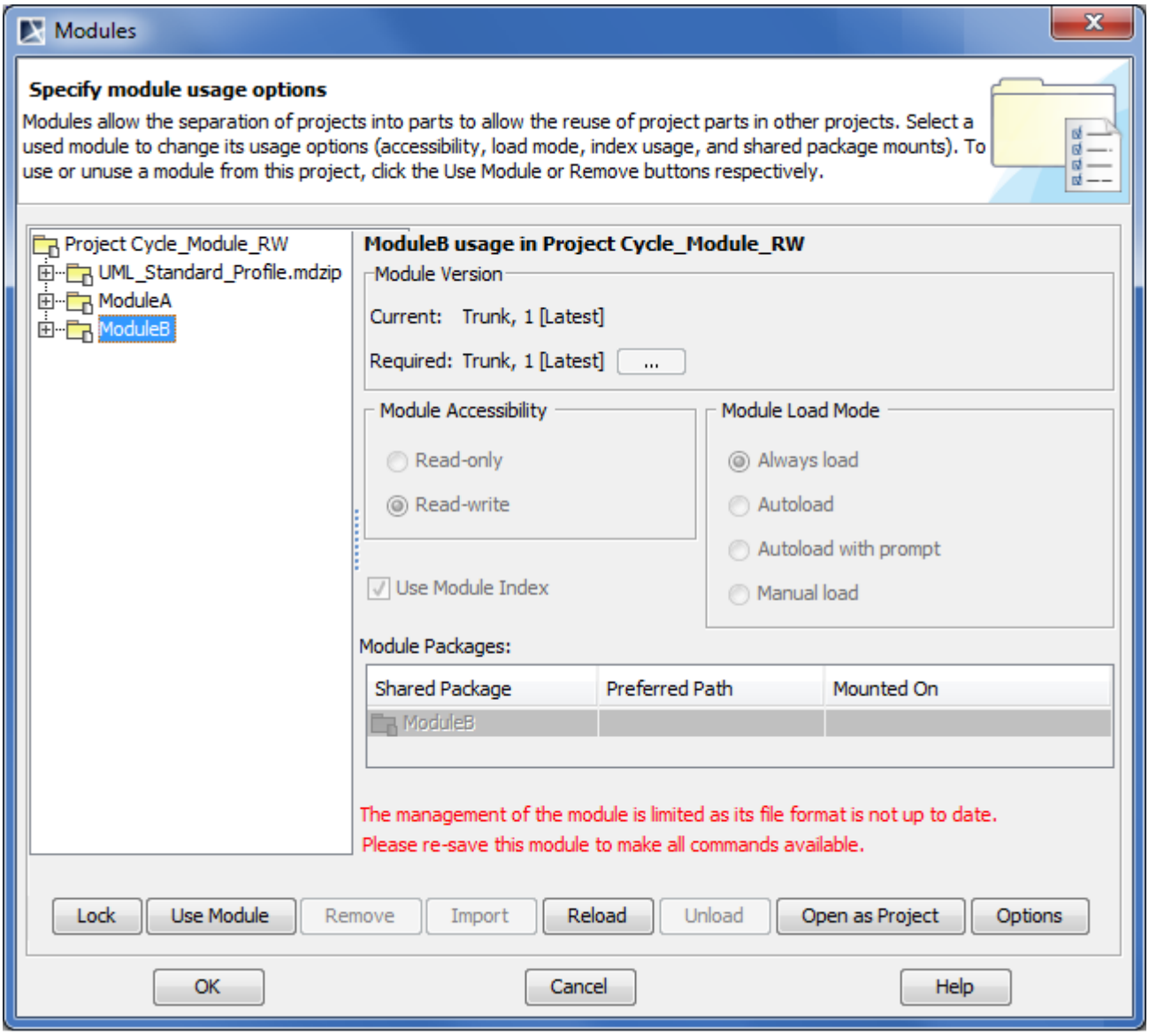


Figure 648 -- Modules dialog. Module of earlier format selected to import

3. Open each module that requires upgrading and resave (commit) it to the server. The module is upgraded to the latest version in this way. You can continue working with upgraded modules as usually.
4. Reopen the main project to finish the upgrading process

Related procedures

Locking model elements and diagrams for editing

Unlocking model elements and diagrams after editing

Related references

Managing Used Projects

Working with Project Properties Dialog

Copyright © 1998-2015 No Magic, Inc.

Unlocking Model Elements and Diagrams

Once the editing process is completed, you need to unlock the model elements so other users could edit them.

Unlocking commands

Availability of unlocking commands depends on the locked item selected for unlocking. Commands are described in the following table:

Command in shortcut menu	What is unlocked	Real-life examples
Unlock Element in Diagram **NOTE:** The command is available only on symbol's shortcut menu.	• The diagram pane whereon the selected symbol is. • The element represented by the selected symbol. • The diagram in the Model Browser. • The selected element in the Model Browser.	You can use this command after you have finished modifying the symbol on the diagram.
Unlock Element	The selected element in the Model Browser.	You can use this command after you have finished modifying the element.
Unlock Elements Recursively	The selected element and its owning elements in the Model Browser.	You can use this command after you have finished modifying all elements owned by the selected elements.
Unlock Diagram Content **NOTE:** The command is available only on symbol's or diagram's shortcut menu.	• Symbols on a diagram pane of a selected diagram. • The diagram pane of the selected diagram. • Elements of the selected diagram in the Model Browser. • The selected diagram in the Model Browser.	You can use this command after you have finished modifying the diagram and its symbols.
Unlock Diagram **NOTE:** The command is available only on symbol's or diagram's shortcut menu.	• Symbols on the diagram pane of the selected diagram. • The diagram pane of the selected diagram. • The selected diagram in the Model Browser.	You can use this command after you have finished modifying diagram representation, but still need to continue working with elements used on the diagram.

Unlocking model elements and diagrams after editing

After you have finished editing, you should unlock model elements and diagram.

To unlock elements or / and diagrams

1. Do one of the following:
   - Select a locked element in the Model Browser.
   - Select a locked diagram in the Model Browser.
   - Select a locked symbol on a diagram pane.
   - Right-click the diagram pane.
2. On the selected item's shortcut menu, point to **Lock** or **Lock Diagram**, and then click a wanted unlocking command. For more information about locking commands, see "Unlocking

Copyright © 1998-2015 No Magic, Inc.

commands" on page 1065.

NOTE The list of commands may vary depending on the item you have selected.

To unlock several elements

1. Do one of the following:
   - Select several locked elements in the Model Browser.
   - Select several locked symbols on the diagram pane.
2. On the selected elements shortcut menu, click **Unlock**. This forbids editing element specification properties in the Specification window.

To unlock all element

1. From the **Collaborate** menu, select **Unlock All**. The **Commit Project to the Server** dialog opens.
2. Click to clear the Keep Locks check box. By default, the check box is selected.
3. Click **Commit** when you are done.

To unlock an element in the Specification window

1. Open the Specification window of the selected locked element or locked symbol.
2. Click the Unlock element button (see the highlighted button in the following figure). This forbids editing element specification properties in the Specification window.

NOTE If the element was modified, you will be requested to commit changes to the server.

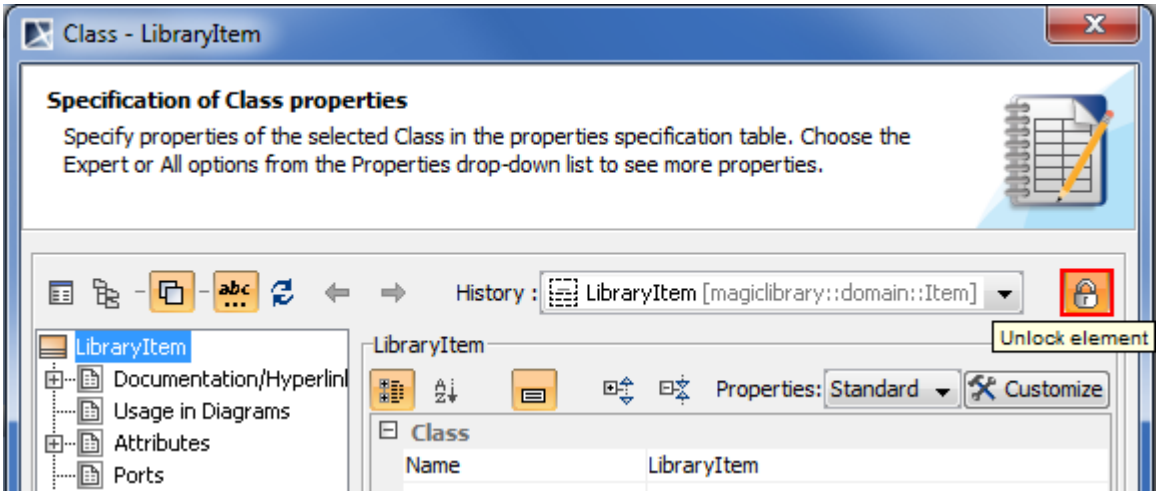


Figure 649 -- Unlock element button (fragment of Specification window)

To unlock an element assigned as a property value

1. Open the Specification window of the selected element or symbol.
2. In the element property list, select the element assigned as a the property value.
3. On the selected value shortcut menu, point to **Lock**, and then click one of available commands:
   - **Unlock Element**.
   - **Unlock Elements Recursively**. This command is available if the selected value is an owner for some elements.

Copyright © 1998-2015 No Magic, Inc.

For more information about unlocking commands, see "Unlocking commands" on page 1065.

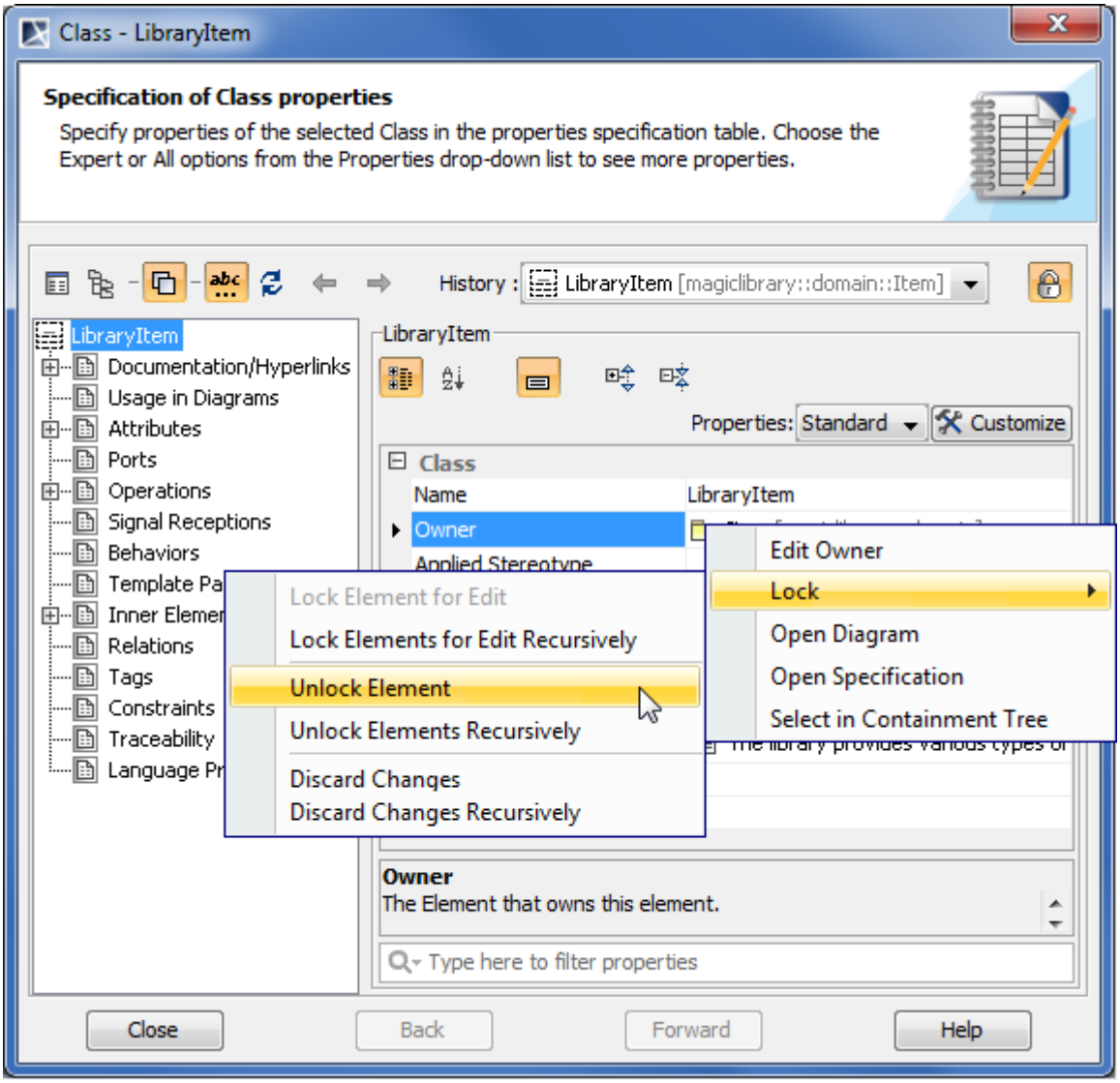


Figure 650 -- Unlocking property value in Specification window

To unlock on project committing

1. From the **Collaborate** menu, select **Commit Project** or press CTRL+K. The **Commit Project to the Server** dialog opens.
2. Click to clear the Keep Locks check box.
3. Click **Commit** when you are done.

Copyright © 1998-2015 No Magic, Inc.

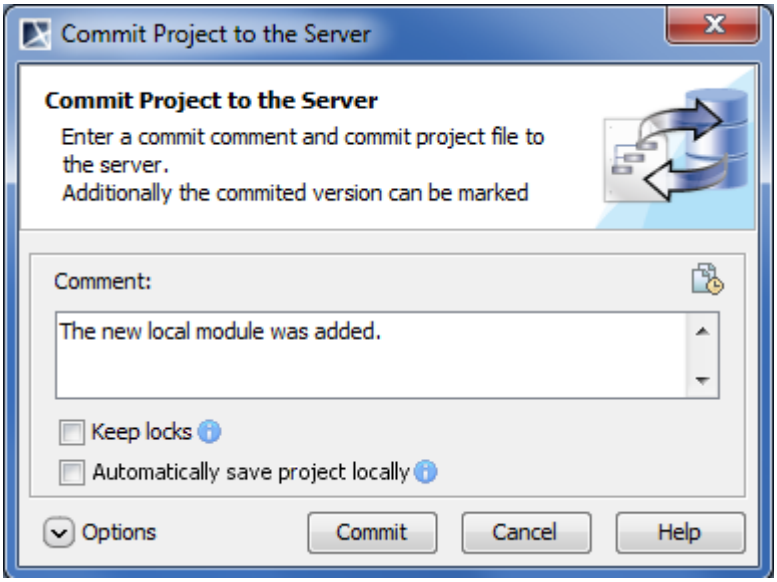


Figure 651 -- Commit Project to the Server dialog

NOTE

Model elements that are not modified are unlocked without committing them to the server.

Unlocking with administrator permissions

The administrator (that is a user named “Administrator”) can forcibly unlock elements locked by other users.

TIP!

The unlocking procedures for administrator are the same as described in section "Unlocking model elements and diagrams after editing" on page 1065.

Forced unlocking is helpful when elements are left locked by the user who is currently unavailable.

IMPORTANT!

Use forced unlocking only in very special cases, as this may cause users working on this project to start their work again.

The user has a possibility to commit changes made to elements that were forcibly unlocked by an administrator. To commit changes, the following conditions should be satisfied:

- The user should have the latest version before locked elements have been lost.
- A new revision of the project has not been committed to the server.

The preceding scenario is very rare, and users should not rely on it. After all, if administrator forcibly unlocked elements, it means there was a need to edit these elements for some purpose.

Changes to elements that were locked and not forcibly unlocked by an administrator, can be committed to Teamwork normally.

To delete an element from the server project

1. Lock the element or elements you want to delete (see "Locking model elements and diagrams for editing" on page 1055).
2. Delete the element.

Copyright © 1998-2015 No Magic, Inc.

3. Commit the project to the Teamwork Server.

IMPORTANT!

If the element contains inner elements which can be locked separately, these inner elements should not be locked by the other users to complete element removal.

For this reason, we suggest to use command **Lock Element for Edit Recursively** to lock all inner elements so nobody could prevent you from removing selected elements.

Committing to Teamwork Server

When committing a server project, new data (data from the MagicDraw application) is merged with the existing one (the current project data on the server). If the current merged element does not yet reside on the server, it should be added to the project without merging.

IMPORTANT!

Only modifier model elements are committed to the server before the new project version is created.

Any new elements added to the parent scope are given a unique name when placed among elements of the same type.

Each new project version is created and saved to the server.

To commit a new project or changes in a project

1. From the **Collaborate** menu, select **Commit Project**, or press CTRL+K.
2. In the **Commit Project to the Server** dialog, type the comment (optional).

NOTE

You can select a comment from the recently used comments. Just click the  button above the comment box, and select the wanted comment from the **Choose Comment** dialog.

3. Add the version tags in the **Version Tags** tab (optional).
4. Select the **Keep Locks** option if you want to keep all the currently locked elements and modules locked.
5. Click **OK**.

NOTE

After committing a project, a new project version that contains changes will be saved in the Teamwork Server.

If you want to save the project locally to your computer, on the **File** menu, click **Save Changes Locally.**

Related procedures

Saving server projects locally

Related references

Commit Project to the Server dialog

Saving server projects locally

You can save your server projects on a specific location in your computer. You can work with the local project and also be able to commit changes to the server. Working with the local copy of the server project helps to improve collaborative work productivity. Even if the connection with Teamwork Server is lost, you can still continue working with the project and commit changes later, when the connection is restored.

Copyright © 1998-2015 No Magic, Inc.

To save a server project locally

1. Do one of the following:
   - From the **File** menu, choose **Save Changes Locally**.
   - From the **Collaborate** menu, choose **Save Changes Locally**.
   - On the **File** toolbar, click  .
   - Press CTRL+S.
2. Specify the name and location, if needed.



If you want to keep your local project copy up-to-date, you have to save changes locally every time committing to the server. For this purpose, select the **Automatically save project locally** check box on the **Commit Project to the Server** dialog.

Related procedures

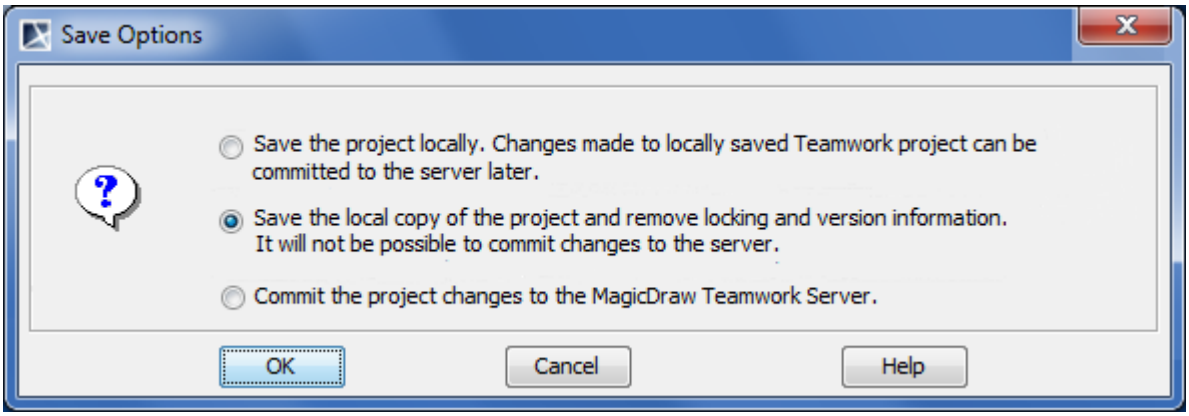
Committing to Teamwork Server

Converting server projects to local

You can convert a server project back to the local copy. After converting, the project will no longer be related with a server and you will not be able to commit changes to the Teamwork Server anymore.

To convert a server project to a local file system

1. Open a server project.
2. On the **File** menu, click **Save Project As**. The **Save Options** dialog opens.
3. In the dialog, select **Save a local copy of the project and remove locking and version information.** The project will be saved on a disk and will not include any locking or version information. A local copy of the project cannot be committed to the Teamwork Server anymore.



4. Click **OK** and specify a location to save the project.

If a server project uses one or more module versions created with an earlier version than MagicDraw 17.0.1, you need to save these modules with MagicDraw 17.0.1 manually. If you cannot change modules because your project uses fixed modules, you should perform the following steps:

1. Log on to Teamwork Server which stores the module.
2. Branch each module of the earlier version from the fixed version to a separate branch.
3. Open the branched version of each module and commit it to the server in order to migrate the module to the latest version.
4. In the project, change versions of each branched module to the branched version instead of original one.
5. Save the project locally again.

Copyright © 1998-2015 No Magic, Inc.

Related procedures

Committing to Teamwork Server

Updating server projects

When *updating* a project, the latest project version from the server is loaded. A new version of a project is created every time a user commits new data to the server.



Model elements that are locked for will not be updated.

When *updating lock information*, the list of elements that are locked by other users is refreshed.

You should update a project when you know that a new project version is available on the server.

To update the whole project

- From the **Collaborate** menu, select **Update Project**, or press CTRL+U.

To update elements that are locked by other users

- From the **Collaborate** menu, select **Update Lock Information**, or press CTRL+SHIFT+U.

Versioning

Every time a project is committed, a new project version is created. You can undo committed changes by setting a previous project version as latest.

To view project versions

1. From the **Collaborate** menu, select **Projects** or **Open Server Project** commands.
2. Click the Project History button. The **Project Versions** dialog opens.

The **Project Versions** dialog lists all versions of the selected project. You can set the selected project version as latest, edit comments, and review other project versions.

Visibility of version tags

Project version tags are visible in the following places:

- Version browser
- Branches browser

Copyright © 1998-2015 No Magic, Inc.

- Module/project name representation in the containment tree/application title.

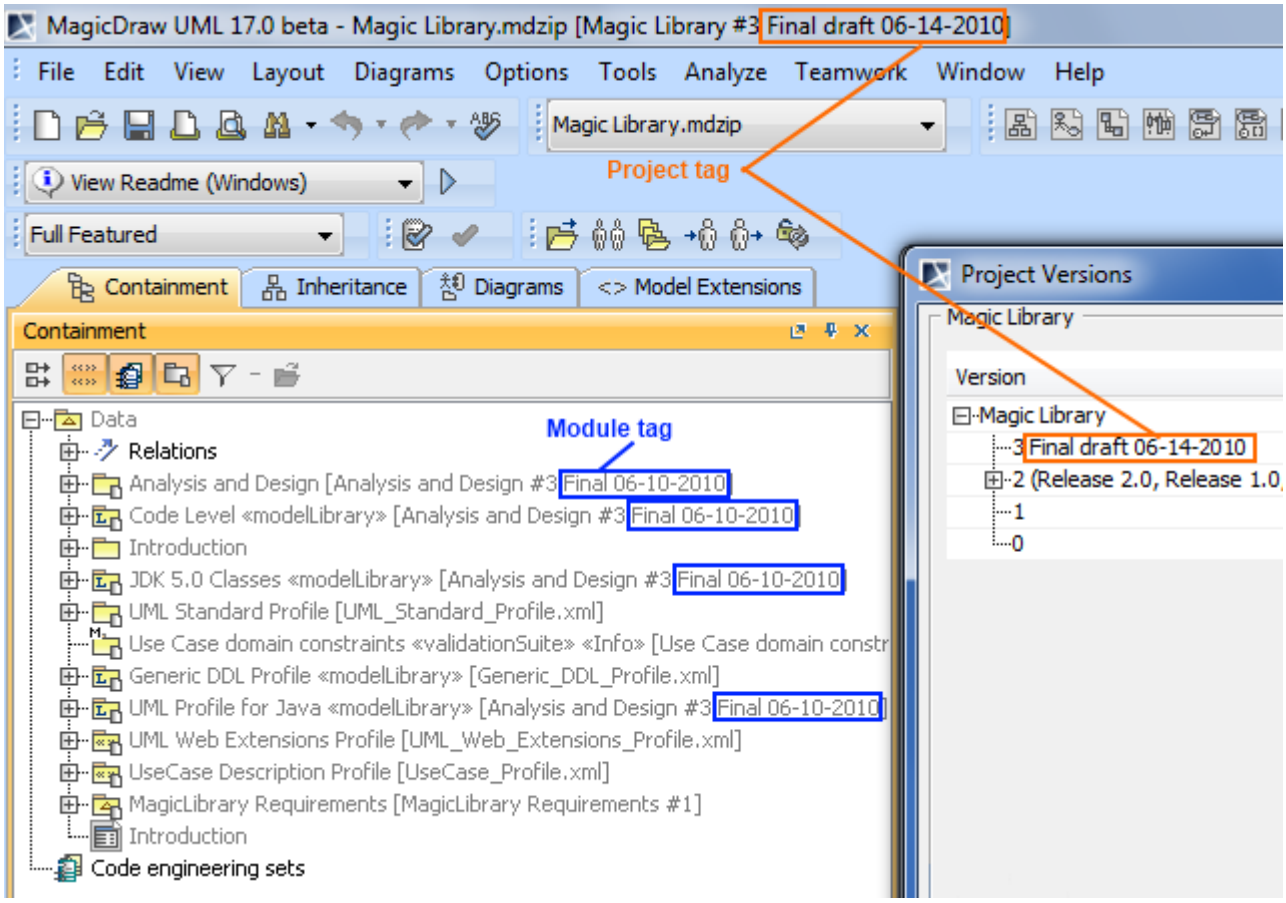


Figure 652 -- Places where projects tags are displayed

Related references

Manage Projects dialog

Open Server Project dialog

Sending messages

While working in the collaborative environment, you can send messages to users who are currently logged on to the server.

Copyright © 1998-2015 No Magic, Inc.

As of MagicDraw version 17.0.3, all the sent and received messages are stored in the Notification Window. You will also receive the notification message informing about the received messages.



Figure 653 -- Example of message displayed in Notification Window and notification message

NOTE

- You can send a message only to users who are on the same Teamwork Server.
- To refresh the list of available users, reopen the **Send Message** dialog.

TIP!

For more information about the Notification Window, see "Notification Window" on page 205.

Copyright © 1998-2015 No Magic, Inc.

To send a message to one or all active Teamwork users

1. From the **Collaborate** menu, select **Send a Message** or press CTRL+SHIFT+M. The **Send Message** dialog opens.

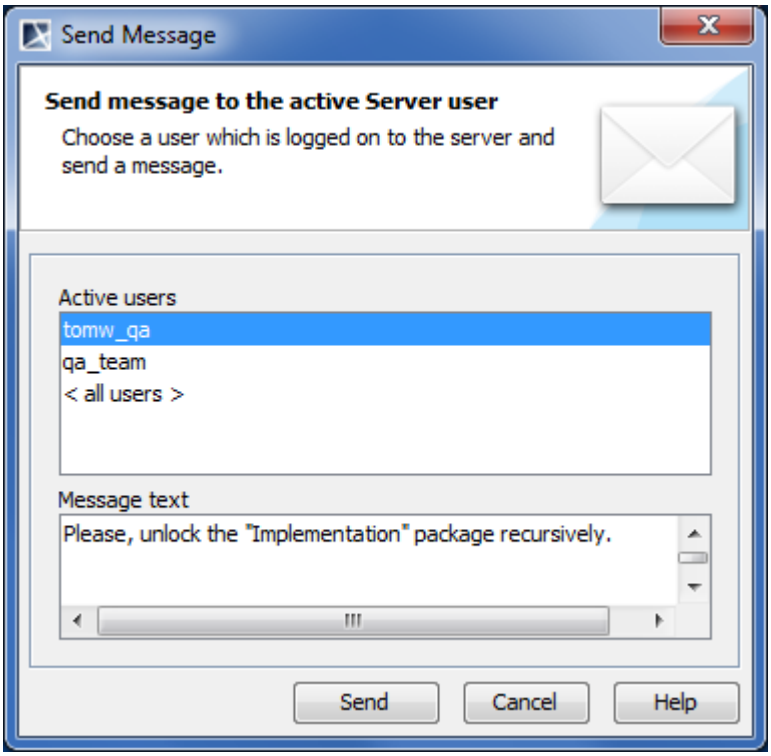


Figure 654 -- Send Message dialog

2. In the **Active users** list, select a user or users to whom you want to send a message. To send a message to all active users, select <all users>.
3. Type the message in the **Message text** box**.**
4. Click **Send**.

Code engineering in server projects

TIP!

For more information about general code engineering functionality, see MagicDraw CodeEngineering UserGuide.pdf which can be found in <MagicDraw installation directory>\manual.

To generate code in the server project

1. From the Code engineering sets shortcut menu in the Browser, select **New** and then select a code engineering language. Type a name for new set.
2. Add elements from the Data branch to the newly created set. All elements (locked and unlocked) can be added to set for code generation. Drag & drop them from the Browser tree.
3. Select the code engineering set, containing elements in the Browser and from the shortcut menu, select **Generate.** The code is generated for all elements.

To reverse code in the server project

1. From the created code engineering set shortcut menu, select **Edit**.
2. Specify the Working Directory and Working Package, to where reversed elements will be placed in the model.

NOTE

Reversing the whole set to a defined package is allowed only if the package is locked.

Copyright © 1998-2015 No Magic, Inc.

3. In the **Add File** tab from the **Round Trip Set** dialog, add code files from the **All Files** list to the **Set** list. Click **OK**.
4. From the modified code engineering set shortcut menu, select **Reverse**. Define options in the **Reverse Options** dialog. Click **OK**.

When performing a round-trip code engineering, code cannot be reversed for unlocked elements - the **Reverse** command will be disabled in shortcut menu.

Project Branching in Teamwork



Project branching

Introduction

Project Branching allows the user to derive one project version from another. It duplicates versions of a project and enables users to work simultaneously on them. Branching increases productivity for those who:

- use MDA tools (for example AndroMDA, openArchitectureWare and other).
- produce different products that are still very interrelated. This is common when a product has several versions and each of them is designed for different types of customers.
- have several development teams working on a single product and have to merge several development branches before releasing the product.

MagicDraw supported Teamwork Server repository features:

1. Teamwork Server stores a list of projects.
2. Multiple hierarchical branches are supported for every single project version.
3. Multiple versions of a single project and a branch are supported with one active (editable) version each.
4. The user can open every version of a project or a branch as read-only and one active version as read-write.
5. A project or branch can be derived from any previous version of that project or branch (either from the same branch or different branch).
6. The user can create a branch from any version of a project or a branch.
7. The user can compare two versions of the same or different projects or branches.
8. Teamwork Server integrates with Subversion and ClearCase.

Copyright © 1998-2015 No Magic, Inc.

The Teamwork Server project repository structures in MagicDraw are shown in the following figure.

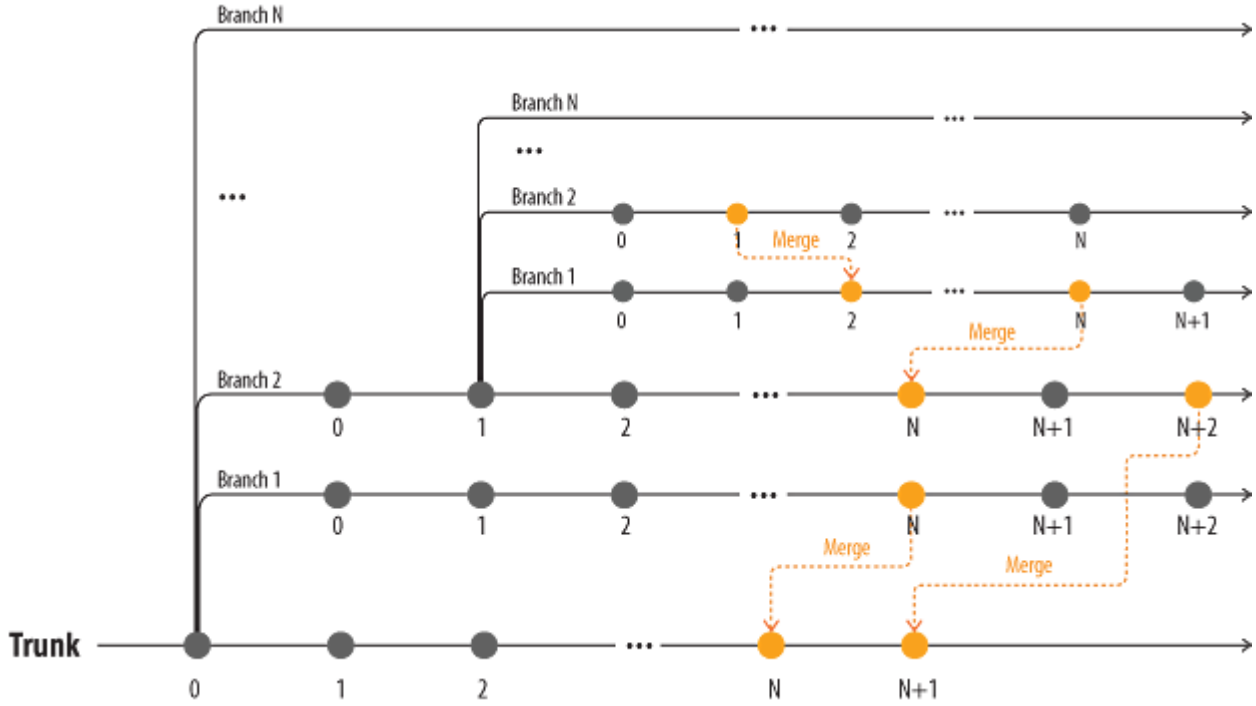


Figure 655 -- Example of MagicDraw project development

As shown above, a branch can be created from any project or branch version. Every project and branch has at least one version (version 0). In MagicDraw every project version also carries information of the version it was derived from. Project and branch versions are assigned by adding 1 (one) to the previous project or branch version number. A new version of a project can be derived from a previous project version or from any branch version of the project. A new branch version can also be derived from any project version it belongs to or from any branch version of that project. Every version can be tagged.

Modules are also projects. Therefore, they can also be branched. All features and rules that apply to projects and project branches apply to modules as well.

A version of a project or a branch stores information of the modules it uses: the module path or module branch and the version of that module or branch. A version may store the “latest” value. This means that it always uses the latest version of the module or module branch.

Copyright © 1998-2015 No Magic, Inc.

An example of a branched repository is shown in Figure 656 on page 1077.

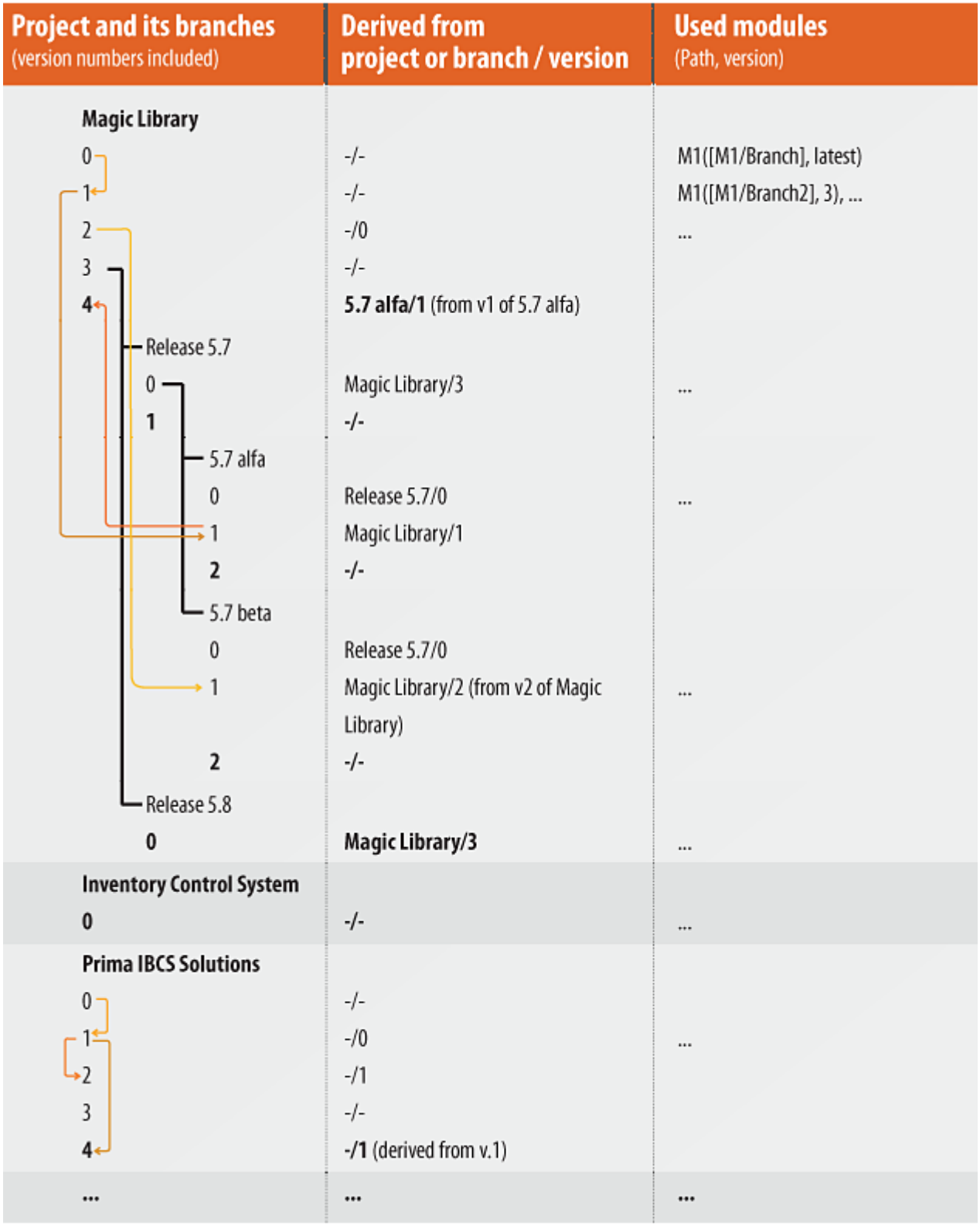


Figure 656 -- Project repository structure (active versions are in bold) in MagicDraw. Arrows show data flow direction

MagicDraw supports two branching scenarios:

1. Branching a project causes branching of all or part of used modules. This scenario is depicted in figure:

Copyright © 1998-2015 No Magic, Inc.

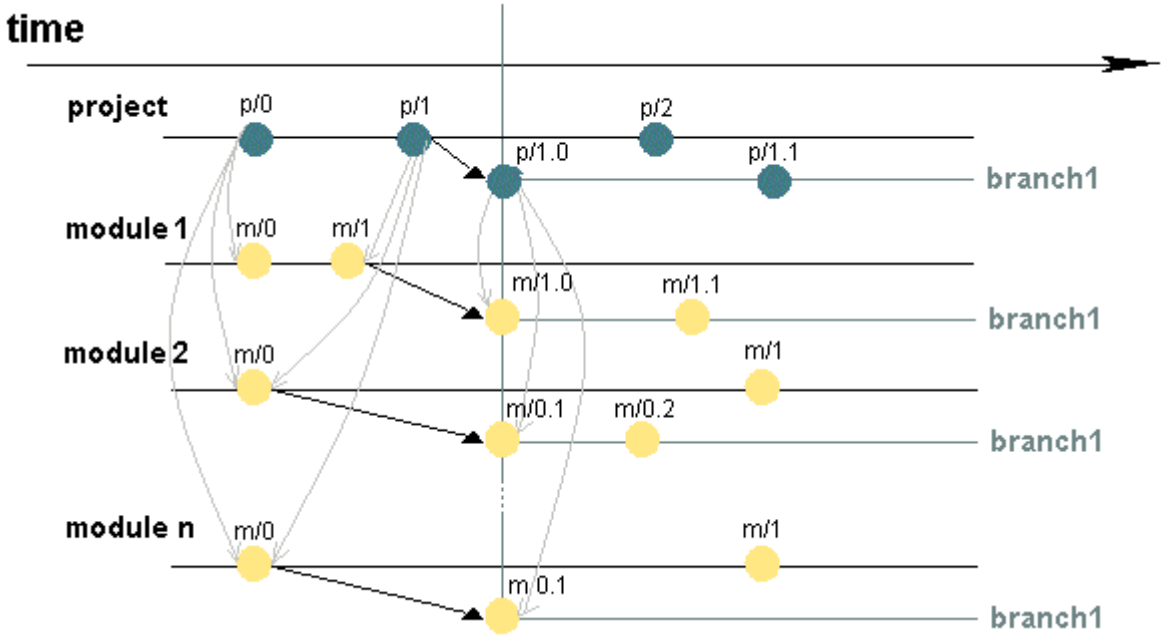


Figure 657 -- Branched project uses branched modules

The list of modules, which are used by a particular project version, is presented in the table below.

Project version	Used modules
p/0	• module 1 - version m/0; • module 2 - version m/0, ... • module n - version m/0.
p/1	• module 1 - version m/1; • module 2 - version m/0, ... • module n - version m/0.
p/1.0	• module1 m/1.0 • module2 m/0.1 • ... • module n m/0.1

2. Branching a project does not cause branching of the modules it uses. The branch of a project uses the same versions of the modules are used by the project it is derived from (it can be the module “latest” version or some explicit version specified by the user). This scenario is depicted in Figure 658 on page 1079.

Copyright © 1998-2015 No Magic, Inc.

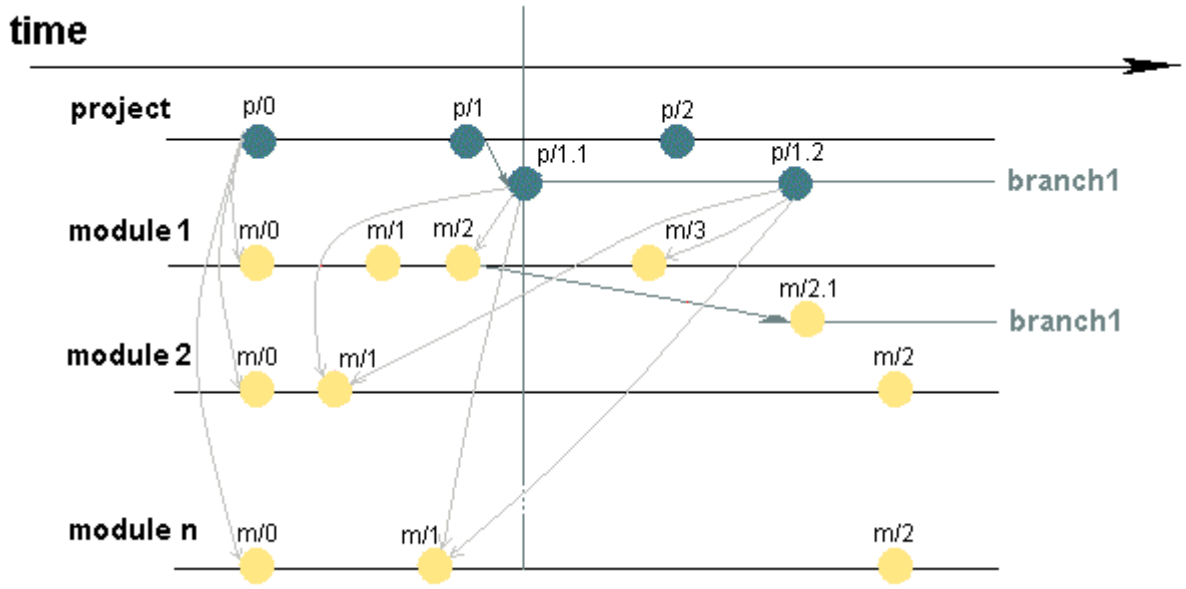


Figure 658 -- Branched project uses the same versions of modules that are used by the project

The list of modules, which are used by a project in a particular project version, is presented in the table below.

Project version	The list of modules, which project uses
p/0	• module 1 - version m/0; • module 2 - version m/0, • module n - version m/0.
p/1.1	• module 1 - version m/2; • module 2 - version m/1, • module n - version m/1.
p/1.2	• module 1 - version m/3; • module 2 - version m/1, • module n - version m/1.

Managing branches

To create a project branch

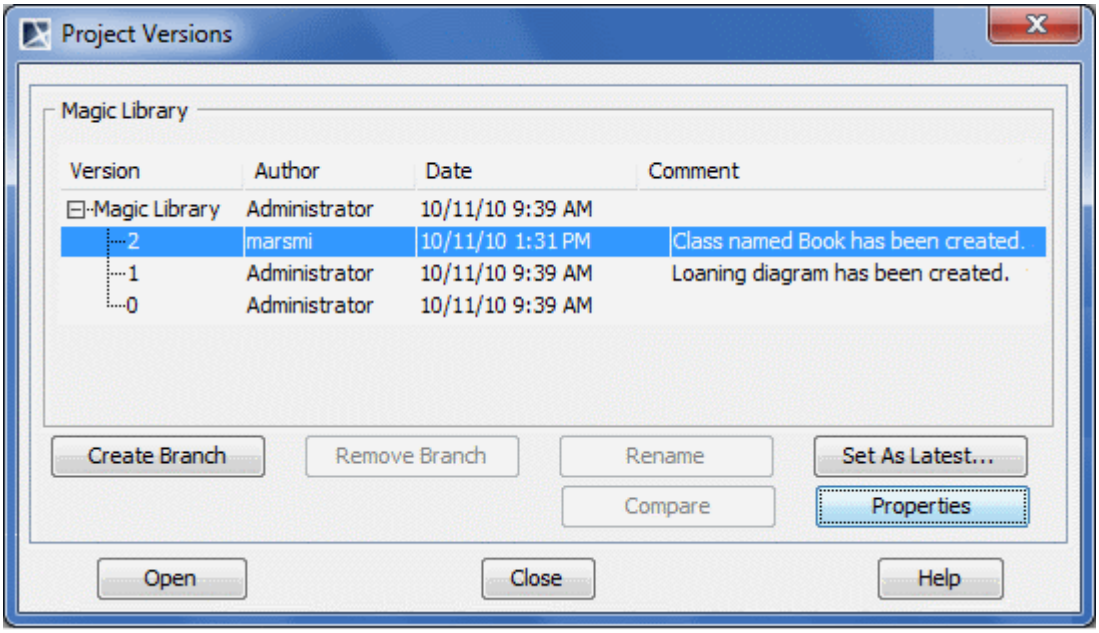


You must have the Administer project permission for branching. For more information about the permissions, see “User permissions” in MagicDraw Teamwork UserGuide.pdf.

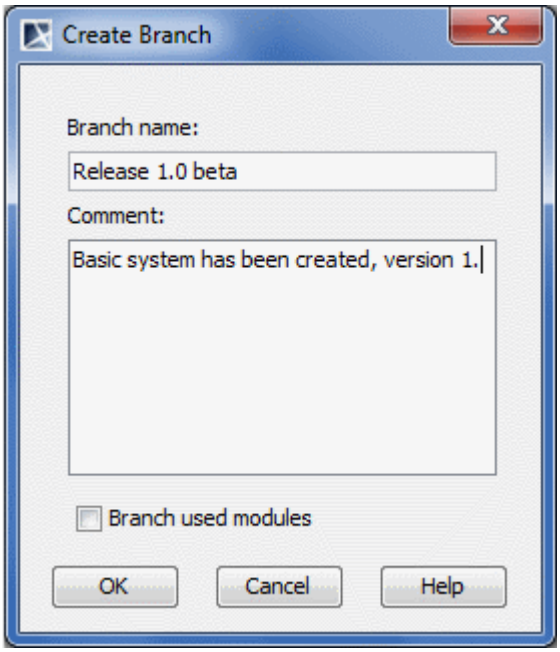
1. From the **Collaborate** menu, select **Projects**. The **Edit Projects** dialog opens.
2. Select the project from which a branch will be created.

Copyright © 1998-2015 No Magic, Inc.

3. Click the Project History button. The **Project Versions** dialog opens.

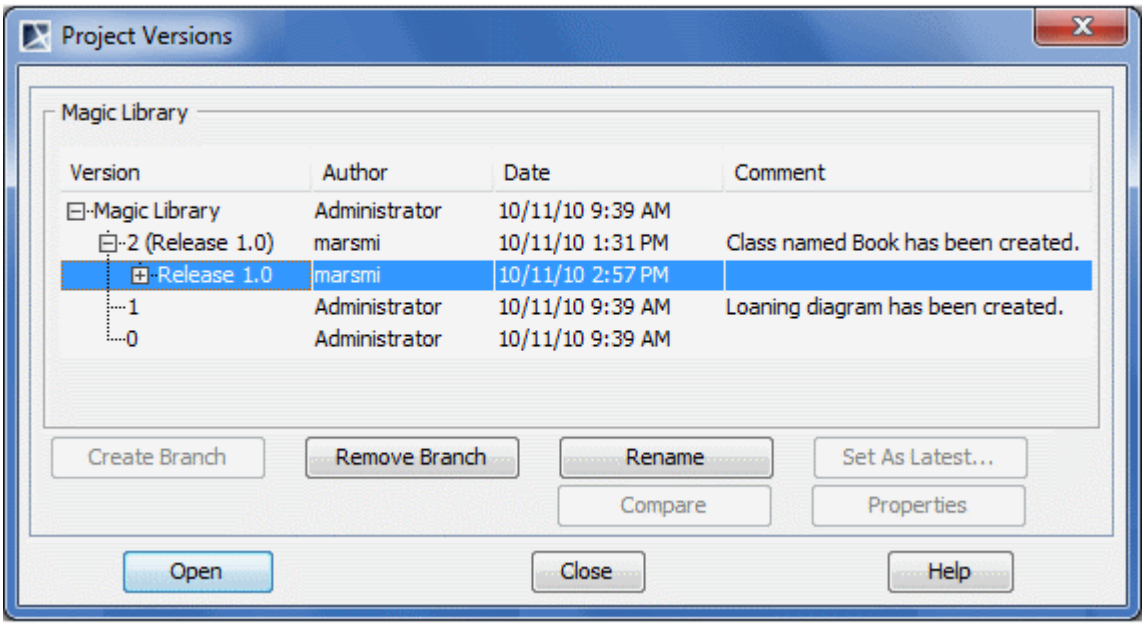


4. Select the project version from which a branch will be derived.
5. Click the **Create Branch** button.
6. In the **Create Branch** dialog, type the branch name and comment.



Copyright © 1998-2015 No Magic, Inc.

7. In the **Create Branch** dialog, click **OK**. A new project branch is created.



TIP!

- You can branch multiple projects together with their modules. To branch multiple projects, in the **Project Versions** dialog select a few projects (use SHIFT or CTRL keys).
- It is possible that several branches are derived from a single project version. In this case a comma-separated list of branches is displayed next to the project version (see the following figure).

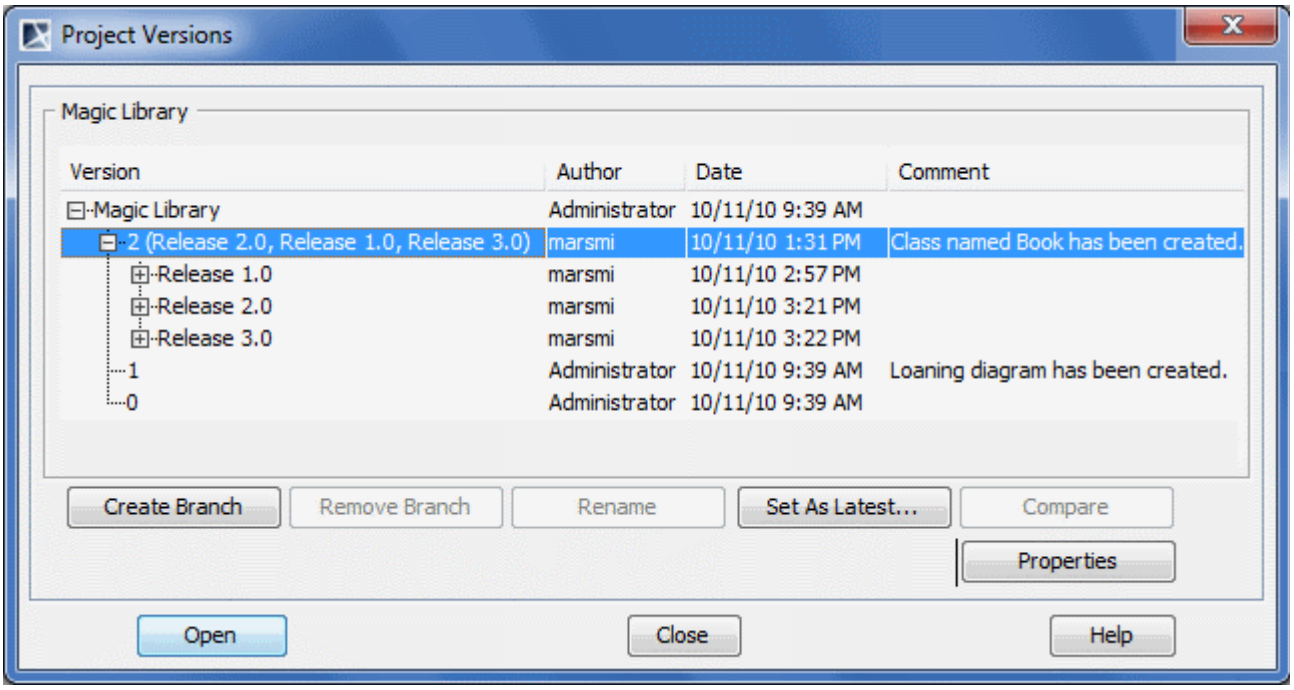


Figure 659 -- Project Versions dialog. Several branches are derived from single project version

Copyright © 1998-2015 No Magic, Inc.

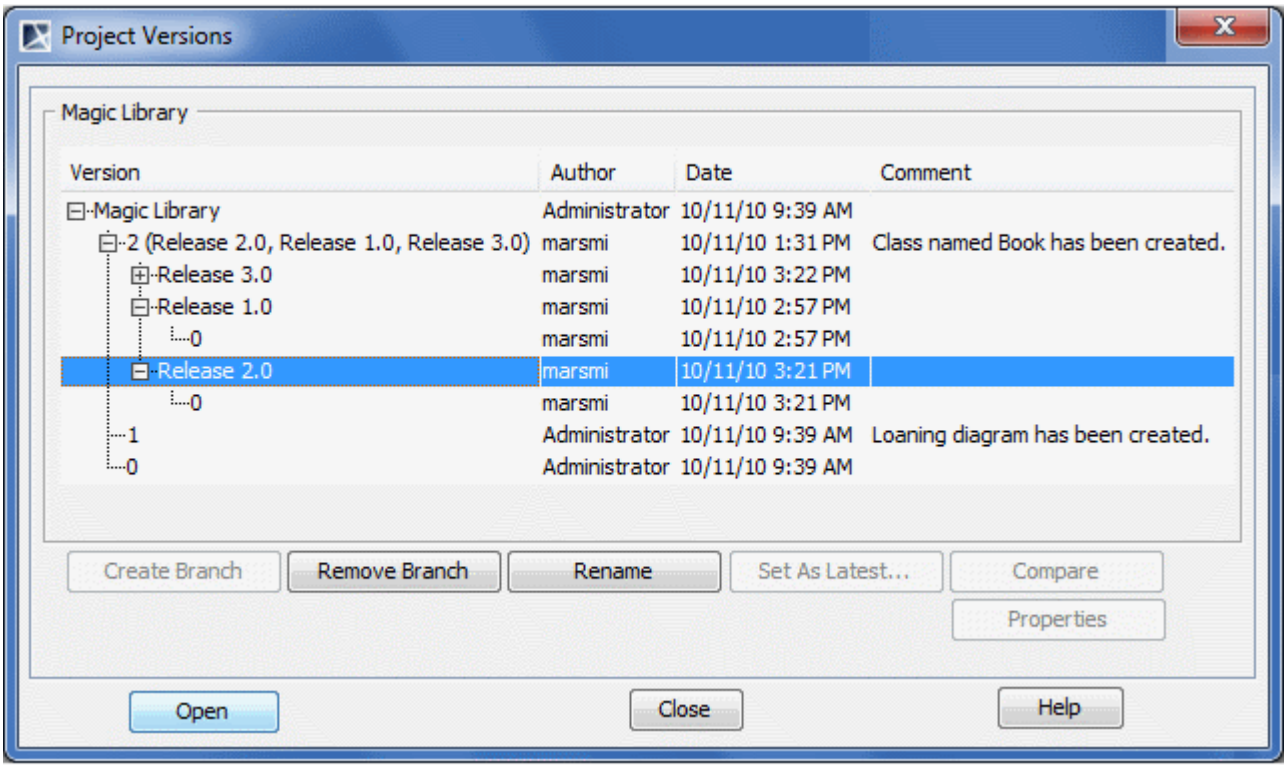
To open a branch

1. Open **Project Versions** dialog.



For more information about creating branches, see the procedure "To create a project branch" on page 1079.

2. In the **Project Versions** dialog, select a branch to open.



3. Click the **Open** button to open the project branch.



It is easy to recognize whether you are working on a branch by looking at the title of the main window: the project name, branch names separated by slashes, and project version in the branch are shown (see the following figure).

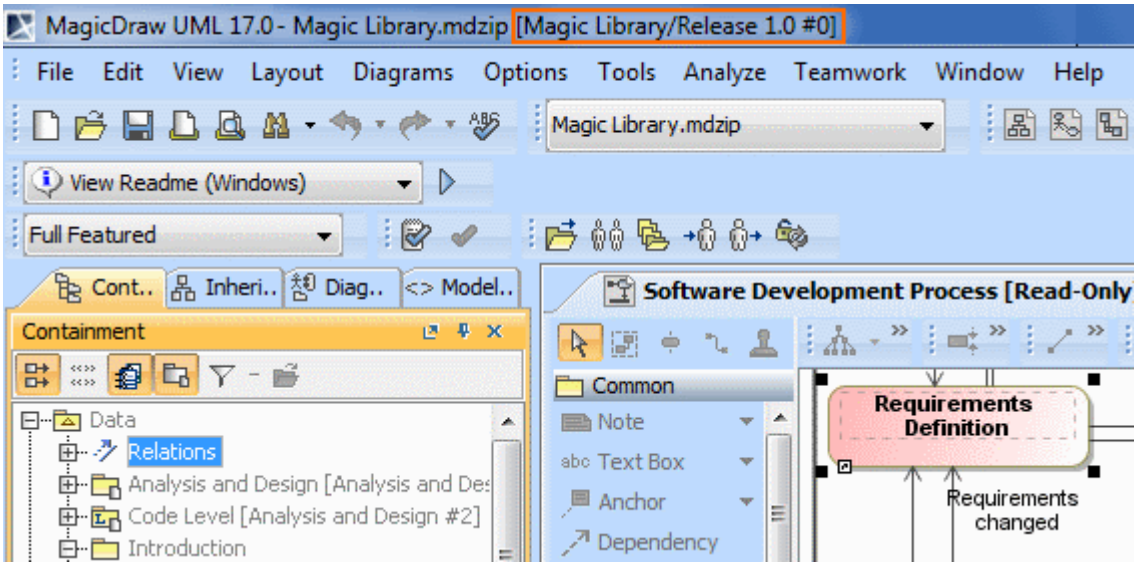
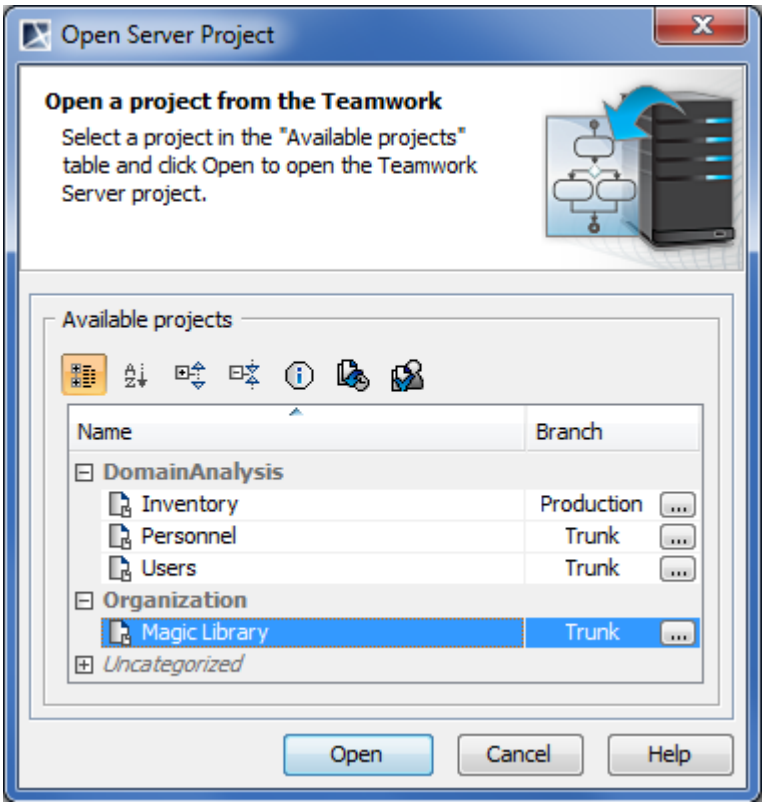


Figure 660 -- Project title of main window: project name, branch name, and project version notation

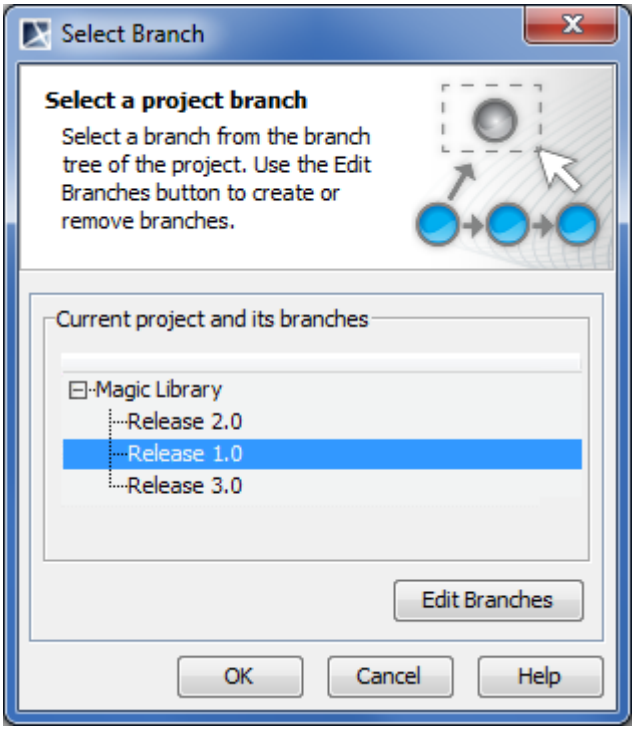
Copyright © 1998-2015 No Magic, Inc.

To quickly open the latest version of a branch using the **Open Server Project** dialog

1. From the **Collaborate** menu, select **Open Server Project**. The **Open Server Project** dialog opens.



2. Select a project.
3. Press the **...** button. The **Select Branch** dialog opens.



4. Select a project branch and click **OK**.
5. In the **Open Server Project** dialog, click **Open**. The project of the selected branch opens.

Copyright © 1998-2015 No Magic, Inc.

The branch opened at the latest is recorded in the **Open Server Project** dialog. You can see the branch name in the **Branch** column. The default branch value is Trunk.

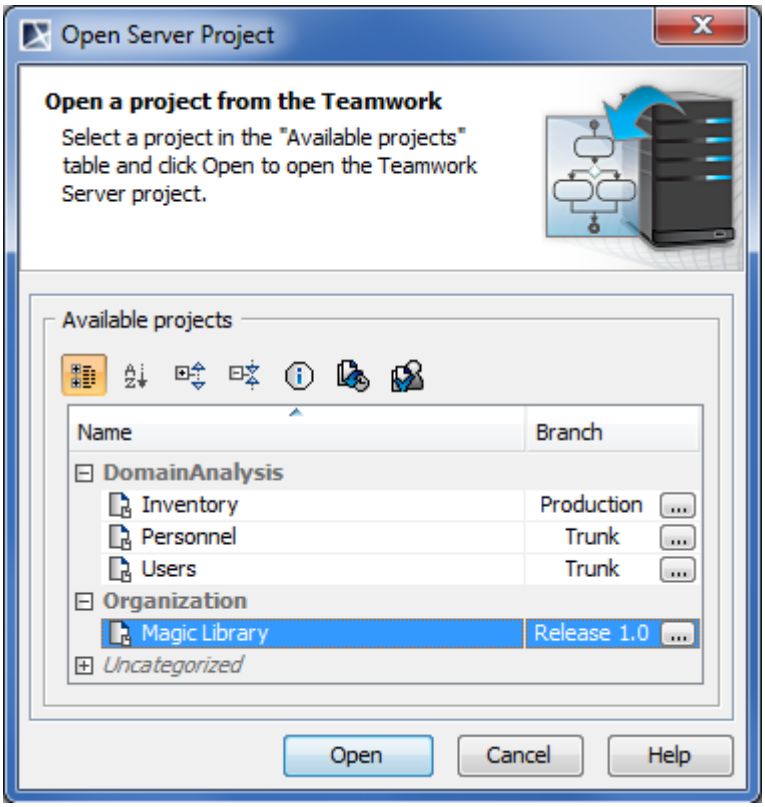


Figure 661 -- Project and its branch selected to open

To branch modules together with project

IMPORTANT!

You must have the Administer project permission for branching. For more information about the permissions, see “User permissions” in MagicDraw Teamwork UserGuide.pdf.

1. Open **Project Versions** dialog.

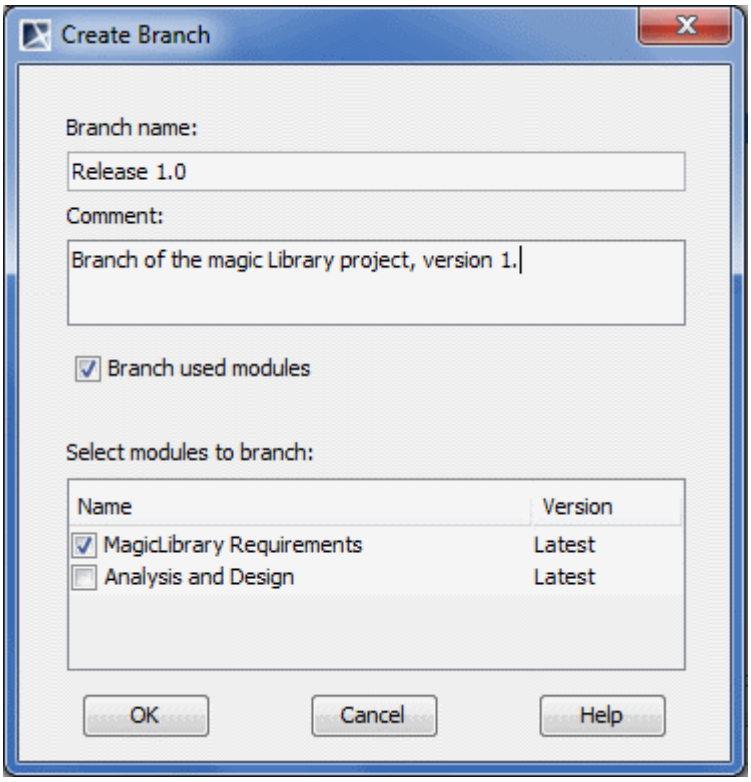
   TIP!

   For more information about creating branches, see the procedure "To create a project branch" on page 1079.

2. In the **Project Versions** dialog, select the project version and click the **Create Branch** button. The **Create Branch** dialog opens.

Copyright © 1998-2015 No Magic, Inc.

3. Select the **Branch Used Modules** check box. The **Select modules to branch** area appears in the dialog.



4. In the **Select modules to branch** list, select the module to be branched together with the project.
5. Click **OK** when you are done.

The workflow of project branching when modules are branched together with the project is depicted in Figure 657 on page 1078.

Related references

Manage Projects dialog

Open Server Project dialog

Teamwork Dialogs

To activate the Teamwork interface, connect to Teamwork Server.

This chapter introduces Teamwork dialogs:

- Manage Projects dialog
- Edit Users dialog
- Add/Edit User dialog
- Open Server Project dialog
- Rename dialog
- Commit Project to the Server dialog
- Project Versions dialog
- Version Information dialog
- Use Server Module wizard

Copyright © 1998-2015 No Magic, Inc.

Manage Projects dialog

You can use the **Manage Projects** dialog to manage projects, assign them to other Teamwork users, and set user permissions for the system and specific projects.

To open the **Manage Projects** dialog

- From the **Collaborate** menu, select **Projects**.
- On the Collaboration toolbar, click the Projects button.

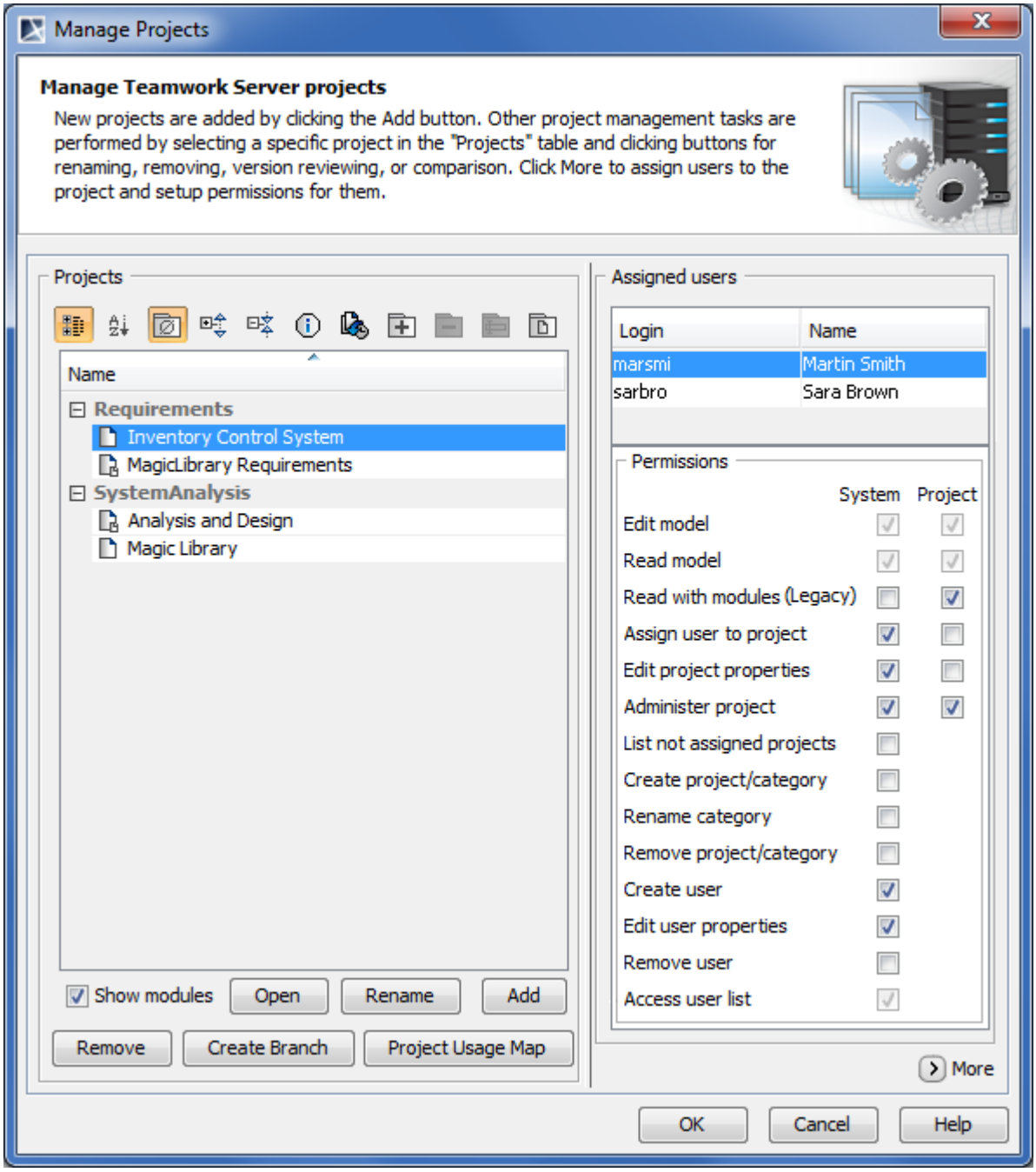


Figure 662 -- Manage Projects dialog

Copyright © 1998-2015 No Magic, Inc.

Element name	Description
Buttons of the Category toolbar	
Categorized view 	Displays the categorized structure of the project list.
Alphabetical view 	Displays the plain project list without any category.
Show Empty Categories 	Shows the categories that have no projects.
Expand 	Expands all categories. The button is available only in the categorized view.
Collapse 	Collapses all categories. The button is available only in the categorized view.
Project Info 	Opens the **Versions** dialog. In this dialog you can see project's information, such as comment, tags, modules, and meta information. The button is available only when the project is selected.
Project History 	Opens the **Project Versions** dialog. The button is available only when the project is selected. For more information about Project Versions dialog, see "Project Versions dialog" on page 1095.
Create Category 	Opens the **Create Category** dialog for creating a new category. The button is available only in the categorized view. For more information about creating categories, see the procedure "To create a new category" on page 1044.
Remove Category 	Opens the dialog asking to confirm or cancel the removal of the selected category. The button is available only in the categorized view, when the category or the project is selected. For more information about removing categories, see the procedure "To remove the category" on page 1045.
Rename Category 	Opens the **Rename Category** dialog for renaming the selected category. The button is available only in the categorized view, when the category or the project is selected. For more information about renaming categories, see the procedure "To rename a category" on page 1044.
Move to Category 	Opens the **Move to Category** dialog for choosing the category to move the selected project into. The button is available only in the categorized view, when the project is selected. For more information about project moving to category, see the procedure "To move a project to another category" on page 1044.
Columns of the Project list	
Name	• The name of the project. • The name of the category.

Copyright © 1998-2015 No Magic, Inc.

Element name	Description
Buttons of the Project list	
Show modules	Shows those projects that contain one or more shared packages.
Open	Opens the selected project.
Rename	Opens the **Rename** dialog for reaming the project. The button is available only when the project is selected.
Add	Opens the **Add Server Project** dialog for creating a new server project. For more information about creating new server projects, see the procedure "To create a new server project" on page 1045.
Remove	Opens the dialog asking to confirm or cancel the removal of the project from the Teamwork Server. The button is available only when the project is selected.
Create Branch	Opens the **Create Branch** dialog for creating a new branch for the selected project or its branch. The button is available only when the project is selected. For more information about creating project branches, see the procedure "To create a project branch" on page 1079.
Project Usage Map	Opens the Project Usage Map of the selected project. The button is available only when the project is selected. For more information about using the project usage map, see "Project Usage Map" on page 1104
Columns of the Assigned users list	
Login	The login name of the user assigned to the selected project.
Name	The full name of the user assigned to the selected project.
Permissions list	A list of permissions given to the selected user for a specific project. For more details about user permissions, see “User permissions” in MagicDraw Teamwork UserGuide.pdf.
Columns of the Available users list	
Login	The Teamwork user's login name.
Name	The Teamwork user's full name.
Buttons between the user lists	
<<	Moves the selected users from the **Available Users** list to the **Assigned Users** list.
>>	Move the selected users from the **Assigned Users** list to the **Available Users** list.

Related procedures

Creating server projects

Project Branching in Teamwork

Creating and managing categories

Versioning

Related references

Project Versions dialog

Edit Users dialog

To open the **Edit Users** dialog

- From the **Collaborate** menu, select **Users**.

Copyright © 1998-2015 No Magic, Inc.

- On the Collaboration toolbar, click the button.

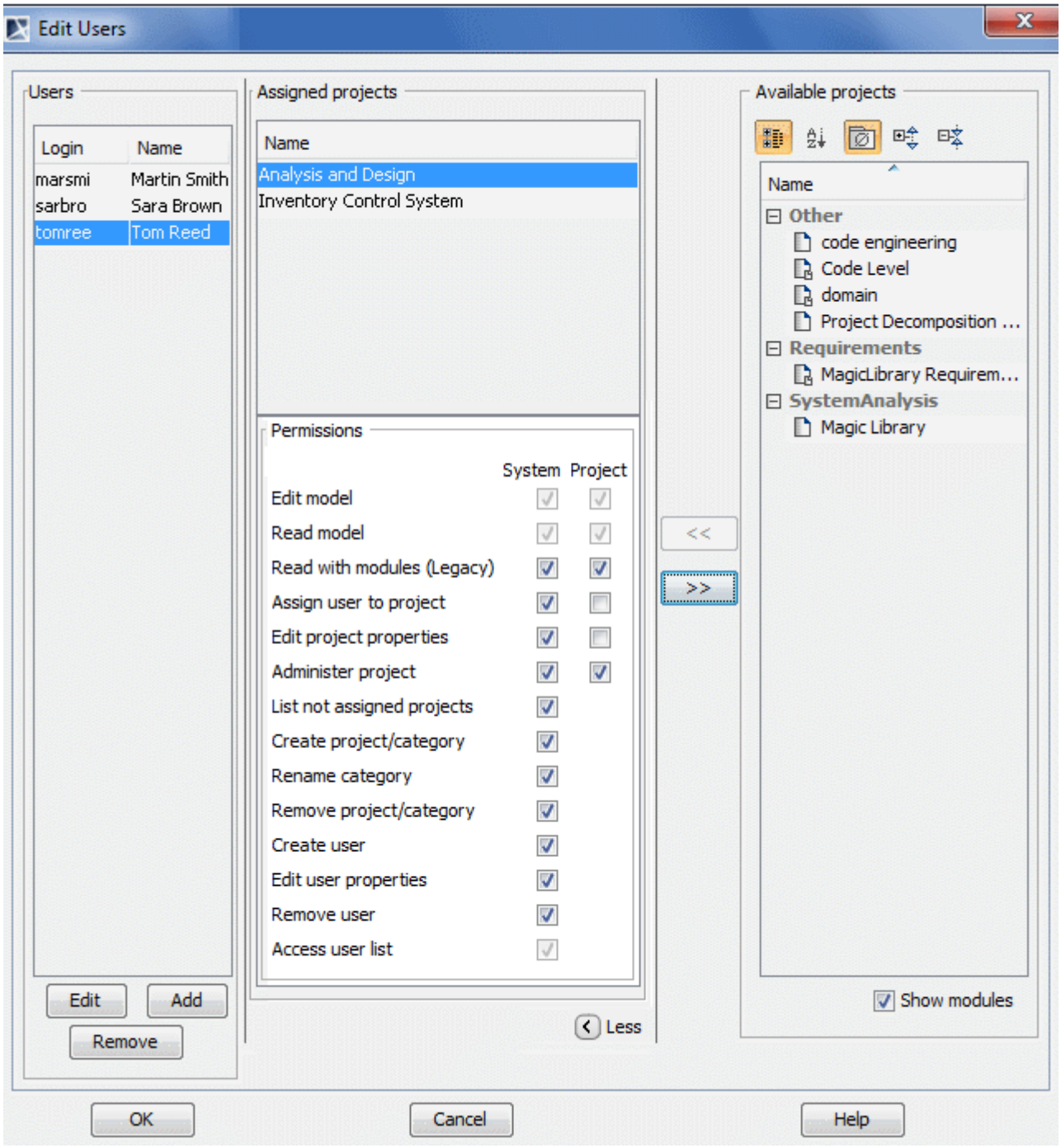


Figure 663 -- Edit Users dialog

Group box/Button	Element name	Function
Users	Login	The user login name.
	Name	The user's full name.
	Edit	Edit the user's name and password.
	Add	Add a new user to the Teamwork Server.
	Remove	Remove a user from the Teamwork Server.
Assigned Projects	Name	List projects assigned to specific users.

Copyright © 1998-2015 No Magic, Inc.

Group box/Button	Element name	Function
Permissions	-	The detailed description of permissions. See “User permissions” in MagicDraw Teamwork UserGuide.pdf.
<<	-	Move the selected project from the **Available Projects** list to the **Assigned Projects** list.
>>	-	Move the selected project from the **Assigned Projects** list to the **Available Projects** list.
Available Projects	Name	List all available server projects.

Related references

Add/Edit User dialog

Add/Edit User dialog

The **Add User** dialog is used for creating a new Teamwork user, and the **Edit User** dialog is used for editing the Teamwork user appropriately. Both dialogs have the same structure, only the **Login** field in the **Edit User** dialog is not editable.

To open the **Add User** dialog

1. From the Collaborate menu, select **Users**.
2. Click the **Add** button.

To open the **Edit User** dialog

1. From the **Collaborate** menu, select **Users**.
2. In the **Users** list, select the user and click the **Edit** button.

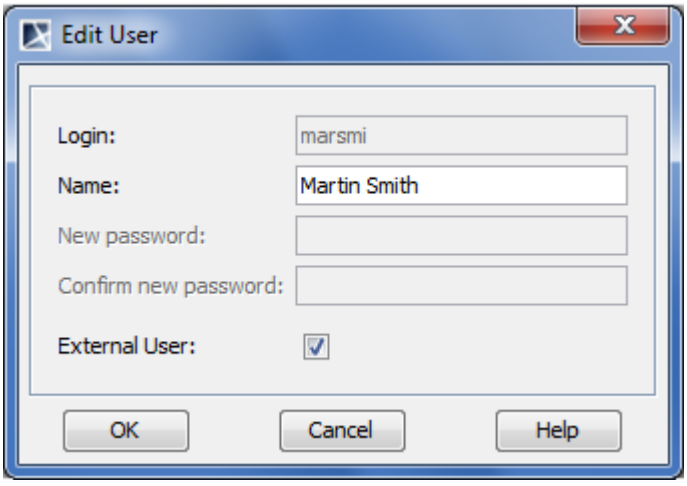


Figure 664 -- Edit User dialog

Element name	Description
Login	The login name of the Teamwork user. **NOTE:** Changing the user’s login name is not allowed.
Name	The full name of the Teamwork user.

Copyright © 1998-2015 No Magic, Inc.

Element name	Description
External User	The check box which indicates whether the user is a native user or an external user. Do one of the following: • Leave the check box deselected for the native user. • Select the check box for the external user. For detailed information about adding native and external users, see the procedure “To add a new native user” and the procedure “To add a new external user” in MagicDraw Teamwork UserGuide.pdf.
New password	A new password. The field will be visible if you add or edit a native user (the **External user** check box is clear). **NOTE:** To create or change a password for an external user, use an appropriate tool for managing the external database (Subversion, ClearCase or LDAP) wherein the user’s account is stored.
Confirm new password	The new password retyped. The field will be visible if you add or edit a native user (the **External user** check box is clear).

Related references

Edit Users dialog

Open Server Project dialog

The **Open Server Project** dialog is used for opening projects and managing them as well.

To open the **Open Server Project** dialog

- From the **Collaborate** menu, select **Open Server Project**.
- On the Collaboration toolbar, click the  button.

Copyright © 1998-2015 No Magic, Inc.

- Press CTRL+SHIFT+O.

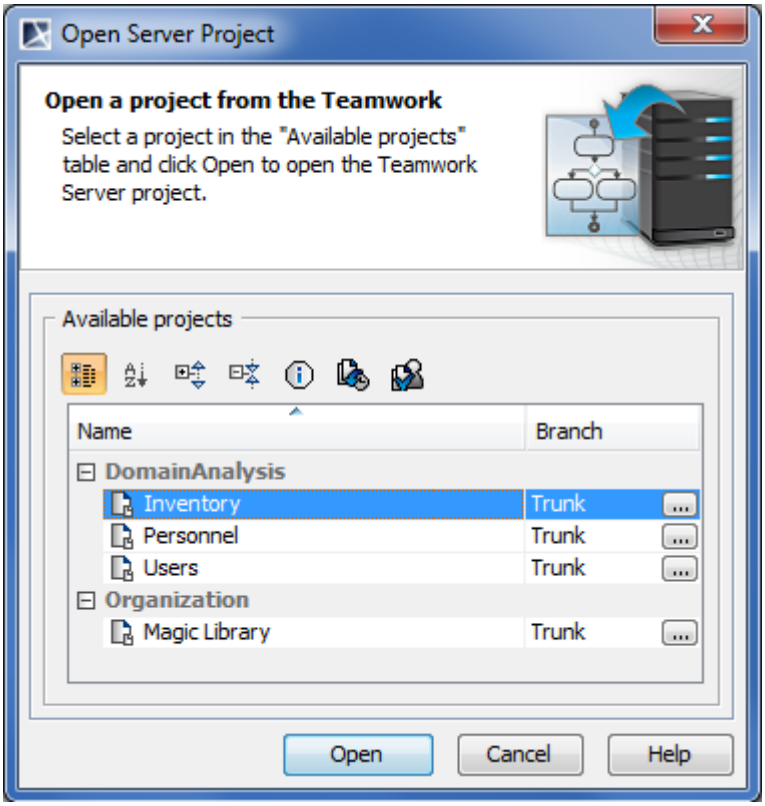


Figure 665 -- Open Server Project dialog

Element name	Description
Buttons of the Category toolbar	
Categorized view 	Displays the categorized structure of the project list.
Alphabetical view 	Displays the plain project list without any category.
Expand 	Expands all categories. The button is available only in the categorized view.
Collapse 	Collapses all categories. The button is available only in the categorized view.
Project Info 	Opens the **Versions** dialog. In this dialog you can see project's information, such as comment, tags, modules, and meta information. The button is available only when the project is selected.
Project History 	Opens the **Project Versions** dialog. The button is available only when the project is selected. For more information about project versions, see: • "Versioning" on page 1071. • "To open the Project Versions dialog" on page 1095.

Copyright © 1998-2015 No Magic, Inc.

Element name	Description
Show only assigned 	Show only the projects that are assigned to the current user.
Columns of the Project list	
Name	• The name of the project. • The name of the category.
Branch	The name of the project branch, last selected using the **"..."** button, or the value "Trunk", if the project isn't branched or if the project itself was last selected.
Common buttons	
Open	Opens the selected project.

Related procedures

Creating server projects
Managing branches
Creating and managing categories
Versioning

Related references

Project Versions dialog

Rename dialog

The **Rename** dialog is used for changing the name of a project or module that is used in the project. In server projects, it is absolutely safe to change module names as modules are identified by a module ID in the Teamwork server, not by a module name.

To open the **Rename** dialog

1. Open the **Open Server Project** dialog.
2. Select a project which name you want to change.
3. Click the **Rename** button.

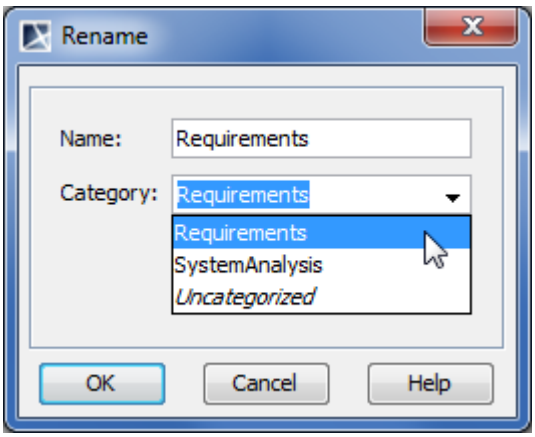


Figure 666 -- Rename dialog

Element name	Function
Name	Type the new project name.

Copyright © 1998-2015 No Magic, Inc.

Element name	Function
Category	Select the category for the project or create a new one.

Related references

Open Server Project dialog

Commit Project to the Server dialog

To open the **Commit Project to the Server** dialog

- From the **Collaborate** menu, select **Commit Project**.
- Press CTRL+K.

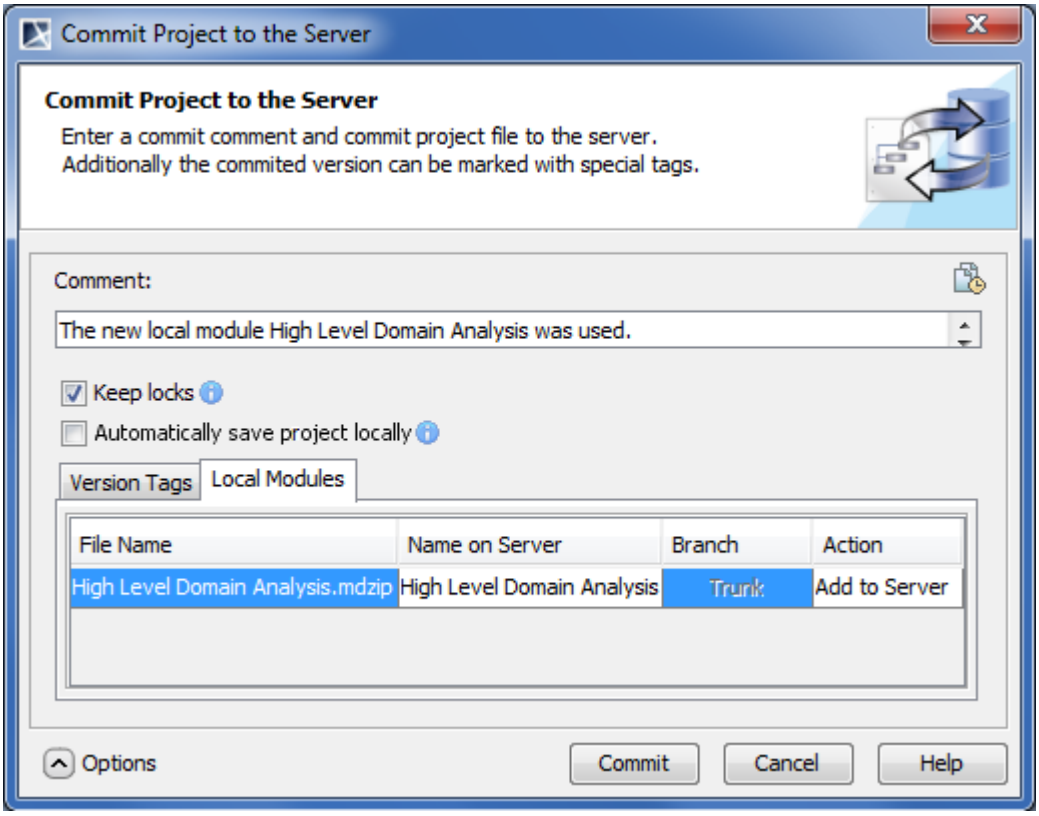


Figure 667 -- Commit Project to the Server dialog

Element name	Function
Choose comment	Opens a Choose comment dialog with the recently added comments. You can select a comment from the dialog.
Comment	Add information about the project, if needed.
Keep locks	Select this option if you want to keep all the currently locked elements and modules locked.
Automatically save project locally	Updates locally saved project with the latest committed version. Select this option, if you want to keep your locally saved project copy up-to-date. **NOTE**. This option becomes available when you save a server project to your computer for the first time.
Version Tags	Add a version tag, if needed.

Copyright © 1998-2015 No Magic, Inc.

Element name	Function
Local Modules	List local modules that were used in the server project. Before committing, select an action you want to perform for modules.
Options	See more or less options.

Related procedures

Committing to Teamwork Server

Project Versions dialog

In this dialog, project versions are listed from the latest one (which is on the top) to the oldest one.

To open the **Project Versions** dialog

1. From the **Collaborate** menu, select **Open Server Project**.
2. In the **Open Server Project** dialog, select a project and click the Project History button .

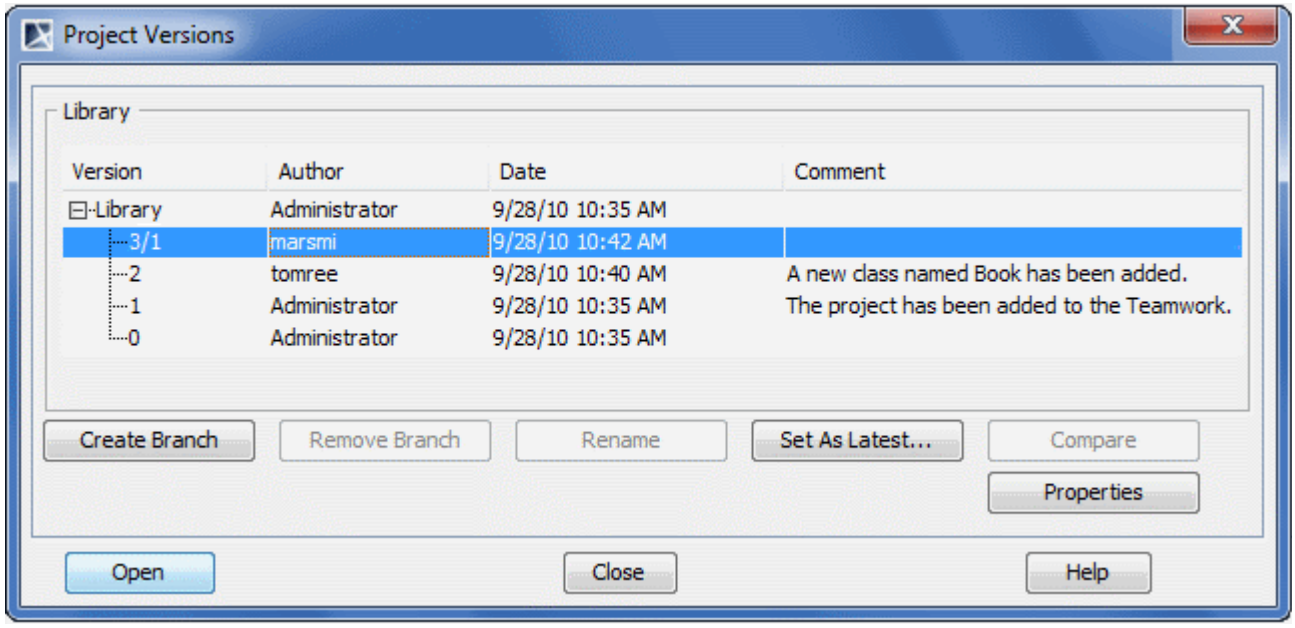


Figure 668 -- Project Versions dialog

Element name	Function
Version	The version number of a project. **NOTE:** In the example above, you can see that the last project version number is denoted as “3/1”. This is the case of setting a previous project version as the latest one. In this particular example, “3” denotes the current project version number and “1” denotes the number of the version, which has been set as the latest one.
Author	The login name of the user, who created the project version.
Date	The date and time, when the project version has been created.
Comment	The description about the changes in the version. You can make a version comment, when committing a version.

Copyright © 1998-2015 No Magic, Inc.

Element name	Function
Create Branch	Opens the **Create Branch** dialog for creating a new branch for the selected version or a branch. The button is available only if a version or a branch is selected. For more information about creating project branches, see section “To create a project branch” on page 1079.
Remove Branch	Opens the dialog asking to confirm or cancel the removal of the selected branch. The button is available only when a branch is selected. **NOTE:** You will not be allowed to undo this action.
Rename	Indicates the branch name for reaming. The button is available only when a branch is selected.
Set As Latest	Set the selected project version as the current version. The created version number consists of two parts: the current version number and the version number that is set as the current version. If a user does not have the right to edit the project, this button is disabled.
Compare	Compares two selected branches. If there are differences between project branches, the result is presented in **Difference Viewer** dialog. The button is available only when two project branches are selected (use SHIFT or CTRL keys).
Properties	Displays information about versions of a project.
Open	Open the selected version of the project. **NOTE:** You can make changes only to the latest version of the project. Earlier versions are read-only.
Close	Exit the dialog.
Help	Display **MagicDraw Help**.

Related procedures

Managing branches

Versioning

Understanding model differences

Related references

Manage Projects dialog

Open Server Project dialog

Version Information dialog

Version Information dialog

To open the **Version Information** dialog

1. Open the **Project Versions** dialog (see the procedure "Project Versions dialog" on page 1095).

Copyright © 1998-2015 No Magic, Inc.

2. Select a project and click **Properties**.

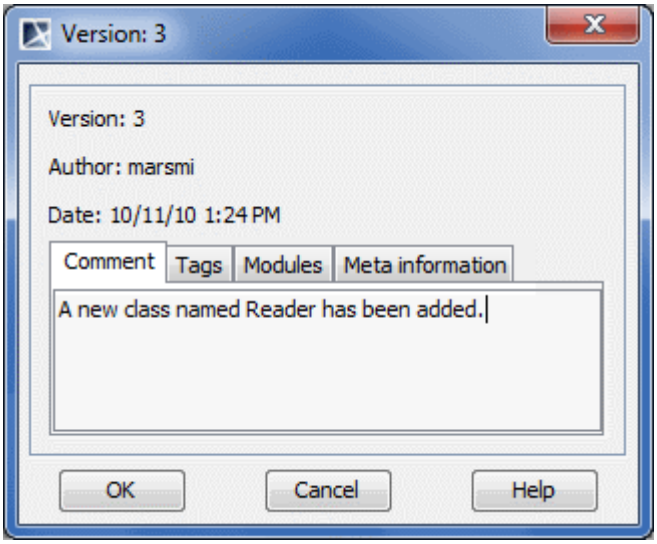


Figure 669 -- Version Information dialog

Element name	Function
Version	The version number of a project.
Author	The author's login name who created the project.
Date	The date and time the selected project version was committed.
Comment	Allow users to type optional descriptions about changes in the committed version. You can enter or edit comments for any project version.
Tags	Shows tags that have been created for the selected project version.
Modules	Displays modules that are used in the selected project version.
Meta information	Displays project's information, such as required plugins and used modules.

Related references

Project Versions dialog

Use Server Module wizard

With a help of the **Use Server Module** wizard you can add a selected server module, such as a library, profile, or other user defined module part.

To open the **Use Server Module** wizard

- On the **Collaborate** menu, click **Server Modules** > **Use Server Module**.
- On the **Options** menu, click **Modules**. In the opened **Modules** dialog, click the **Use Module** button.
- In the **Containment Tree**, open the selected module shortcut menu and click **Modules** > **Module Options**. In the opened **Modules** dialog, click the **Use Module** button.

The wizard consists of two steps:

1. Selecting a module.
2. Specifying selected module settings.

Copyright © 1998-2015 No Magic, Inc.

STEP #1: Selecting a module

After the wizard is opened, you see the list of available server modules. Select the particular server module and do one of the following:

- Click **Next**, if you need to change default module settings. The next wizard step opens.
- Click **Finish**. The selected server module is added to you project.

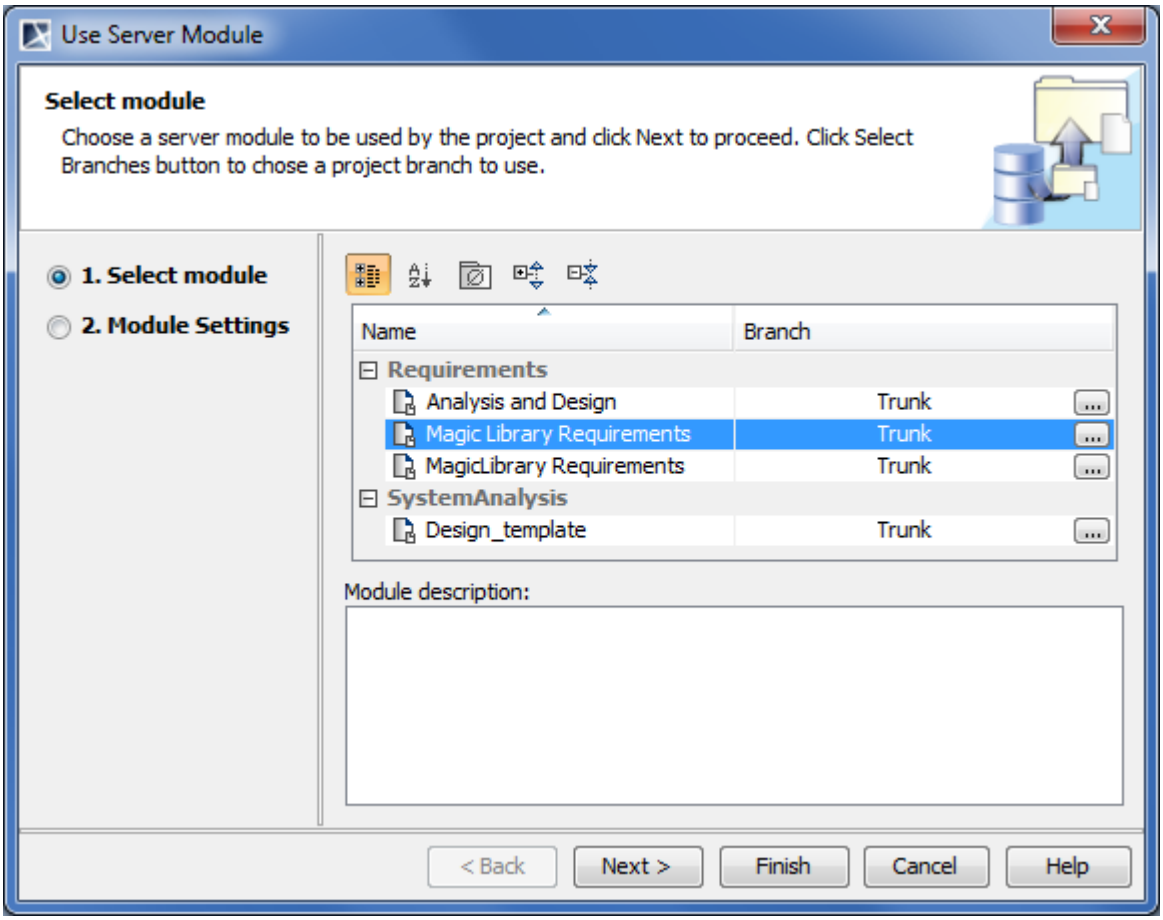


Figure 670 -- Use Server Module wizard. Selecting a server module

Dialog element name	Element type	Description
Name	Text box	A module name.
Version	Text box	A number of the latest version.
Module description	Text box	A description of the last project version.

STEP #2: Specifying module settings

In this step, you can specify shared module accessibility and packages on which the corresponding module share is placed.

Copyright © 1998-2015 No Magic, Inc.

Click **Finish** after you have specified module settings. The selected server module is added to you project

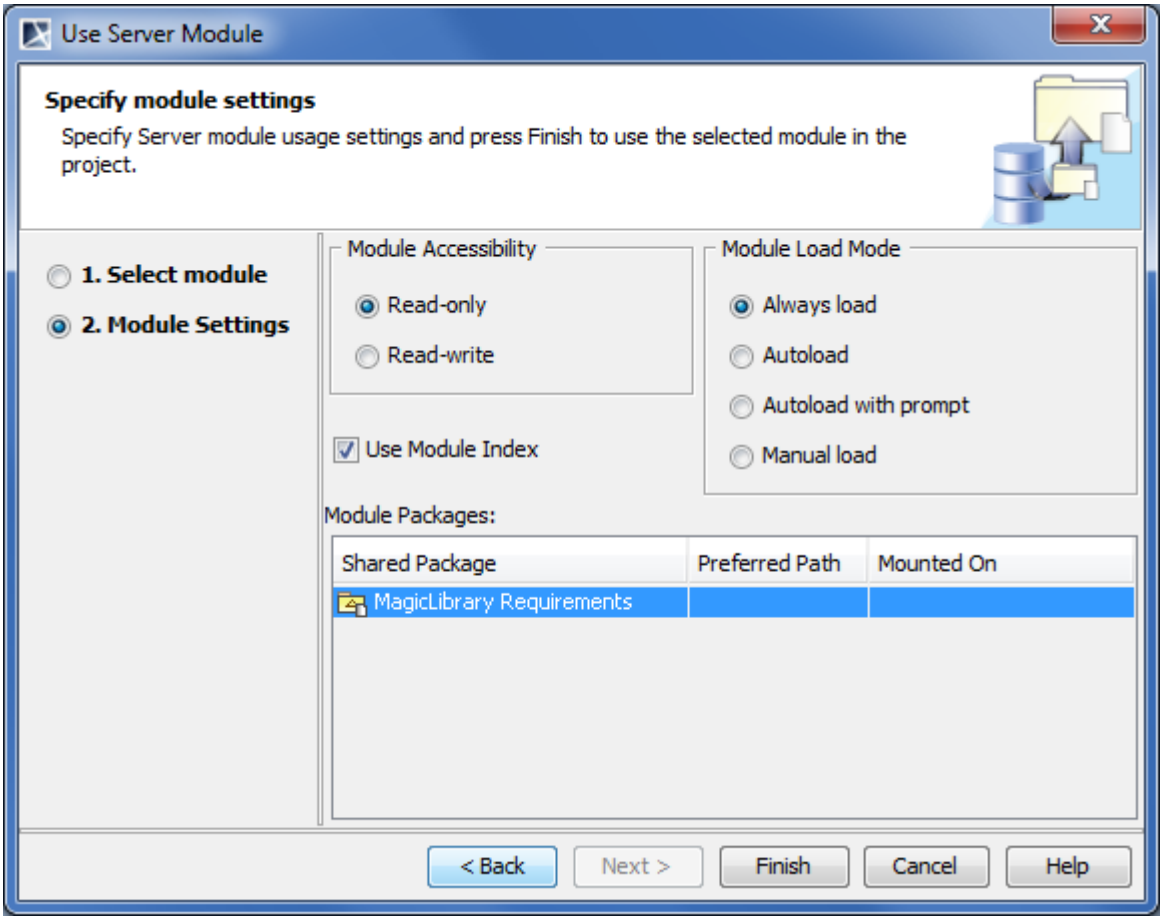


Figure 671 -- Use Server Module wizard. Specifying module settings

Dialog element name	Element type	Description
Module Accessibility	Option button	Specifies the way a module can be accessed in a project: • **Read-only**. Modules are loaded for reading only and cannot be edited in the project. Elements of the module can be used and referenced in the project. Usually, libraries are used in the read-only accessibility mode in projects. • **Read-write**. Modules can be directly edited in the project wherein the modules are used.

Copyright © 1998-2015 No Magic, Inc.

Dialog element name	Element type	Description
Modules Load Mode	Option button	By selecting an appropriate load mode, you can save a memory used by a program and project: • **Always load** (default). A selected module is always loaded when the project is opened. • **Autoload**. A selected module is not loaded when the project is loaded. MagicDraw monitors user activities in the project and loads a particular module on the demand by the project • **Autoload with prompt**. This mode is similar to an autoload mode. The difference is that MagicDraw asks the user a confirmation before loading it. • **Manual load**. A selected module is not loaded when a project which uses this module is loaded. The model integrity is not broken, as all required elements of the module exists, just simplified versions (that is, loaded as proxies) of the elements are used in the project. This load mode is recommended for all modules that are stable or rarely modified. **NOTE:** Reload the project after changing the module load mode.
Use Module Index	Check box	Select this check box to load indexed elements from the not loaded module. This is valid only when the module is not loaded. Using indexed elements in the project increases the performance, as only simplified version (that is, proxies) of elements are loaded. For more information about indexing, see "Indexing" on page 151.
Module Packages	Table	This table lists all shared packages of the module and paths where these packages are mounted in the project.
Shared Package	Text box	A name of a shared package.
Preferred Path	Text box	Displays a suggestion where to mount the package in the project. This location is used by default if not specified otherwise.
Mounted On	Text box	Displays the real location where the package is currently mounted in the project. Click the **...** button to change a package location.

Related concepts

Data Partitioning on Teamwork Server

Related procedures

Working with server modules

Adding projects to server

Data Partitioning on Teamwork Server

Data partitioning is the segmentation of a MagicDraw project into multiple independent parts or modules. Since these modules are saved using the same file format as regular MagicDraw projects, they can be opened and edited in any MagicDraw client application. This approach allows easy reuse of the same basic model libraries in multiple projects.



Please notice that a server project opens module versions that were used on a project commitment but not the latest module version.

Copyright © 1998-2015 No Magic, Inc.



For more information about data partition, see "Project Partitioning" on page 147.

Related concepts

Project Partitioning

Related procedures

Creating and managing categories

Related references

MOF Support

Use Server Module wizard

Creating server modules

To create a server module using the **Collaborate** menu

1. Open a server project.
2. From the **Collaborate** menu, select **Server Modules** and then click **Export Server Module**. The **Export Module** dialog opens.



You cannot export a package as a module if another module is mounded inside.

3. In the **All data** list, select one or more packages from which you want to create a server module and click the **Add** button. The packages will be added to the **Selected packages** list.
4. Click **OK**.



In this step you may be asked to confirm or cancel the dependency checking between the exported packages and the rest of the project (including shared packages that belong to the project and used modules). The dependency analysis step is described in "Use Project wizard" on page 153.

5. Enter the name of the module for Teamwork Server, if there is such need.
6. Either select an existing category or type a new category name.
7. Click **OK**. The module will be created in Teamwork Server as a separate file.

To create a server module using the package shortcut menu

1. Open a server project.
2. In the Containment tree, select one or more packages from which you want to create a server module.
3. From the package shortcut menu, select **Modules** and then click **Export Server Module**. The **Export Module** dialog opens. The packages you have selected in the Containment Tree are added to the **Selected Packages** list.



You cannot export a package as a module if another module is mounded inside.

4. You may decide to add more packages. In this case, perform step #3 that is described in the procedure "To create a server module using the Collaborate menu" on page 1101.
5. Perform step #4 and step #5 that are described in the procedure "To create a server module using the Collaborate menu" on page 1101.

Copyright © 1998-2015 No Magic, Inc.

Once the server module is created, all its packages and the elements they contain are read-only. Module name and version number are displayed in the brackets next to each exported package name in the Containment Tree (see the picture below).



Working with server modules

In Teamwork Server, a project can be divided in several parts. You may perform several actions with the part of the model:

- Reuse in other server projects
- Edit it separately.
- Export it as a server module.



You are not allowed to import a module into the server project, if a module has cycle dependencies or it is used both by a project and by the other module that is used in the project.

To use a server module in the server project

1. From the **Collaborate** menu, select **Server Modules** and then **Use Server Module**. The **Use Server Module** dialog opens.
2. Select the module you want to use in your project. Click **Next.**
3. Specify **Module Settings** in the next step and click **OK**.

The module will appear as read-only or read-write. It is now available for use in your project. Open the module as a separate project to edit it.



For a detailed description about editing a server module, see "To edit a server module" on page 1103.

To commit a project containing non-server modules to Teamwork Server

1. From the **Collaborate** menu, select **Commit Project**. The **Commit Project to the Server** dialog opens (see "Commit Project to the Server dialog" on page 1094).
2. If you want to save this local module as a separate server module, perform the following steps:
    3. Click the **Local Modules** tab.
    4. In the **Action** list, select **Add to Server**. Perform this step for all modules you want to add to Teamwork server.



MagicDraw standard modules cannot be added to server as they already exist there.

5. Click **Commit**.

Copyright © 1998-2015 No Magic, Inc.

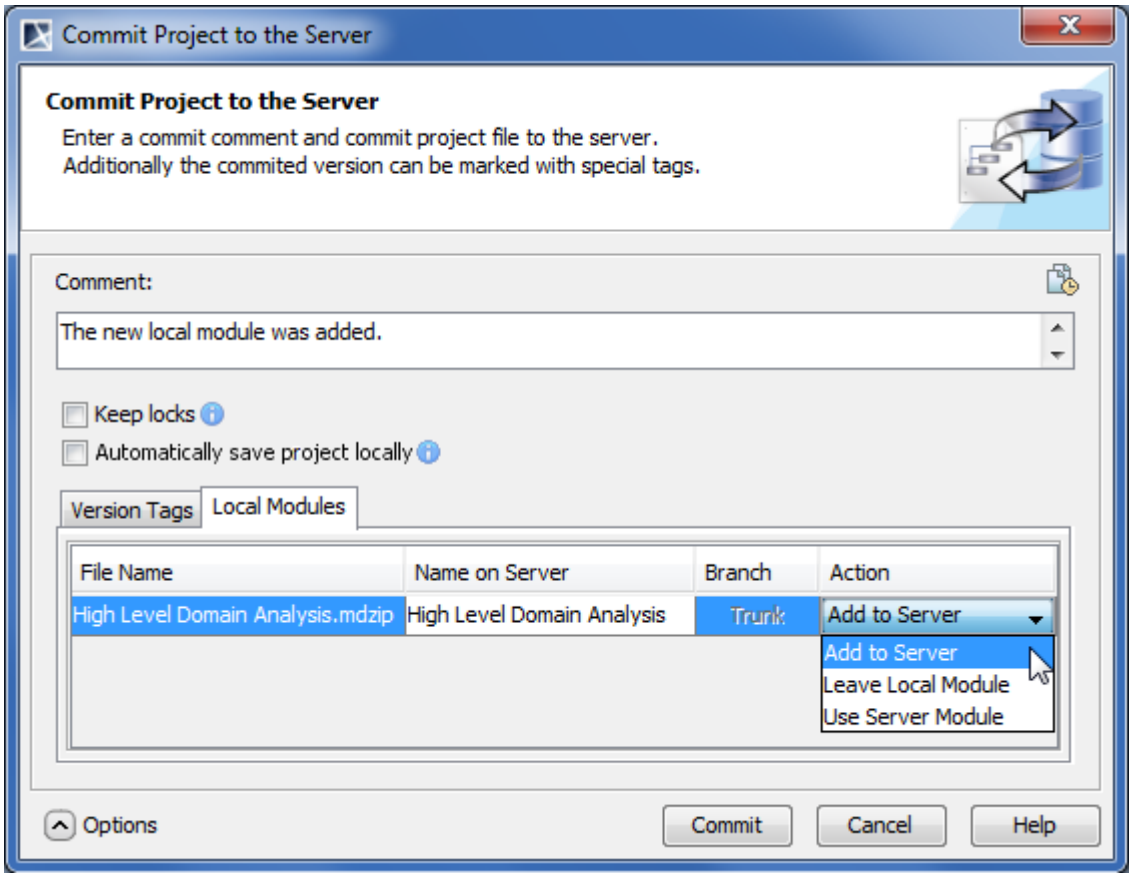


Figure 672 -- Local Modules tab on Commit Project to the Server dialog

Now the project is committed to the Teamwork Server, and the modules are created.

To edit a server module

1. In the Containment tree, select a module.
2. From the module shortcut menu, select **Modules** > **Open Module As Project**. The module is opened as a new project.
3. Lock the module elements you want to edit.

   TIP! For more information about locking, see "Locking model elements and diagrams for editing" on page 1055.
4. When unlocking an edited element, the **Commit Project to the Server** dialog opens. Click **Commit**.

A new version of this module on the Teamwork Server is created.

IMPORTANT! Please notice that a server project opens module versions that were used on a project commitment but not the latest module version.

Copyright © 1998-2015 No Magic, Inc.

To reopen a server module

- From the module shortcut menu, select **Modules** > **Reload Module.** The version number of the latest version is shown beside the module name (see the following figure).



This procedure allows you to use a new version of a module that has been uploaded to the Teamwork Server. You can use reloaded module in your own server project.

Project Usage Map

The Project Usage Map is a live visual graph that represents Teamwork Server project usages as well as identifies potential problem areas.

The Project Usage Map allows for representing projects and their dependencies in two views:

- All Projects view that shows all projects and all the dependencies among them.
- Individual project view that shows a particular project along with other directly and indirectly used modules.

Using the Project Usage Map you can easily do the following:

- Identify, analyze, and validate dependencies among projects (for example, you can find out easily all the projects, wherein a particular module is used).

Copyright © 1998-2015 No Magic, Inc.

- Identify cyclic dependencies among projects. Identify and fix inconsistent dependencies among projects.

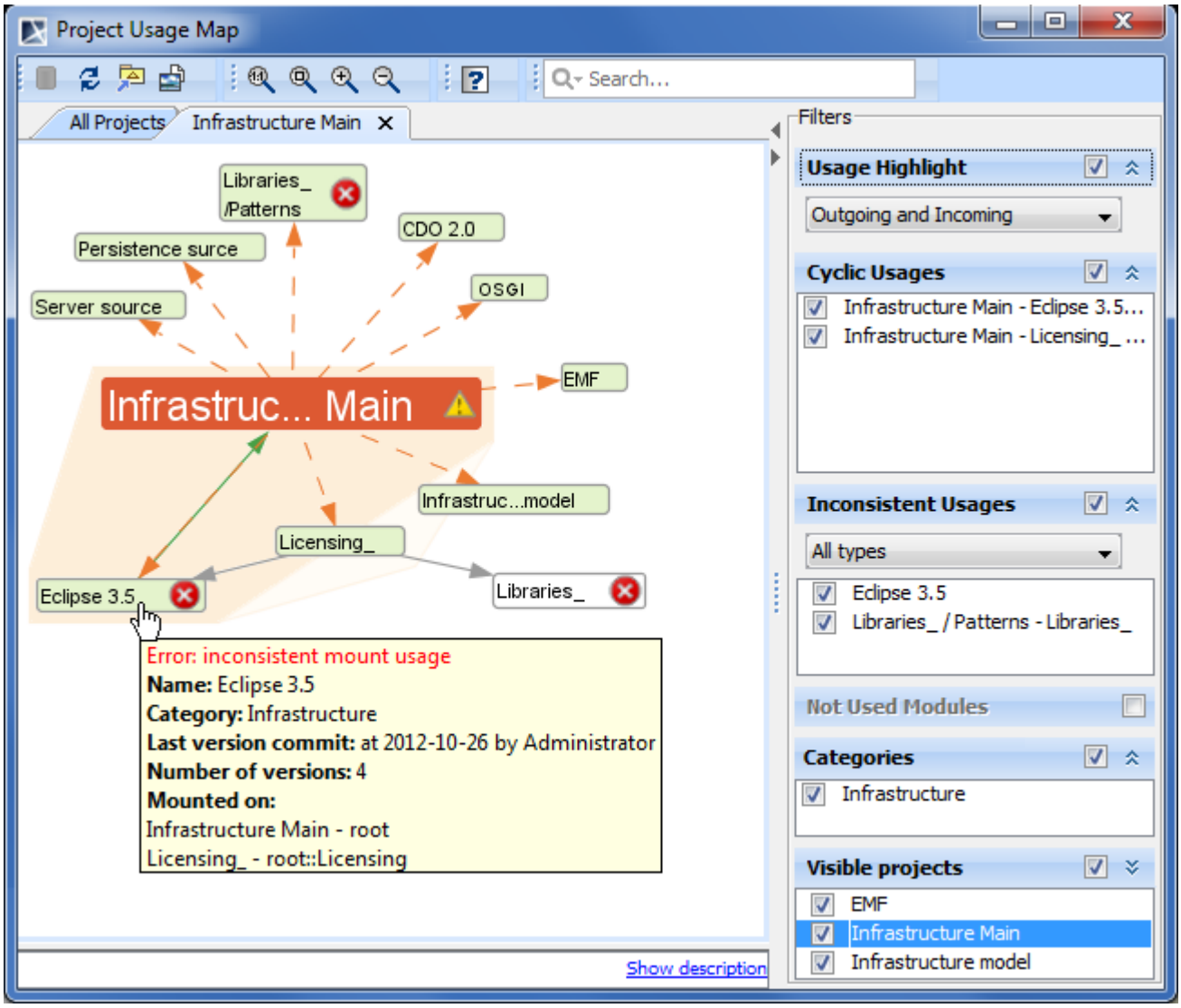


Figure 673 -- Example of Project Usage Map

Concepts

Cyclic usages

Several parts of the project are directly or indirectly using the main project.

Inconsistent mount

The project is mounted on different packages of the main project.

Inconsistent version or branch usage

The main project uses different versions or branches of the same module.

Module

A project with shared parts.

Not converted project

A project whose usage information cannot be retrieved and analyzed by the Project Usage Map since it was created using MagicDraw version 17.0 or earlier. Open the project and add it back to the Teamwork Server so it will be converted into the newest format.

Copyright © 1998-2015 No Magic, Inc.

Non-sharing usage

A usage that makes a project invisible to other projects despite their using it as a module.

Sharing usage

A usage that makes a project visible to other projects wherein this project is used as a module.

Unconfirmed usage

The usage has been created automatically and is not yet confirmed by the user.

Unused modules

The project that has shared parts that are not used in any other project.

Working with Project Usage Map

MagicDraw provides Project Usage Map to identify, analyze, and validate dependencies between projects. It is enough to be connected to your Teamwork Server repository from MagicDraw to open Project Usage Map.

Using the Project Usage Map, you can:

- Review the content of the entire Teamwork Server repository in the All Projects view.
- Narrow the analysis scope by hiding in the view any types of conflicts and / or projects, in which you have no interest.
- Review usages of a particular project in the Individual project view.
- Open projects and resolve problems.

To open a current project usage map

Do one of the following:

- From the active project's **Collaborate** menu, select **Project Usage Map** > **Current Project Map**.
- In the **Edit Projects** dialog, select the project and then click the **Project Usage Map** button.

To open the all projects usage map

- From the **Collaborate** menu, select **Project Usage Map** > **All Projects Map**.

To identify project usage problems

Do one of the following:

- Select one or several problems in the filter area.
- Select a project with an error icon in the map area.



For more information about problem identification and solutions, see "Solving Project Usage Problems" on page 1113.

In the Project Usage Map, you can easily identify project usages. This means, that you can see which projects uses or are used by the selected project.

To identify project usages

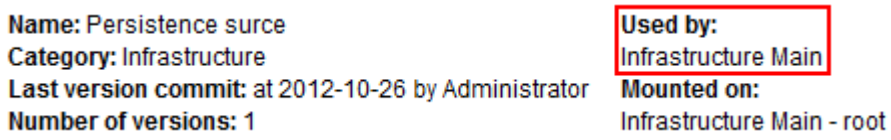
1. In the **Usage Highlight** filter, select one of the following:

Copyright © 1998-2015 No Magic, Inc.

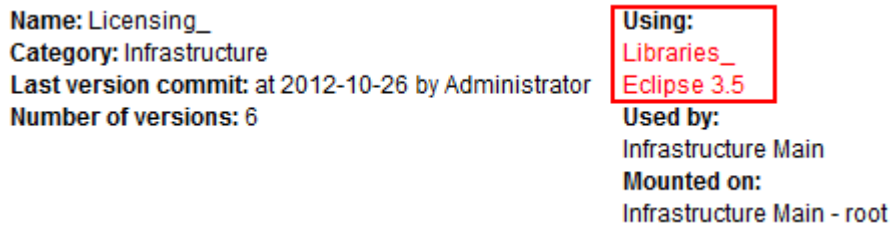
- **Outgoing and Incoming**
- **Outgoing**
- **Incoming**

2. Select any project in the map.

A green arrow represents the incoming usage. The incoming usage is drawn between the selected project and the project that is using it. In the description area, the project that is using the selected project is listed under **Used by**.



An orange arrow represents the outgoing usage.The outgoing usage is drawn between a selected projects and the project that is used by it. In the description area, pre project that is used by the selected project is listed under **Using**.



To narrow results visible in a project usage map

Do one of the following:

- Select a project in the map area and press Delete
- From the project shortcut menu, select Hide
- In the Filter area, click to clear specific project, category, or problem check box
- In the Search box, type the project name.
- Open the specific project's usage map.

You can export a project usage map to the model. This allows you for saving the current server project usages representation in the class diagram and compare it with the latest representation (see the following figure).



For more information about comparing the projects, see "Projects Comparison" on page 579.

To export a project usage map to the model

- In the **Project Usage Map** window toolbar, click the Export to Model button.



Any changes in the exported project usage map will not affect the Project Usage Map view.

Copyright © 1998-2015 No Magic, Inc.

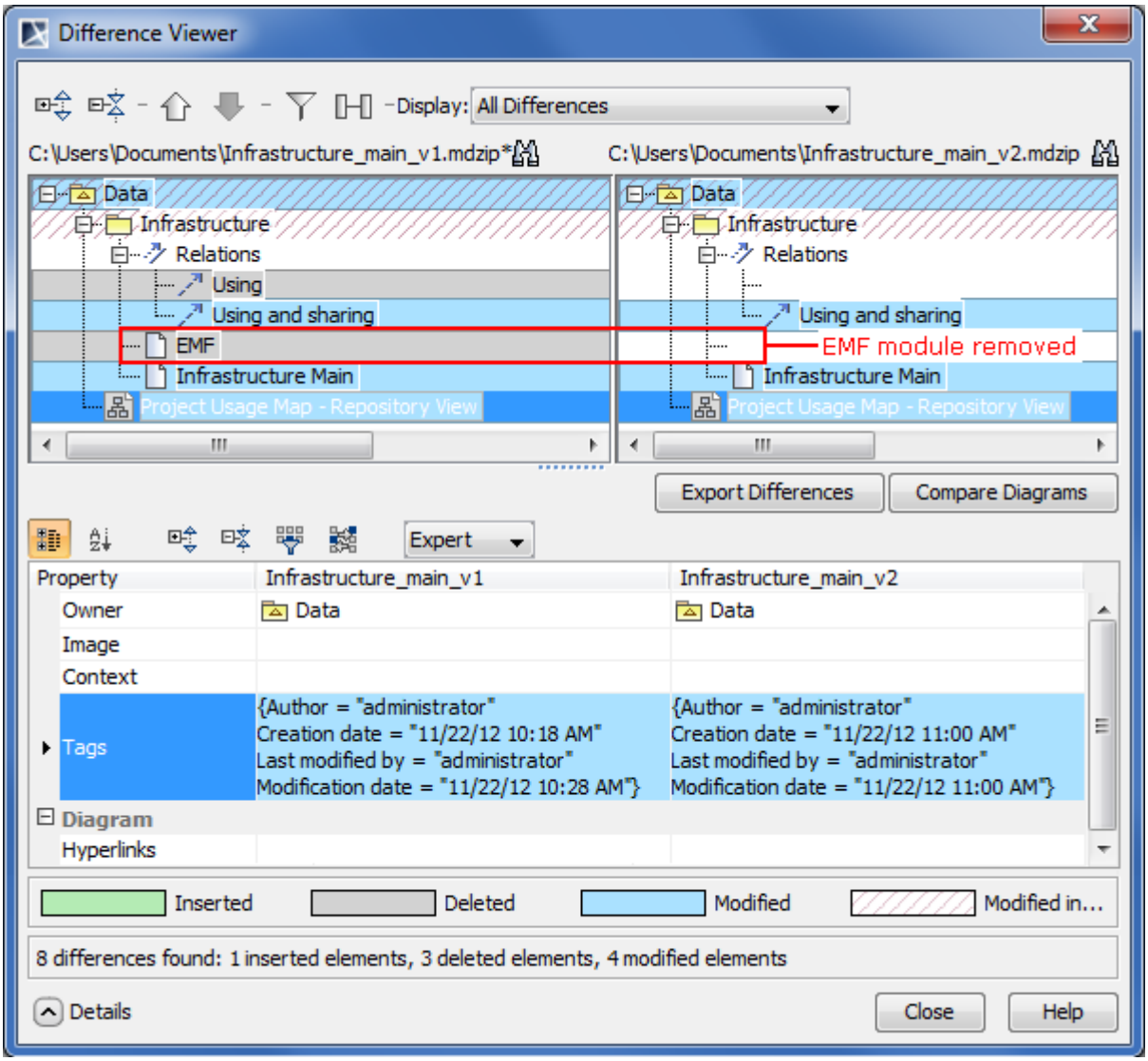


Figure 674 -- Example of exported maps comparison

To save the project usage map as an image

1. In the **Project Usage Map** window, click the Copy as BMP Image button. The map view is copied to the clipboard.
2. Paste in any image editor.
3. Save to the selected location.

To select a particular project in the map

Do one of the following:

- Click a project in the map
- Select a project from the Visible projects filter
- In the Search box, type the project name.
- Select the project and then click the project name in the description area.

TIP!

You can select one of the search criteria:

- Match from start
- Match anywhere

Copyright © 1998-2015 No Magic, Inc.

Refresh the Project Usage Map to see the latest view of the Teamwork Server repository.

To refresh the project usage map

- In the **Project Usage Map** window toolbar, click the Refresh button.



All the layouts and selected filters will be cleared after refreshing.

To open the individual project view

Do one of the following:

- Double-click a project
- Select Show Project Structure from the project shortcut menu

Solve the inconsistent usages and other problems by opening projects directly from the Project Usage Map.

To open as a project

1. Select the project which usages you want to fix.
2. From the shortcut menu, select **Open Project**.

Project Usage Map window

In the Project Usage Map window, you can analyze the contents of the entire Teamwork Server repository, as well as individual projects. You will be able to explore the displayed projects, analyze both project usages and identified problems.

Copyright © 1998-2015 No Magic, Inc.

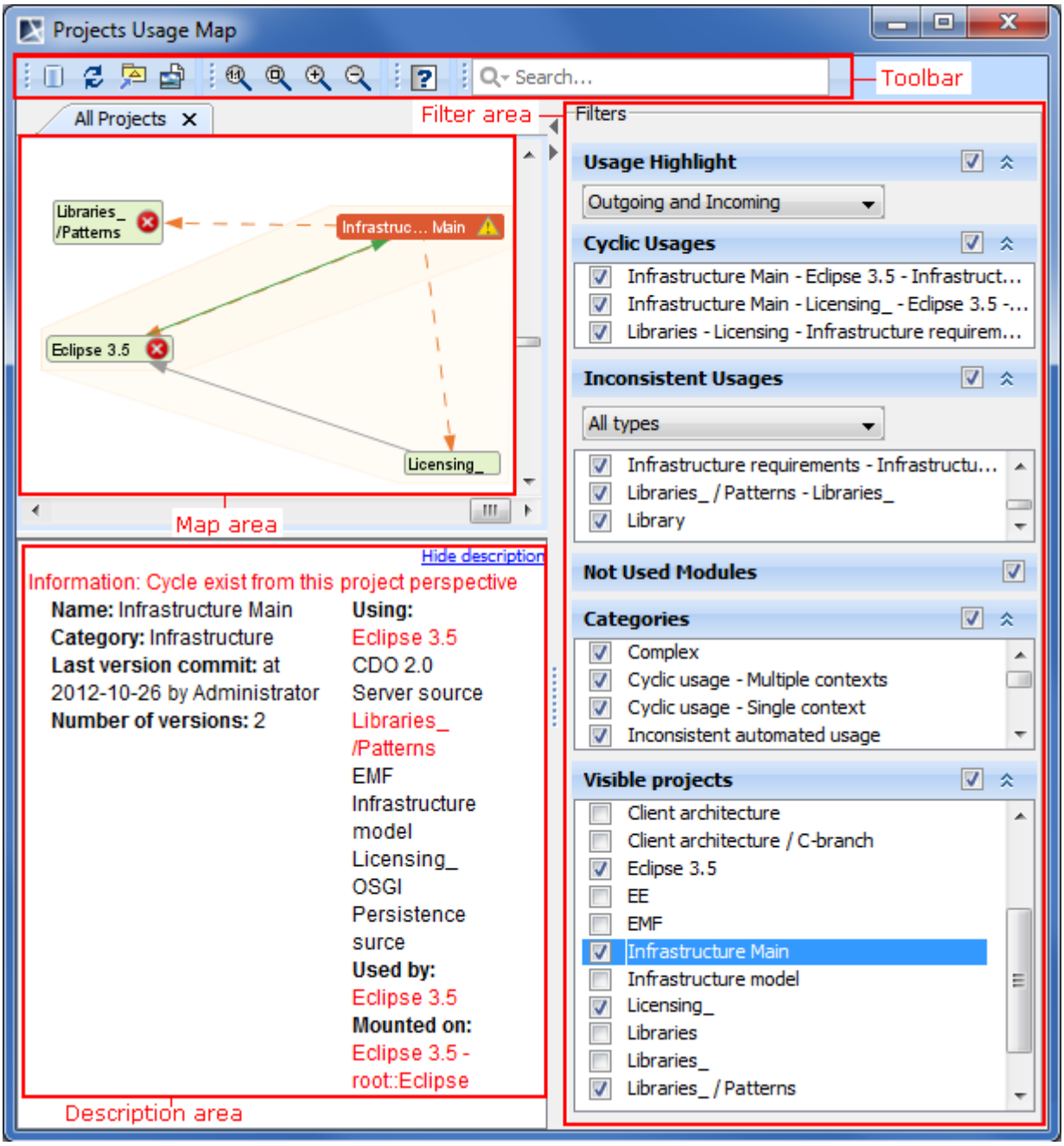


Figure 675 -- Project Usage Map window

Copyright © 1998-2015 No Magic, Inc.

Toolbar

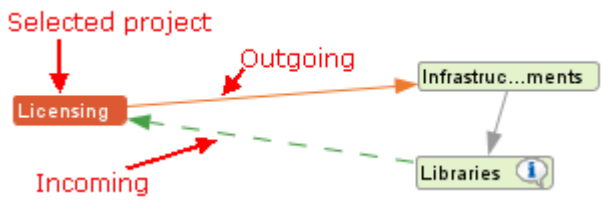
Toolbar element	Description
Open All Projects Map	Click this button, if you closed the All Project view in the Project Usage Map. This button is unavailable if the All Project view is opened.
Refresh	Click this button if you want to see the latest version of the repository view. All the layouts and filters will be reset.
Export To Model	Click this button to export an active usage map view to the Class diagram. You can save the exported usage map for the later comparison of the latest version.
Copy as BMP Image	Click to copy the active usage map. The usage map view is copied to the clipboard. You can paste it in any image editor.

Filters area

Window element	Description
Usage Highlight	Select which usages you want to highlight: incoming, outgoing, or both.
Cyclic Usages	Select which cyclic usages you want to be displayed in the map view.
Inconsistent Usages:	Select which type of inconsistent usages you want to be displayed.
None	Select if no inconsistent usages should be displayed.
Version	Select if you want to see version inconsistencies.
Branch	Select if you want to see branch inconsistencies.
Mount Point	Select if you want to see mount point inconsistencies.
Unconfirmed	Select if you want to see unconfirmed usage inconsistencies.
All Types	Select if you want to see all type inconsistencies.
Not Used Modules	Select if you want to see only the projects with not used modules.
Categories	Specify the particular categories of projects.
Visible Projects	Select the projects you want to be displayed in the map. By default, all the projects are selected. Right-click in the Visible Projects list area and select Show all or Hide all. Press the Delete key to unselect a project.
Description area	See the information of the selected project. All the problems are marked in red.

Map area: understanding the contents

The representation of the selected project, incoming and outgoing usages is depicted in the following figure.



An orange arrow represents outgoing usages. A green arrow represents incoming usages.

Copyright © 1998-2015 No Magic, Inc.

The representation of the Sharing usage is depicted in the following figure by the solid grey arrow.



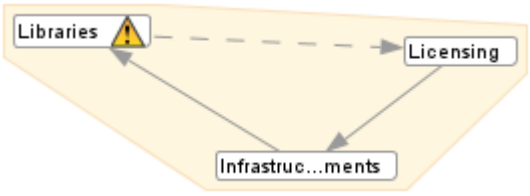
The representation of the Non-sharing usage is depicted in the following figure by the dashed gray arrow.



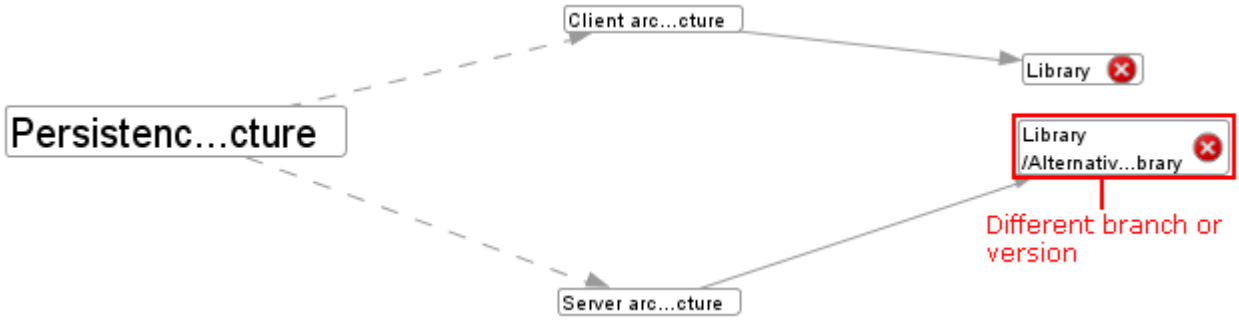
The representation of a Not converted project is depicted in the following figure.



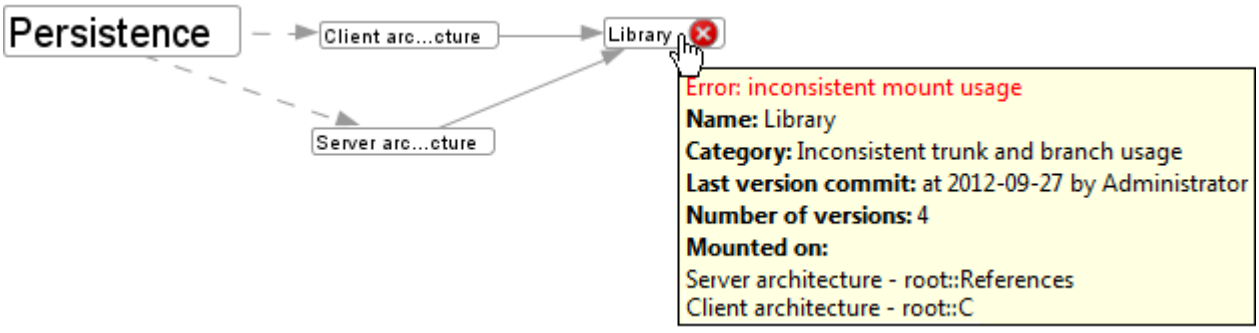
The representation of Cyclic usages is depicted in the following figure.



The representation of a branch or version inconsistencies is depicted in the following figure.

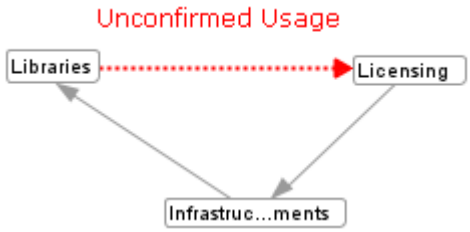


The representation of the mount point inconsistencies is depicted in the following figure.

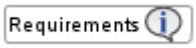


Copyright © 1998-2015 No Magic, Inc.

The representation of Unconfirmed usage is depicted in the following figure.



The representation of Unused modules is depicted in the following figure.



Solving Project Usage Problems

Using the Project Usage Map, you can identify usage problems. Then you can easily select and open projects that need to be fixed.

Resolving cyclic usages

Problem

When a project decomposition is used, the project is split into smaller projects. You can benefit from this decomposed project - you need to load only small part of the project instead of all of it. This reduces complexity and even improves performance.

In addition to this, you would not expect and most likely not care to load the remaining parts of the main project each time you open a single part of it. This can happen if you have cyclic usages i.e. project parts are using the main project. Typically, such usages increase complexity and reduce performance.

Also, if the project takes part in a cycle, it can't be reused as a totally independent part in another project. Other projects taking part in the cycle will be automatically used as well.

Copyright © 1998-2015 No Magic, Inc.

Cycles are often a symptom of unintentional usages, created unknowingly by the user.

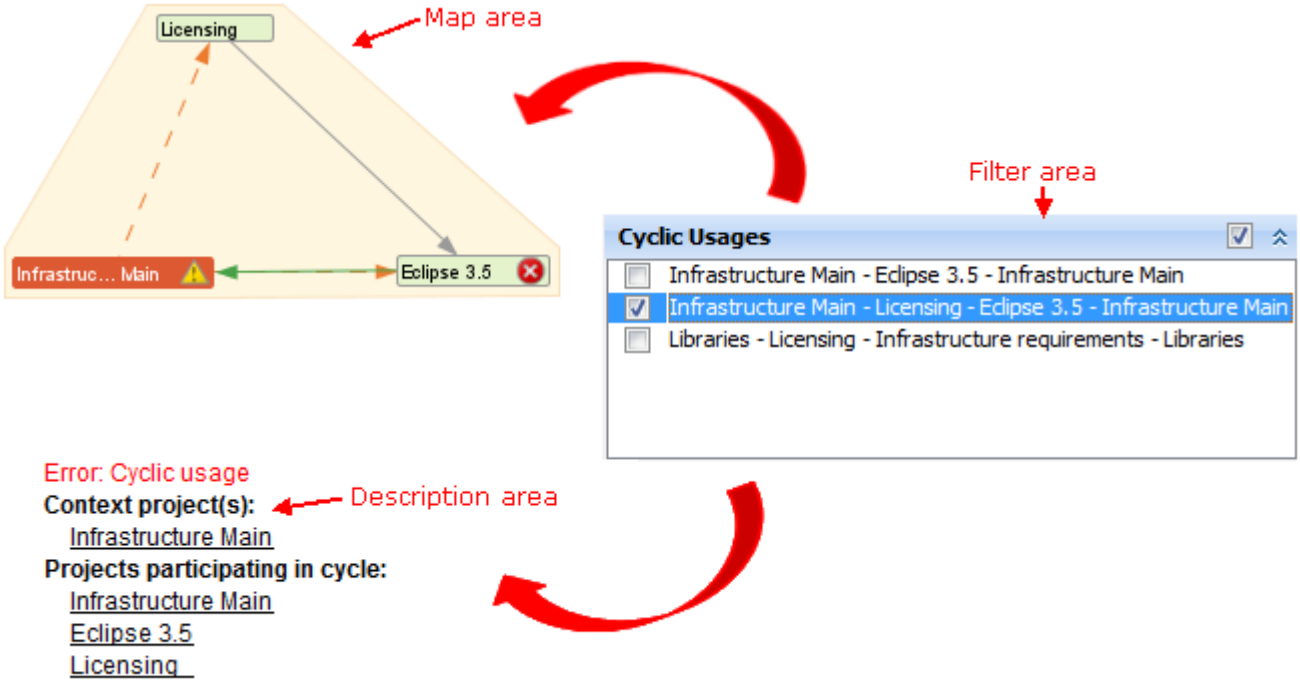


Figure 676 -- Identifying cyclic usages

Solution

The Project Usage Map automatically identifies cycles and highlights them in the Repository View. You can then open a Project Usage Map for the suspected projects participating in a cycle to analyze it more closely and if necessary - break usages that cause cycles.

To resolve the cyclic usages



Be sure to check and resolve the model level usages between the project and module.

1. Open the main project of the cycle.
2. From the **Options** menu, select **Modules**. The **Modules** dialog opens.
3. Select a module which should be removed, and click the **Lock** button.
4. Click the **Remove Module** button.

Copyright © 1998-2015 No Magic, Inc.

5. Click **OK** when you are done.

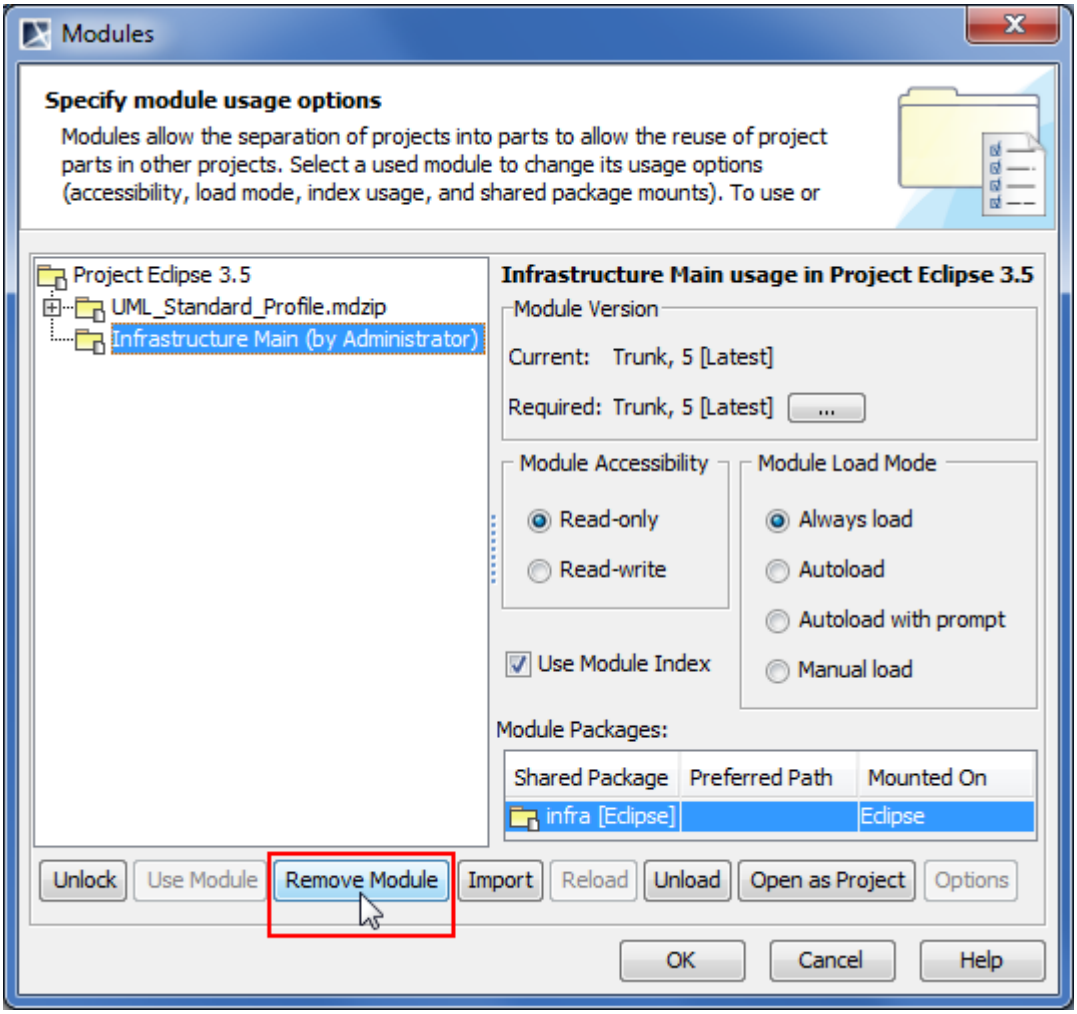


Figure 677 -- Removing module to break cyclic usage

Copyright © 1998-2015 No Magic, Inc.

Resolving version inconsistencies

Problem

You are using different versions of the same project in your main project.

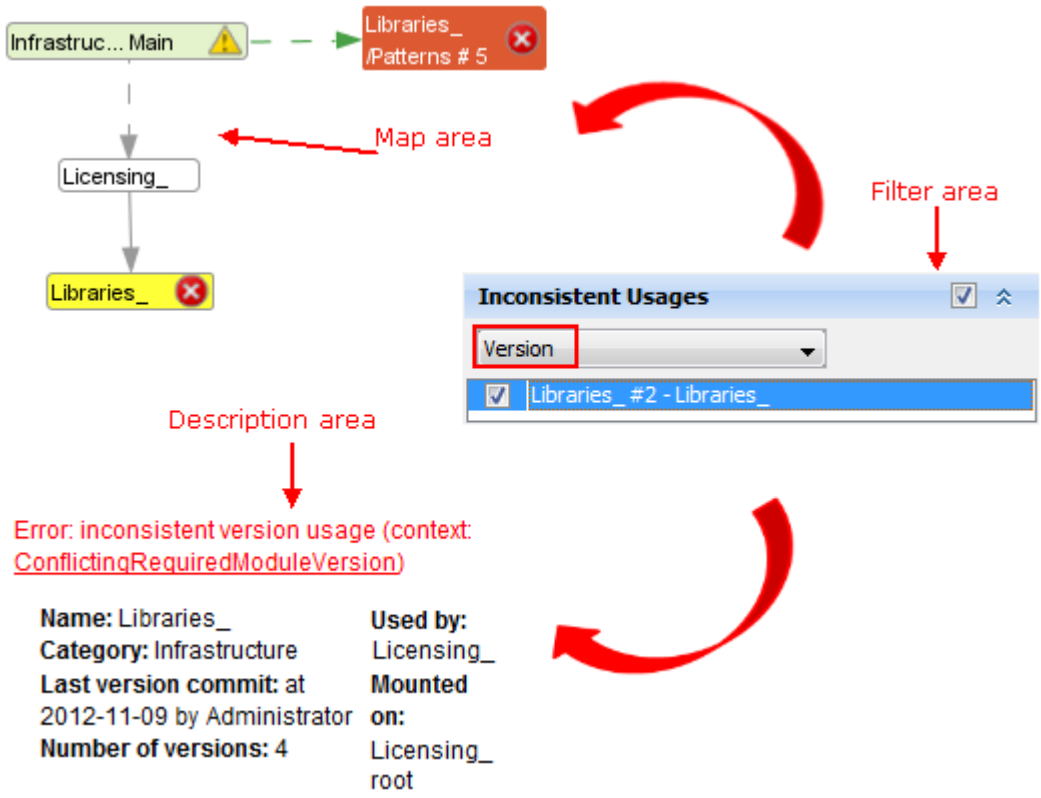


Figure 678 -- Identifying version inconsistency

Solution

The Project Usage Map highlights these inconsistencies. You can then open projects with inconsistent usages and fix them by unifying the used project version.

To resolve a version inconsistency

NOTE Be sure to check and resolve the model level usages between the project and module.

1. Open a project using the projects of inconsistent version.
2. From the **Options** menu, select **Modules**. The **Modules** dialog opens.
3. Select a module which version you want to change, and click the **Lock** button.
4. In the **Module Version** area, click the **...** button. The **Edit Branches** dialog opens.
5. Select the wanted version.
6. Click **OK** when you are done.

Copyright © 1998-2015 No Magic, Inc.

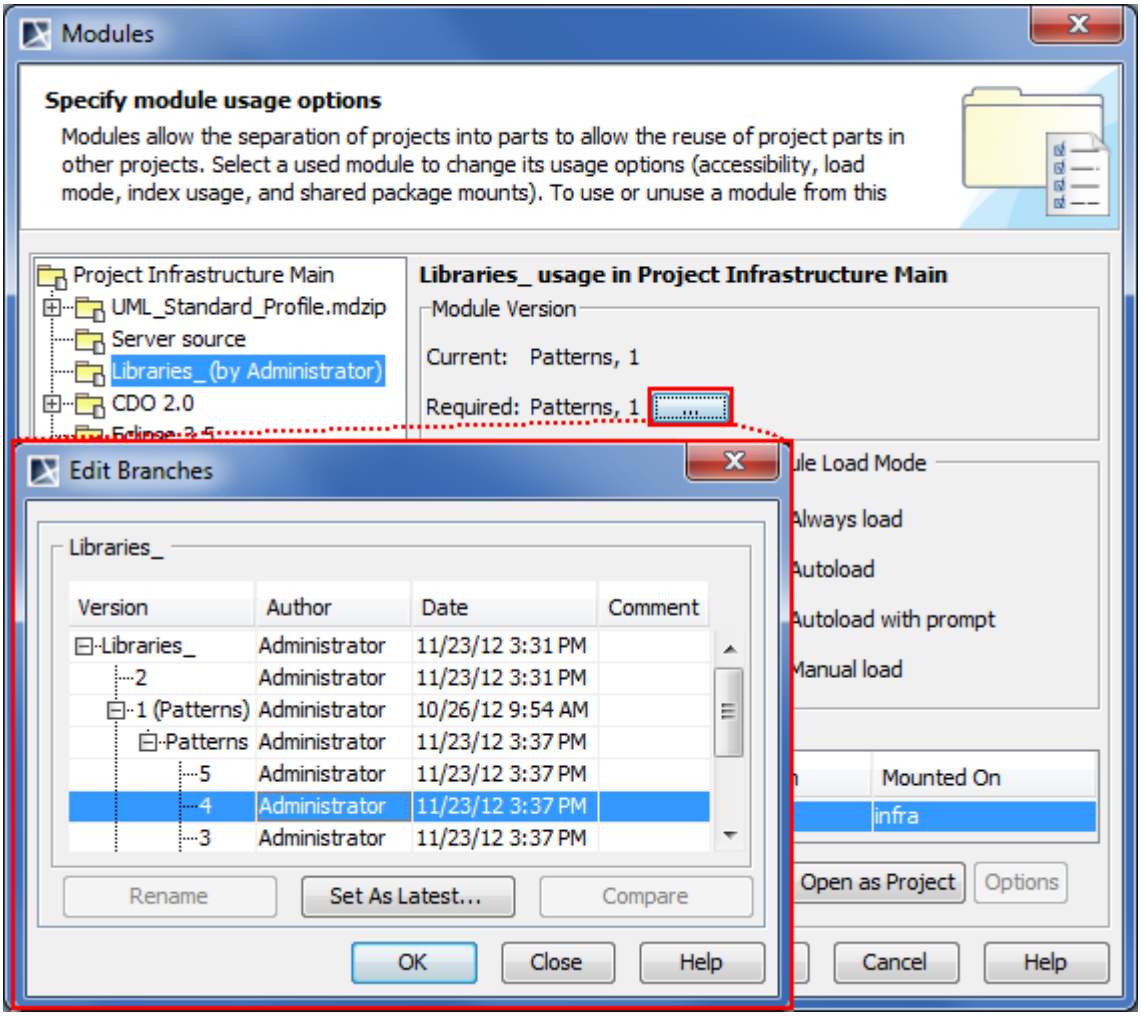


Figure 679 -- Changing module version

Resolving branch inconsistencies

Problem

You are using a version from the trunk and branch of the same project in your main project

Copyright © 1998-2015 No Magic, Inc.

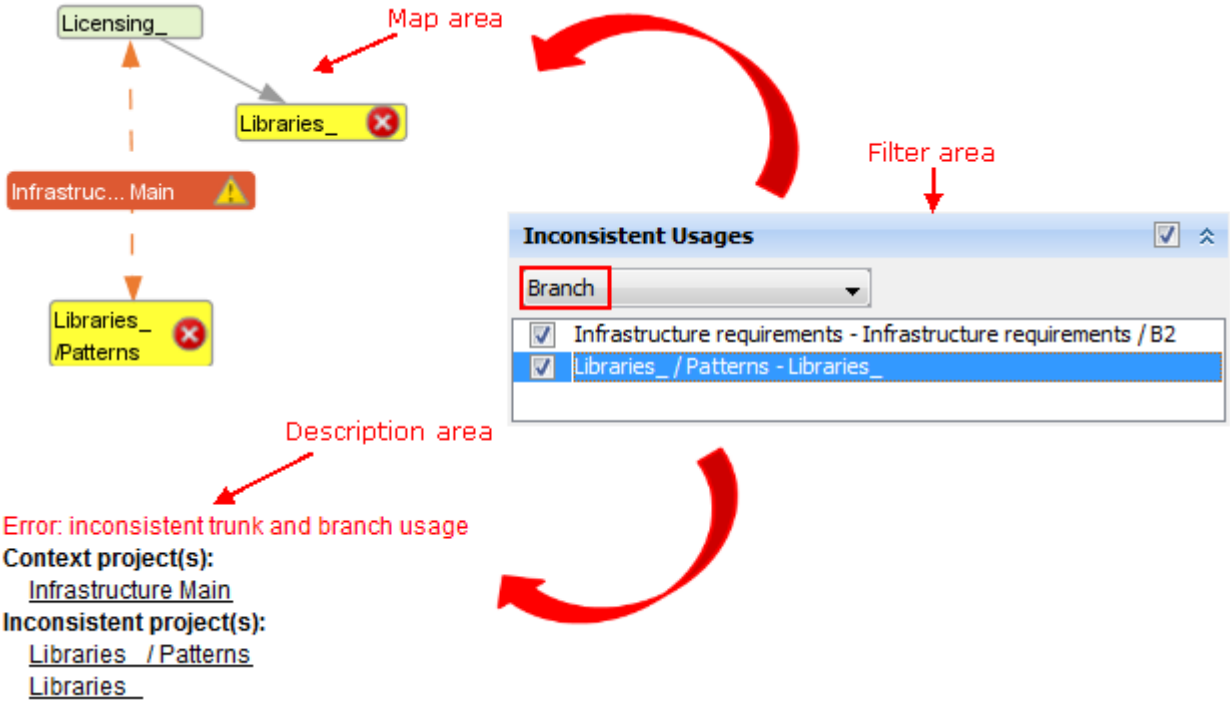


Figure 680 -- Identifying branch inconsistency

Solution

The Project Usage Map highlights these inconsistencies. You can then open projects with inconsistent usages and fix them by unifying the used project branch.

To resolve a branch inconsistency

NOTE Be sure to check and resolve the model level usages between the project and module.

1. Open a project using the projects of inconsistent version.
2. From the **Options** menu, select **Modules**. The **Modules** dialog opens.
3. Select a module which version you want to change, and click the **Lock** button.
4. In the **Module Version** area, click the **...** button. The **Edit Branches** dialog opens.
5. Select the wanted branch or trunk.
6. Click **OK** when you are done.

Copyright © 1998-2015 No Magic, Inc.

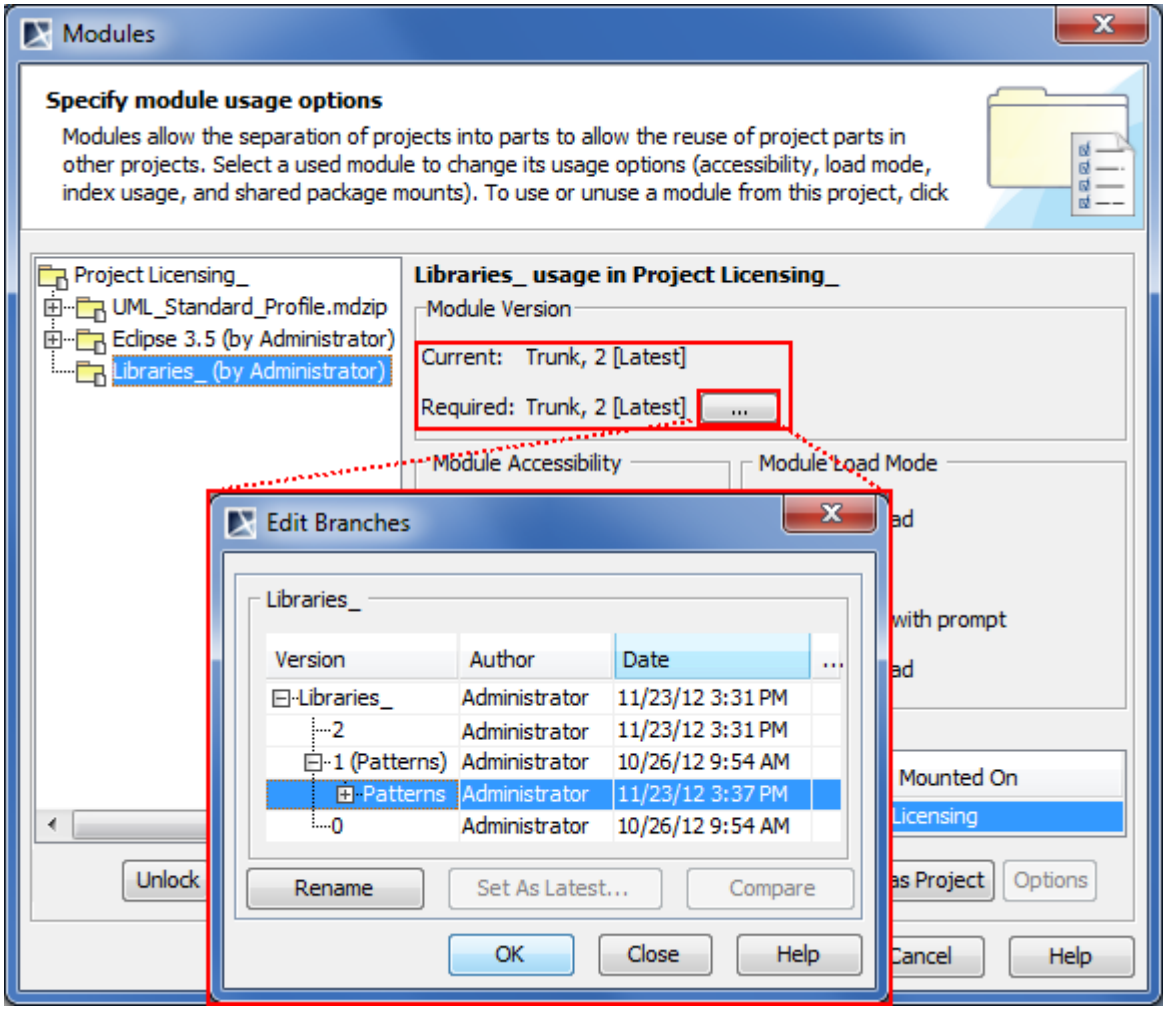


Figure 681 -- Changing module branch

Resolving mount point inconsistencies

Problem

You are mounting (mount - the other project usage in a particular package of the main project) the used project in different packages in your main project

Copyright © 1998-2015 No Magic, Inc.

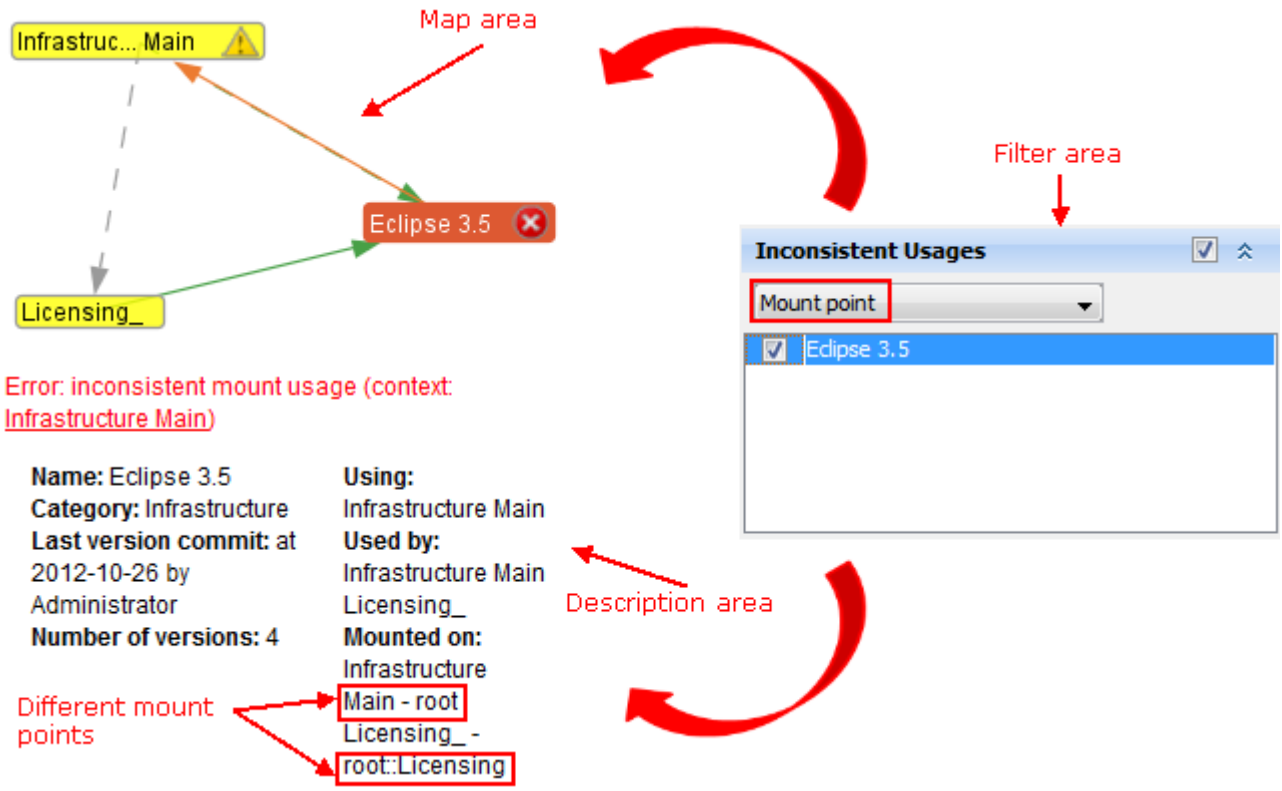


Figure 682 -- Identifying mount point inconsistency

Solution

The Project Usage Map highlights these inconsistencies. You can then open projects with inconsistent usages and fix them by unifying the used project mounted package information.

To resolve a mount point inconsistency

NOTE Be sure to check and resolve the model level usages between the project and module.

1. Open a project using the projects of inconsistent version.
2. From the **Options** menu, select **Modules**. The **Modules** dialog opens.
3. Select a module which version you want to change, and click the **Lock** button.
4. In the **Module Packages** table, select a wanted module.
5. In the Mount On cell, click the **...** button. The **Select Package** dialog opens.
6. Select the wanted package and click **OK**.
7. Click **OK** when you are done.

Copyright © 1998-2015 No Magic, Inc.

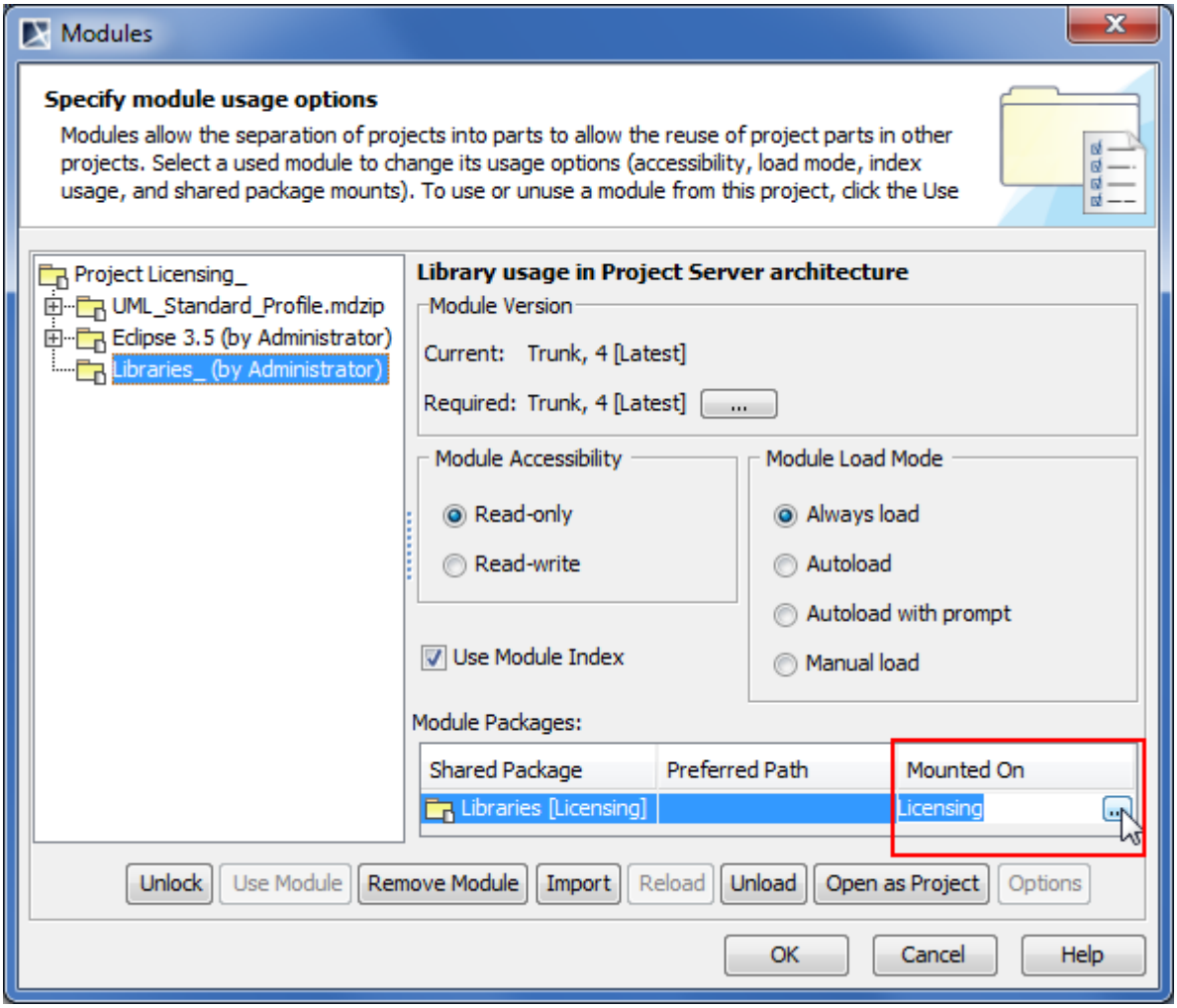


Figure 683 -- Changing mount point

Resolving unconfirmed usages

This type of usages is created automatically.

Copyright © 1998-2015 No Magic, Inc.

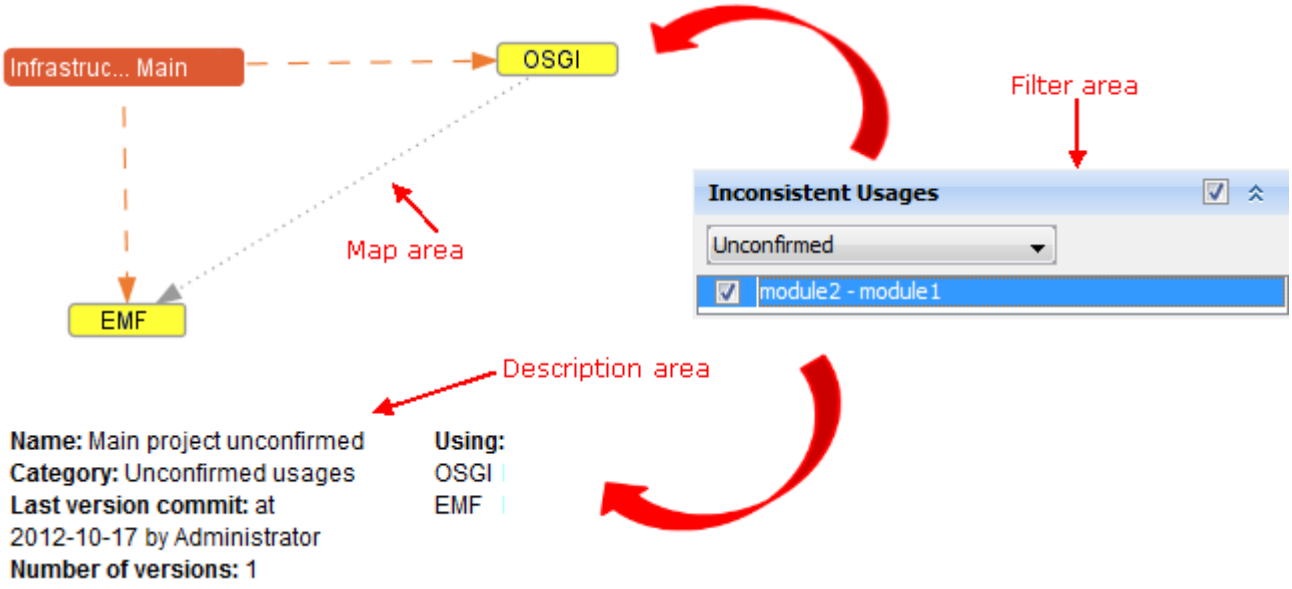


Figure 684 -- Identifying unconfirmed usages

Solution

NOTE

Be sure to check and resolve the model level usages between the project and module.

There are two ways to solve unconfirmed module usage situation - either confirm it or reject it.

If the module usage A => B is good and necessary according to the end-user policy, it can be confirmed. I.e. the user-defined usage is to be created in place of the current unconfirmed automated module usage.

To do that, use the Confirm and use the module into <module_name> solver of the validation result. This solver opens the standard Use Module wizard and pre-selects the required module. When wizard is completed, the necessary user-defined module usage is created.

If the usage A => B is not good according to the end-user policy (for example – leads to module usage cycles or is incorrect because of semantically there should be no dependency between these modules), then it needs to be rejected and removed. To remove the usage, the model-level references, that are causing this automated module usage need to be changed – either removed, or redirected to different elements.

TIP!

For more information about resolving the unconfirmed usages, see "Resolving unconfirmed module usages" on page 165.

Resolving not used modules problem

Problem

When the number of projects in the repository grows it is common for some of the projects to become outdated or not used anymore. You would prefer removing them BUT you are not sure if they are not used by some other, still-active project.

Copyright © 1998-2015 No Magic, Inc.

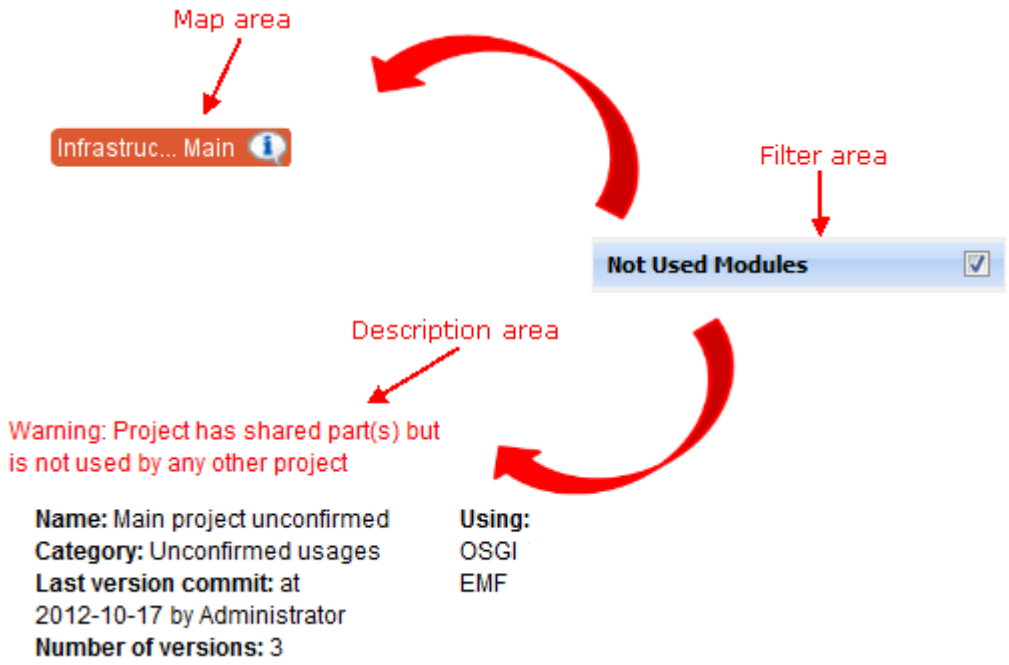


Figure 685 -- Identifying not used modules

Solution

NOTE

Be sure to check and resolve the model level usages between the project and module.

The Project Usage Map highlights unused modules. Based on this information, you can move all of the unused modules into the deprecated category or remove them entirely from the repository.

Copyright © 1998-2015 No Magic, Inc.

13 APPENDIX I: SHORTCUT KEYS, MENUS, BUTTONS, AND ICONS

This chapter includes the descriptions of the menu system, the toolbar buttons and the Browser window buttons of the MagicDraw graphical user interface (GUI), and also the descriptions of the icons used in modules and profiling mechanism.

- "Commonly Used Shortcut Keys" on page 1125
- "Menu System" on page 1127
- "Main Toolbars" on page 1140
- "Diagram Toolbars" on page 1143
- "Model Browser" on page 1145
- "Icons of general elements" on page 1146
- "Icons of relationships" on page 1154
- "Icons from Modules and Profile mechanism" on page 1158

Copyright © 1998-2015 No Magic, Inc.

Commonly Used Shortcut Keys

What do you want to do?	Shortcut keys	
	Windows OS	**Mac OS**
Open the **Find** dialog	CTRL+F	Cmd+F
Open the **Quick Find** dialog	CTRL+ALT+F	Cmd+ALT+F
Open the **Find and Replace** dialog	CTRL+R	Cmd+R
Remove both a symbol from an active diagram and the element from the model	CTRL+D	Cmd+D
Remove only a symbol from an active diagram but not the model element	DELETE	Del
Select all shapes on an active diagram	CTRL+A	Cmd+A
Select an element in the Containment tree	ALT+B	ALT+B
Add more symbols to a selection	SHIFT+Left mouse button	SHIFT+Left mouse button
Select the same type elements	ALT+Left mouse button	ALT+Left mouse button
Open a list of possible elements to assign when typing on a symbol	CTRL+SPACEBAR	CTRL+SPACEBAR
Add a new line in the compartment of a shape, for example, create an attribute for a class.	SHIFT+ENTER	SHIFT+ENTER
Open element's Specification window	ENTER	ENTER
Open element's **Symbol Properties** dialog	ALT+ENTER	ALT+ENTER
Open the **Report Wizard**	CTRL+SHIFT+G	Cmd+SHIFT+G
See the list of diagrams history	F12	–
Cut a selected item	CTRL+X	Cmd+X
Copy a shape or text	CTRL+C	Cmd+C
Copy a shape	Drag the shape to the empty place on the diagram, while holding down CTRL.	–
Paste	CTRL+V	Cmd+V
Paste with new data	CTRL+E	Cmd+E
Save	CTRL+S	Cmd+S
Commit changes to the server	CTRL+K	Cmd+K
Undo	CTRL+Z	Cmd+Z
Redo	CTRL+Y	Cmd+Y
Print	CTRL+P	Cmd+P
Assign a shortcut key to print a report	ALT+ number 1 to 9	ALT+ number 1 to 9

Copyright © 1998-2015 No Magic, Inc.

What do you want to do?	Shortcut keys	
	Windows OS	Mac OS
Turn on the Full Screen mode	F11	–
Open the recently closed diagram	CTRL+SHIFT+T	–
Paste a symbol style	CTRL+SHIFT+V	Cmd+SHIFT+V

Copyright © 1998-2015 No Magic, Inc.

Menu System

File menu

Command	Button / Shortcut keys	Function
New Project	 CTRL+ Shift+N	Creates a new project. Choose an icon to create a new Blank Project, create a Project from an Existing Source, or create a Project from a Template. A project is nameless until you close or save it by choosing the **Save Project** or **Save Project As** commands. You may simultaneously create as many new projects as you wish, without saving and closing the previously created or open projects. You may work on only one open or created project at a time. If you wish to work with another project, select the project name in the **Projects** list on the main window toolbar.
Open Project	 CTRL+O	Opens an existing project. The **Open** dialog box appears. Select a project you wish to open. You may open as many projects as you wish. If you wish to work with another project, select the project name in the **Projects** list on the main window toolbar.
Save Project	 CTRL+S	Saves the open project. To save the current project for the first time, type the name of the project and select the format of the file you wish to save.
Save Project As		Saves the project giving it a name. Use this command to save the current project for the first time or under a different name, the **Save** dialog box appears. Type the name of the project and select the format of the file you wish to save.
Close Project		Closes an open project. If the open project has unsaved changes, MagicDraw displays a dialog box asking whether the changes should be saved before the project is closed.
Close All Projects		Closes all open projects. If the open projects have unsaved changes, MagicDraw displays a dialog box asking whether the changes should be saved before the each project is closed.
Open Element from URL		You can open any elements through their URLs by clicking the **Open Element from URL** command and the element will be highlighted in the Containment tree or in the diagram. For more information about element URL, see “Copying/Opening Element URLs” on page 429.
Use Module		The **Use Module** dialog box appears. Choose profile or module for use in the project.

Copyright © 1998-2015 No Magic, Inc.

Command	Button / Shortcut keys	Function
Import From		Imports data from: • Another project - The **Import** dialog appears. Select the project you wish to import. You may import as many projects as you wish. Imports an existing project to a previously open project. This is the recommended command for importing an existing project to the teamwork server. • CSV file • UML XMI 2.1/2.4 file • MagicDraw Native XML file • MOF XMI File • CA ERwin Data Modeler 7.x file • Eclipse UML 2 (v1.x, v2.x, v3.x, v4.x) XMI file • Enterprise Architect 2.1 XMI 2.1 • Rational Software Architect/Modeler project • Rational Rose *.mdl project file
Export To		Exports the project to: • Module. The content of the selected objects is saved in a separate file. • Template. The project is saved as a template project. • UML XMI 2.4 file • MagicDraw Native XML file • EMF Ecore file • MOF XMI file • Eclipse UML2 (v1.x, v2.x, v3.x, v4.x) XMI file. Exports the model to EMF based UML 2 compatible XMI file.
Share Packages		Saves the package as a separate module.
Save As Image		Saves the open diagram as an image of in the selected formats. The **Save As Image** command is used for saving any item or group of items selected in the diagram pane, in one of the various image file formats.
Print	 CTRL+P	The **Print** dialog opens. Printing of active diagram, selected symbols, selected diagrams, or all opened diagrams is available. Select the printer, set the options for the printer, specify the number of copies, and select the specific pages to be printed.
Print Preview		Preview the diagram before printing. The **Print Preview** window opens.
Print Options		The **Print Options** dialog box opens.

Copyright © 1998-2015 No Magic, Inc.

Command	Button / Shortcut keys	Function
Project Properties		The **Project Properties** dialog opens. This dialog contains the following information: project location, file size, date created, date modified for the last time, and number of diagrams in the project. In the **Description** tab, type the description of the project or other important information. The **Modules** tab shows a list of modules the project is using. The list shows a specified number of recent project files. Specify the number of files in the **Recent Files List Size** property in the **Environment Options** dialog box.
Switch Projects		A list of open projects.
Exit		Exits the application. All open projects are closed. If an open project has unsaved changes, the MagicDraw displays a dialog box asking whether the changes should be saved before closing the project.

The **File** menu contains a list of recent projects. The shortcuts with numbers to recent projects are displayed. If the shortcut is selected from the **File** menu, the recent project will open instantly.

Edit menu

MagicDraw allows you to use the **Edit** menu commands while drawing the diagrams on the Diagram pane. The commands allow selecting, cutting, copying, and pasting of items and entire diagrams, reversing the actions you have taken while drawing, and finding an item in the current project.

Command	Button/ Shortcut keys	Function
Undo	 CTRL+Z	Reverses the last action you have performed while drawing the diagram on the Diagram pane (moving, dragging, resizing, cutting, copying, pasting, deleting, selecting, editing shapes, setting project and shape properties, etc.). Actions are reversed in the opposite order you have performed them, starting with the most recent. By default, the limit of the undo mechanism is 100 steps backwards. To change the limit, choose **Environment** from the **Options** menu. The **Environment Option**s dialog box appears. Change the **Undo List Size** property. The **Undo** command is unavailable until you perform any action after loading an the existing project or creating a new project. Each command has an easily recognized name. You will be able to see the command history and undo or redo one action or a group of actions. The main window will have two lists of commands: one for the undo commands, another one for the redo commands.

Copyright © 1998-2015 No Magic, Inc.

Command	Button/ Shortcut keys	Function
Redo	 CTRL+Y	Reverses the action of the **Undo** command (moving, dragging, resizing, cutting, copying, pasting, deleting, selecting, etc.). The **Redo** command is unavailable until you use the **Undo** command.
Cut	 CTRL+X	Cuts the selected items or group of items on the Diagram pane. The cut items are placed in the clipboard. Later they can be pasted back to the Diagram pane of the current or to another project. The **Cut** command is unavailable until you select any item or any group of items on the Diagram pane of the current project.
Copy	 CTRL+C	Copies the selected items or group of items on the Diagram pane. The copied items are placed in the clipboard. The cut items can be pasted back to the Diagram pane or to another project. The **Copy** command is unavailable until you select any item or any group of items on the Diagram pane of the current project.
Copy URL		Copy a project element URL to a clipboard and share it with other as a quick reference to model elements. For more information about copying element URL, see “Copying/Opening Element URLs” on page 429.
Paste	 CTRL+V	Pastes the cut or copied items or group of items from the clipboard to the Diagram pane of the current project. MagicDraw creates shapes for items, or a group of items, in the current project. You will see the data and shapes of the pasted items and diagrams in the Browser window.
Paste with New Data	CTRL+E	Creates new and pastes the cut or copied items or groups of items from the Clipboard to the Diagram pane of the current project. MagicDraw creates new data and data shapes in the current project.
Paste Style	 CTRL+SHIFT+V	Pastes the symbol style on the selected symbol. For more information about pasting symbol style, see "NEW! Displaying rake icon on symbol" on page 322.
Paste as Diagram Overview		Creates a new diagram overview with represented selected elements in it. For more information about the diagram overview, see "Diagram Overview" on page 1030.
Delete	 CTRL+D	Deletes data together with symbol. It is unavailable until you select any item or group of items in the current project.
Remove from Diagram	DELETE	Deletes a symbol of model element from the diagrams, by leaving it in the model.
Select All	CTRL+A	Selects all items on the Diagram pane of a particular project.

Copyright © 1998-2015 No Magic, Inc.

Command	Button/ Shortcut keys	Function
Select All of the Same Type	CTRL+ALT+A	Selects all shapes of the selected types in active diagram. Enabled, when one or more shapes are selected in active diagram.
Copy as BPM	CTRL+SHIFT+B	Copies the selected model elements to the system clipboard. If no model elements are selected, the active diagram is copied. Note that **Copy as BMP Image** is available only under Windows system.
Copy as EMF	CTRL+SHIFT+E	Copies the selected model elements to the system clipboard. If no model elements are selected, the active diagram is copied. Note that Copying as EMF is available only under Windows system.
Copy as JPG	CTRL+SHIFT+J	Copies the selected model elements to the system clipboard as a JPG image. If no model elements are selected, the active diagram is copied. Note that copying as JPG is available only under Windows system.
Copy as PNG	CTRL+SHIFT+P	Copies the selected model elements to the system clipboard as a PNG image. If no model elements are selected, the active diagram is copied. Note that copying as PNG is available only under Windows system.
Find	 CTRL+F	Opens the **Find** dialog box.
Quick Find	CTRL+ALT+F	Performs a quick search of class/interface, classifier, or diagram.
Find TODO		Performs a search for the TODO tagged value. Results are displayed in the Browser, **Search Results** tab.
Find and Replace	Ctrl+R	Replaces one specified model value with another.
Paths		• **Path Style** - choose the line style for drawing a path. • **Rectilinear** – drawing rectilinear lines. • **Oblique** - drawing free form lines. • **Bezier** - in computer graphics, a curve that is calculated mathematically to connect separate points in smooth, free-form curves. • **Change Path Style** – switches in series between rectilinear, oblique, and bezier path line style. Shortcut is CTRL+L. • **Reset Labels Positions** – resets the changed path labels (name, roles, etc.) to the default position. • **Remove Break Points** – removes the break points of the path and makes the path a line straight.
Symbol		All commands that are available through the shortcut menu for a particular symbol.

Copyright © 1998-2015 No Magic, Inc.

View menu

Command	Button / Shortcut keys	Function
Fit In Window	CTRL+W	Reduces the size of the whole diagram to fit in the Diagram pane.
Zoom In	 CTRL+NUMPAD PLUS SIGN (+)	To change the zoom step size choose **Environment** from the **Options** menu and set the **Zoom Step Size** property in the **Environment Options** dialog box.
Zoom Out	 CTRL+NUMPAD MINUS SIGN (-)	Decreases the size of the selected objects in the diagram pane by x percent. To change the zoom step size, choose **Environment** from the **Options** menu and set the **Zoom Step Size** property in the **Environment Options** dialog box.
Zoom 1:1	 CTRL+NUMPAD SLASH MARK (/)	Restores the original size of the selected diagram symbols.
Zoom To Selection	CTRL+NUMPAD ASTERICS MARK (*)	Increases the size of the selected model element on the Diagram pane to the maximum visible size.
Refresh	CTRL+Shift+F	Repaints all diagram shapes.
Grid		Set grid options. Every diagram may have its own grid settings: • **Show Grid** - show/hide grid • **Snap Paths to Grid** - use/do not use grid for drawing paths. • **Snap Shapes to Grid** - use/do not use grid for drawing shapes. • **Grid Size -** change the grid size. Type the number.
Reopen Closed Tabs	CTRL+SHIFT+T	Opens diagram tabs starting from the last closed.
Show Tabs in Full Screen	F11	Shows opened diagram tabs the full screen mode. Click **Close Full Screen** to return to the previous view.

Copyright © 1998-2015 No Magic, Inc.

Command	Button / Shortcut keys	Function
Main Toolbars: • Menu Bar • File • Switch Projects • Perspectives • Collaboration • External Tools • Validation • Rearrangable • Hidable • Floatable • Expert Menu Mode • Expert Toolbar Mode • Customize		Clear the check boxes of the toolbars you want to hide or select/clear check boxes to rearrange toolbar modes.
Diagram Toolbars: • Navigation • Layout • Edit • Symbol Editing • Shape Editing • View • Rearrangable • Hidable • Floatable • Expert Mode • Customize		Clear the check boxes of the toolbars you want to hide or select/clear check boxes to rearrange toolbar modes.
Status Line		Select the check box of **Show Status Bar** or **Show Memory Monitor** to display this info on the status line.

Layout menu

Use the commands of the **Layout** menu for managing the layout of the shapes on the current Diagram pane. You must select more than one shape for other Layout menu commands. A class diagram should be open and activate before using the **Class Diagram** command in this menu.

Command	Function
Layout Options	Opens the **Diagram Layout Options** dialog box. Layout options for the diagram can be set.
Quick Diagram Layout (CTRL+Q)	Applies recommended layout with default options on the active diagram.
Layout Class Diagram Style	Applies layout, which uses specific layout algorithms to improve class diagram readability.
Layout Activity Diagram Style	Applies layout, which uses specific layout algorithms to improve activity diagram readability.

Copyright © 1998-2015 No Magic, Inc.

Command	Function
Layout Business Process Diagram Style	Applies layout, which uses specific layout algorithms to improve business process diagram readability.
Layout Hierarchic Style	Applies layout that portrays the main direction or flow of directed graphs. It is ideal for many application areas, especially for Workflow, Software Engineering, Customer relationship management, Configuration management, Process modeling, Database Modeling, and Bio informatics.
Layout Tree Style	Applies layout, which specializes in the layout of tree-structured graphs. The need to visualize directed or undirected trees arises in many application areas, e.g. Dataflow analysis, Software Engineering, Network management, Bio informatics.
Layout Orthogonal Style	Applies layout that is well suited for medium sized sparse diagrams. It produces compact drawings with no overlapping shapes, few crossings, and few bends. All edges will be routed in an orthogonal style, i.e. only rectilinear style links will be used.
Layout Organic Style	Applies layout of the organic style graph.
Layout Circular Style	Applies layout of the algorithm that portrays interconnected ring and star topologies. It is excellent for applications in social networking (criminology, economics, ...), network management, WWW visualization, and eCommerce.
Route Paths Rectilinearly (Ctrl+Shift+Q)	Applies layout, which routes the links of a diagram using only vertical and horizontal line segments, while keeping the positions of the shapes in the diagram fixed. The routed links will usually not cross through any shapes and not overlap any other links.
Route Paths Obliquely	Applies layout, which routes the links of a diagram using oblique link style, while keeping the positions of the shapes in the diagram fixed. The routed links will usually not cross through any shapes and not overlap any other links.
Make Same Width	Applies layout to the selected shapes according to their width. After the layout, the width of the shapes is equal (according to the widest).
Make Same Height	Applies layout to the selected shapes according the their height. After the layout, height of the shapes is equal (according to the tallest).
Align • Right • Left • Top • Bottom	Aligns the selected shapes: • Aligns the selected shape(s) vertically, starting with the rightmost shape(s). • Aligns the selected shape(s) vertically, starting with the leftmost shape(s). • Aligns the selected shape(s) across from the uppermost shape(s). • Aligns the selected shape(s) across from the lowermost shape(s).
Center • Horizontally • Vertically	Centers the selected shapes: • Centers the selected shape(s) on a horizontal line. • Centers the selected shape(s) on a vertical line.
Space Evenly • Horizontally • Vertically	Sets spaces among the selected shapes evenly. • Spaces between the selected shapes becomes even on a horizontal line. • Spaces between the selected shapes becomes even on a vertical line.

Copyright © 1998-2015 No Magic, Inc.

Diagrams menu

Command	Shortcut keys	Function
Create Diagram		The Create Diagram dialog opens.
Customize		The **Customize Diagrams** dialog opens.
Diagram Wizards		Wizards for creating diagrams may be opened: • Class Diagram Wizard • Package Dependency Diagram Wizard • Package Overview Diagram Wizard • Activity Decomposition Hierarchy Wizard • Hierarchy Diagram Wizard • Realization Diagram Wizard • Sequence Diagram from Java Source Wizard • Content Diagram Wizard
History	F12	The **History** dialog opens.
Load All Diagrams		If there are unloaded diagrams in the project, this command loads all diagrams.

Options menu

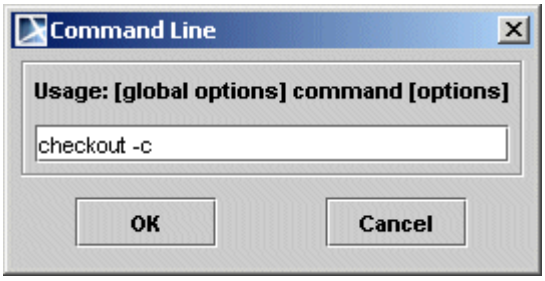
Command	Function
Project	The **Project Options** dialog box opens.
Modules	The **Modules** dialog box opens.
Environment	The **Environment Options** dialog box opens.
Perspectives	Choose a command from the submenu - to switch to different **Perspective** or to **Customize**.
Look and Feel	Choose a style for the MagicDraw (GUI) from the following list: • Metal • CDE/Motif • Vsnet (Windows) • Office 2003 (Windows) • Office 2007 (Windows) • Eclipse (Windows) • Xerto The **Look and Feel Themes** command allows you to personalize the GUI: set your favorite colors and fonts.

Copyright © 1998-2015 No Magic, Inc.

Tools menu

Command	Shortcut key	Function
Apply Pattern		The **Pattern Wizard** dialog box appears. Create the design pattern for the selected class, interface.
Model Transformations		Opens the **Model Transformations Wizard** dialog box with a list of all available transformations.
Hyperlinks		Opens the **Hyperlink** dialog where you can add hyperlinks to any model elements.
Report Wizard		The Report Wizard is the new report engine for MD 14.0 and above. It is designed to solve the several problems of the old engines (XSL/XSLT and JPython). It supports text based templates to generate the output file. The format of output file depends on the type of the template file. The type of template files that the Report Wizard supports are normal text, RTF, HTML, Spreadsheet template (need to be saved as HTML format), and XML template (DocBook or FO). All commercial MagicDraw editions will have full use of all features within the Report Wizard. For more details, see MagicDraw ReportWizard UserGuide.pdf.
Quick Reverse		Choose the language you need (Java, Java Bytecode, C++, C#, CIL, CIL Disassembler, IDL, CORBA IDL, DDL, XML Schema, WSDL). Opens the **Round Trip Set** dialog box. (For more details, see Code Engineering User's Guide).
Generate Code Framework	CTRL+G	Generates code for the selected items in the current diagram. Opens the **Notification Window** with the information appears (For more details, see Code Engineering User's Guide).
Check Syntax	CTRL+T	Checks syntax in the model according to the default code engineering language. Opens the **Notification Window** with the information.
Set empty tags to defaults		Set default tag value to tag with empty value. This functionality is needed when the stereotype is already assigned to an element and the new mandatory tag definition with default value is created for the stereotype. After creating such a tag definition, the model elements that have the modified stereotype applied will have newly created tags unset. For more information see "To create default tag values" on page 994.
Integrations		Opens the **Integrations** dialog box with a list of tools for possible integration with MagicDraw.

Copyright © 1998-2015 No Magic, Inc.

Command	Shortcut key	Function
Requirements • **DataHub**	CTRL+ SHIFT+R	• Starts Cameo DataHub, if it is enabled. More information about Camae DataHub at http://www.nomagic.com/products/cameo-datahub.html.
CVS • Command Line		Performs operations with CVS (for detailed description of integration with CVS, see MagicDraw Integrations UserGuide.pdf) • Allows the user to enter a CVS command line (like “checkout -c”) whenever the CVS command is not available through the menus. The **Command Line** dialog box opens. 
• Checkout Module		• Use this option to checkout a new module on your disk. The **Checkout Module** dialog box opens.
• Add Project to CVS		• Adds a new project to CVS. The **Add Project** to CVS dialog box opens. **NOTE:** You can add, update or commit projects to CVS only if they are saved in some checked out directory or subdirectory.
• Commit Project to CVS		• Commits a project to CVS. The **Commit Project to CVS** dialog box opens.
• Update CVS Project		• Updates the current project by loading the latest project version from CVS.
Check Spelling		Checks spelling for the whole project or for the selection. The **Check Spelling** dialog opens.
Scripts		Runs the selected script. Select the script from the **Scripts** dialog.
ParaMagic Plugin		Starts ParaMagic Plugin, if it is enabled. More about ParaMagic Plugin at http://www.nomagic.com/products/sysml-plugin/paramagic-plugin.html.
MDA		Starts CameoMDA plugin, it is enabled. More about CameoMDA Plugin at http://www.nomagic.com/products/magicdraw-addons/cameo-mda.html.

Copyright © 1998-2015 No Magic, Inc.

Analyze menu

Command	Shortcut keys	Function
Model Visualizer		Opens the **Model Visualizer** dialog box with a list of all available wizards.
Metrics		Metrics feature allows the measurement of a project by different viewpoints. Specify the metrics scope in the **Metrics Options** dialog box.
Compare Projects		Opens the **Compare Projects** dialog box, where you can choose projects to perform model differencing.
Dependency Checker		Opens the **Dependency Checker** dialog box to configure the options for the dependency analysis of the whole project.
Validation		Validation functionality allows the completeness and correctness evaluation of the models, created by the user, according to constraints defined in Object Constraint Language (OCL) or java code. For more information about validation, see “Validation” on page 612.
Display Paths		Displays paths among shapes that are already created in the model data.
Display Related Elements		Displays elements related to the selected element.
Create Relation Map		Creates the Relation Map for the selected element. More information about relation map, see “Relation Map” on page 544.
Used By	CTRL+ALT+U	Finds a list of all elements that reference the current element.
Depends On	CTRL+ALT+D	Finds a list of the elements that depend on the current element.
Go To		This is a feature that allows you to find model elements associated with the selected model element. Enabled when a selected element is related to another model element.

Collaborate menu

These commands are available when you are working with server projects and are connected to the server. For a detailed information about working with server projects, see section "Working with Server Projects" on page 1039.

Copyright © 1998-2015 No Magic, Inc.

Window menu

You may use commands of the **Window** menu to manage the layout of the windows.

Command	Shortcut key	Function
Containment		Opens the **Containment Tree** tab in the Browser window.
Inheritance		Opens the **Inheritance Tree** tab in the Browser window.
Diagrams		Opens the **Diagrams Tree** tab in the Browser window.
Model Extensions		Opens the **Model Extensions Tree** tab in the Browser window.
Search Results		Opens the **Search Results** tab in the Browser window.
Documentation		Opens the **Documentation** tab in the Browser window.
Zoom		Opens the **Zoom** tab in the Browser window
Properties		Opens the **Properties** tab in the Browser window
Notification Window	CTRL+M	Opens the **Messages Window**. The **Messages Window** is used for displaying the warnings and errors that may appear in the project. It appears automatically and is intended to display the warnings and errors after saving, loading, exporting and importing the project.
Reset Windows Configuration		All Browser tabs are placed in their default position.
List of open diagrams		Displays a list of open diagrams.
Close All Tabs But Current	CTRL+ SHIFT+ F4	Closes all open diagrams except the one you currently being used.
Close All Tabs	CTRL+ ALT+F4	Closes all open diagrams.

The **Window** menu contains a list of open diagrams in the project. The list shows the specified number of the recent diagrams. This number can be customized in the **Recent Windows List Size** property in the **Environment Options** dialog box. For a detailed description on this dialog box, see the Section “Customizing Environment Options” on page 96.

Help menu

Command	Function
Help Contents	Displays a table of contents for the MagicDraw Help.
MagicDraw User Manual	Opens the MagicDraw User Manual.
Other Documentation	Displays the list of all available user’s guides and user’s manuals.

Copyright © 1998-2015 No Magic, Inc.

Command	Function
Tip of the Day	Displays the **Tip of the Day** screen.
Report an Issue	No Magic always welcomes your initiative. Submit bugs, suggestions, and new feature requests through the **Report an Issue** dialog directly to Online Support System.
View and Submit Internal Errors	View errors received during work with MagicDraw. Send a bug report to the No Magic support team. For more information about submitting a bug see "Reporting issues directly from MagicDraw" on page 41.
Check for Updates	Opens The HTTP Proxy Server Connection dialog box. Set the data for connection to start the MagicDraw update.
License Manager	Select evaluation key, activate commercial license, or use the floating license.
Finish Offline Floating Session	If you are using offline Floating License, finish the offline session.
Show Welcome Screen	If no project is open, you may choose this command to open the Welcome screen.
What's New	Opens the New and Noteworthy section: http://www.nomagic.com/news/news-noteworthy.html
News Reader	Read the latest news about new updates, products, resources, and No Magic Inc. events.
Resource/Plugin Manager	Check for available updates and new resources in the **Resource/Plugin Manager** window.
MagicDraw on the Web • Online Support • Online Demo • New and Noteworthy • MagicDraw Home Page • UML Stuff • No Magic Home Page	The web pages where you can find additional information about MagicDraw and UML. Get online support.
Entertainment with UML	• **Memory** game. Open pictures to find pairs with as few tries as possible. • **Puzzle** game. Transpose the separate parts to get the whole picture.
About MagicDraw	Displays the information screen about the application.

Main Toolbars

The main toolbars are located at the top of the MagicDraw window, below the main menu. They contain the commands for commonly used tasks, so helping to speed up your work with MagicDraw.



Figure 1 -- The fragment of the Main toolbar

For more information about customizing toolbars, see “Customizing toolbars” on page 73.

Copyright © 1998-2015 No Magic, Inc.

File Toolbar

IMPORTANT! If you cannot see some of the buttons, please, check the perspective and the mode you are working on.

Button	Title (shortcut keys)	Description
	New Project (Ctrl+Shift+N)	To create a new blank project, press the **New Project** button: • The **New Project** dialog box opens. Select the **Blank Project** icon. • Specify the file name in the Name text box. • Click the "..." button to select a location to store the newly created project in your computer. Click OK.
	Open Project (Ctrl+O)	To open an existing project, press the **Open Project** button.
	Save Project (Ctrl+S)	To save the current project.
	Print Active Diagram (Ctrl+P)	To print an open diagram.
	Print Preview	The **Print Preview** dialog box opens showing how your diagram looks before printing.
	Find (Ctrl+F) Find and Replace (Ctrl+R) Find TODO Quick Find (Ctrl+Alt+F)	Search for an element, symbol or diagram in the project according to your selected criteria.
	Undo action (Ctrl+Z)	Undo the last action you performed while drawing the diagram on the Diagram pane (moving, dragging, resizing, cutting, copying, pasting, deleting, selecting, editing shapes, setting project and shape properties, etc.). Actions are reversed in the order you have performed them. The **Undo** command is unavailable until you perform an action after loading an existing project or creating a new project. By default, the limit of the undo mechanism is 100 steps backwards. To change the limit, select **Environment** from the **Options** menu. The **Environment Options** dialog box opens. Change the **Undo List Size** property. Each command has an easily recognized name. You will be able to see the command history and undo or redo action history. The main window will have two lists of commands: one for the undo commands, another one for the redo commands.

Copyright © 1998-2015 No Magic, Inc.

Button	Title (shortcut keys)	Description
	Redo action (Ctrl+Y)	Restore the Undo action (moving, dragging, resizing, cutting, copying, pasting, deleting, selecting, etc.). The **Redo** command is unavailable until you use the **Undo** command.

Switch Projects Toolbar

To switch quickly from one recently opened project to the other, use the **Switch Projects** toolbar.



If you cannot see some of the buttons, please, check the perspective and the mode you are working on.

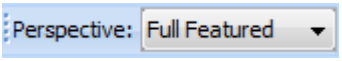
Toolbar button	Title (shortcut keys)	Description
	List of open projects	To see a list of all open projects, click the button. You will see the name and location of the currently open projects.

Perspectives Toolbar

To change the MagicDraw user perspective, use the **Perspectives** toolbar.



If you cannot see some of the buttons, please, check the perspective and the mode you are working on.

Toolbar button	Title (shortcut keys)	Description
	List of the available MagicDraw user perspectives	Click to change the MagicDraw user perspective.

Validation Toolbar

To validate the created models, use the buttons on the **Validation** toolbar.



If you cannot see some of the buttons, please, check the perspective and the mode you are working on.

Toolbar button	Title	Description
	Validate	Click to open the **Validation** dialog box.
	Run Last Validation	Click to run the model validation on options saved in the **Validation** dialog box.

Copyright © 1998-2015 No Magic, Inc.

Create Diagram Toolbar

To create a diagram, use the button on the **Create Diagram** toolbar.



If you cannot see some of the buttons, please, check the perspective and the mode you are working on.

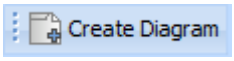
Toolbar button	Title	Description
	Create Diagram	Click to open the Create Diagram dialog. For more information, see "Creating Diagrams" on page 198.

Diagram Toolbars

The diagram toolbar contains buttons for working with symbols on the diagram pane. Select any symbol or path on the diagram pane and the required buttons from the diagram main toolbar become active.



Figure 2 -- The Diagram toolbars

Use the diagram main toolbar to change the symbol layout, path style, symbol properties style, diagram zoom as well as symbol copy/paste, cut, or delete actions.

Button	Title	Description
	Back	Navigate back to the previous diagram that was open in the current diagram tab.
	Forward	Navigate forward to the diagram in the current diagram tab.
Navigation group		
	Select in Containment Tree	Open the Containment tab and select the current diagram or symbol in the Containment tree.
Layout group		
	Quick diagram layout	Apply the recommended layout tool with default options to the active diagram. Click the small arrow next to the Quick Diagram Layout button to see other available layouts.
	Align	Align the selected shapes according to their size.
	Make same width	After the layout, the same width will be apply to the shapes (according to the widest).

Copyright © 1998-2015 No Magic, Inc.

	Make same height	Layout the selected shapes according the their height. After the layout, the same height will be applied the shapes (according to the highest).
	Make same size	Layout the selected shapes according to their width and height. After the layout, the same width and height will be applied to the shapes according to the widest and highest shape.
	Center Horizontally	Center the selected shapes on a horizontal line.
	Center Vertically	Center the selected shapes on a vertical line.
	Space Evenly Horizontally	Space the selected shapes evenly. The space between the selected shapes is equally distributed on a horizontal line.
	Space Evenly Vertically	Space the selected shapes evenly. The space between the selected shapes is equally distributed on a vertical line.
	Remove Break Points	To remove all angles of a path, press the Remove Break Points button.
	Rectilinear	To change a path style to rectilinear lines, press the Rectilinear button.
	Oblique	To change a path style to oblique lines, press the Oblique button.
	Bezier	To change a path style to bezier lines, press the Bezier button.
	Reset Labels Positions	To reset the path labels to the default position, press the **Reset Labels Position** button.
Edit group		
	Copy, Paste, Delete, and Remove from Diagram	Using the Edit group buttons, you can copy or delete symbols from the diagram pane. **NOTE** After a symbol is removed from the diagram pane, it does not mean that the element is deleted from the project.
Symbol Editing group		
	Fill Color	Press the button to apply the selected fill color or press the small arrow near the button to select another color.
	Set Selected Symbol Style as Default	If a new style was set, it will be applied for all newly created elements after drawing them on the diagram pane.

Copyright © 1998-2015 No Magic, Inc.

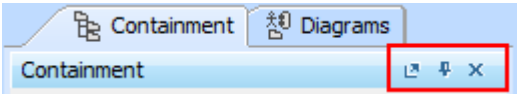
	Apply Default Symbol Style	The selected symbol style will be changed to it's default style. (The default symbol style is defined in the **Project Options** dialog box, **Symbol property styles** branch).
	Select All of the Same Type	All shapes of the same style are selected, for example, all classes will be selected.
	Paste Style	Paste a symbol style on another symbol. For more information about pasting symbol style, see "NEW! Displaying rake icon on symbol" on page 322.
Shape Editing group		
	Suppress All	Suppress all compartments.
	Show as Box	Display the shape as the box, that is, suppress all compartments and do not display as image.
	Show as Image	Display the assigned image instead of the shape.
	Make Preferred Size	Resize the shape so that it uses minimum space.
View group		
	Fit in Window, Zoom 1:1, Zoom In, Zoom Out, and Percentage drop down box	Using the view group buttons, you can change the view of the diagram by zooming it in and out. **NOTE** You can also change the diagram view in the Browser, Zoom panel.

Model Browser

The Model Browser is a hierarchical navigational tool that allows you to manage your model data, including packages, components, classes, all UML diagrams, extension mechanisms, and other data. For more information about Model Browser, see "Model Browser" on page 75.

Floating, Auto-hide, and Close Buttons

All trees of the Model Browser have Toggle Floating, Toggle auto-hide, and Close buttons. You can move or hide a tree using these buttons.



Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Toggle Floating	Press the **Toggle Floating** button, the window is split and you can move the window to any desired position.
	Toggle auto-hide	Press the **Toggle auto-hide** button, the current Browser window is hidden. The tab of the hidden window is displayed on the left side of MagicDraw (in vertical position). Bring the mouse over the hidden window tab and the window is displayed.
	Close	Close the current window.
	Toggle Floating	The floating window is docked again.
	Toggle auto-hide	The auto-hide enabled window is displayed again.

The Model Browser consists of two parts:

1. Containment tree/Inheritance tree/Diagrams tree/Model Extensions tree/Search Results tab, which is located at the top of the Model Browser.
2. **Zoom**/**Documentation**/**Properties** panel, which is located at the bottom of the Model Browser.

Icons of general elements

This section lists all element icons displayed in the Model Browser of MagicDraw. To make your icons search easier, the icons of relationships are listed in the next section “Icons of relationships” on page 1154, and the icons used in modules/profiling mechanism are listed in the section “Icons from Modules and Profile mechanism” on page 1158.

All descriptions are duplicated from the OMG UML Specification with permission.

Icon	Title	Description
	Abstract class	An abstract class is a class, which represents the conceptual set of the classifiers. In the **Class** specification dialog box select the **Is Abstract** option and a class icon changes to the icon of the abstract class.
	Accept Event Action	An accept event action is an action that waits for the occurrence of an event that meets the specified conditions.
	Action	An action is a named element that is the fundamental unit of an executable functionality. The execution of an action represents some transformation or processing in the modeled system, be it a computer system or otherwise.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Activity	An activity element is created on the activity diagram creation.
	Activity Parameter Node	An activity final node is a final node that stops all flows in an activity.
	Actor	An actor represents the roles played by the human users, external hardware, and other subjects.
	Artifact	An artifact represents a physical piece of information that is used or produced by a software development process.
	Artifact Instance	An instance of an artifact.
	Association Class	An Association Class can be seen as an association that also has class properties, or as a class that also has association properties. Not only it connects a set of classifiers, but also defines a set of features that belong to the relationship itself, not to any of the classifiers.
	Attribute	An attribute is a named property of a class that describes a range of values that can be held by instances of that class.
	Call Operation Action	The call operation action transmits an operation call request to the target object, where it may cause the invocation of an associated behavior.
	Central Buffer Node (Object Node)	An object node is an activity node that indicates an instance of a particular classifier, possibly in a particular state, may be available at a particular point in the activity.
	Choice	The choice vertices, when reached, result in the dynamic evaluation of the guards of the triggers of its outgoing transitions.
	Class	A class is the descriptor for a set of objects with similar structure, behavior, and relationships.
	Collaboration	A collaboration is represented as a kind of classifier and it defines a set of cooperating entities to be played by instances (its roles) as well as a set of connectors that define communication paths between the participating instances.
	Collaboration Use	A collaboration use represents a particular use of collaboration to explain the relationships between the properties of a classifier.
	Combined Fragment	A combined fragment defines an expression of interaction fragments. It is defined by an interaction operator and the corresponding interaction operands.
	Comment	A comment gives an ability to display different remarks on diagrams. It may be attached to multiple elements.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Component	A component represents all kinds of elements that pertain to piecing together software applications. They can be simple files, such as DLLs or executables.
	Component Instance	An instance of a component that may reside on a node instance.
	Composite State	A composite state is a state with one region. There is a set of two or more states placed within a region. These are known as substates. The substates describe states within a state.
	Orthogonal State	An orthogonal state is a state with two or more regions. In each region, you may draw one life cycle with both a beginning and an end.
	Conditional Node	A conditional node is a structured activity node that represents an exclusive choice among some number of alternatives.
	Connection Point Reference	A connection point reference represents the use of entry/exit points.
	Constraint	A constraint is a condition or restriction expressed in a natural language text or in a machine readable language for the purpose of declaring some of the semantics of an element.
	Data Store	A data store keeps all tokens that enter it, copies them when they are chosen to move downstream. Incoming tokens containing a particular object replace any tokens in the object node containing that object.
	Data Type	A data type is a type whose instances are identified only by their value.
	Deep History	A deep history represents the most recent active configuration of the composite state that directly contains this pseudostate.
	Decision Node in activity diagram	A decision node is a control node that chooses between outgoing flows. Each token arriving at a decision node can traverse only one outgoing edge. Decisions are made using guard conditions. They help protect transitions that depend on a guarding condition.
	Merge Node in activity diagram	A merge node has multiple incoming edges and a single outgoing edge. It is not used to synchronize concurrent flows but to accept one among several alternate flows.
	Deployment	A node deploys and provides a place to store and/or execute an artifact.
	Deployment Specification Instance	An instance of a deployment specification element.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Device	A device is a physical computational resource. For example; a piece of hardware such as a desktop computer, a processor, a server, or a human work unit such as a department or team.
	Device Instance	An device instance is an instance of a device.
	Duration	A duration defines a value specification that specifies the temporal distance between two time instants.
	Duration Constraint	A duration constraint defines a constraint that refers to a duration interval.
	Element Value	A value specification is an abstract metaclass used to identify a value or values in a model. It may make reference to an instance or it may be an expression denoting an instance or instances when evaluated.
	Entry Point	An entry point connection point reference as the target of a transition implies that the target of the transition is the entry point pseudostate as defined in the submachine of the submachine state.
	Enumeration	An enumeration is a kind of data type, whose instances may be any of a number of user-defined enumeration literals.
	Enumeration Literal	An enumeration literal defines an extension element of an enumeration data type.
	Event (all type of events)	An event is the specification of some occurrences that may potentially trigger effects by an object.
	Execution Environment	An execution environment is a node that offers an execution environment for specific types of components that are deployed on it in the form of executable artifacts.
	Execution Environment Instance	An instance of execution environment.
	Exit Point	An exit point connection point reference as the source of a transition implies that the source of the transition is the exit point pseudostate as defined in the submachine of the submachine state that has the exit point connection point defined.
	Expansion Node	An expansion node is an object node used to indicate a flow across the boundary of an expansion region.
	Expansion Region	An expansion region is a structured activity region that executes multiple times corresponding to the elements of an input collection.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Final State	A special kind of state signifying that the enclosing region is completed. If the enclosing region is directly contained in a state machine and all other regions in the state machine also are completed, then it means that the entire state machine is completed.
	Activity Final	An activity may have more than one activity final node. The first one reached stops all flows in the activity.
	Function Behavior	A function behavior is an opaque behavior that does not access or modify any objects or other external data.
	Hyperlink	An icon with a small arrow image on the left-bottom side indicates that element has a hyperlink to another element/symbol, file, or web page. For more information about hyperlinks, see “Defining Hyperlinks” on page 340.
	Information Flows	The InformationFlows package provides mechanisms for specifying the exchange of information between entities of a system at a high level of abstraction.
	Information Item	An information item is an abstraction of all kinds of information that can be exchanged between objects.
	Initial Node	An initial node is a control node at which a flow starts when the activity is invoked.
	Input Pin Output Pin	An input pin is a pin that holds input values to be consumed by an action. An output pin is a pin that holds output values produced by an action.
	Instance	An instance specification is a model element that represents an instance in a modeled system.
	Interaction	An interaction is a unit of behavior that focuses on the observable exchange of information between connectable elements.
	Interaction Operand	An interaction operand is contained in a combined fragment. It represents one operand of the expression given by the enclosing combined fragment.
	Interaction Use	An interaction use refers to an Interaction. The InteractionUse is a shorthand for copying the contents of the referred Interaction where the interaction use is.
	Interface	An interface is a specifier for the externally-visible operations of a class, component, or other classifier (including subsystems) without a specification of the internal structure.
	Interruptible Region	An interruptible region contains activity nodes. When a token leaves an interruptible region via edges designated by the region as the interrupting edges, all tokens and behaviors in the region are terminated.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Junction	The junction pseudo state corresponds to the merge and the static conditional branch.
	Lifeline	A lifeline represents an object. The lifeline runs from the beginning of the interaction, at the top of the diagram, to the end of the interaction at the bottom of the line.
	Literal Boolean	A literal boolean is a specification of a boolean value.
	Literal Integer	A literal integer is a specification of an integer value.
	Literal Null	A literal null specifies the lack of value.
	Literal Real	A literal real is a specification of a real value.
	Literal String	A literal string is a specification of a string value.
	Literal Unlimited Natural	A literal unlimited natural is a specification of an unlimited natural number.
	Loop Node	A loop node is a structured activity node that represents a loop with the setup, test, and body sections.
	MetaClass	A class whose instances are classes. Metaclasses are typically used to construct metamodels.
	Method (Operation)	An operation is a behavioral feature of a classifier that specifies the name, type, parameters, and constraints for invoking an associated behavior.
	Model	A model is an abstraction of a physical system from a particular point of view.
	System Boundary	A system boundary element consists of use cases related by the exclude or include (uses) relationships.
	Model Library	This icon depicts Model with applied Model Library stereotype. The model library is a package that contains model elements that are intended to be reused by other packages.
	N-ary Association	An n-ary association is an association among two or more classes (a single class may appear more than once).
	Node	A node is a computational resource upon which artifacts may be deployed for execution. The nodes can be interconnected through communication paths to define the network structures.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Node Instance	A node instance is an instance of a node where the component instances may reside.
	Opaque Action	An opaque actions is an action with implementation-specific semantics.
	Opaque Behavior	A behavior with implementation-specific semantics. The Opaque Behavior is introduced for implementation-specific behavior or for use as a place-holder before one of the other behaviors is chosen.
	Opaque Expression	An opaque expression is an uninterpreted textual statement that denotes a (possibly empty) set of values when evaluated in a context.
	Package Model Library	This icon depicts a Package with applied Model Library stereotype. The model library is a package that contains model elements that are intended to be reused by other packages.
	Package	A package groups classes and other model elements together.
	Parameter	A parameter is a specification of an argument used to pass the information on to or out of an invocation of a behavioral feature.
	Port	A port is a property of a classifier that specifies a distinct interaction point between that classifier and its environment or between the (behavior of the) classifier and its internal parts.
	Primitive Type	A primitive type defines a predefined data type, without any relevant substructure.
	Protocol State Machine	A protocol state machine is always defined in the context of a classifier. It specifies which operations of the classifier can be called in which state and under which condition, thus specifying the allowed call sequences on the classifier's operations.
	Signal Reception (Reception)	A reception is a declaration stating that a classifier is prepared to react to the receipt of a signal.
	Region	A region is an orthogonal part of either a composite state or a state machine. It contains states and transitions.
	Send Signal Action	A send signal action is an action that creates a signal instance from its inputs, and transmits it to the target object, where it may cause the discharge of a state machine transition or the execution of an activity.
	Sequence Node	A sequence node is a structured activity node that executes its actions in order.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Shallow History	A shallow history represents the most recent active substate of its containing state.
	Shared Package	Not all module contents are visible in the using project. A module has a shared part and a not shared part. Only the contents of the shared part are visible in the working project. The shared packages are marked with a hand image. For more information about project partitioning, see “Project Partitioning” on page 147.
	Signal	A signal is a specification of send request instances communicated between objects.
	Slot	A slot specifies that an entity modeled by an instance specification has a value or values for a specific structural feature.
	State Machine	A state machines can be used to express the behavior of part of a system.
	State	A state specifies how the object reacts to events occurring around it.
	Stereotype	A stereotype is an extension mechanism that defines a new and more specialized element of the model based on an existing element.
	Structured Activity Node	A structured activity node is an executable activity node that may have an expansion into the subordinate nodes as an activity group.
	Submachine State	A submachine state specifies the insertion of the specification of a submachine state machine.
	Subsystem	A subsystem is a unit of hierarchical decomposition for large systems.
	Subsystem Instance	An instance of the subsystem.
	Swimlane	Actions and subactivities can be organized into a Swimlane in the activity diagrams. The swimlanes are used to organize responsibility for actions and subactivities according to a class.
	Fork Horizontal or Vertical	A fork node is a control node that splits a flow into multiple concurrent flows.
	Join Horizontal or Vertical	A join node is a control node that synchronizes multiple flows.
	Template Parameter	A template parameter exposes a parameterable element as a formal template parameter of a template.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Template Parameter Substitution	A template parameter substitution relates the actual parameter(s) to a formal template parameter as part of a template binding.
	Template Parameter Signature	A template signature bundles the set of formal template parameters for a templated element.
	Time Constraint	A time constraint specifies the combination of min and max timing interval values.
	Time Expression	A time expression defines a value specification that represents a time value.
	Time Observation	An time observation is a reference to a time instant during an execution. It points out the element in the model to observe and whether the observation is made when this model element is entered or when it is exited.
	Time Event	A time event specifies a point of time by an expression. The expression might be absolute or might be relative to some other point of time.
	Trigger	A trigger specification may be qualified by the port on which the event occurred. A trigger relates an event to a behavior that may affect an instance of the classifier.
	Use Case Instance	A use case instance is an instance of a use case.
	Use Case	A use case is a kind of behavior-related classifier that represents a declaration of an offered behavior.
	Value Pin	A value pin is an input pin that provides a value by evaluating a value specification.
	Variable	A variable is considered as a connectable element.
NEW! 	Rake Icon	Rake icon on symbol informs about its internal diagram.

Icons of relationships

Icon	Title	Description
	Relations branch	Most of the relationships are included in the Relations branch in the Browser.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Abstraction	An abstraction is a dependency relationship that relates two elements or sets of elements that represent the same concept at different levels of abstraction or from different viewpoints.
	Aggregation association	An aggregation describes a special type of association designed to help cope with complexity.
	Assembly Connector	An assembly connector is a connector between two components that defines that one component provides the services that another component requires.
	Association	An association answers to the question why two classes of objects need to know about one another.
	Communication Path	The communication path is a subclass of association. It specifies the relationship between nodes by defining the number of nodes that may be connected (multiplicity), and the nature of the connection, via the name of the path or a stereotype.
	Association Class	An association class can be seen as an association that also has class properties, or as a class that also has association properties. Not only it connects a set of classifiers, but also defines a set of features that belong to the relationship itself, not to any of the classifiers.
	Call Message	A call message represents the request to invoke a specific operation.
	Composition	A composition is used for aggregations where the life span of the member object depends on the life span of the aggregate.
	Connector	A connector specifies a link that enables communication between two or more instances.
	Constraint	A constraint is a condition or restriction expressed in a natural language text or in a machine readable language for the purpose of declaring some of the semantics of an element.
	Control Flow	A control flow is an edge that starts an activity node after the previous one is finished.
	Create Message	A create message is message designating the creation of another lifeline object.
	Delegation Connector	A delegation connector is a connector that links the external contract of a component (as specified by its ports) to the internal realization of that behavior by the component parts.
	Dependency	A dependency indicates a semantic relationship between two model elements (or two sets of model elements).

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Deployment	A deployment is a relationship between the location and the artifact.
	Delete Message	A delete message is message designated to terminate another lifeline.
	Direct Association	A directed relationship represents a relationship between a collection of source model elements and a collection of target model elements.
	Directed Aggregation Association	An aggregation describes a special type of association designed to help cope with the complexity. A directed relationship represents a relationship between a collection of source model elements and a collection of target model elements.
	Direct Composition Association	A composition is used for aggregations where the life span of the member object depends on the life span of the aggregate. A directed relationship represents a relationship between a collection of source model elements and a collection of target model elements.
	Element Import	An element import is defined as a directed relationship between an importing namespace and a packageable element.
	Exception Handler	An exception handler is an element that specifies a body to execute in case the specified exception occurs during the execution of the protected node.
	Extend	A relationship from an extending use case to an extended use case that specifies how and when the behavior defined in the extending use case can be inserted into the behavior defined in the extended use case.
	Extension	An extension point identifies a point in the behavior of a use case where that behavior can be extended by the behavior of some other (extending) use case, as specified by an extend relationship.
	Generalization	A generalization is the relationship from the child element (the more specific element, such as a subclass) to the parent (the more general element, such as a super class) that is fully consistent with the first element and that provides additional information.
	Generalization Set	A generalization set defines a particular set of generalization relationships that describe the way in which a general classifier may be divided using specific types.
	Include	An include (uses) relationship from the use case A to the use case B indicates that an instance of the use case A will also contain the behavior as specified by B. The behavior is included at the location which is defined in A.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Information Flow	An information flow specifies that one or more information items circulates from its sources to its targets.
	Interface Realization	An interface realization is a specialized realization relationship between a behavioral classifier and an Interface.
	Link	An instance specification whose classifier is an association represents a link and is shown using the same notation as for an association, but the solid path or paths connect the instance specifications rather than the classifiers.
	Manifestation	A manifestation is the concrete physical rendering of one or more model elements by an artifact.
	Package Merge	A package merge is a directed relationship between two packages that indicates that the contents of the two packages are to be combined.
	Message	A message is a named element that defines one specific kind of communication in an Interaction. A communication can be, for example, raising a signal, invoking an operation, creating or destroying an Instance. The message specifies not only the kind of communication given by the dispatching execution specification, but also the sender and the receiver.
	Send Message	The message was generated by an asynchronous send action.
	Non-navigable Association	An association with non-navigable association ends.
	Object Flow	An object flow is a technique used to capture how objects participate in activities and how they are affected by the activities.
	Package Import	A package import is defined as a directed relationship that identifies a package whose members are to be imported by a namespace.
	Profile Application	A profile application is used to show which profiles have been applied to a package.
	Protocol Transition	A protocol transition specifies a legal transition for an operation.
	Realization	A realization is a specialized abstraction relationship between two sets of model elements, one representing a specification (the supplier) and the other represents an implementation of the latter (the client).
	Component Realization	A component realization concept is specialized in the Components package to (optionally) define the Classifiers that realize the contract offered by a component in terms of its provided and required interfaces.

Copyright © 1998-2015 No Magic, Inc.

Icon	Title	Description
	Reply Message	A reply message is a reply message to an operation call.
	Role Binding	A role binding is a mapping between features of the collaboration type and features of the classifier or operation. This mapping indicates which connectable element of the classifier or operation plays which role(s) in the collaboration.
	Substitution	A substitution is a relationship between two classifiers which signifies that the substitutingClassifier complies with the contract specified by the contract classifier.
	Template Binding	A template binding represents a relationship between a templateable element and a template.
	Transition	A state transition usually has an event attached to it, but it is not necessary to attach one. If an event is attached to a state transition, the state transition will be performed when the event occurs.
	Usage	A usage is a relationship in which one element requires another element (or set of elements) for its full implementation or operation.

Icons from Modules and Profile mechanism

Module, Profile, and Shared package model elements have their own icons that are represented in the Model Browser. You can see the icons, their titles, and descriptions in the following table.

Icon	Title	Description
	Module	If you have (or developing) a large model, which has several weakly dependent parts, it is advisable to split it into several module files. Partitioning has a package level granularity.
	Profile	A Profile is a kind of Package that extends a reference metamodel. The primary extension construct is the Stereotype, which is defined as part of Profiles.
	Profile Model Library exported as module	This icon depicts a profile with applied Model Library stereotype. A package that contains model elements that are intended to be reused by other packages.
	Shared Package	Not all module contents are visible in the working project. A module has a shared part and not shared part. Only the contents of the shared part are visible in the working project. The shared packages are marked with a hand image.



For more information about working with modules and profiling mechanism, see "Project Partitioning" on page 147.

Copyright © 1998-2015 No Magic, Inc.

14 APPENDIX II: UML 2.5 SUPPORT

Introduction

MagicDraw always supports the latest versions of UML standard. The latest version of MagicDraw is no exception.

"Appendix II: UML 2.5 Support" introduces the supported by MagicDraw changes of the UML specification from 2.4.1 to 2.5, including both metamodel and notation changes.

The appendix consists of the following sections:

- "Metamodel Changes" on page 1159
- "Notation Changes" on page 1162
- "Opening UML 2.4.1 models with MagicDraw 18.0 or later" on page 1162

Metamodel Changes

This section summarizes the UML metamodel changes from 2.4.1 to 2.5 supported in MagicDraw and gives some brief information about the impact of each change.

All the changes are grouped by the categories:

- "New Property Names Added" on page 1160
- "Properties Renamed" on page 1160
- "Derivation Changes" on page 1160
- "Order Changes" on page 1161
- "Other Changes" on page 1161
- "Properties Removed" on page 1161

Copyright © 1998-2015 No Magic, Inc.

Updates legend

Notation	Meaning
text	addition
~~text~~	removal

New Property Names Added

Property	Description / Comment
Classifier::conveyingFlow	-
DeployedArtifact::deploymentForArtifact	
PackageableElement::import	

Properties Renamed

Property	Description / Comment
NamedElement::~~classifier~~inheritingClassifier	-
Classifier::~~class~~nestingClass	
ParameterableElement:: ~~parameterSubstitution~~owningTemplateParameterSubstitution	

Derivation Changes

Property	Description / Comment
Property::/classifier : Classifier [0..1]{readOnly, union, subsets Feature::featuringClassifier, subsets RedefinableElement::redefinitionContext}	All these properties have been made automatically derived and read-only unions.
InputPin::/action : Action[0..1]{readOnly, union, subsets Element::owner}	
Action::/input : InputPin[*]{ordered, readOnly, union, subsets Element::ownedElement}	
OutputPin::/action : Action[0..1]{readOnly, union, subsets Element::owner}	
NamedElement::/namespace : Namespace[0..1]{readOnly, union, subsets Element::owner}	
DirectedRelationship::/source : Element [1..*]{readOnly, union, subsets Relationship::relatedElement}	Both properties have been made automatically derived and read-only unions. This feature has been supported in earlier versions of MagicDraw.
DirectedRelationship::/target : Element [1..*]{readOnly, union, subsets Relationship::relatedElement}	
NamedElement::/clientDependency	The **Client Dependency** property of a named element has been made automatically derived. This feature has been supported in earlier versions of MagicDraw.

Copyright © 1998-2015 No Magic, Inc.

Order Changes

Property	Description / Comment
AcceptEventAction::result : OutputPin [0..*]{ordered, subsets Action::output}	The ability to order output pins holding the values received from an event occurrence, has been available in earlier versions of MagicDraw.
Association::/endType : Type [1..*]{~~ordered~~, subsets Relationship::relatedElement}	The set of classifiers that are used as end types of the association, is no longer ordered.
Classifier::/attribute : Property [0..*]{ordered, union, subsets Classifier::feature}	The ability to order the attributes, which are direct properties of a classifier has been available in earlier versions of MagicDraw. Though the **Attribute** property cannot be edited, its value displays the same order of the attributes as *is defined* in other places of the classifier specification, such as the **Owned Attribute** property or the **Attributes** property group. Thus to change the **Attribute** property value, change the value of a relevant property.
ConnectableElement::/end : ConnectorEnd [0..*]{~~ordered~~}	A set of connector ends that attach to the particular connectable element, is no longer ordered.
DurationObservation::event : NamedElement [1..2]{ordered}	Event elements are now automatically ordered.
ReplyAction::♦ replyValue : InputPin [0..*]{ordered, subsets Action::input}	The set of input pins can now be ordered. In the **Reply Value** property value cell, click the ⇅ button to open the dialog for reordering.
UnmarshallAction::♦result : OutputPin [1..*]{ordered, subsets Action::output}	The set of output pins can now be ordered. In the **Result** property value cell, click the ⇅ button to open the dialog for reordering.

Other Changes

Property	Description / Comment
LoopNode::♦loopVariable : OutputPin [0..*]{ordered, subsets Element::ownedElement}	The **Loop Variable** property has been made composite.
OutputPin::loopNode : LoopNode[0..1]{subsets Element::owner}	
Activity::structuredNode : StructuredActivityNode [0..*]{readOnly, subsets Activity::node, Activity::group}	The property has been made read-only in earlier versions of MagicDraw.

Properties Removed

Property	Description / Comment
Property::/default : String[0..1]	The **Default** property no longer appears in a property specification. The value of the **Default Value** property will be used instead of the removed property value.

Copyright © 1998-2015 No Magic, Inc.

Notation Changes

The inherited members are denoted with the caret “^” sign.

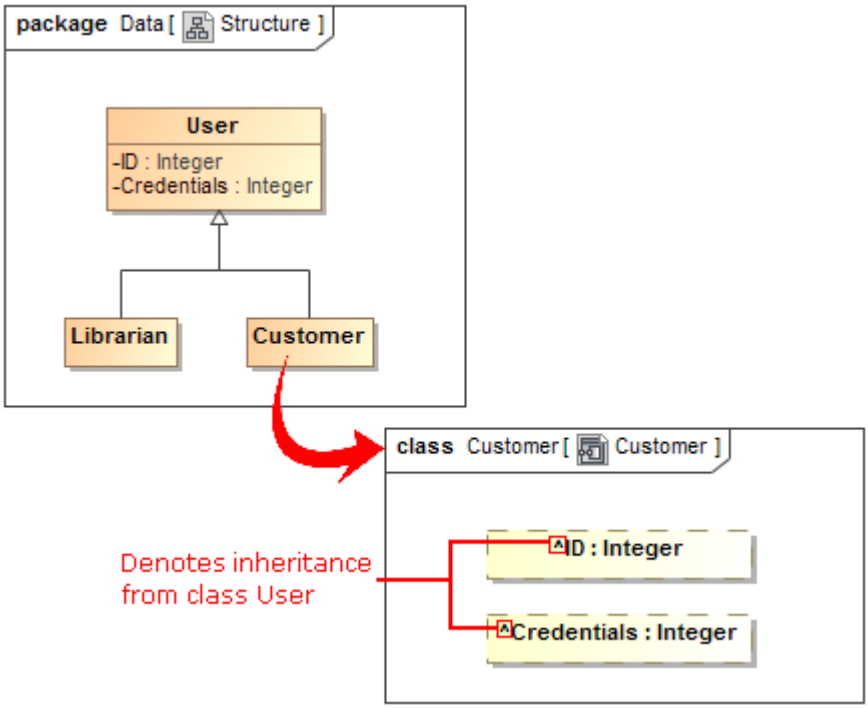


Figure 1 -- Identified inherited members

Opening UML 2.4.1 models with MagicDraw 18.0 or later

UML 2.4.1 models, which are saved with MagicDraw 17.0.5 or earlier, can be successfully modified with MagicDraw 18.0 or later after converting the models to UML 2.5 format.

To convert the older models, simply open them with MagicDraw 18.0 or later and then re-save them. The models will be automatically converted to the latest format and the next time will open as UML 2.5 models. After the conversion to UML 2.5, all property values of these models will persist.

Copyright © 1998-2015 No Magic, Inc.

INDEX

A

About (command) 108, 112, 1089
abstract
    class 877, 898
    operation 994
Abstract (generalizable element) 343, 941
action state 731
activation bar 710, 966
active
    class 898
activity diagram 729
actor 704
    working with 876
aggregation 698
    creating 885
Align (command) 98, 1079
assigning
    classifier to classifier role 965
    classifier to collaboration 905
    classifier to instance 954
    model element to a package 386
    state to object flow state 988
association 698
    in class diagram 878
    in use case diagram 878
    n-ary 881
    navigability of 885
    with a role 893
association class 880
association end
    multiplicity of 886
    qualifier of 887
    visibility of 886
asynchronous message/stimulus 968
attribute 890
    creating new 890
    defining initial value of 954
    multiplicity 895
    scope of 894
    show only public 902
    type modifier of 894
    type of 893, 1040
attributes
    controlling the list of 898
    representing as association 893
    sorting of 902
    suppressing compartment 902

B

binding dependency 923
browser
    changing position 117
    changing size 117
    closing or reopening 117
    code engineering sets in 123
    Containment tree 117
    creating model elements and diagrams in 132
    Diagrams tree 126
    displaying full information in 118
    Documentation 134
    functions of 114
    Inheritance tree 128
    Model Extensions tree 129
    multiple selections 133
    sorting alphabetically 117
    structure 115
    Zoom 135
Browser (command) 106, 1087

C

Call (event type) 1060
Center (command) 98, 1079

Copyright © 1998-2015 No Magic, Inc.

Change (event type) 1060
changeable 895
Check Syntax (command) 101, 1082
class 895
    active 898
    creating setters and getters 495
    defining as abstract, leaf, root and/or active 877, 898
    description of 895
    design patterns 900
    inner elements of 897
    owner of 386
    show/hide package name 386
    suppressing attributes 902
    suppressing operations 902
class diagram 694
    elements in 695
    wizard 539
classifier
    assign to an instance 954
    assigning to a collaboration 905
    assigning to classifier role 965
classifier role
    assigning classifier to 965
    in sequence diagram 710
Close All Diagrams (command) 107, 1088
Close All Projects (command) 88, 1069
closing
    all opened projects 179
    diagram 262
code engineering
    checking syntax 124
    creating new set 123
    editing set 123
code generation 124
    changing properties of 124
collaboration
    assigning classifier to 905
collaboration diagram
    changing numbering in 972
compartments of class 895
component 744, 907
    diagram 742
    instance 744
Component View package 178
composition 885
concurrency of operation 996
concurrent
    operation 996
constraints
    show-hide on symbol 360
    showing on class 903
content diagram 795
Contents (command) 108, 111, 1089
Copy (command) 91, 1072
Copy as EMF (command) 92, 1073
Copy as JPG (command) 92, 1073
copying/cutting
    among different projects 132
    in the Browser 132
    of text 295
    symbol on a diagram 294
    using drag and drop 293
CORBA IDL diagram 806, 818
creating
    backup file 186
    displaying already created relationships 576
    elements and diagrams in Browser 132
    new attribute 890
    new code engineering set 123
    new diagram 260
    new operation 989
    new project style 365
    paths between shapes 279
creating:stereotypes 844
Cut (command) 91, 1072

D

data
    creating several shapes of the same 275
Data package 178
data types 915
DDL diagram 808

Copyright © 1998-2015 No Magic, Inc.

defining
    model elements 341
Delete (command) 92, 1073
deleting
    all model elements 133
    from the Browser 132
    symbol or model element 277
dependency
    binding 923
    permission 924
    usage 924, 925
deployment diagram 743
design patterns 900
destroying sequence object 966
diagram
    closing 262
    creating 260
    defining properties of 371
    information le 273
    opening 261
    renaming 263
    saving as image 331
diagrams
    activity 729
    class 694
    collaboration 706
    content 795
    CORBA IDL 806, 818
    DDL 808
    implementation 742
    robustness 800
    sequence 709
    state 718, 724
    use case 703
    web 802
    WSDL 812
    XML Schema 814, 817, 825
Diagrams menu 99, 1080
display
    related elements 575
documentation of MagicDraw 39
drag and drop
    copying 293
    from browser to diagram 293
    multiple symbols 293
    source code files 294
drawing
    more than one shape 274
    shape 274
    symbol from the Browser 132

E

Edit menu 90, 1071
editing
    code engineering set 123
editions of MagicDraw 24
Entertainment with UML (command) 108, 1089
enumeration 917, 1028
enumeration literal 919
Environment (command) 100, 1081
event 719
Exit (command) 89, 1070
extend 928
extension point 1061
    adding to use case 1063

F

features of MagicDraw 24
File menu 87, 1068
Fit In Window (command) 94, 1075

G

generalizable elements 941
    defining as 941
generalization
    grouping into tree 940
generate
    code from the selected set 124
Generate Framework (command) 101, 1082
getter 495
grid 307
    size 308
    snapping to 307

Copyright © 1998-2015 No Magic, Inc.

style 308
visibility of 307
Grid (command) 95, 1076
grouping
diagrams in Browser 126
generalization paths into tree 940
of model elements 996
realizations into tree 1036
Guarded 996

H

Help menu 108, 1089
HTML text 392
hyperlink 382
for model element 380
hyperlink: for model element 380

I

image
available formats 330
saving as 331
implementation diagram 742
Import Project (command) 88, 1069
include 705, 947
initial value
defining for attribute 954
showing on attribute 902
inner elements
of class (inserting) 897
installation
other platforms 54
system requirements 51
UNIX 54
using no install 54
Windows 54
instance 748
assign classifier to 954
of component 744
of node 745
interface 747
interface style
making changes to 174
Interface Style (command) 100, 1081
internal transition 963

J

JVM (Java Virtual Machine)
for Solaris 53
for Windows 53

L

layout 308
Layout menu 96, 1077
Leaf
class 898
Leaf (generalizable element) 941
lifeline 966
link
in class diagram 701
in collaboration diagram 707
in implementation diagram 751
link attribute 988
link to self 707
Look and Feel (command) 100, 1081
Look and Feel themes 175

M

Mac OS X
installation 54
Magic Draw UML on the Web (command) 108, 1089
MagicDraw
appearance of 174
documentation 39
editions 24
features 24
support 45
updating 70
Make Same Height (command) 98, 1079
menus 86, 1068
message
changing numbering of 972
defining action for 971
diagonal 713

Copyright © 1998-2015 No Magic, Inc.

in collaboration diagram 708, 709
asynchronous 968
synchronous 969
in sequence diagram 976, 979
predecessor of 975
recursive 714
to self 713
uninterpreted 712
Message Window (command) 107, 1088
model
development of 23
static structural 694
model element
defining 341
saving as image 331
model management 694
moving
of all model elements 133
symbol 293
MS Office
copying to 294
175
multiplicity
of an attribute 895
of association end 886

N

name
specifying for a shape 275
n-ary association 881
navigability
of association 885
New Project (command) 87, 1068
Next Diagram (command) 99, 1080
node 744, 980
note 838, 840
adding hyperlink to 382
retrieving documentation 842
text as HTML in 392
note anchor 840
numbering
changing 972
numbers
show/hide on collaboration diagram 971

O

object flow state
assigning state 988
defining classifier for 988
Open Project (command) 87, 1068
opening
diagram 261
diagram from Browser 127
last project on startup 188
project 187
specification dialog box 342
operation
concurrency 996
creating new 989
defining 989
defining as abstract, leaf, or root 995
functions of 989
parameter of 992
return type 994
return type of 991
show only public 902
showing signature of 901
visibility 995
operations
controlling the list of 898
generation of setters and getters 897
managing 498
sorting of 902
suppressing compartment 902
Options menu 100, 1081
ordering
of attributes 902
of operations 902

P

package 696, 746, 996
adding inner elements 997
changing header position 998

Copyright © 1998-2015 No Magic, Inc.

Component View 178
Data 178
dependency wizard 544, 548
showing assigned model elements 998
parameter
of an operation 992
parameterized class 1056
defining of 1056
partition 1051
paste 294
with new data 294
Paste (command) 91, 1072
Paste With New Data (command) 91, 1072
path 259
changing style of 280, 281
defining properties of 371
drawing 279
Paths (command) 92, 1073
patterns See design patterns
permission 924
predecessor 975
presentation options
for class 900
Previous Diagram (command) 99, 1080
primitive 920
Print Active Diagram 89, 1070
Print Options (command) 89, 1070
Print Preview (command) 89, 1070
printing 334
project
dividing into modules 211
multiple projects 178
opening of 187
packages of 178
Project (command) 100, 1081
Projects (command) 89, 1070
properties
changing code generation 124
pseudostate 719

Q

qualifier
of association end 887
Quick Reverse (command) 101, 1082

R

realization
grouping into tree 1036
Redo (command) 90, 1071
Refresh (command) 94, 1075
registering MagicDraw 70
relationship
displaying on diagram 576
renaming
diagram in Browser 127
Retrieve DB Structure (command) 101, 1082
return type
of an operation 994
reverse
changed files 125
code engineering set 124
robustness diagram 800
Root
class 898

S

Save (dialog box) 185
Save As Image (dialog box) 331, 332, 333
Save Project As (command) 87, 1068
saving 185
as image 331
formats 185
in .mdf format 185
project 185
project as template 189
Select All (command) 92, 1073
selection 288
all symbols of the same type 288
multiple 288
multiple (in the Browser) 133
of all model elements 133
of all symbols 288
of symbol 288
separator 841

Copyright © 1998-2015 No Magic, Inc.

adding hyperlink to 382
sequence diagram
model elements in 710
overview 709
Sequential 996
setter 495
shape
defining properties of 371
definition 259
drawing more than one shape 274
drawing of 274
shortcut keys
assigning 176
Signal (event type) 1060
signal receipt
trigger event for 868, 872
Solaris
JVM 53
sorting
of attributes 902
of operations 902
Space evenly (command) 98, 1079
state 720, 725, 731, 1042
state diagram
model elements in 719, 725
state machine 719
statechart diagram See state diagram
stereotypes
defining properties of 371
show/hide on a symbol 360
showing on class 903
stereotypes:creating of 844
stimulus
changing numbering 971
changing numbering of 972
defining action for 971
main information about 708, 709
Report an Issue (command) 108, 111, 1089
support for MagicDraw 45
swimlane 1051
symbols 259
defining properties 359
formatting 371
moving 293
presentation of
constraint 295
note 838, 295
separator 295
text box 295
synchronous 969
system boundary 705

T

tagged values
show/hide on a symbol 360
showing on class 903
template
saving as 189
text box 295
adding hyperlink 382
text as HTML in 392
text editor
choosing 125
Time (event type) 1060
Tip of the Day (command) 108, 111, 1089
toolbars 109, 1090
Tools menu 100, 1081
transition 735
to self 724, 729
type modifier 894
type of an attribute
showing full path 902

U

Undo (command) 90, 1071
Unix
installation 54
Unlock Key (command) 108, 1089
updating MagicDraw 70
usage 924, 925
use case
adding extension point 1063

Copyright © 1998-2015 No Magic, Inc.

extension point 1061

V

View menu 94, 1075
visibility
for operation 995
of association end 886
showing on attribute 902
showing on operation 902

W

web diagram 802
Window menu 106, 1087
Windows
JVM 53
wizards 404, 537
workspace 178
WSDL diagram 812

X

XMI 186
XML Schema diagram 814, 817, 825

Z

Zoom 1:1 (command) 94, 1075
Zoom In (command) 94, 1075
Zoom Out (command) 94, 1075
Zoom To Selection (command) 94, 1075
zooming 306
adjusting step size 307
fit in window 306
to maximum size 307
to original size 306
using Browser 135
zoom in 306
zoom out 306

Copyright © 1998-2015 No Magic, Inc.