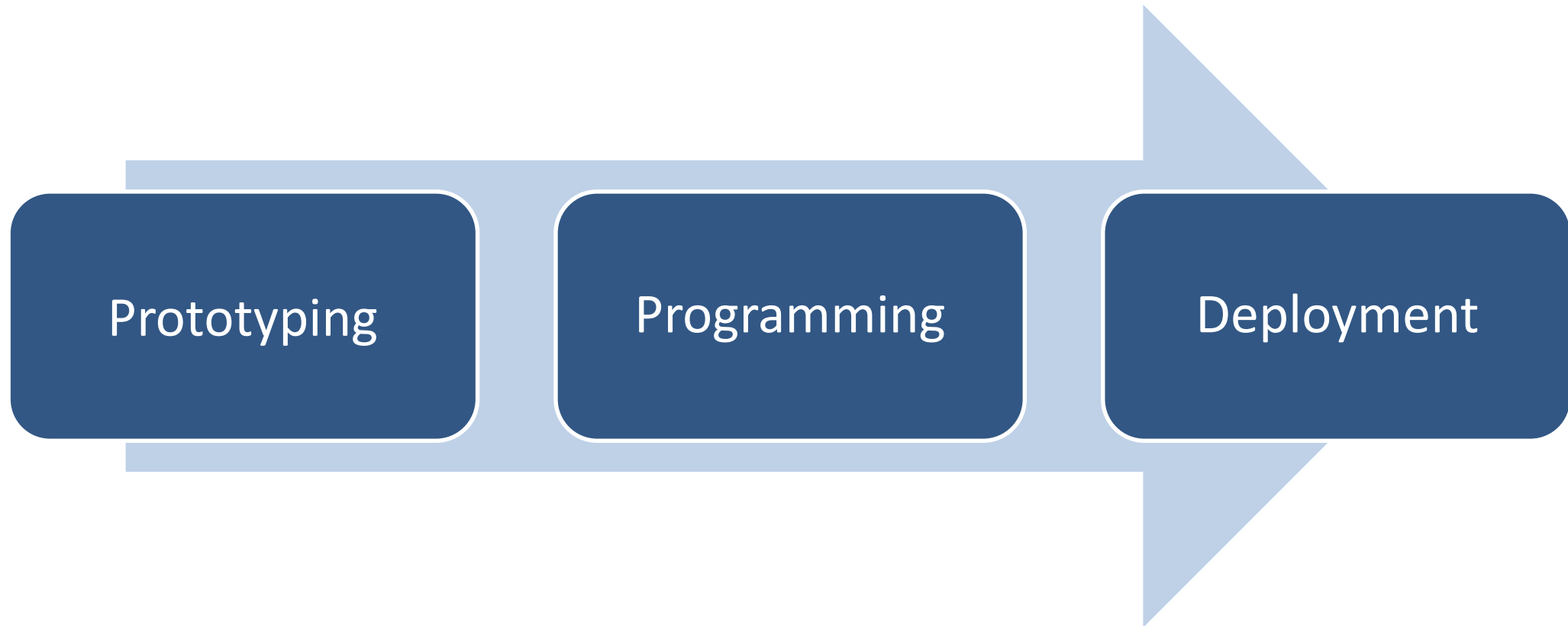


Programming with MATLAB

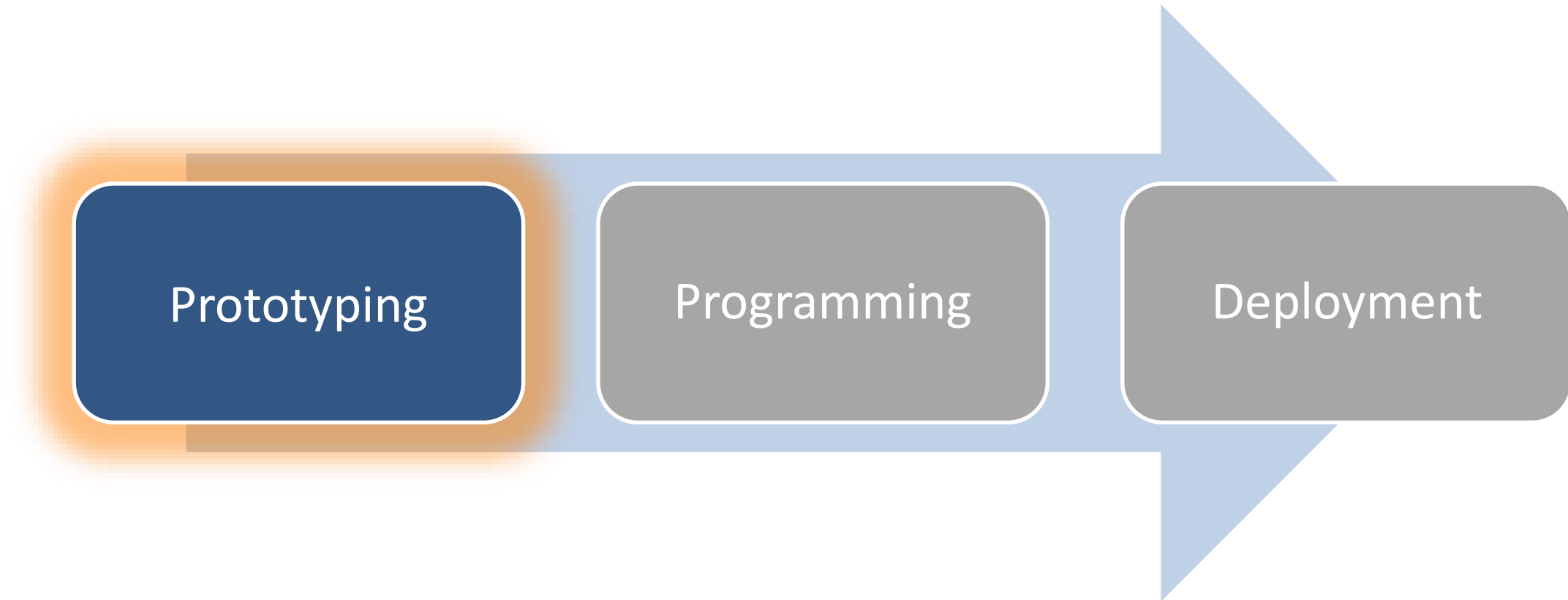
Paolo Fabbri
Senior Engineer



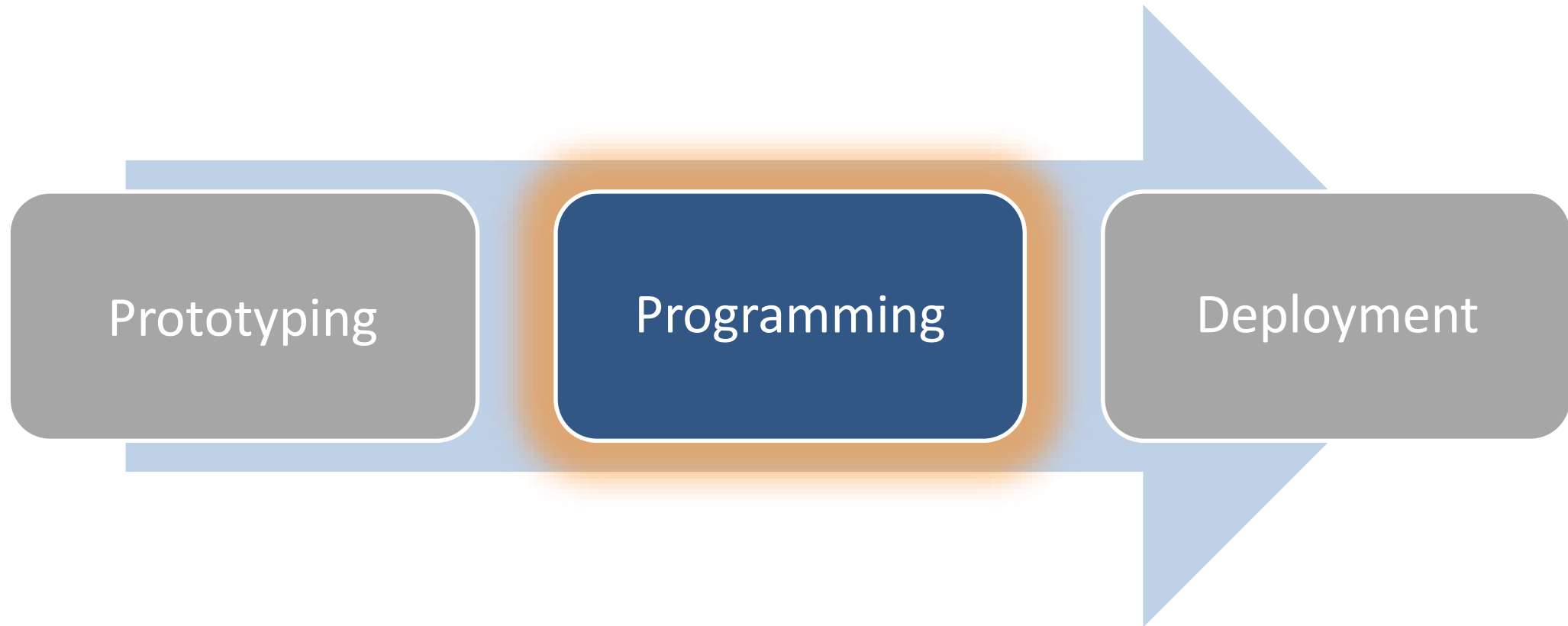
MATLAB Application Development Landscape



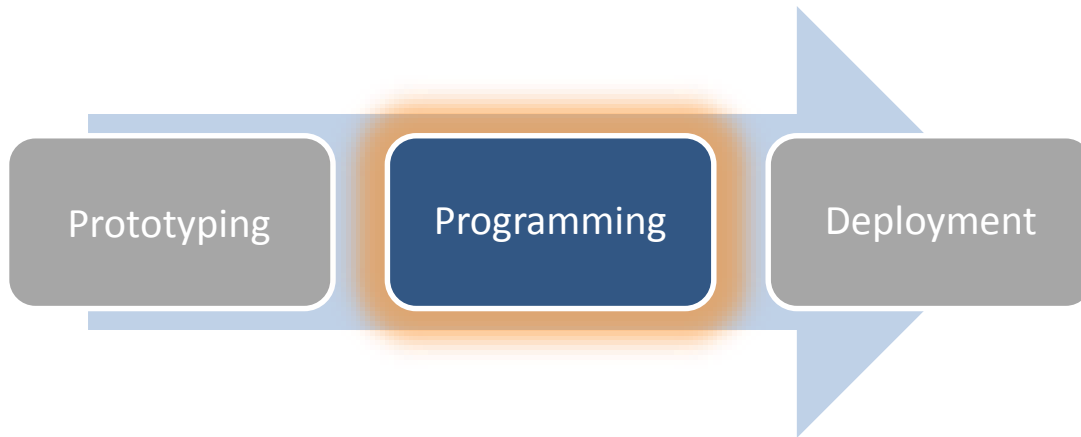
MATLAB Application Development Landscape



MATLAB Application Development Landscape



Today Focus



Object-Oriented Programming

Unit Test Framework

Source Control Integration

Programming Interfaces

What is a program?

What is a program?

Code

```
x = 12
while (x < 100)
    x = x+1
    if (x == 23)
        disp('Hello')
    end
end
```

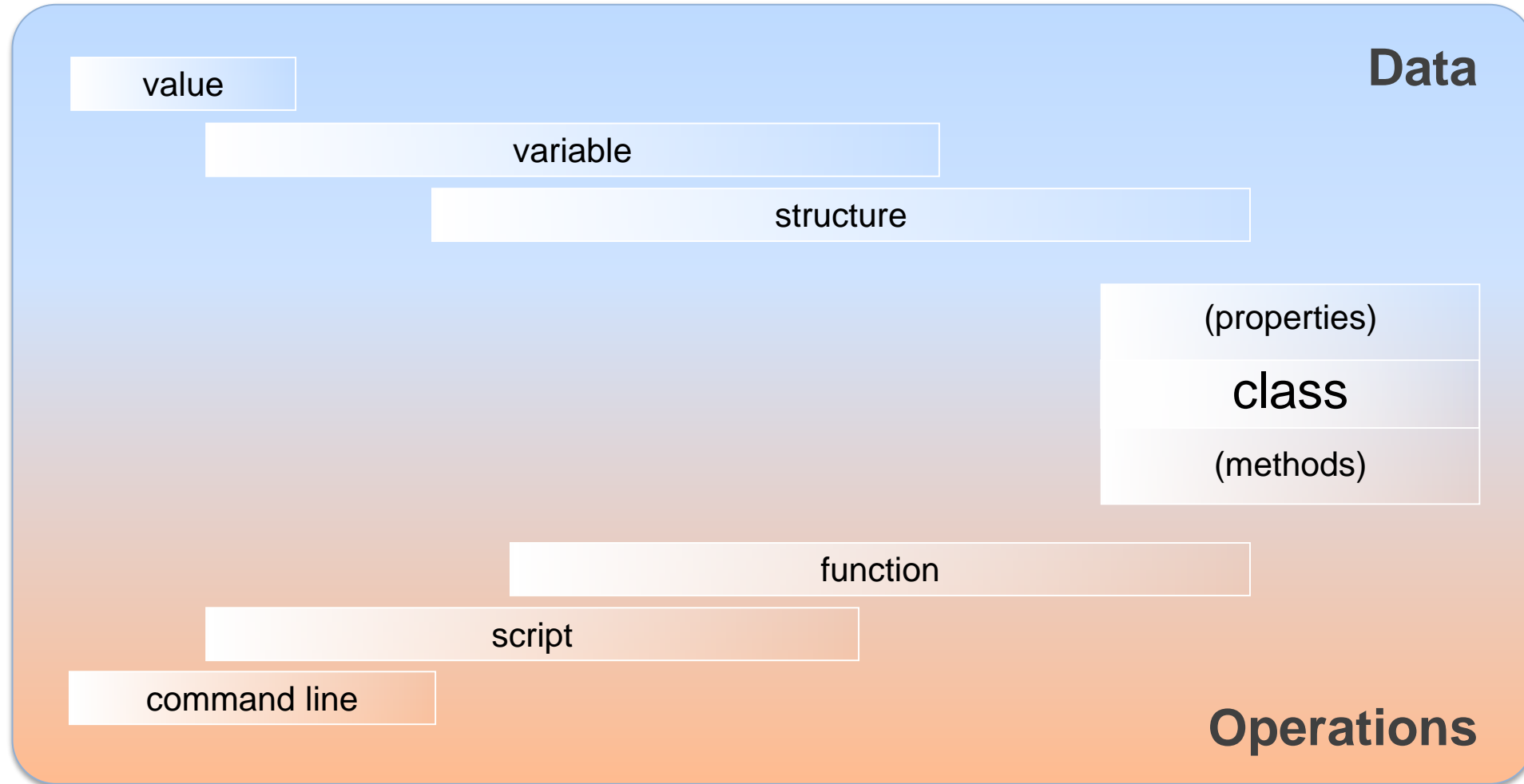
Data

```
x = 12
while (x < 100)
    x = x+1
    if (x == 23)
        disp('Hello')
    end
end
```

Operations

```
Assignment
Looping Test
Increment
Test to Act
    Take Action
End
End
```

Range of Programming Techniques



Classes and Objects

People	
Properties	Methods
<ul style="list-style-type: none"> •FirstName •LastName •BirthDate •Address •Contacts 	<ul style="list-style-type: none"> •getFullName •getAge

Paolo	
Properties	Methods
<ul style="list-style-type: none"> •FirstName = Paolo •LastName = Fabbri •BirthDate = ... •Address = ... •Contacts = ... 	<ul style="list-style-type: none"> •getFullName •getAge

Davide	
Properties	Methods
<ul style="list-style-type: none"> •FirstName = Davide •LastName = Ferraro •BirthDate = ... •Address = ... •Contacts = ... 	<ul style="list-style-type: none"> •getFullName •getAge

Object-Oriented Programming Basics

properties

encapsulate object data

methods

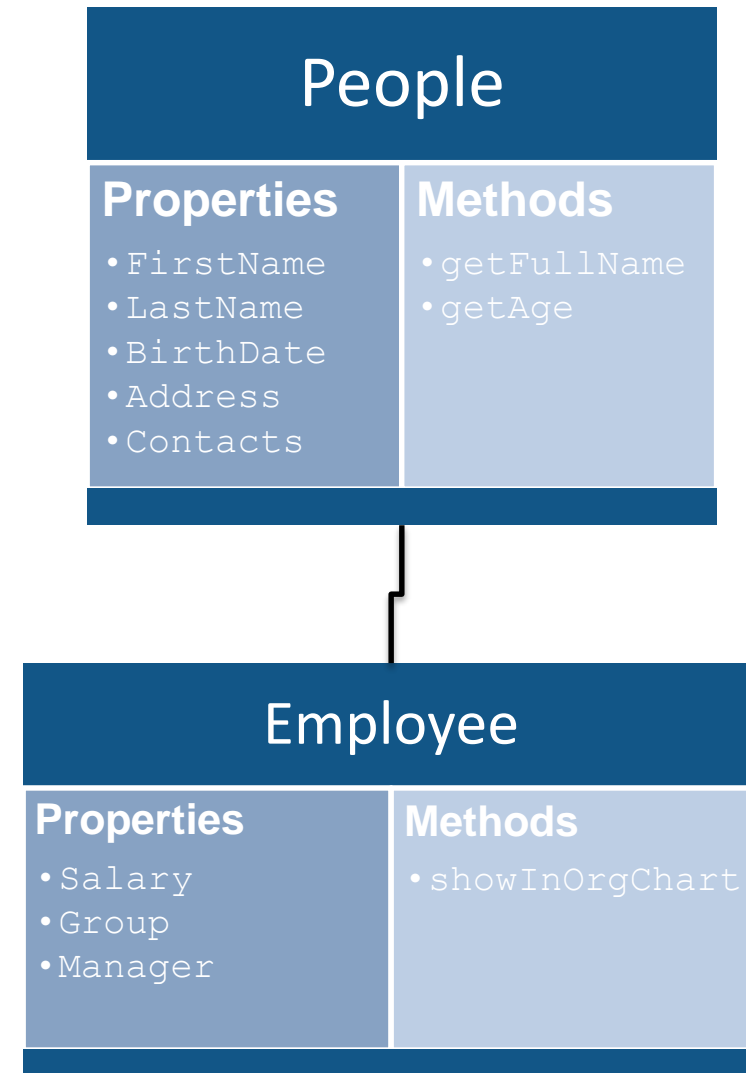
implement the object behavior

events and listeners

implement objects communication

inheritance

allows composition and reusability





Object-Oriented Programming with MATLAB

properties

encapsulate object data

methods

implement the object behavior

events and listeners

implement objects communication

inheritance

allows composition and reusability

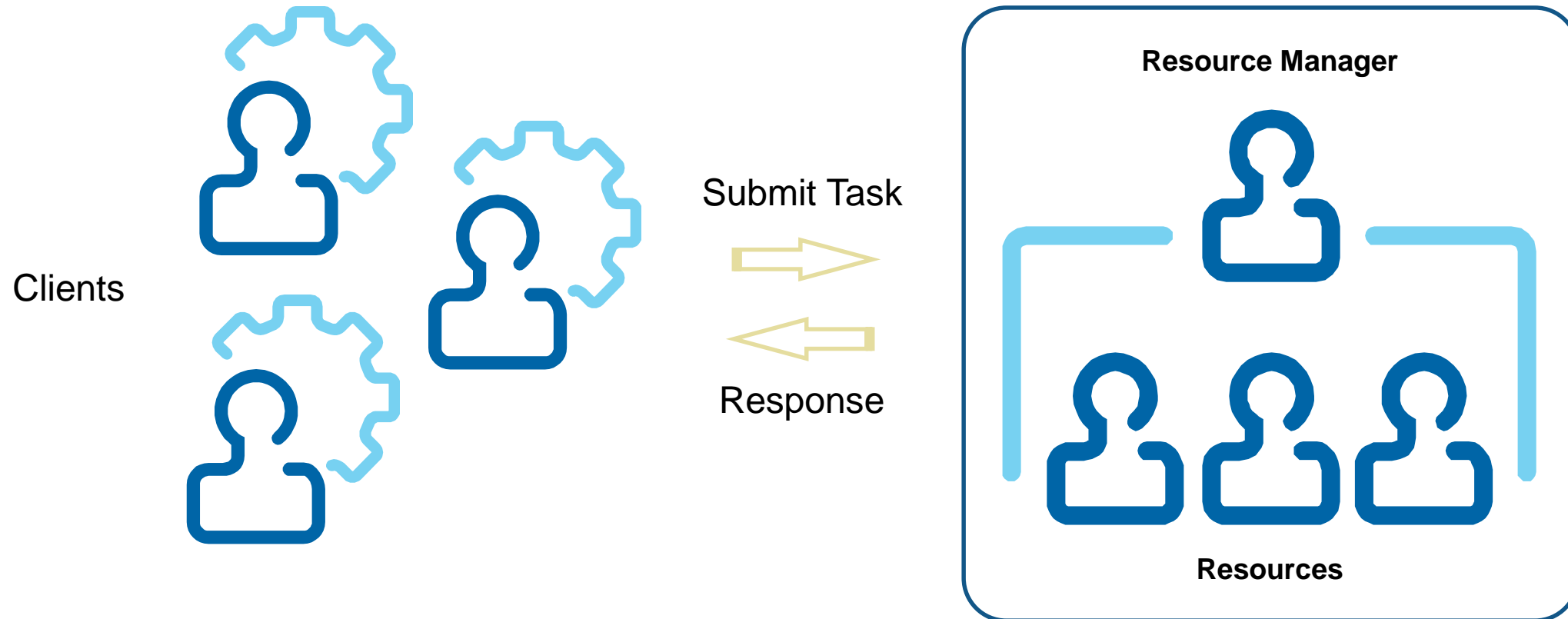
The screenshot shows the MATLAB R2013a Editor window. The title bar indicates the file path: `C:\Data\mathworks\activities\engineering\matlab\demos\windfarm\+components\@Farm\Farm.m*`. The editor displays the following code:

```

1  classdef Farm < handle
2      %FARM Class <Class description goes here>
3
4      %% Properties
5      properties (Access = protected) ...
12
13      properties (GetAccess = public, SetAccess = protected) ...
29
30      properties (Dependent) ...
43
44
45      %% Public Interface
46      methods
47
48      %% Class Constructor

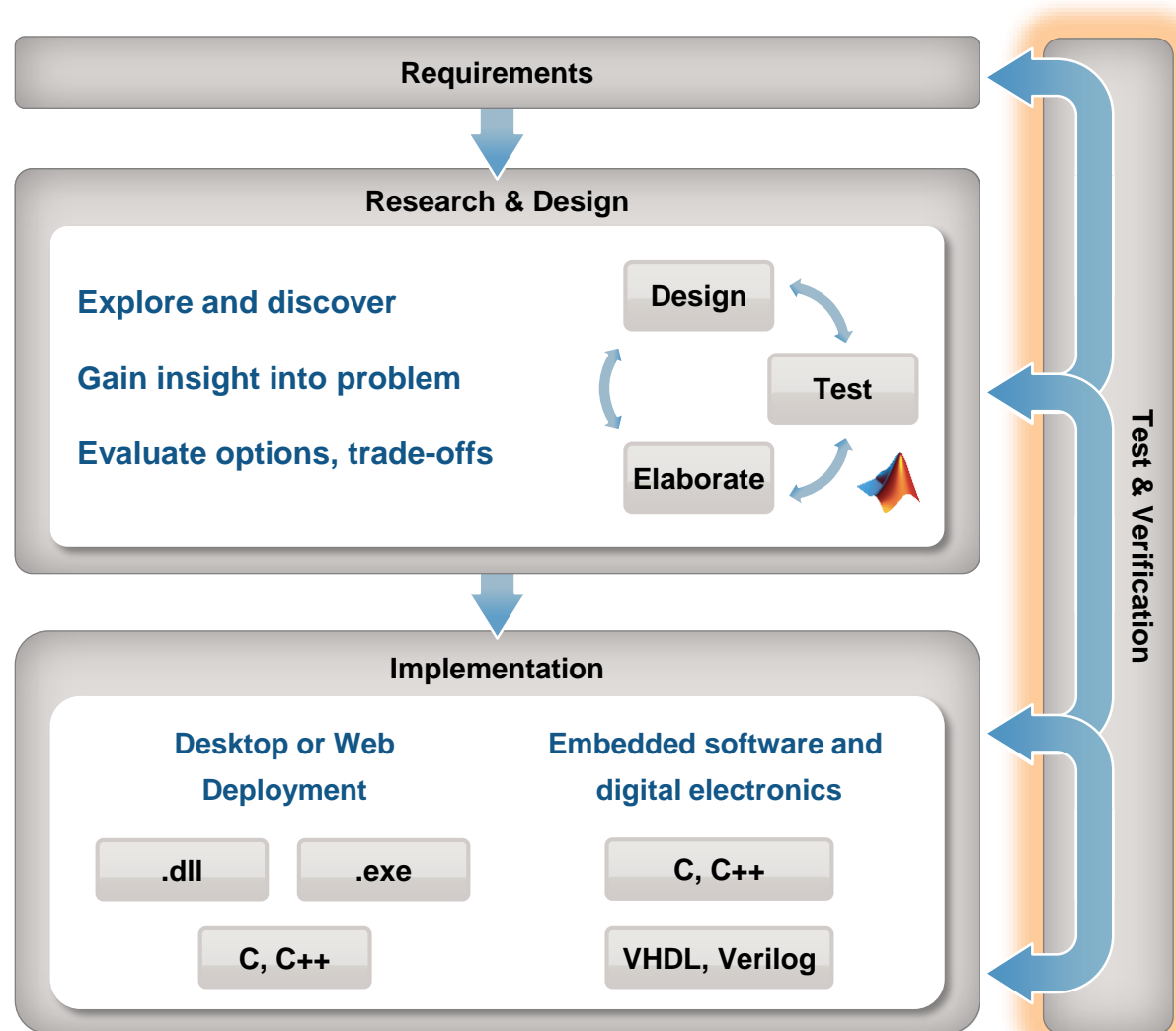
```

Case Study: Team Resources Assignment



MATLAB Unit Test Framework

Application Development Process

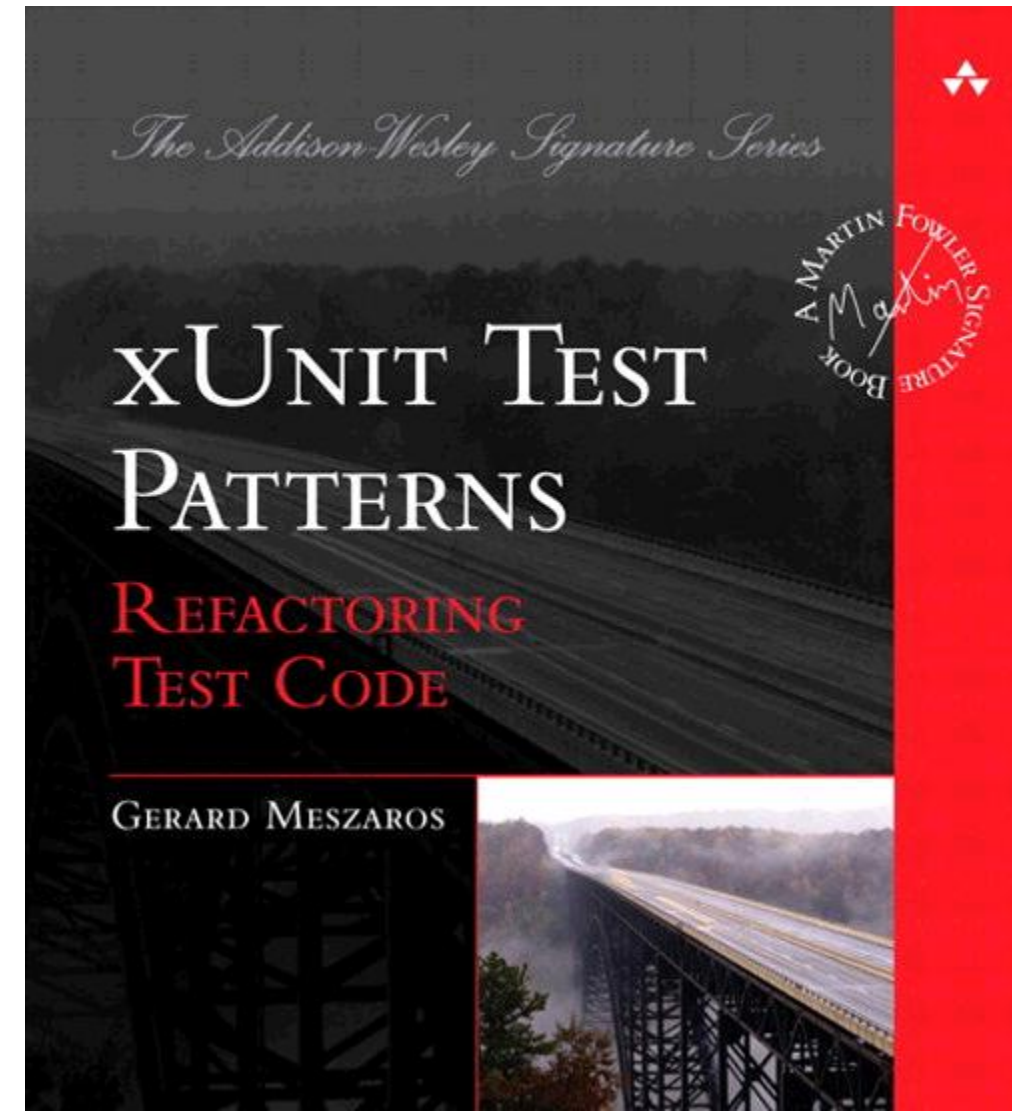




xUnit Patterns

Many testing Best Practices are emerging built around xUnit

- Consistent nomenclature
- Consistent architecture
- Platform for robust testing methodologies
- Setup/Teardown capabilities





MATLAB Unit Test Framework

Write, run and analyze tests for MATLAB programs

Write in function form or **xUnit-style** class form

Run tests **individually** or grouped into a **test suite**

Analyze values using **qualification functions**

Setup and **teardown** to pretest and restore a state

Advanced testing capabilities

```
1 classdef SolverTestClass < matlab.unittest.TestCase
2     % SolverTest tests solutions to the quadratic equation
3     % a*x^2 + b*x + c = 0
4
5     methods (Test)
6
7         %Test real solution
8         function testRealSolutionNew(testCase)
9             %Modify something here...
10            actSolution = quadraticSolver(1, -3, 2);
11            expSolution = [2, 1];
12            testCase.verifyEqual(actSolution, expSolution);
13        end
14
15        %Test imaginary solution
16        function testImaginarySolution(testCase)
17            actSolution = quadraticSolver(1, 2, 10);
18            expSolution = [-1+3i, -1-3i];
19            testCase.verifyEqual(actSolution, expSolution);
20        end
21    end
end
```




Types of Qualifications

Type	Action
Verify	Fail & Continue Execution
Assert	Fail & Halt Current Test, Continue to Next
Fatal assert	Fail & Halt Framework Execution
Assume	Filter Current Test

		Assumption	Assertion	Fatal Assertion
		<code>assumeTrue</code>	<code>assertTrue</code>	<code>fatalAssertTrue</code>
		<code>assumeFalse</code>	<code>assertFalse</code>	<code>fatalAssertFalse</code>
	Value is equal to specified value.	<code>assumeEqual</code>	<code>assertEqual</code>	<code>fatalAssertEqual</code>
	Value is not equal to specified value.	<code>assumeNotEqual</code>	<code>assertNotEqual</code>	<code>fatalAssertNotEqual</code>
	Two values are handles to same instance.	<code>assumeSameHandle</code>	<code>assertSameHandle</code>	<code>fatalAssertSameHandle</code>
	Value is not handle to specified instance.	<code>assumeNotSameHandle</code>	<code>assertNotSameHandle</code>	<code>fatalAssertNotSameHandle</code>
	Function returns true when	<code>assumeReturnsTrue</code>	<code>assertReturnsTrue</code>	<code>fatalAssertReturnsTrue</code>

Source Control Integration

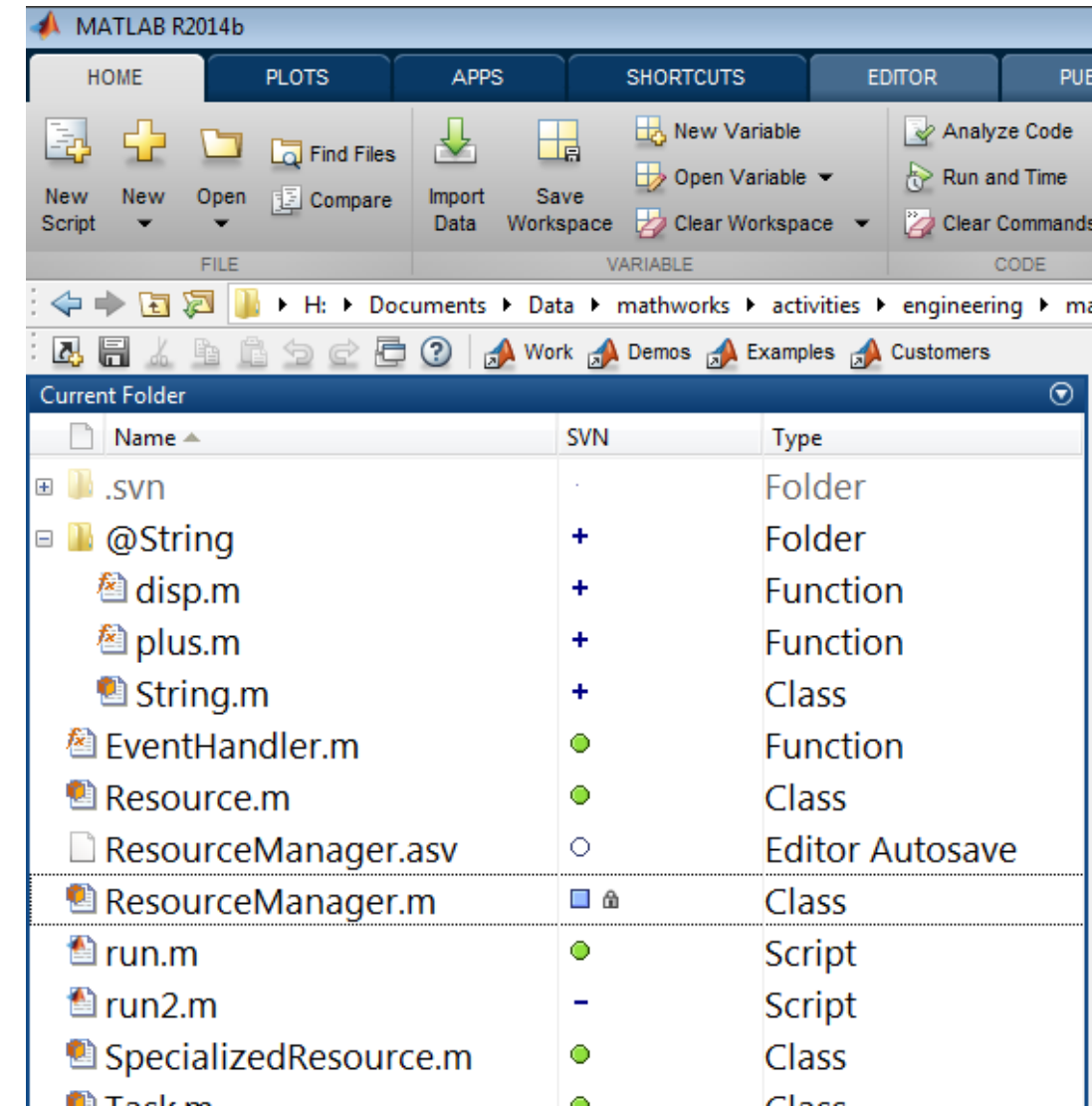


Source Control Integration in MATLAB R2014b

Stay inside MATLAB for development workflow

GIT and **Subversion** Integration in Current Folder

Integrated with tools designed for MathWorks file types (i.e. compare and merge)



Programming Interfaces



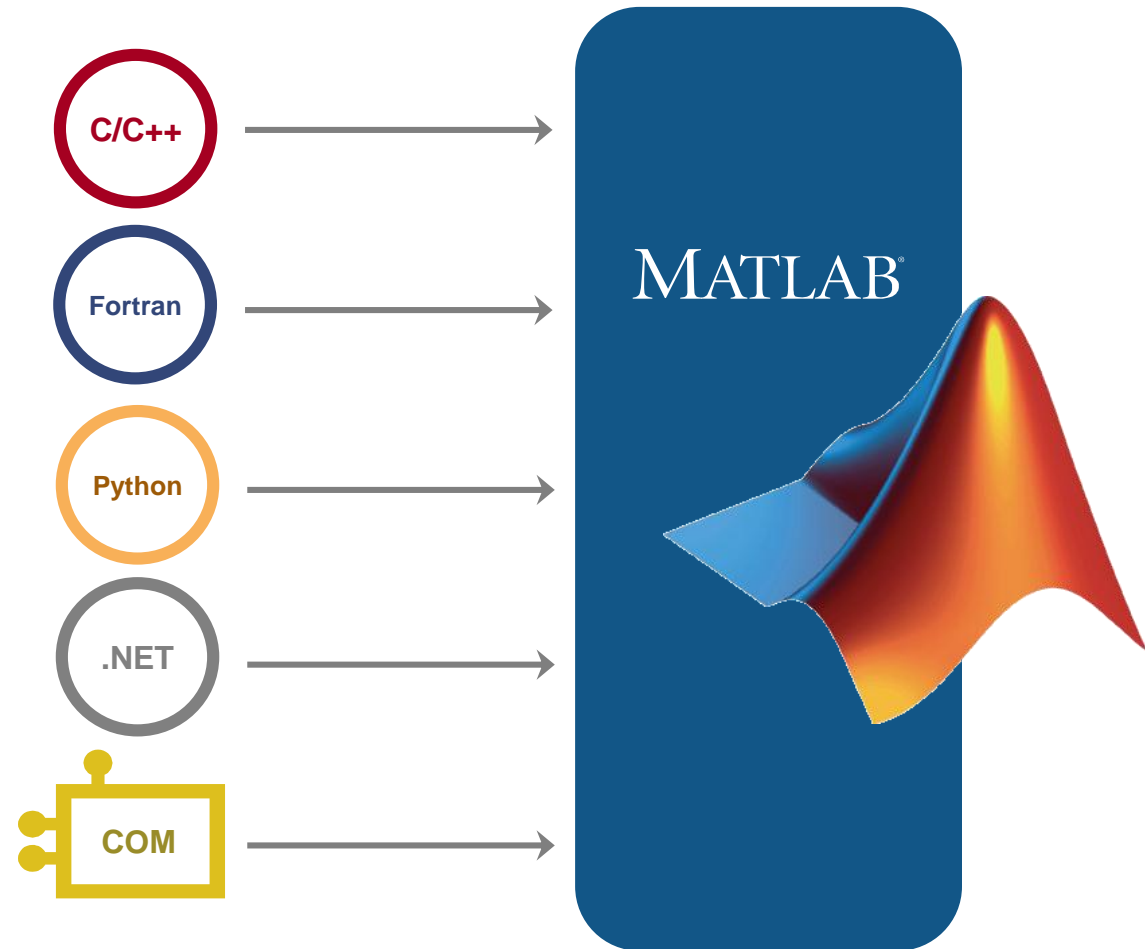
Calling MATLAB from Other Languages

Engine interface

- C/C++
- Fortran
- Python® **R2014b**

Automation server

- COM
- .NET





Calling Other Languages from MATLAB

MATLAB C and Fortran API

```
>> mex mycode.c
```

MATLAB interface to C shared library

```
>> loadlibrary('foo', 'foo.h')
```

MATLAB COM client support

```
>> actxserver('Excel.Application')
```

MATLAB .NET interface

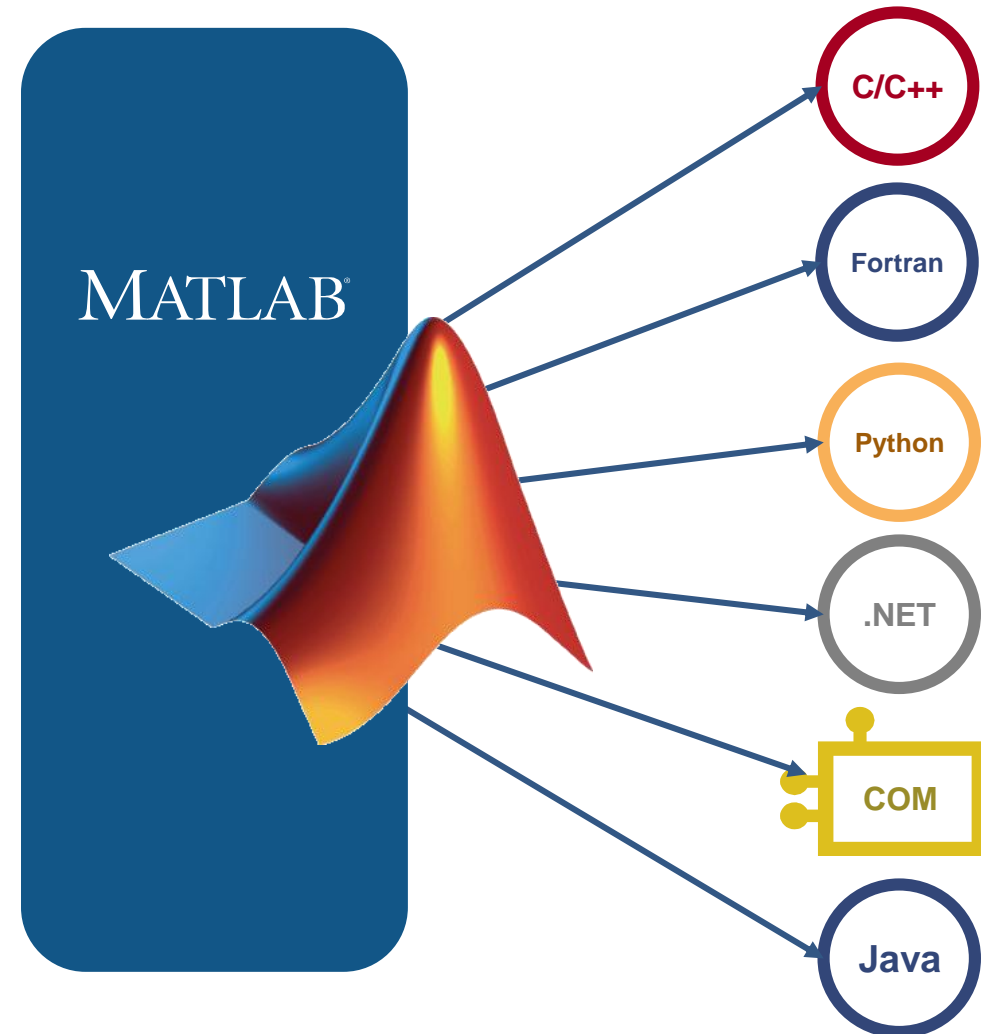
```
>> NET.addAssembly('System.Speech')
```

MATLAB Java interface

```
>> java.lang.String('boo')
```

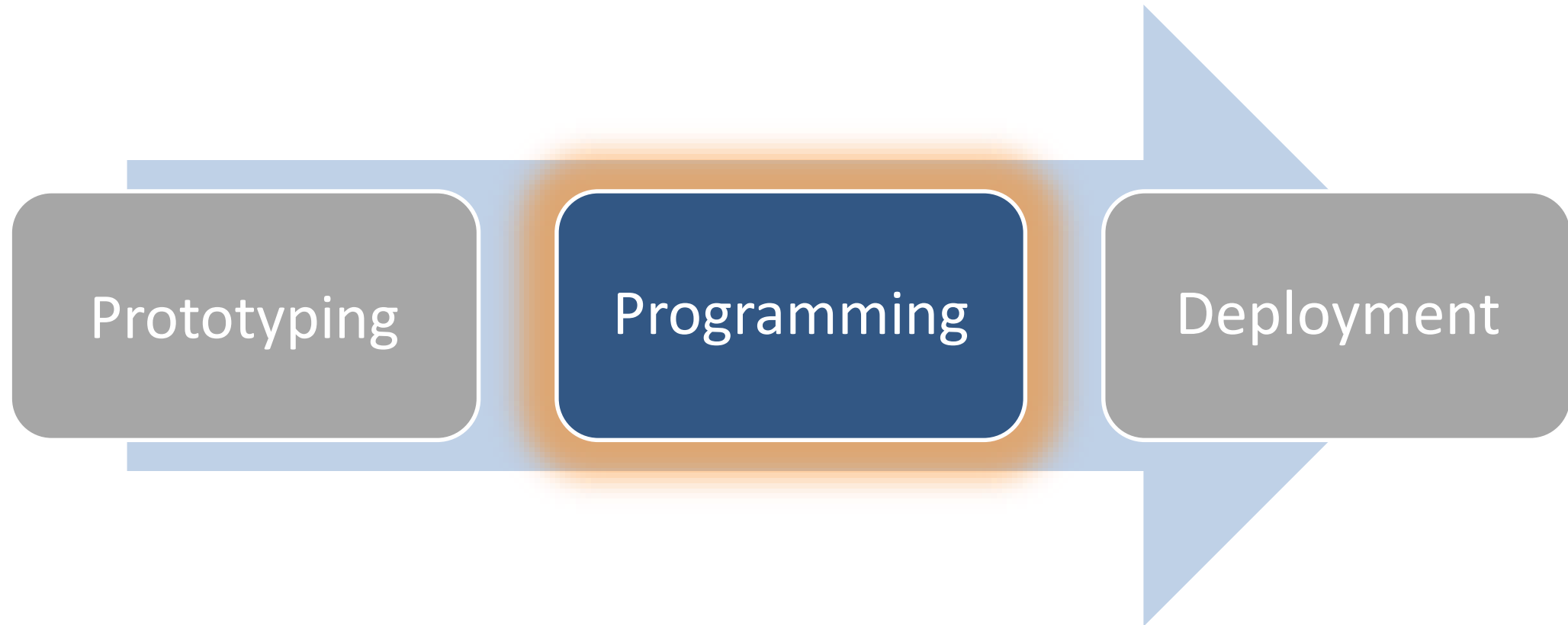
MATLAB Python interface

```
>> py.textwrap.wrap('Text'); R2014b
```

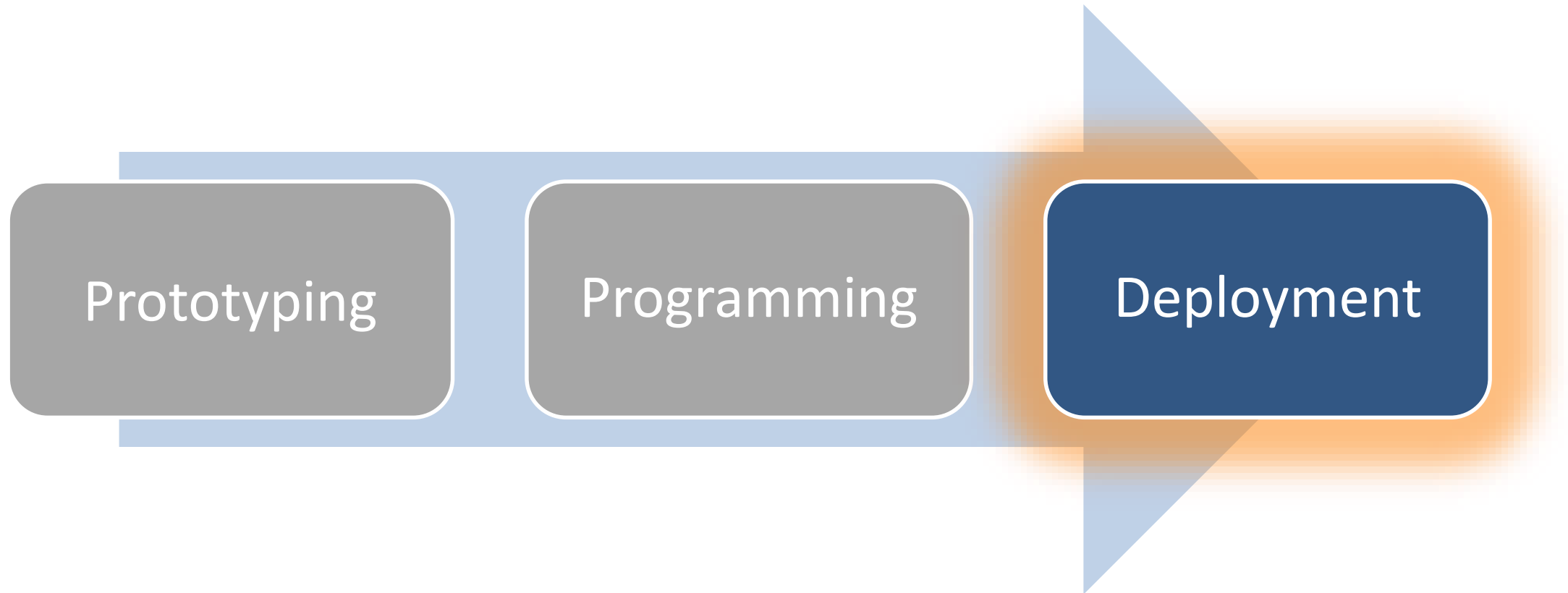


Target-Specific Implementation and Large Scale Distribution

MATLAB Application Development Landscape



MATLAB Application Development Landscape



Thank You!