BioDWH: A Data Warehouse Kit for Life Science Data Integration

Thoralf Töpel[1,2], Benjamin Kormeier[1], Andreas Klassen[1] and Ralf Hofestädt[1]

[1] Bielefeld University, Bioinformatics Department PO Box 100131, D-33501 Bielefeld, Germany

Summary

This paper presents a novel bioinformatics data warehouse software kit that integrates biological information from multiple public life science data sources into a local database management system. It stands out from other approaches by providing up-to-date integrated knowledge, platform and database independence as well as high usability and customization. This open source software can be used as a general infrastructure for integrative bioinformatics research and development. The advantages of the approach are realized by using a Java-based system architecture and object-relational mapping (ORM) technology. Finally, a practical application of the system is presented within the emerging area of medical bioinformatics to show the usefulness of the approach.
The BioDWH data warehouse software is available for the scientific community at `http://sourceforge.net/projects/biodwh/`.

1 Introduction

The volume of biological knowledge is increasing significantly. Determining the biological function of genes and understanding the interactions of metabolism has become a major challenge in the post-genomic era. Several software tools allow scientists to integrate complex metabolic networks that take place in the living cell. Various biological databases have been created to represent, analyze and gain deeper insights into the complex processes and interactions inside organisms. The latest Nucleic Acids Research database issue counts about 1,000 different molecular biology databases [1].

The importance of data integration in bioinformatics has been recognized for many years. Molecular biology research without analysis and data management is unthinkable. So, it is essential for scientists to access and analyze information from multiple heterogeneous data sources to meet their objectives. Therefore, the challenges of database integration are to combine diverse and multiple data and to bring them into a homogenous, consistent state.

For that purpose, BioDWH is introduced as a Java-based open source toolkit for building life science data warehouses using common relational database management systems such as MySQL, Oracle, and PostgreSQL. By using the object-relational mapping (ORM) technology based on Java Persistence API (JPA), it is no longer necessary to select the local database management system based on the restrictions of the integration software. In fact, most popular databases systems are supported by the JPA framework. Additionally, BioDWH provides a number of ready-to-use parsers to extract data from public life science data sources and to store the content in a data warehouse. This process is supported by an intuitive graphical user interface that reduces time and effort to a minimum.

[2] To whom correspondence should be addressed. E-mail: `thoralf.toepel@uni-bielefeld.de`

doi:10.2390/biecoll-jib-2008-93

2 Related Works

The integration of life science data from heterogeneous, autonomous and distributed data sources is an important research field with special challenges regarding the large heterogeneity of the databases on the semantic and technical level. Existing systems are based on different data integration techniques, mainly text indexing systems (e.g., SRS), multi database and federated database systems (e.g., DiscoveryLink, CompareGRID), and data warehouses (e.g., Atlas, BioWarehouse, BioMart, Columba, Systomonas, ONDEX).

SRS [2] is based on local copies of each integrated data source that can be indexed by using a general description language (Icarus), which describes the structure of the data format of the source files. Recent versions of SRS also allow access to data held in relational databases. SRS has a web interface and is popular among biologists because it is relatively easy to use and was one of the first bioinformatics data access tools to offer a consistent and flexible interface. Various output formats are possible (HTML or ASCI text). One problem with the presentation of the results from SRS is the necessity to parse the outputs for further computer-based processing. The absence of any scheme integration limits SRS as a general data integration platform.

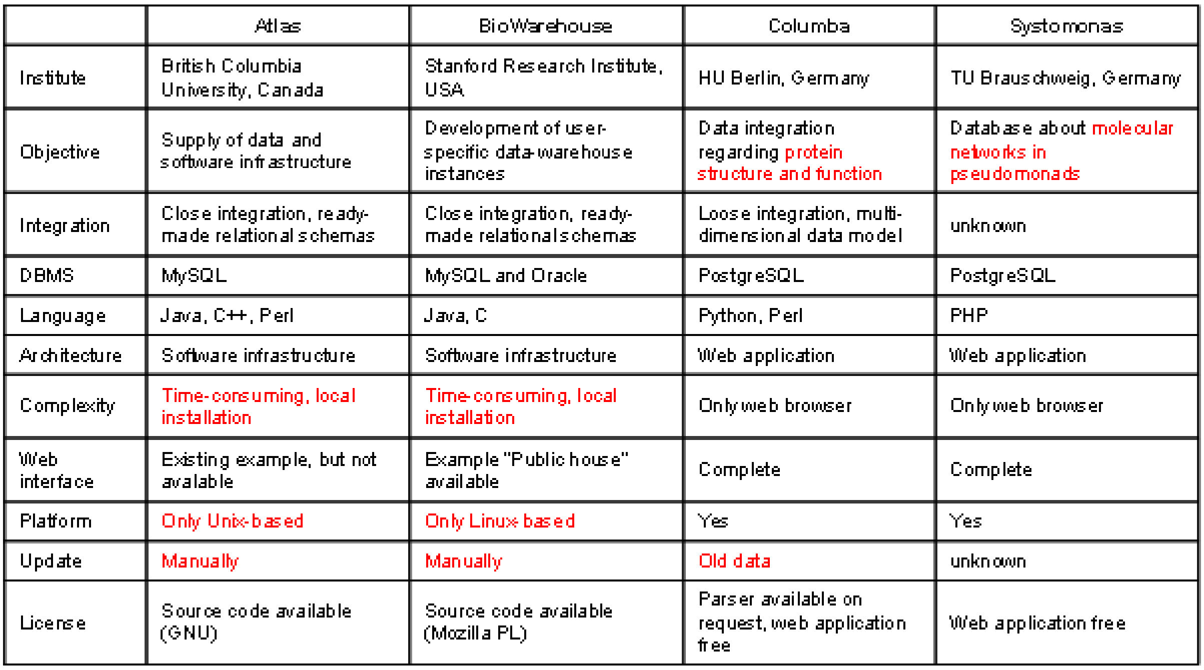
The DiscoveryLink [3] system was developed by IBM to access multiple heterogeneous data sources through single SQL queries. It is based on federated database techniques, so it requires the development of a global data schema. DiscoveryLink accesses its original data sources through wrappers and views. Read-only SQL is supported as a query language. The system is now part of IBM's Websphere Information Integrator. It has been proposed as a platform for integrating life science data, but has not been used in the academic research community.

Data warehouses in bioinformatics can be roughly separated into two groups: general software infrastructures for further customization within new bioinformatics applications (e.g., ONDEX [4], Altas [5], BioWarehouse [6], BioMart [7]) and project-oriented data warehouse implementations for particular biological questions (e.g., Systomonas [8], Columba [9]). The choice of which integration solution is best depends on the nature of the application. Future developments will likely see hybrid solutions emerging as a pragmatic solution to this dilemma. Data integration in life science is also supported by various interdisciplinary standardization and open source activities (e.g., BioRDF, BioJava, BioPerl, EMBOSS).

However, all these approaches have the disadvantage that they are time-consuming when installed locally or they are only available via the web. Additionally, most of them are restricted to an operating system such as Unix/Linux or to specific programming languages. Update strategies constitute another problem; the data is either old or the data warehouse has to be updated manually. Table 1 illustrates these main restrictions of current bioinformatics data warehouse projects. BioDWH intends to increase customization of the data warehouse concept with the advantages of better performance, scalability, up-to-dateness and data quality.

3 The BioDWH System

The system consists of several components that contribute to the data integration process in different ways. These single software components and their collective interaction enable the realization of the following main objectives of BioDWH.



	Atlas	BioWarehouse	Columba	Systomonas
Institute	British Columbia University, Canada	Stanford Research Institute, USA	HU Berlin, Germany	TU Brauschweig, Germany
Objective	Supply of data and software infrastructure	Development of user-specific data-warehouse instances	Data integration regarding protein structure and function	Database about molecular networks in pseudomonads
Integration	Close integration, ready-made relational schemas	Close integration, ready-made relational schemas	Loose integration, multi-dimensional data model	unknown
DBMS	MySQL	MySQL and Oracle	PostgreSQL	PostgreSQL
Language	Java, C++, Perl	Java, C	Python, Perl	PHP
Architecture	Software infrastructure	Software infrastructure	Web application	Web application
Complexity	Time-consuming, local installation	Time-consuming, local installation	Only web browser	Only web browser
Web interface	Existing example, but not available	Example "Public house" available	Complete	Complete
Platform	Only Unix-based	Only Linux based	Yes	Yes
Update	Manually	Manually	Old data	unknown
License	Source code available (GNU)	Source code available (Mozilla PL)	Parser available on request, web application free	Web application free

Table 1: Comparison of four exemplary bioinformatics data warehouses. Main restrictions of these approaches are highlighted in red.

3.1 Main Objectives

The main objective of this approach is to establish a general and flexible data warehouse infrastructure for biological and life science data that is independent from the underlying relational database management system (RDBMS). Configuration of the infrastructure and its tools is possible via Extensible Markup Language (XML), because it is human readable, well-formatted, easy to access and standardized.

Relational database systems are universalized and engrained, because of their flexibility and robustness. The consistent theoretical basis of the relational model makes it possible to protect and guarantee the integrity of data for relational databases. Several relational database management systems have advantages and disadvantages, so that the BioDWH infrastructure has to be independent from the underlying database management system. Consequently, a persistence layer is necessary to achieve this independence from the RDBMS.

Most integrated database systems and data warehouse approaches available are not up-to-date or must be manually updated (see Table 1). Hence, a monitoring utility is needed to keep the system up-to-date. The BioDWH architecture provides a monitoring component that observes multiple data sources, so that different update strategies are configurable. The monitor is configured using XML, too. The monitor component downloads the data from the original source and extracts the compressed data into a local directory. Afterwards, it is possible to automatically start the whole integration process for updating the data warehouse.

A logging mechanism watches the integration process and starts a simple recovery process, if necessary. This error recovery and logging algorithm guarantees a consistent state of the data warehouse. Finally, the data warehouse approach provides an easy-to-use graphical user interface for administration and configuration. The system architecture of BioDWH mainly follows the general data warehouse design. A schematic illustration is shown in Figure 1.

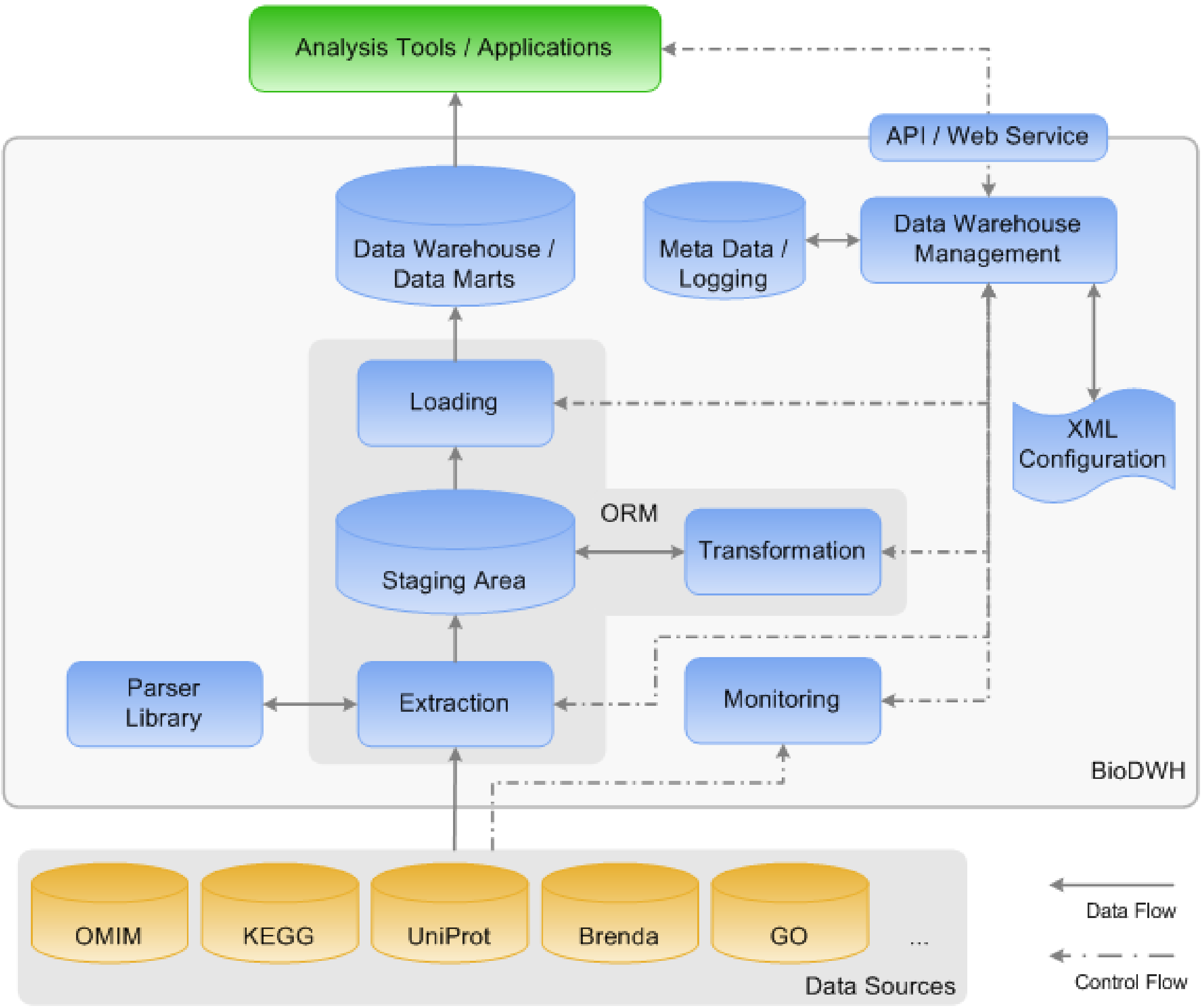


Figure 1: Schematic illustration of the BioDWH system architecture following the general data warehouse design

3.2 Realization

BioDWH is implemented in Java and uses a relational database management system in its backend, e.g., Oracle or MySQL. It provides an easy-to-use Java application for parsing and loading the source data into the data warehouse. Several ready-to-use parsers for popular life science information systems are already available, such as: UniProt, KEGG, OMIM, GO, Enzyme, BRENDA, PDB, MINT, SCOP, EMBL-Bank, and PubChem. Furthermore, an XML-configurable monitor for data source updates is part of the system. For status requests to the data warehouse, we have developed a graphical user interface that works with every web browser.

A well-engineered, object-relational mapping tool called Hibernate was used as a persistence layer, which performs well and is independent from manufacturers like MySQL or Oracle. Additionally, the Hibernate framework fits perfectly into the Java-based infrastructure of the data warehouse. A Java interface and the object-relational mapping using Hibernate persistence or Java Persistence Architecture (JPA) constitute an easy plug-in architecture for integration of new parser.

This object-relational mapping (ORM) is an automated and transparent persistence method of Java application for tables in a relational database system, whereas a mapping between objects and metadata of the database is described. In principle, ORM works with reversible transformation of data from one representation into another. An ORM solution consists of four parts: first, an application programming interface (API) that executes simple CRUD (create, retrieve, update, delete) operations using objects of persistent classes; second, a programming language or API to formulate queries that depend on Java entity classes or properties of classes; third, a facility for mapping metadata; finally, techniques of an ORM implementation to handle interactions of dirty checking, lazy association fetching and other optimization functions of transactional objects.

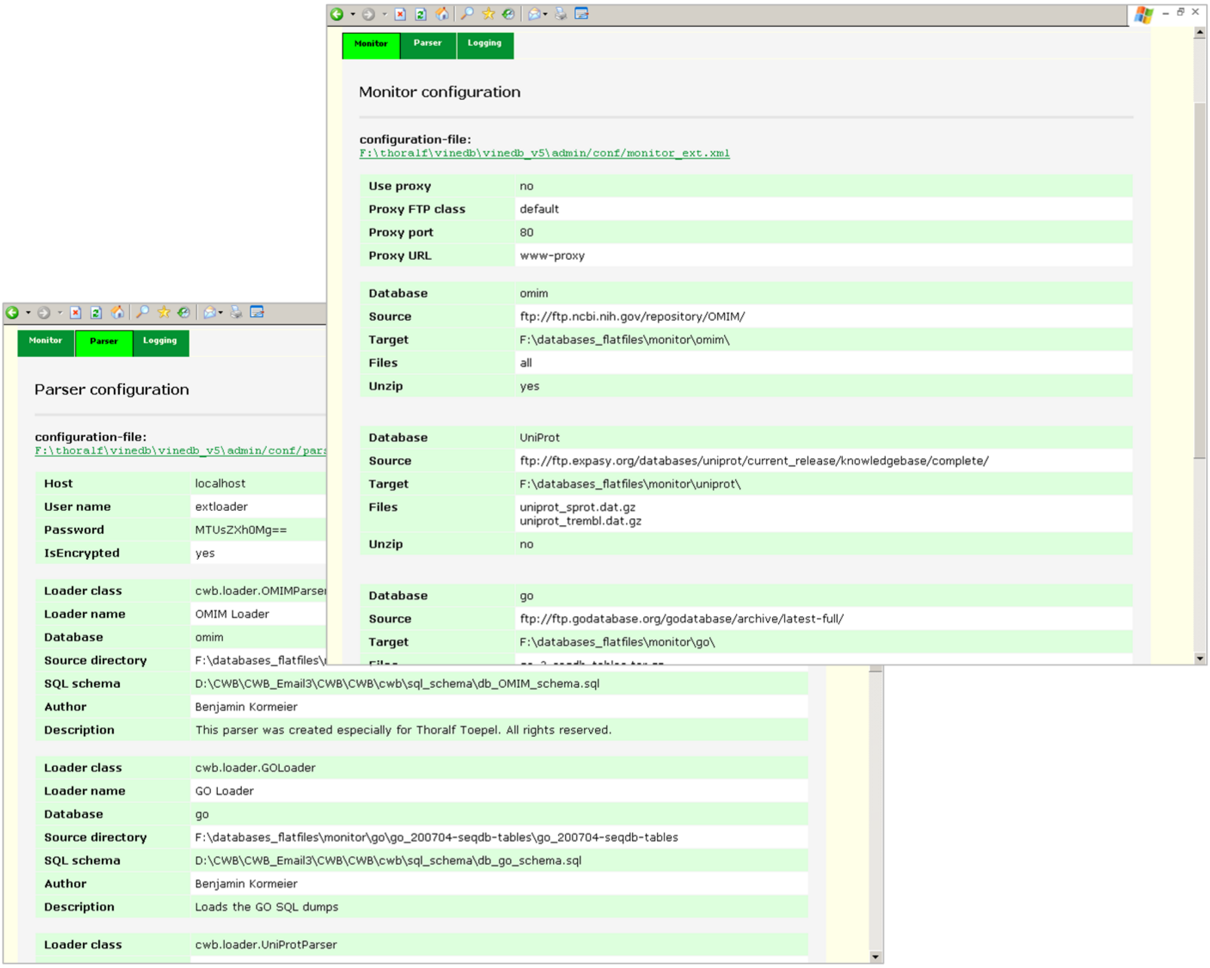
There are different options for implementing ORM: straight relational, light object mapping, medium object mapping and full object mapping. Straight relational means the whole application, including the user interface, is designed based on the relational model and SQL based operations. It is possible to turn SQL in any direction, but there are major problems with portability and maintainability. However, this approach could be a good solution for huge applications. In most cases, these applications use stored procedures and shift tasks from the business layer to the database. Entities are represented as classes in the light object mapping approach. They are manually mapped to relational tables. Manual coded SQL is hidden from the application logic. This approach is very popular and works well for applications with a small number of entities. Medium object mapping applications are based on an object model. SQL statements will be compiled at runtime using framework code or will be generated by a code generator tool. Objects are linked via persistence layer and queries, which can be specified by an object-oriented language. Objects are cached by the persistence layer. The medium object mapping is usually used by mid-size applications that deal with complex transactions. Compatibility between different database brands is granted. Full object mapping supports elaborate object modeling for composition, inheritance, polymorphism and persistence by reachability. Transparent persistence is implemented by a persistence layer. Persistent classes have to implement a particular interface and can not inherent special classes. Lazy, eager and pre-fetching as well as caching strategies are implemented transparent in the application layer. Several open source and commercial Java ORM frameworks reach this level of quality.

The different features of BioDWH are usable by a graphical user interface. It enables the configuration of the monitor and parser for the different public life science data sources as well as the local database management system. A web-based user interface is under construction, using Java Servlets and JavaServer Pages (JSP). Accordingly, it is possible to administrate the entire infrastructure online using a common browser. Figure 2 shows details of the resulting XML-based configuration files. These specify various parameters to access and download the flatfiles from the original data sources, and control the extraction of the downloaded files for integration in the data warehouse afterwards.

4 An Application in Medical Bioinformatics

The BioDWH software infrastructure was already used successfully in several bioinformatics projects. To show the capability of this approach, an application case in the emerging research field of genotype-phenotype correlations will be presented in the following.

Gene mutations cause a number of common as well as rare inherited disorders. Collecting these genotypes along with their phenotypes is extremely important in biomedical research to



Monitor configuration

configuration-file:
F:\thoralf\vinedb\vinedb_v5\admin/conf/monitor_ext.xml

Use proxy	no
Proxy FTP class	default
Proxy port	80
Proxy URL	www-proxy

Database	omim
Source	ftp://ftp.ncbi.nih.gov/repository/OMIM/
Target	F:\databases_flatfiles\monitor\omim\
Files	all
Unzip	yes

Database	UniProt
Source	ftp://ftp.expasy.org/databases/uniprot/current_release/knowledgebase/complete/
Target	F:\databases_flatfiles\monitor\uniprot\
Files	uniprot_sprot.dat.gz uniprot_trembl.dat.gz
Unzip	no

Database	go
Source	ftp://ftp.godatabase.org/godatabase/archive/latest-full/
Target	F:\databases_flatfiles\monitor\go\

Parser configuration

configuration-file:
F:\thoralf\vinedb\vinedb_v5\admin/conf/pars

Host	localhost
User name	extloader
Password	MTUsZXh0Mg==
IsEncrypted	yes

Loader class	cwb.loader.OMIMParse
Loader name	OMIM Loader
Database	omim
Source directory	F:\databases_flatfiles\
SQL schema	D:\CWB\CWB_Email3\CWB\CWB\cwb\sql_schema\db_OMIM_schema.sql
Author	Benjamin Kormeier
Description	This parser was created especially for Thoralf Toepel. All rights reserved.

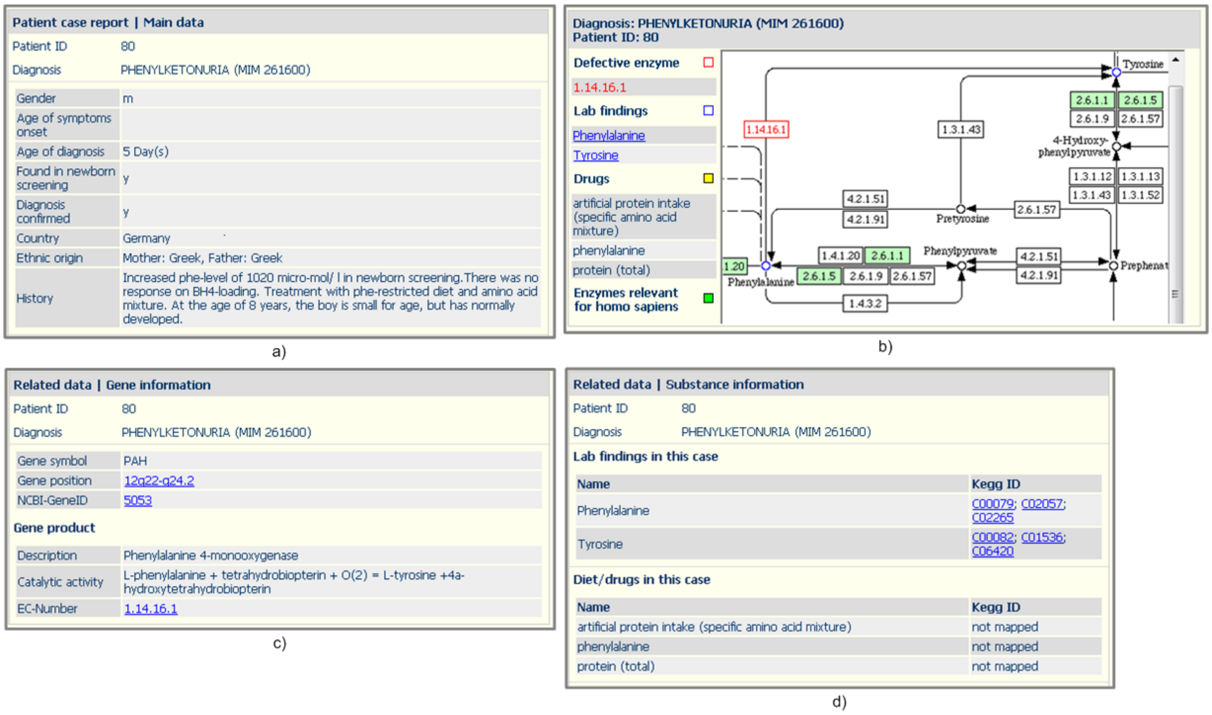
Loader class	cwb.loader.GOLoader
Loader name	GO Loader
Database	go
Source directory	F:\databases_flatfiles\monitor\go\go_200704-seqdb-tables\go_200704-seqdb-tables
SQL schema	D:\CWB\CWB_Email3\CWB\CWB\cwb\sql_schema\db_go_schema.sql
Author	Benjamin Kormeier
Description	Loads the GO SQL dumps

Loader class	cwb.loader.UniProtParser

Figure 2: Screenshot of the web-based administration tool with XML configuration files. a) The monitor configuration mainly consists of parameters that specify the proxy server and the observed data sources identified by ftp directories and relevant files. b) A parser configuration contains the connection parameters for the local database management system and details of the parser that extract and store the data in the DBMS.

provide specific diagnosis tools and new therapeutic approaches. The RAMEDIS system [10] is a platform independent, web-based information system for inherited diseases on the basis of individual case reports. It was developed in close cooperation with clinical partners and collects information on rare metabolic diseases with extensive details, e.g., about occurring symptoms, laboratory findings, therapy and genetic data. RAMEDIS enables co-operative studies and is expected to lead to advances in epidemiology, combining molecular and clinical facts, and generating rules for therapeutic intervention and identification of new diseases. RAMEDIS is available at `http://www.ramedis.de`.

Based on the RAMEDIS information system, data integration was used to bring together clinical data from single case reports such as gene mutation, laboratory values and symptoms with biomedical information on involved genes, enzymes and biochemical pathways. Several databases are available that present considerable information of different aspects of metabolic pathways, such as ExPASy, MetaCyc, Reactome, PANTHER and KEGG, providing both an online map of metabolic pathways and the ability to focus on metabolic reactions in specific organisms. Some have focused more specifically on molecular functions, others on cellular function or on the integration of both levels of function. Several databases have been designed



Patient case report | Main data

Patient ID	80
Diagnosis	PHENYLKETONURIA (MIM 261600)

Gender	m
Age of symptoms onset	
Age of diagnosis	5 Day(s)
Found in newborn screening	y
Diagnosis confirmed	y
Country	Germany
Ethnic origin	Mother: Greek, Father: Greek
History	Increased phe-level of 1020 micro-mol/ l in newborn screening.There was no response on BH4-loading. Treatment with phe-restricted diet and amino acid mixture. At the age of 8 years, the boy is small for age, but has normally developed.

a)

Diagnosis: PHENYLKETONURIA (MIM 261600)
Patient ID: 80

Defective enzyme 🟥

1.14.16.1

Lab findings 🟦

Phenylalanine

Tyrosine

Drugs 🟨

artificial protein intake (specific amino acid mixture)

phenylalanine

protein (total)

Enzymes relevant for homo sapiens 🟩

b)

Related data | Gene information

Patient ID	80
Diagnosis	PHENYLKETONURIA (MIM 261600)

Gene symbol	PAH
Gene position	12q22-q24.2
NCBI-GeneID	5053

Gene product

Description	Phenylalanine 4-monooxygenase
Catalytic activity	L-phenylalanine + tetrahydrobiopterin + O(2) = L-tyrosine +4a-hydroxytetrahydrobiopterin
EC-Number	1.14.16.1

c)

Related data | Substance information

Patient ID	80
Diagnosis	PHENYLKETONURIA (MIM 261600)

Lab findings in this case

Name	Kegg ID
Phenylalanine	C00079; C02057; C02265
Tyrosine	C00082; C01536; C06420

Diet/drugs in this case

Name	Kegg ID
artificial protein intake (specific amino acid mixture)	not mapped
phenylalanine	not mapped
protein (total)	not mapped

d)

Figure 3: Screenshots of the web-based graphical user interface to access the integrated biomedical data in RAMEDIS with links to original data sources. a) General information of the case report, submitted by the author of the report. b) Case-specific representation of the related KEGG map with highlighted elements in different colors: affected enzyme (red), quantified laboratory substances (blue). c) Related genetic information for specific disorder of the case report. d) Overview of biochemical substances in case report, appeared as laboratory substances or drugs.

to represent the function of a specific class of proteins in detail. Enzyme, BRENDA, and EMP describe enzymes; CSNDB and SPAD focus on signaling pathways; TRRD as a part of GeneExpress, Transfac, Transpath and RegulonDB deal with gene regulatory processes.

In order to take advantage of the potential of various valuable biology databases, it must be considered that bioinformatics is an inherently integrative discipline, requiring access to data from a wide range of sources. Consequently, the integration of several life science databases is necessary to provide information with a new level of quality. On the basis of the BioDWH infrastructure, a data integration process was performed to extract the valuable information from OMIM, KEGG, NCBI und ENZYME. Figure 3 shows the graphical user interface of RAEMDIS with the integrated information for a case report of the disorder Phenylketonuria (PKU). Complementary data on involved genes, enzymes and biochemical pathways is provided, depending on the specific case report.

5 Conclusion

Data integration in bioinformatics research is an urgent problem that primarily addresses the heterogeneity and increasing volume of information in life science databases. BioDWH is a Java-based open source toolkit for building life science data warehouses using common relational database management systems. Based on object-relational mapping technology, most relational database management systems can be used for local data storage. Additionally, BioDWH provides a number of ready-to-use parsers to extract data from public life science data sources and to store the content in a data warehouse. Future development will include transformation methods to support user-driven schema integration and the development of plug-in mechanisms to use BioDWH in a broad range of biological applications.

BioDWH has already been used in the context of a medical bioinformatics project to integrate biomedical information with clinical data of rare metabolic diseases. This integrative approach enabled the assembly of a comprehensive view to valuable genotype-phenotype information.

Acknowledgements

This work is supported by Sixth Framework Programme Priority 1 Life Sciences, Genomics and Biotechnology for Health: “CardioWorkBench - Drug Design for Cardiovascular Diseases: Integration of in Silico and in Vitro Analyses” (Proposal/Contract no.: PL 018671).

References

[1] M. Y. Galperin. The molecular biology database collection: 2008 update. *Nucleic Acids Research*, 36(Database issue):D2–D4, 2008.

[2] T. Etzold, A. Ulyanov, and P. Argos. SRS: Information retrieval system for molecular biology data banks. *Methods in Enzymology*, 266:114–128, 1996.

[3] L. M. Haas, P. M. Schwarz, P. Kodali, E. Kotlar, J. E. Rice, and W. C. Swope. DiscoveryLink: A system for integrated access to life science data sources. *IBM Systems Journal*, 40(2):489–511, 2001.

[4] J. Köhler, J. Baumbach, J. Taubert, M. Specht, A. Skusa, A. Rueegg, C. Rawlings, P. Verrier, and S. Philippi. Graph-based analysis and visualization of experimental results with ondex. *Bioinformatics*, 22(11):1383–1390, 2006.

[5] S. P. Shah, Y. Huang, T. Xu, M. M. S. Yuen, J. Ling, and B. F. F. Ouellette. Atlas – a data warehouse for integrative bioinformatics. *BMC Bioinformatics*, 6:34, 2005.

[6] T. J. Lee, Y. Pouliot, V. Wagner, P. Gupta, D. W. J. Stringer-Calvert, J. D. Tenenbaum, and P. D. Karp. BioWarehouse: a bioinformatics database warehouse toolkit. *BMC Bioinformatics*, 7:170, 2006.

[7] S. Durinck, Y. Moreau, A. Kasprzyk, S. Davis, B. De Moor, A. Brazma, and W. Huber. BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. *Bioinformatics*, 21(16):3439–40, 2005.

[8] C. C. Choi, R. Münch, S. Leupold, J. Klein, I. Siegel, B. Thielen, B. Benkert, M. Kucklick, M. Schobert, J. Barthelmes, C. Ebeling, I. Haddad, M. Scheer, A. Grote, K. Hiller, B. Bunk, K. Schreiber, I. Retter, D. Schomburg, and D. Jahn. SYSTOMONAS – an integrated database for systems biology analysis of Pseudomonas. *Nucleic Acids Research*, 35(Database issue):D533–D537, 2007.

[9] S. Trißl, K. Rother, H. Müller, T. Steinke, I. Koch, R. Preissner, C. Frömmel, and U. Leser. Columba: an integrated database of proteins, structures, and annotations. *BMC Bioinformatics*, 6:81, 2005.

[10] T. Töpel, R. Hofestädt, D. Scheible, and F. Trefz. RAMEDIS: the rare metabolic diseases database. *Applied Bioinformatics*, 5(2):115–118, 2006.

[11] G. Joshi-Tope, M. Gillespie, I. Vastrik, P. D'Eustachio, E. Schmidt, B. de Bono, B. Jassal, G. R. Gopinath, G. R. Wu, L. Matthews, S. Lewis, E. Birney, and L. Stein. Reactome: a knowledgebase of biological pathways. *Nucleic Acids Research*, 33(Database issue):D428–D432, 2005.

[12] M. Kanehisa, S. Goto, S. Kawashima, Y. Okuno, and M. Hattori. The KEGG resource for deciphering the genome. *Nucleic Acids Research*, 32(Database issue):D277–D280, 2004.

[13] C. J. Krieger, P. Zhang, L. A. Mueller, A. Wang, S. Paley, M. Arnaud, J. Pick, S. Y. Rhee, and P. D. Karp. MetaCyc: a multiorganism database of metabolic pathways and enzymes. *Nucleic Acids Research*, 32(Database issue):D438–D442, 2004.

[14] P. Karp. A strategy for database interoperation. *Journal of Computational Biology*, 2(4):573–586, 1995.