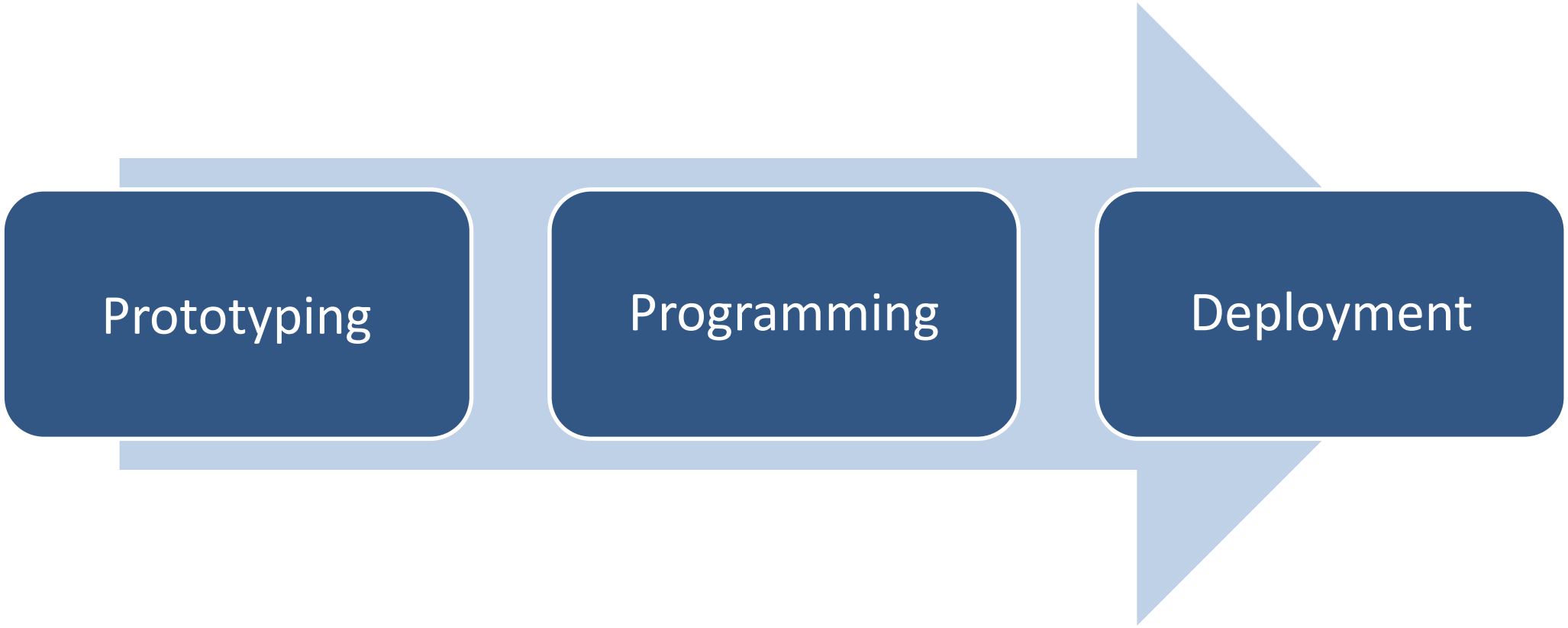
Programming with MATLAB

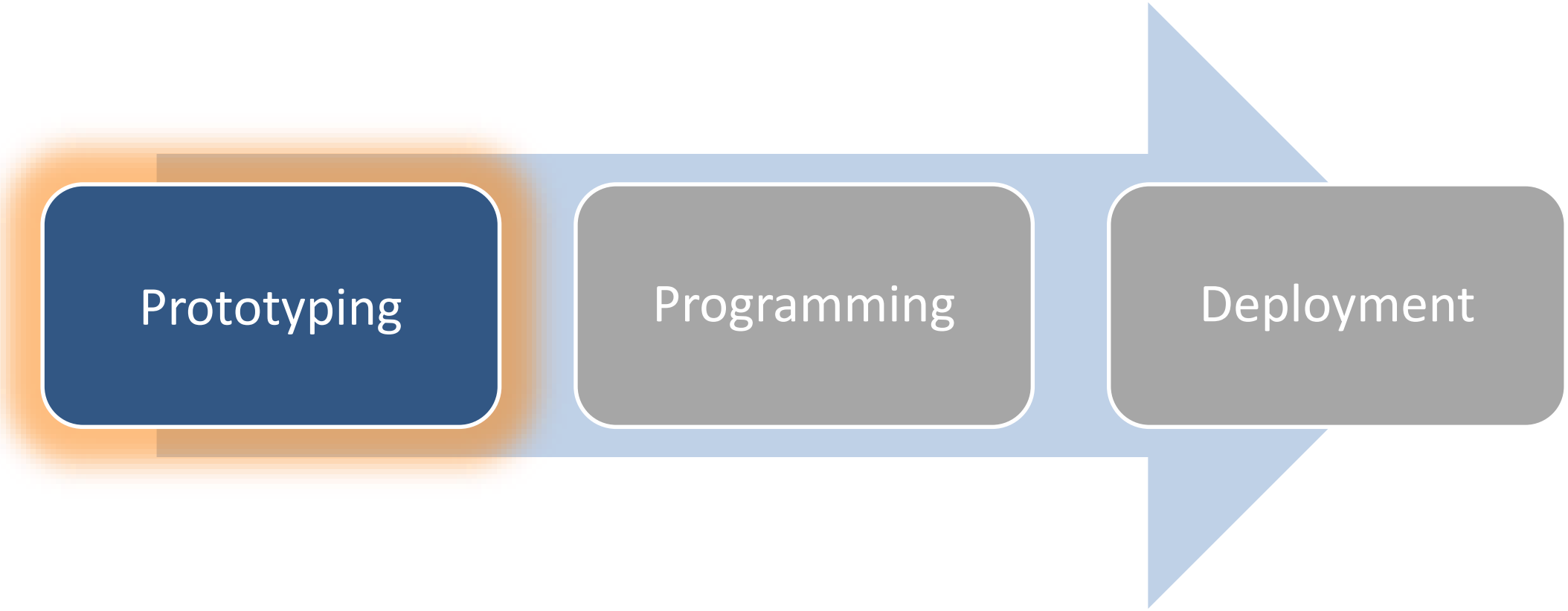
Paolo Fabbri

Senior Engineer

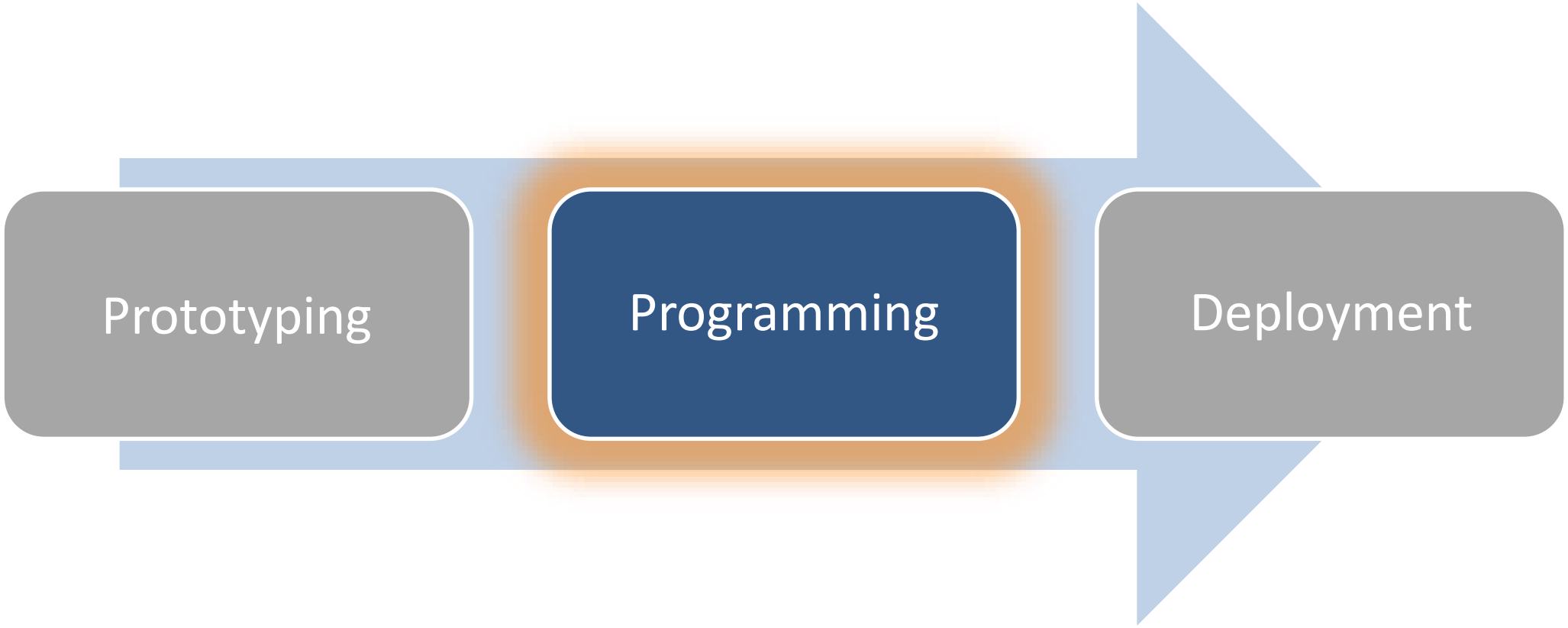
MATLAB Application Development Landscape



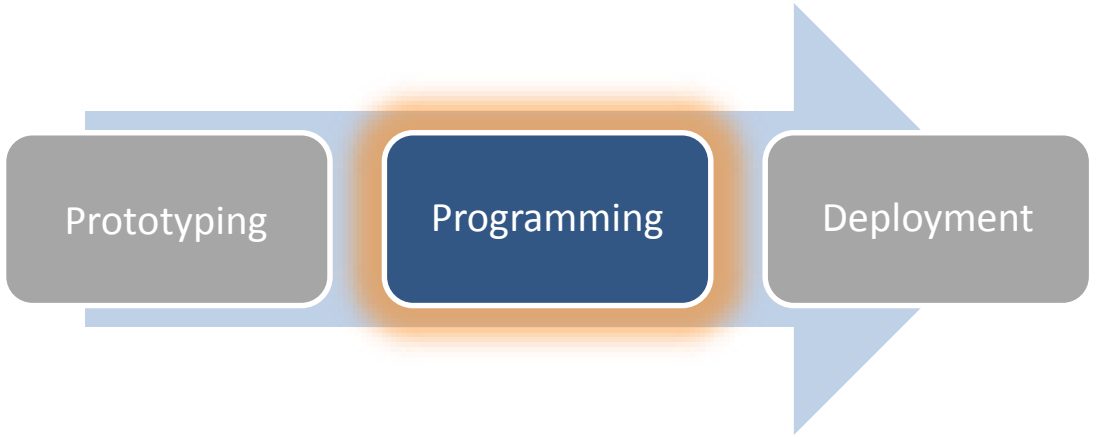
MATLAB Application Development Landscape



MATLAB Application Development Landscape



Today Focus



Object-Oriented Programming

Unit Test Framework

Source Control Integration

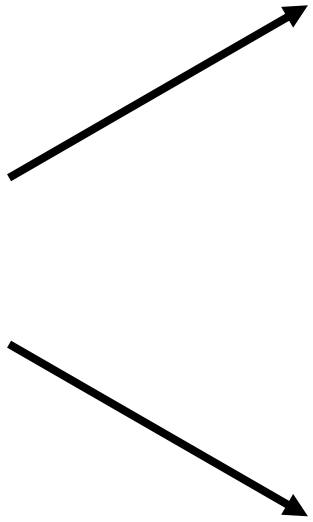
Programming Interfaces

What is a program?

What is a program?

Code

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



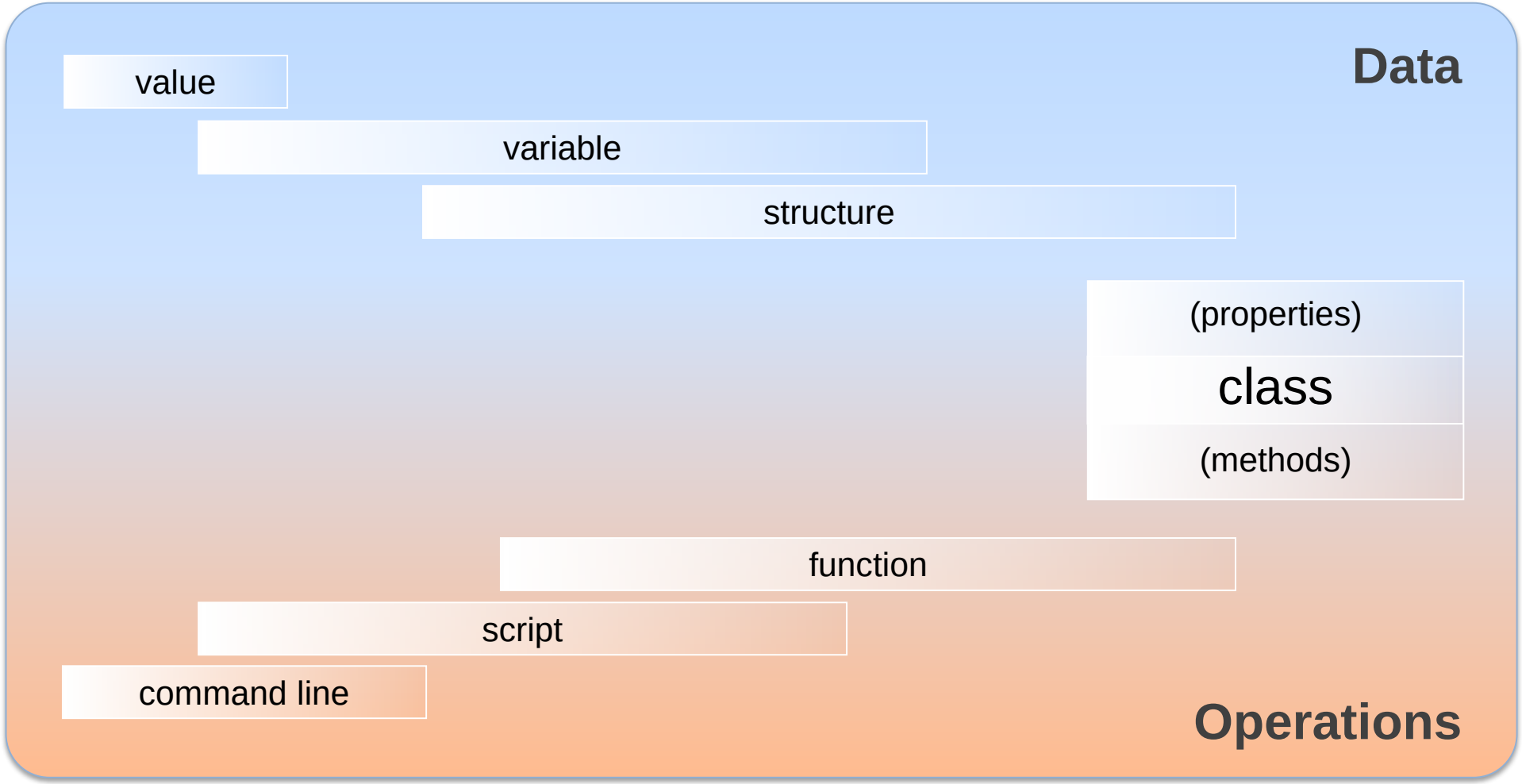
```
x = 12
while (x < 100)
    x = x+1
    if (x == 23)
        disp('Hello')
    end
end
```

Data

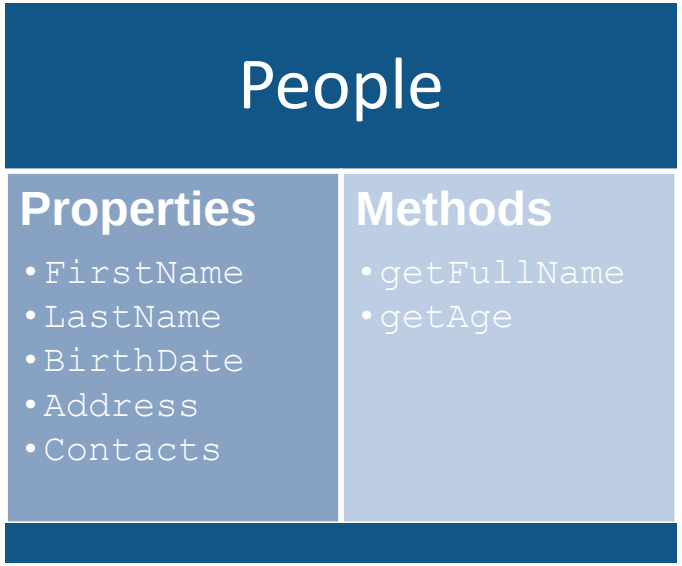
```
Assignment
Looping Test
    Increment
    Test to Act
        Take Action
    End
End
```

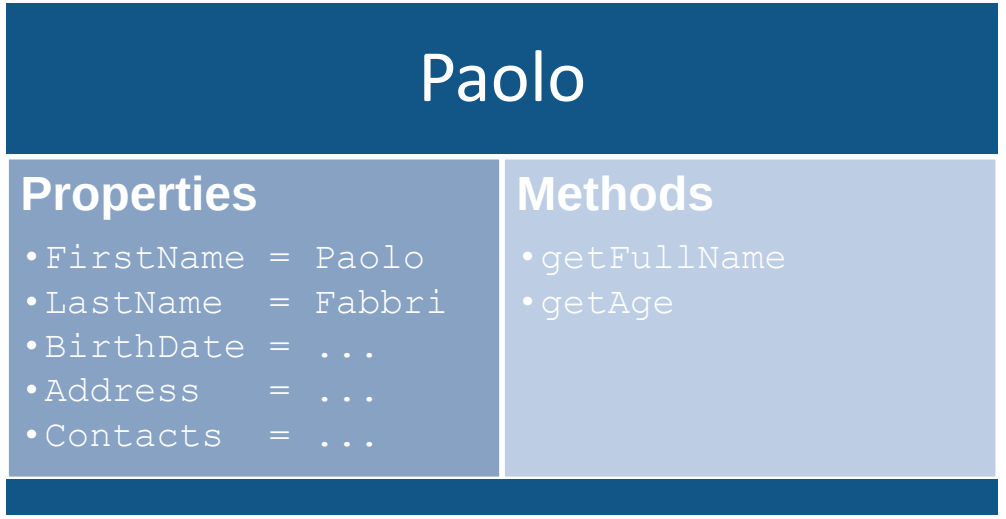
Operations

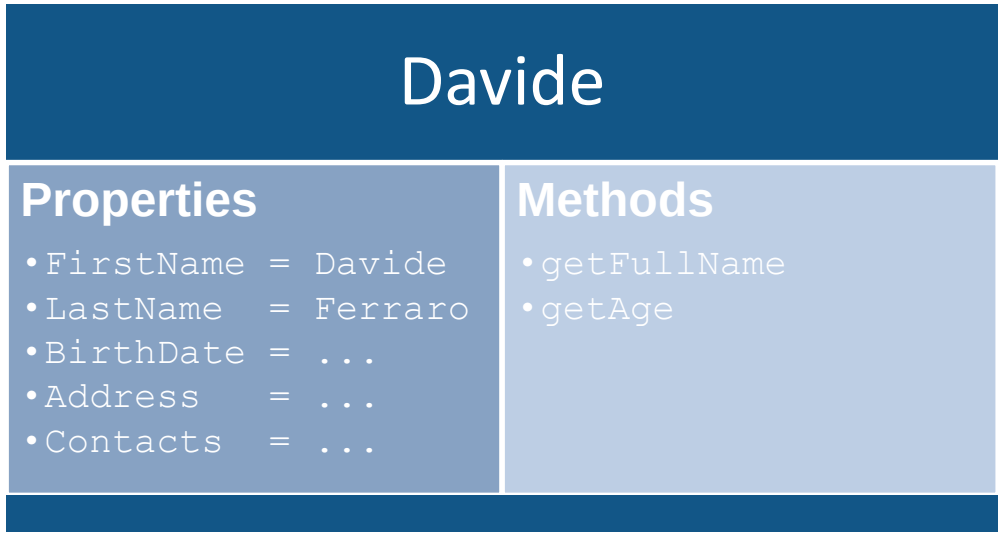
Range of Programming Techniques



Classes and Objects







Object-Oriented Programming Basics

`properties`

encapsulate object data

`methods`

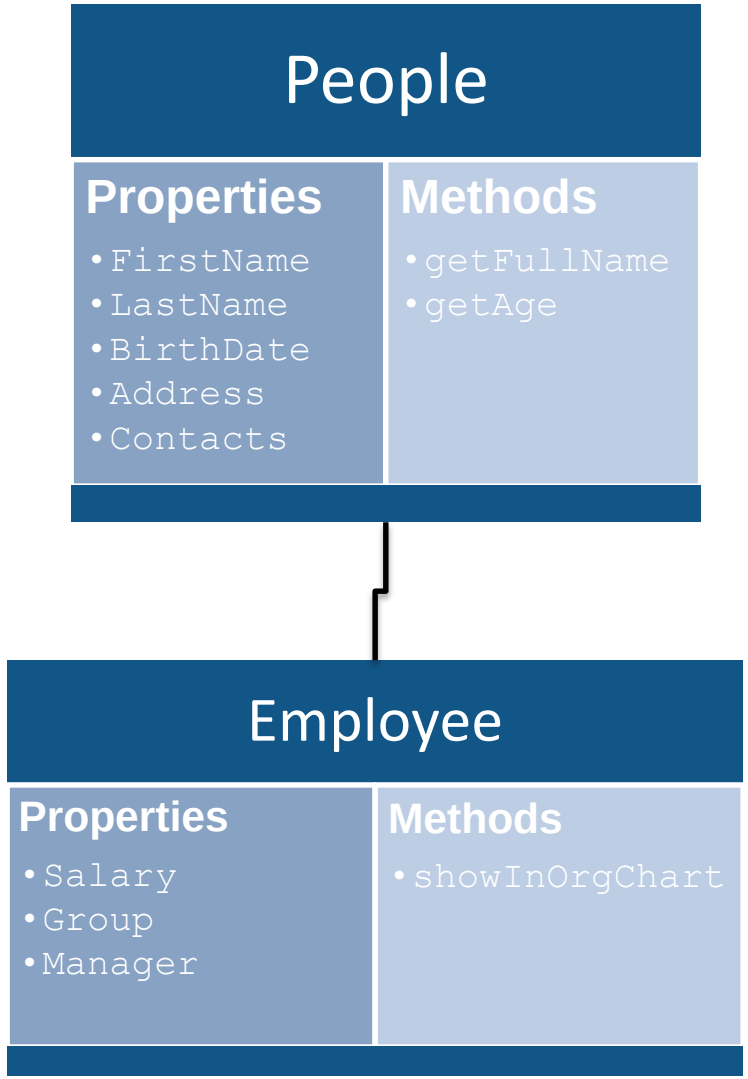
implement the object behavior

`events` and **`listeners`**

implement objects communication

`inerithance`

allows composition and reusability



Object-Oriented Programming with MATLAB

properties

encapsulate object data

methods

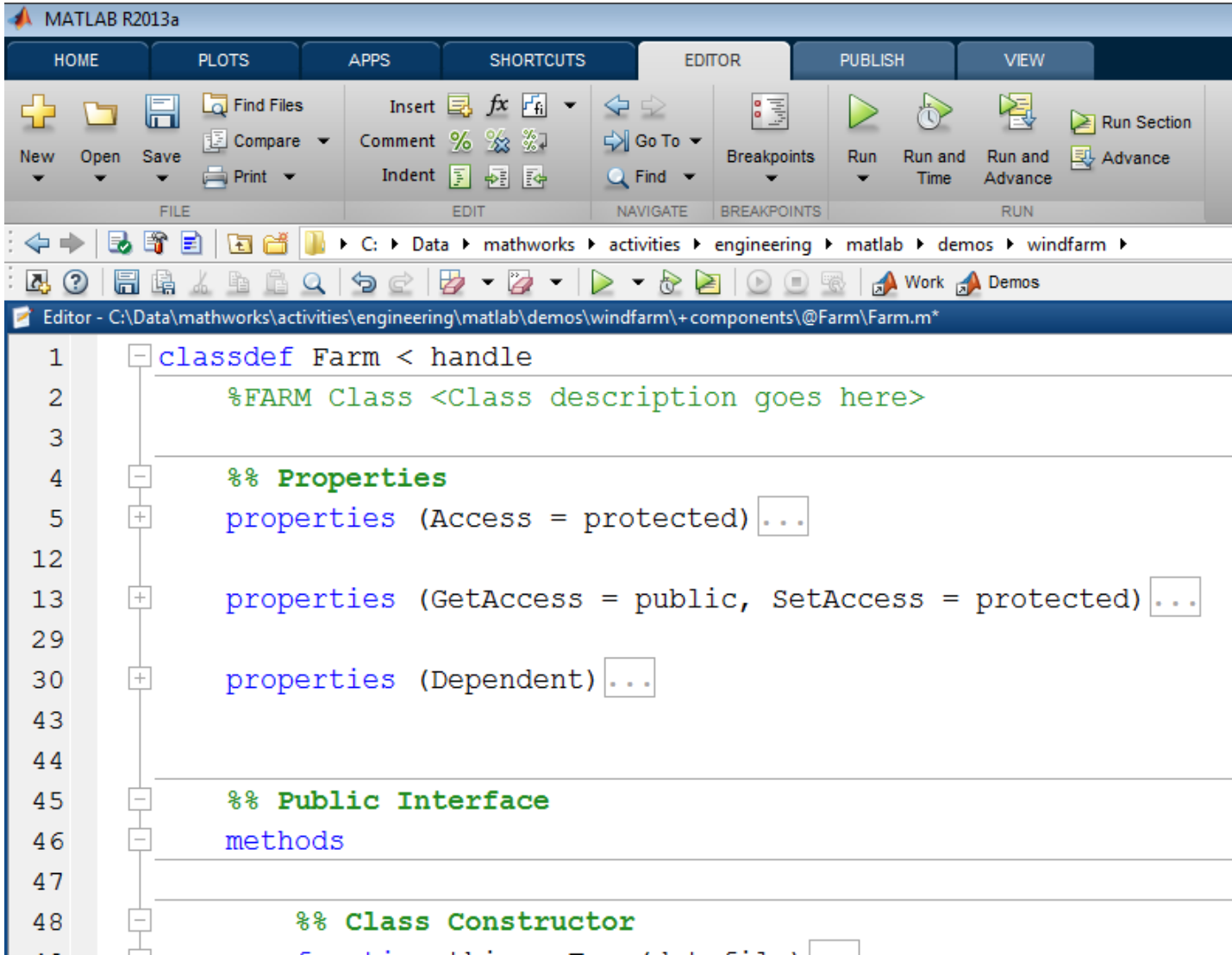
implement the object behavior

events and **listeners**

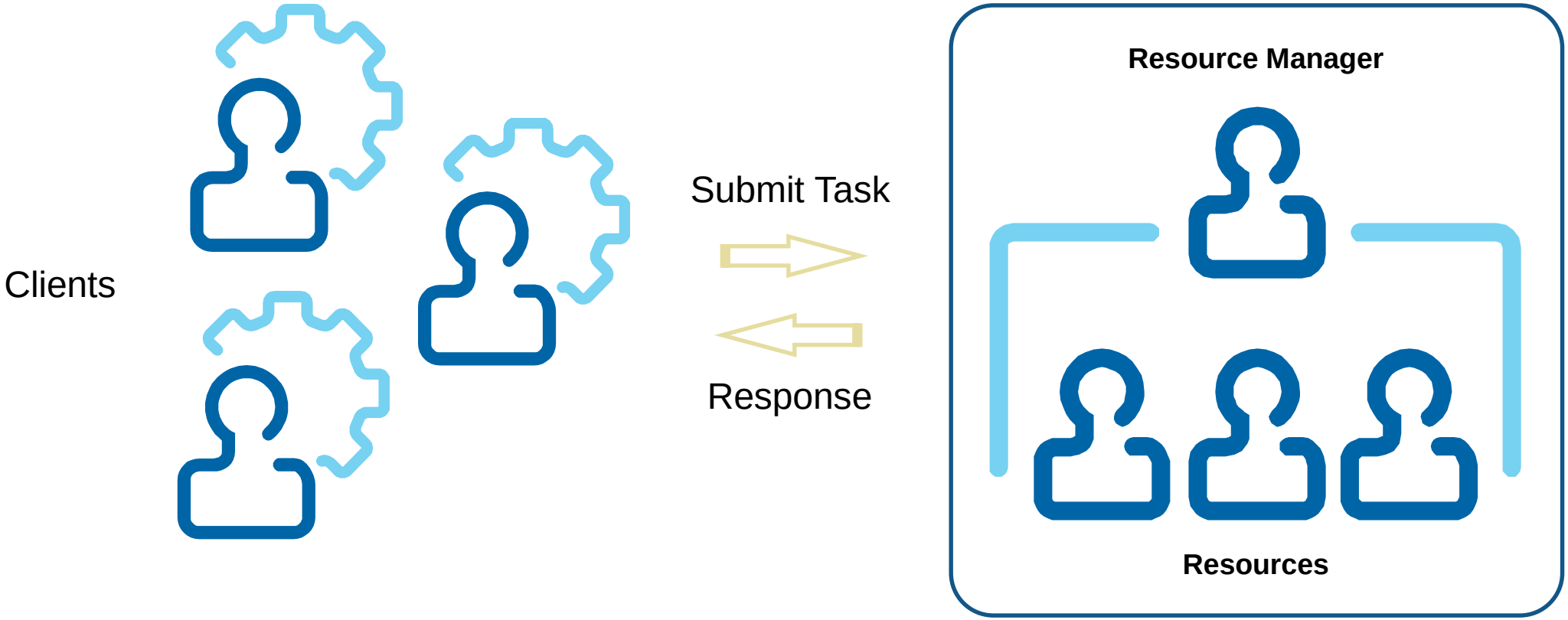
implement objects communication

inerithance

allows composition and reusability

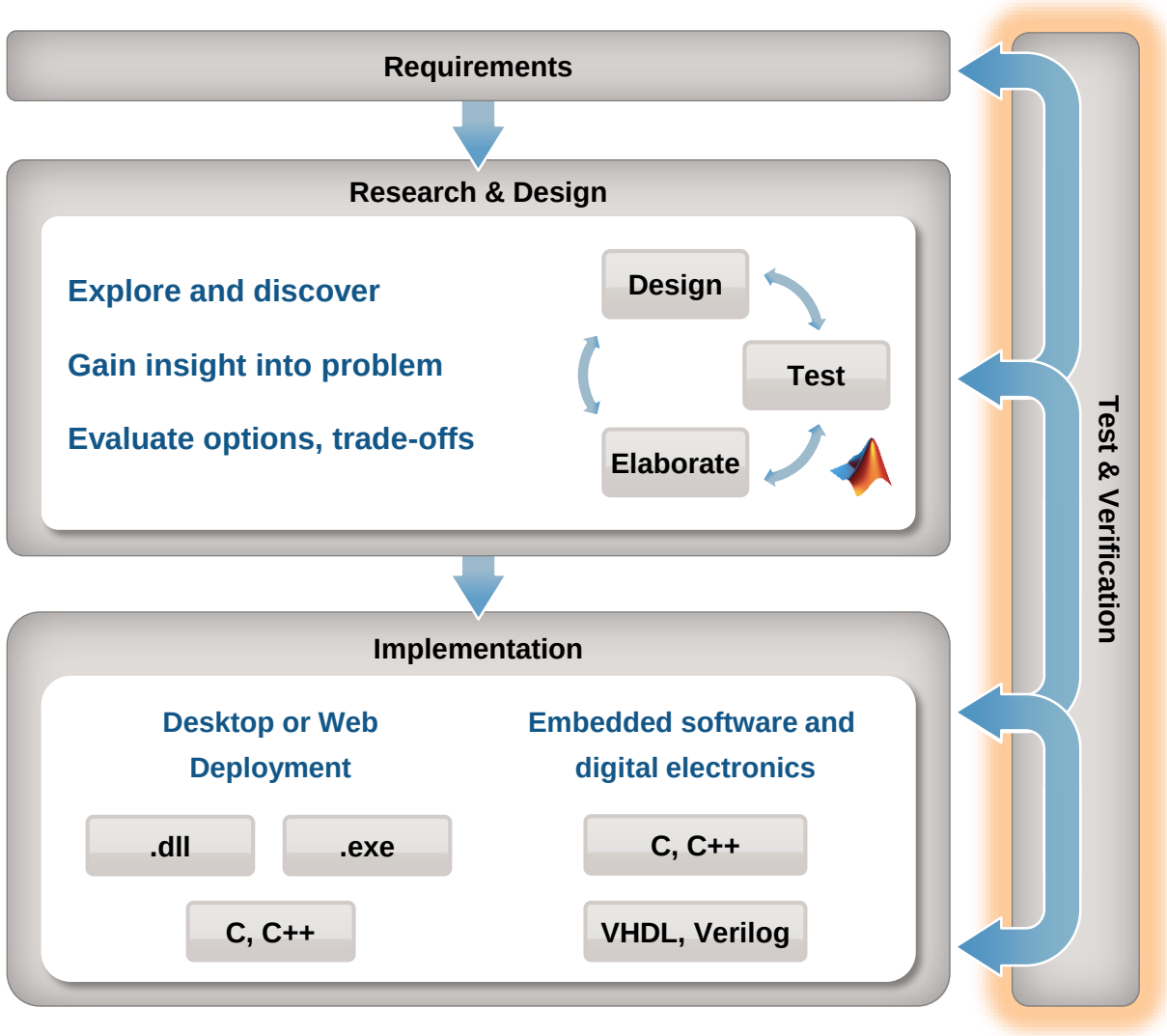


Case Study: Team Resources Assignment



MATLAB Unit Test Framework

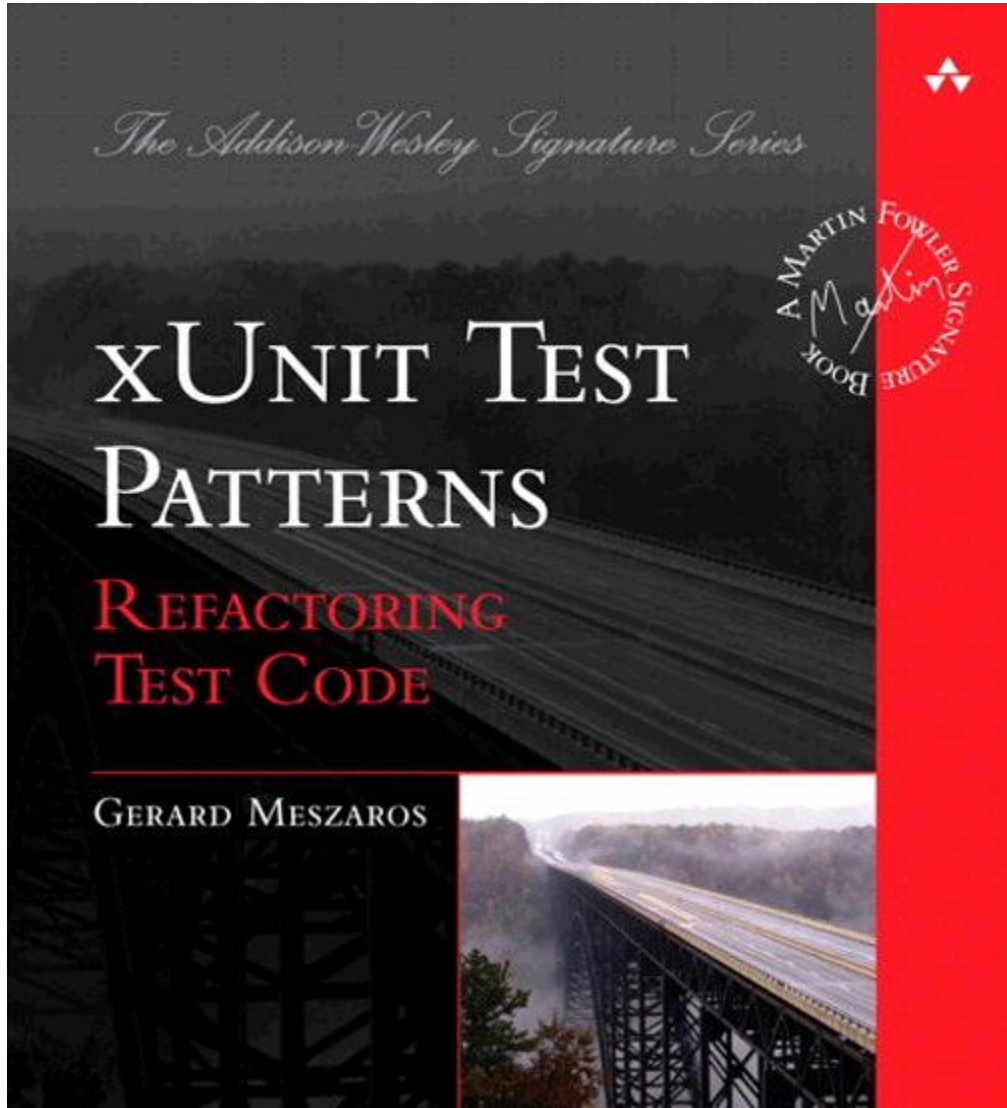
Application Development Process



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

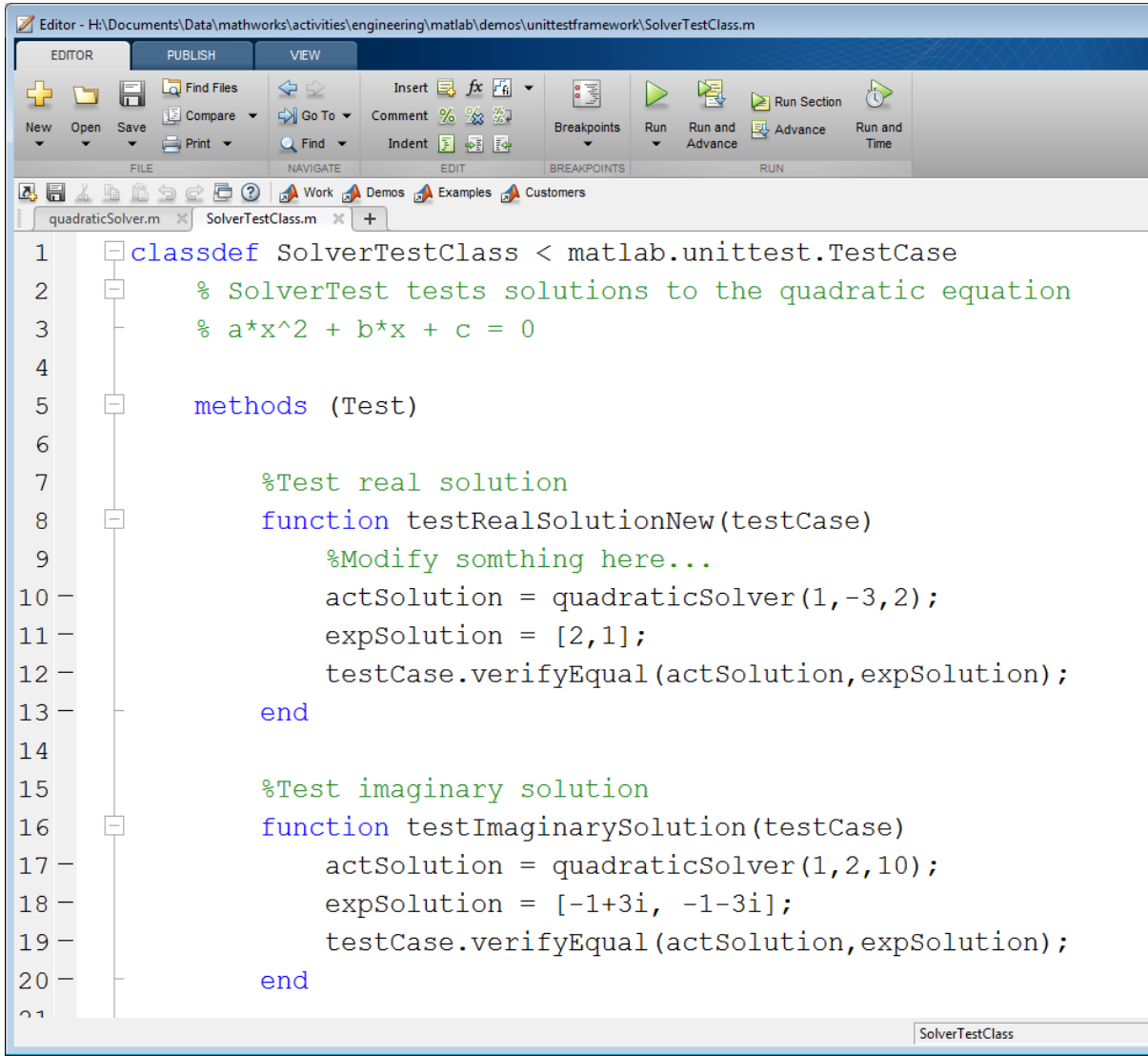
Write in function form or **xUnit-style** class form

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



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
		sumeTrue	assertTrue	fatalAssertTrue
		sumeFalse	assertFalse	fatalAssertFalse
Value is equal to specified value.	verifyEqual	assumeEqual	assertEqual	fatalAssertEqual
Value is not equal to specified value.	verifyNotEqual	assumeNotEqual	assertNotEqual	fatalAssertNotEqual
Two values are handles to same instance.	verifySameHandle	assumeSameHandle	assertSameHandle	fatalAssertSameHandle
Value is not handle to specified instance.	verifyNotSameHandle	assumeNotSameHandle	assertNotSameHandle	fatalAssertNotSameHandle
Function returns true when	verifyReturnsTrue	assumeReturnsTrue	assertReturnsTrue	fatalAssertReturnsTrue

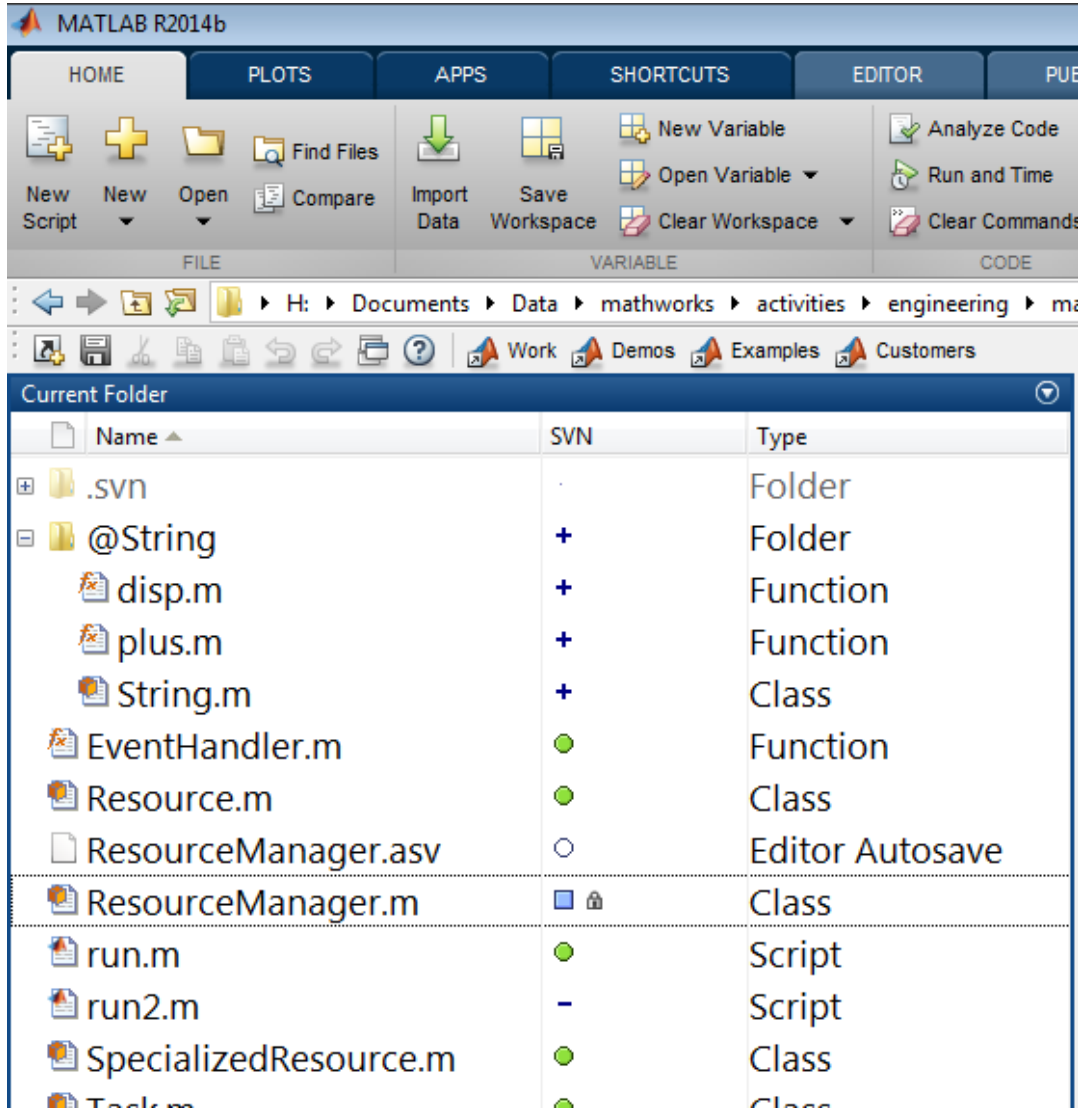
Source Control Integration

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

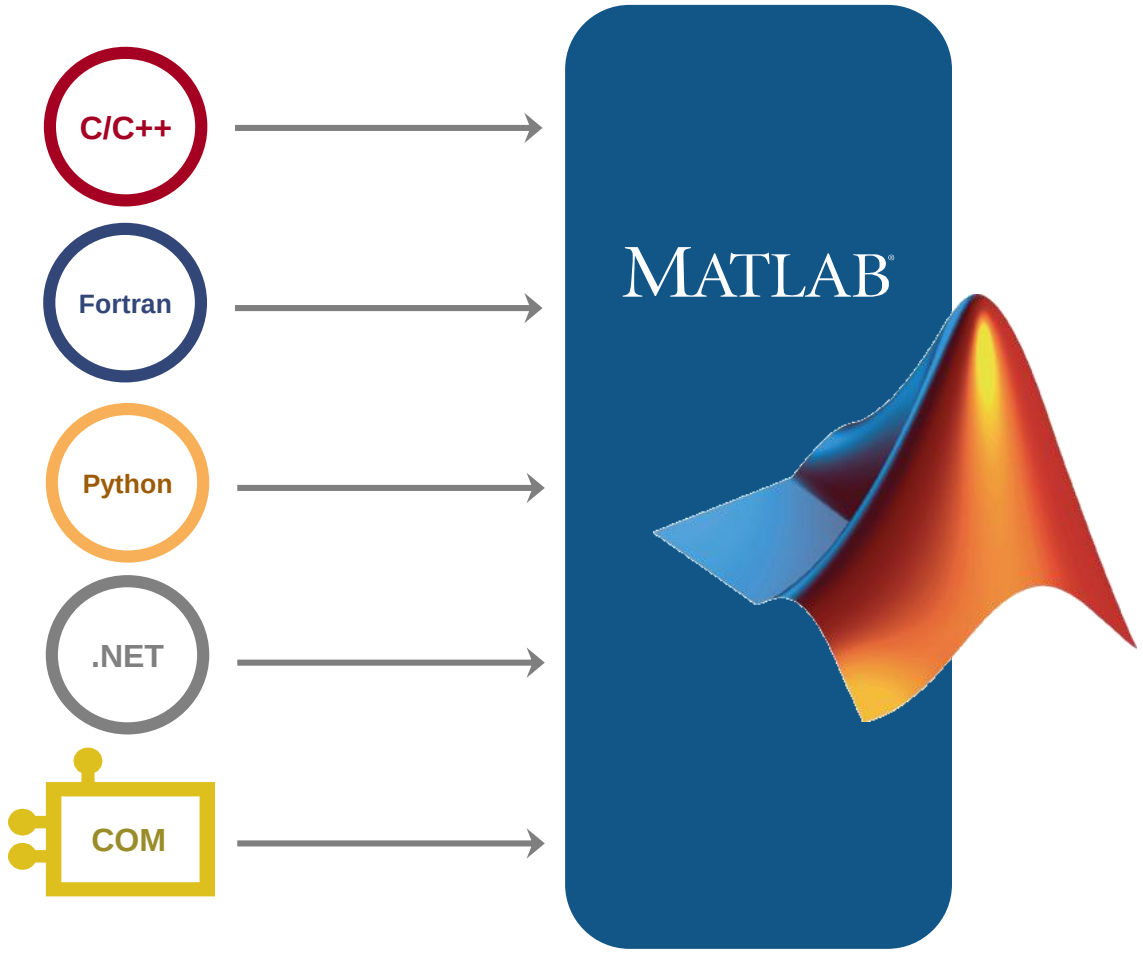
Calling MATLAB from Other Languages

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

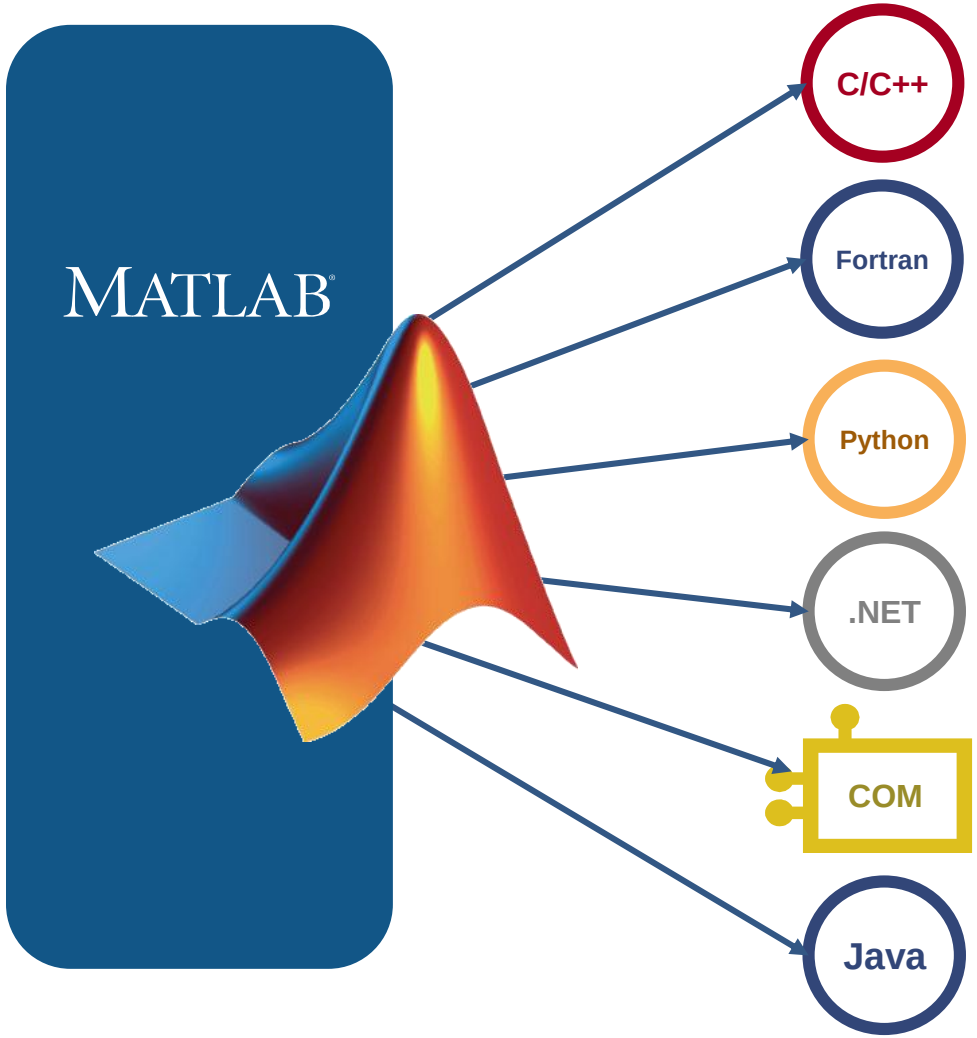
```
>> NET.addAssembly('System.Speech')
```

MATLAB Java interface

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

MATLAB Python interface R2014b

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

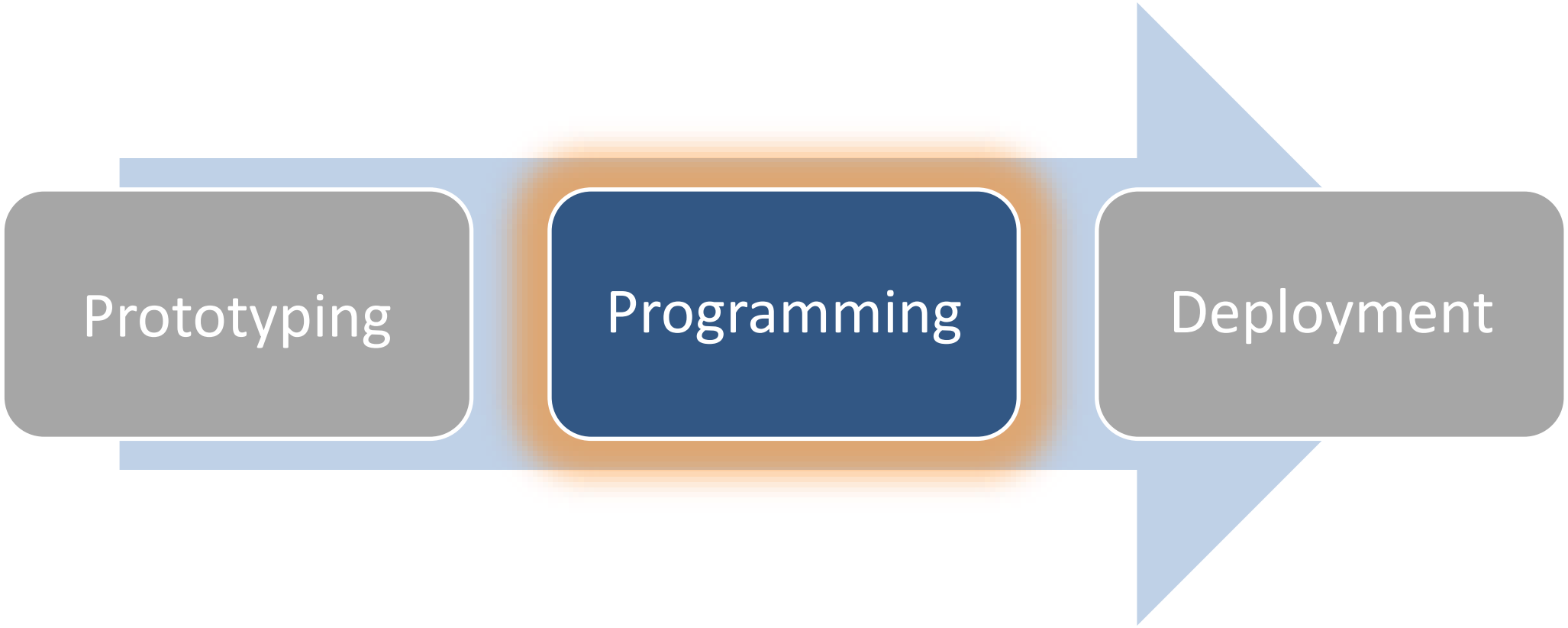


Target-Specific Implementation

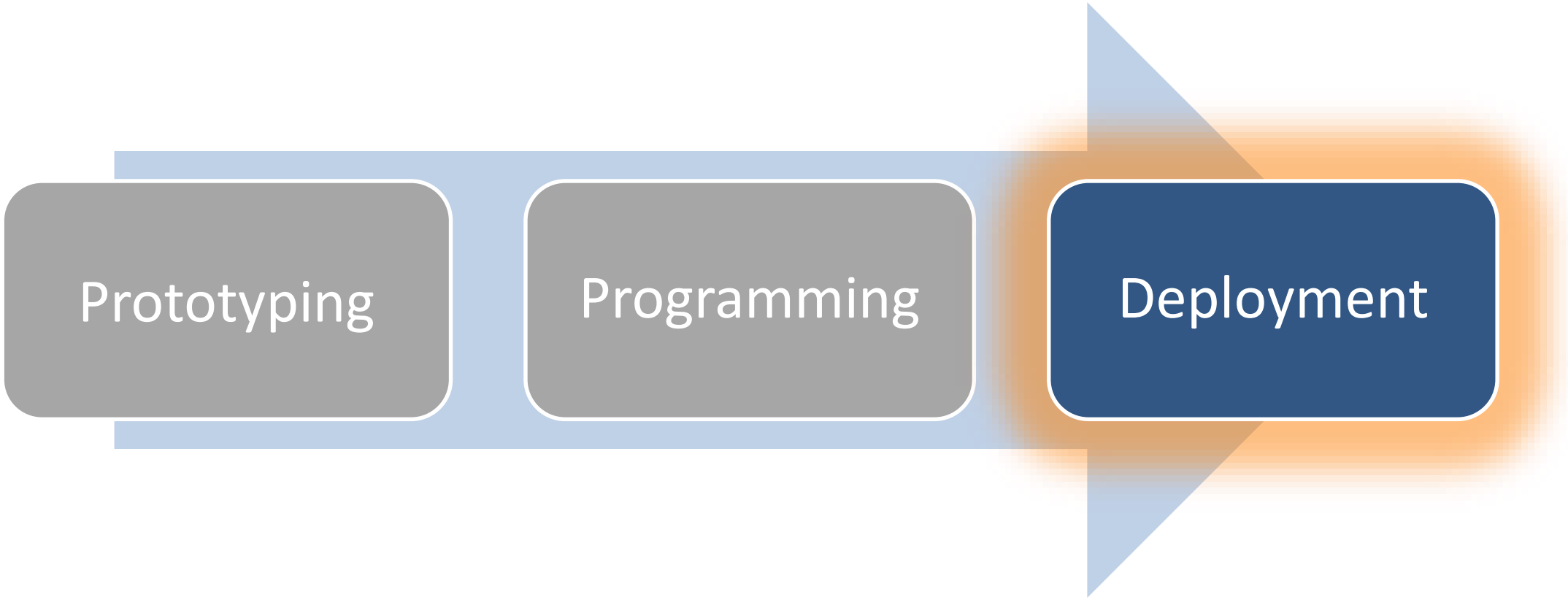
and

Large Scale Distribution

MATLAB Application Development Landscape



MATLAB Application Development Landscape



Thank You!