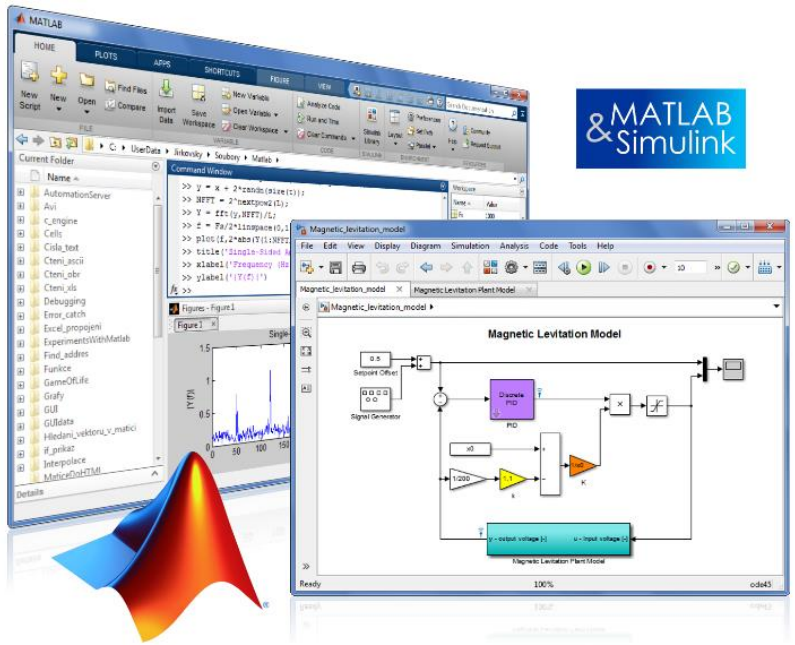
HUMUSOFT®

Master Class

Application Development Workflow

From Algorithm to Production Systems



Michal Blaho

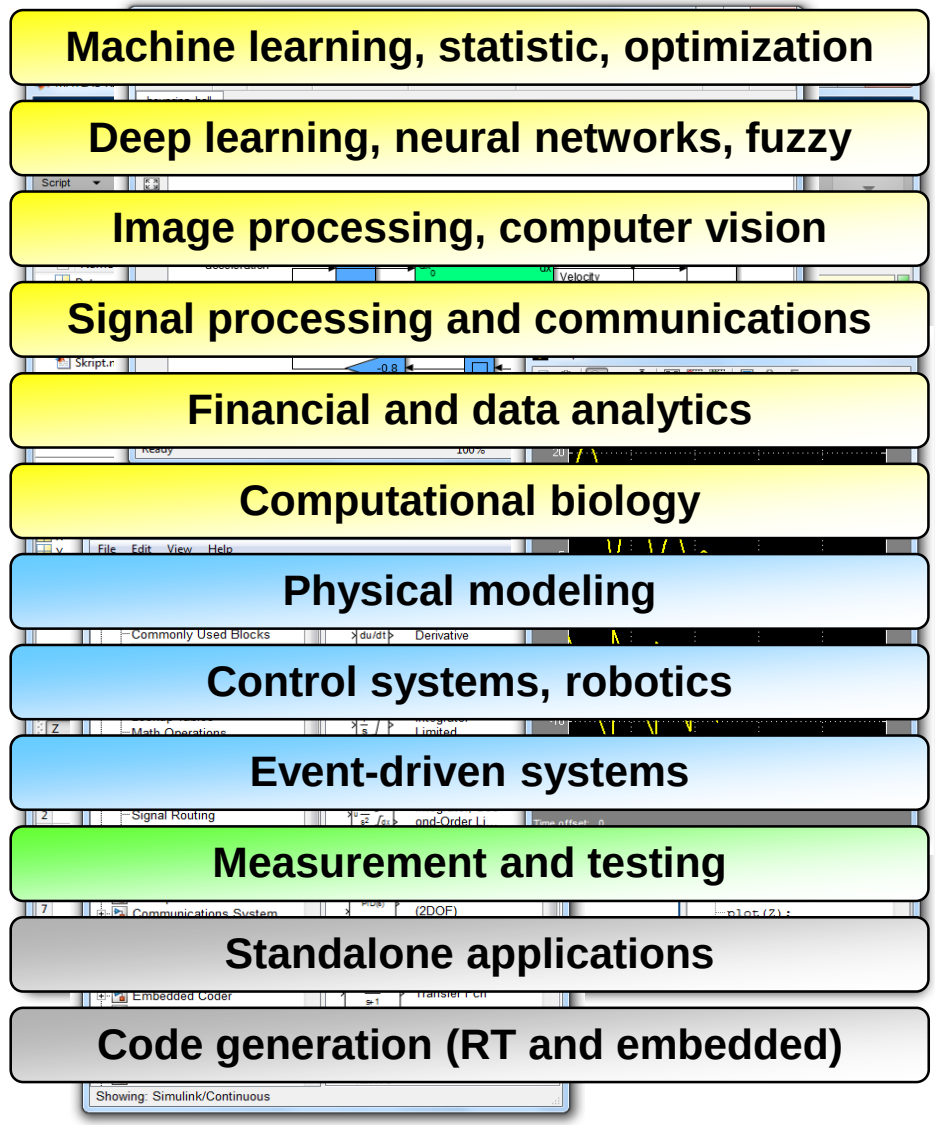
blaho@humusoft.sk

www.humusoft.cz
info@humusoft.cz

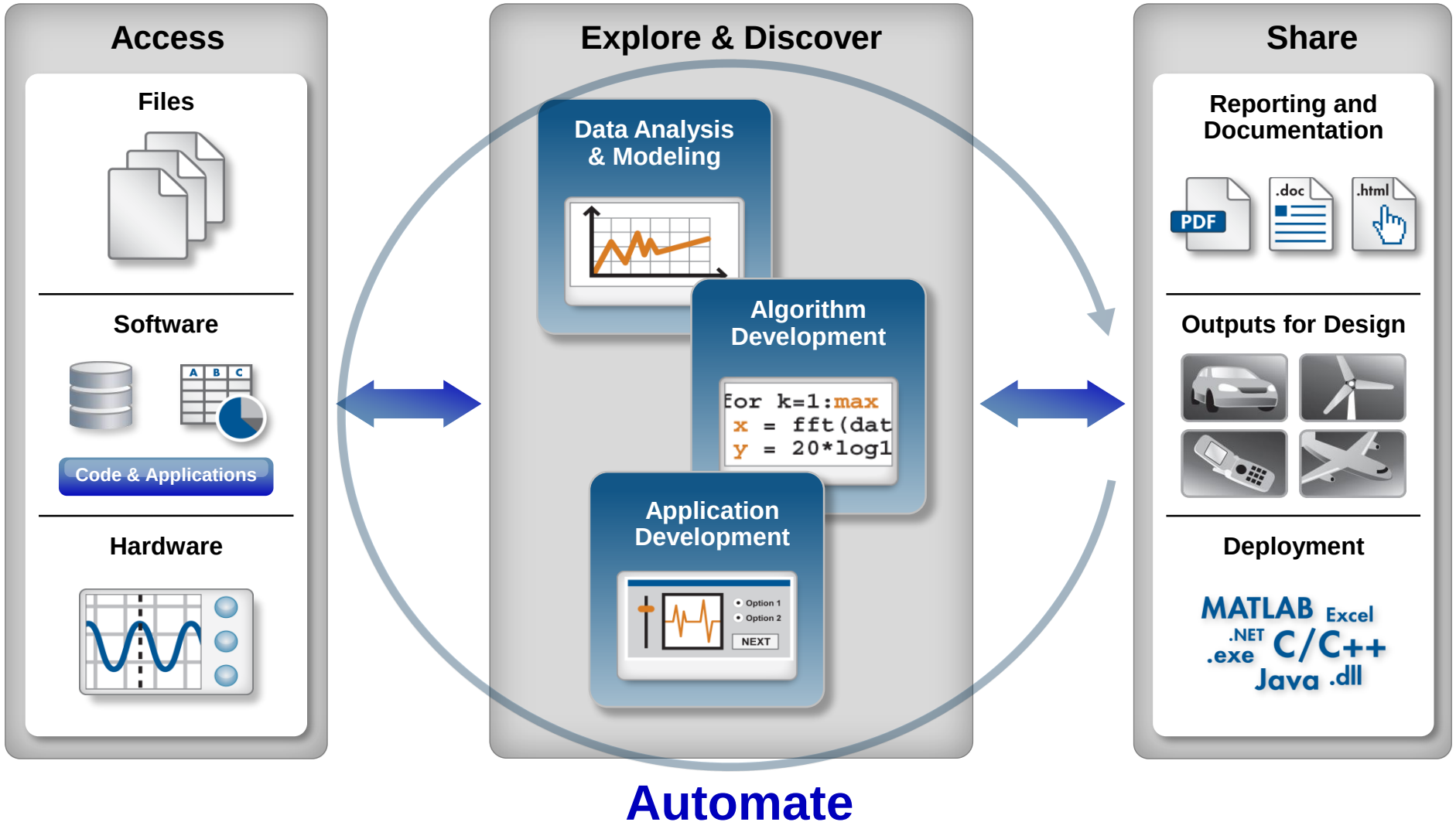
www.mathworks.com

MATLAB and Simulink

- **MATLAB**
  - Engineering tool and interactive environment for scientific and technical computations
  - built-in functions – calculations, graphics
  - Graphical user interface (GUI, APPS)
  - Open system
- **Simulink**
  - MATLAB extension
  - Model, simulate and analyze your dynamic systems
  - Block diagrams
  - Model Based Design platform
- **Toolboxes**

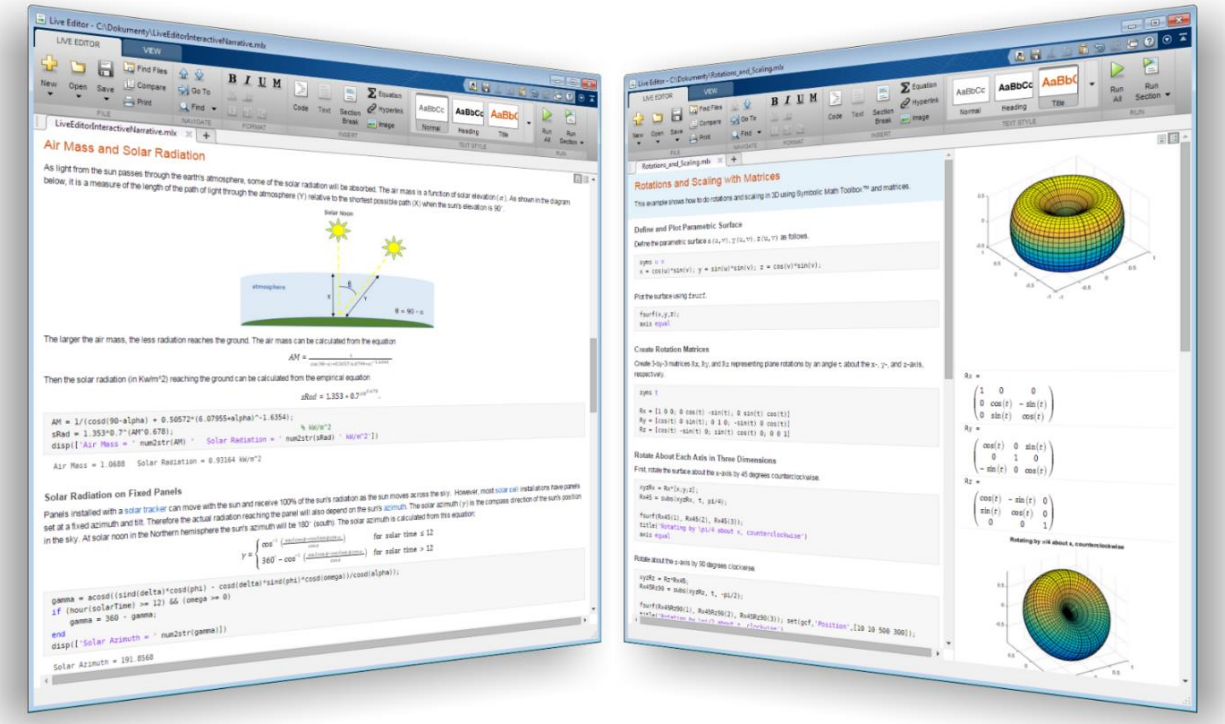


Data Analysis Workflow



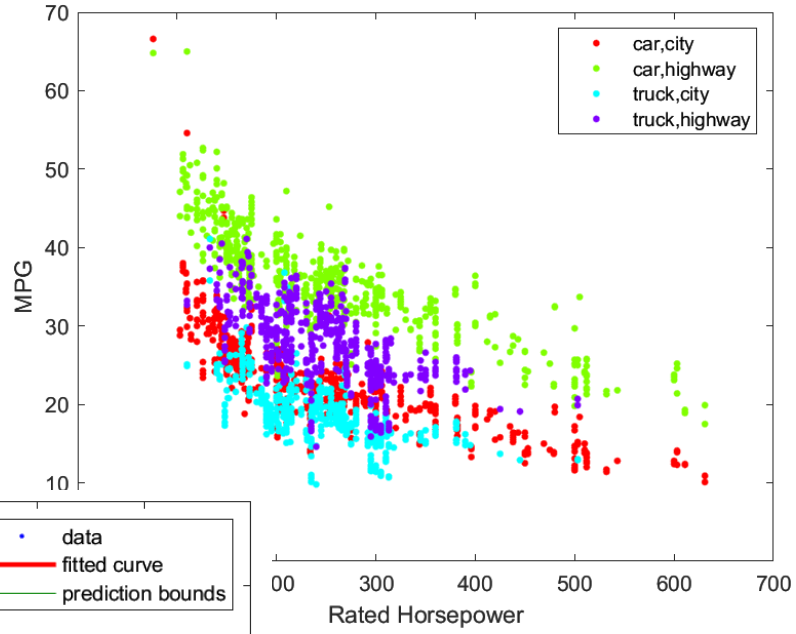
MATLAB Live Editor

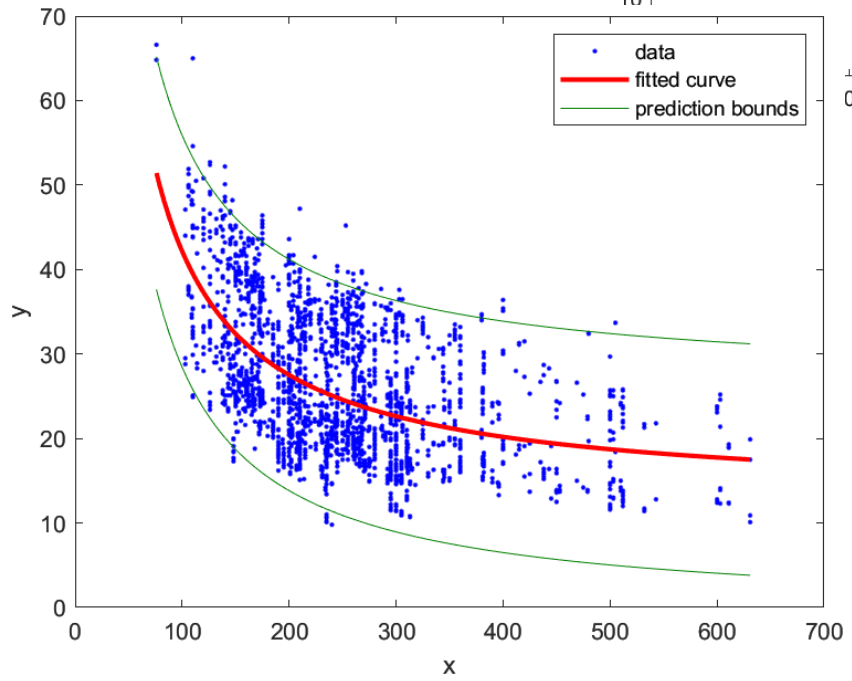
- **Scripts that combine code and outputs in an executable notebook**
  - Code, outputs
  - Title, headings, formatted text
  - Equations, images, hyperlinks
  - Publish as HTML, PDF, Latex, Word
- **Symbolic math**
  - Formatted equations
- **Interactive**
  - Lectures
  - Presentations
- **Live editor webinar**
  - http://www.humusoft.cz/events/www-seminars/



Demo – Fuel Economy Analysis

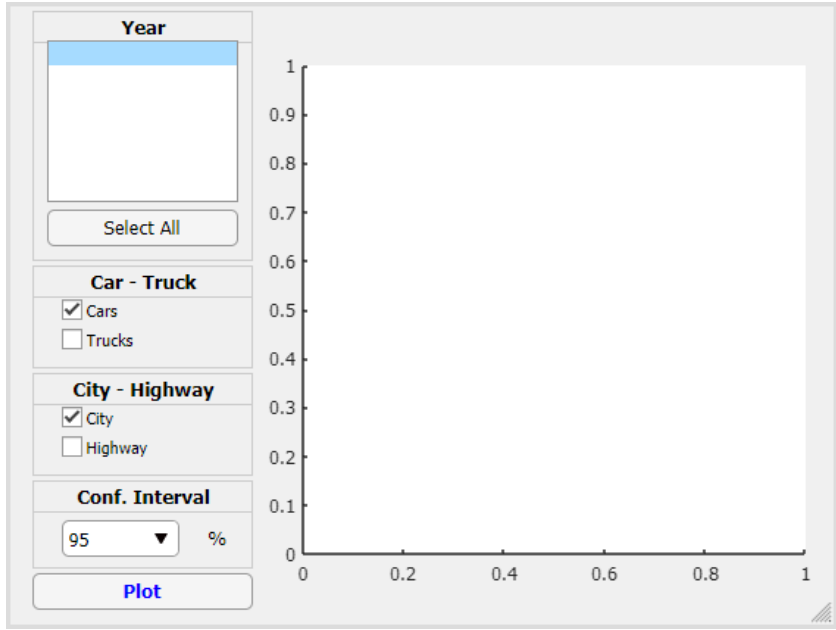
- **Historical fuel economy data**
- **Various cars**
  - **built from year 2000 up to 2012**
- **Data import**
- **Visualization**
  - **All cars**
  - **Grouped visualizations**
- **Curve fitting**
- **Model visualization**
  - **All cars**
  - **Grouped Visualizations**





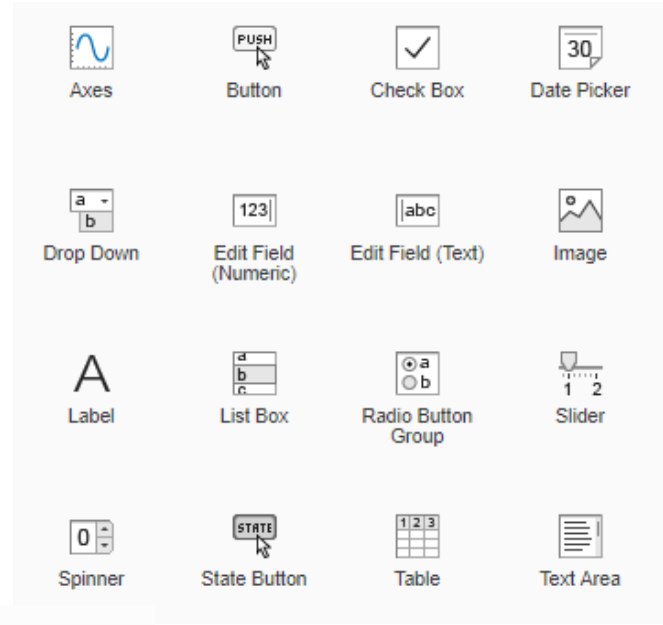
Graphical user interfaces in MATLAB

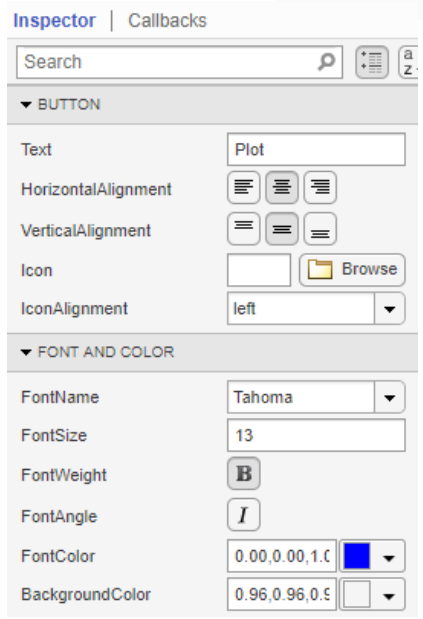
- Point-and-click control of analysis with GUI front end
- Algorithm "runs in background"
- Creating a MATLAB App
  - App Designer
  - GUIDE
  - Programmatically
- From GUIDE to App Designer
  - GUIDE to App Designer Migration Tool for MATLAB
- Difference between GUIDE and App Designer
  - Comparing GUIDE and App Designer
- App Designer recommended for new applications



GUI building – Design view

- Canvas
- Component library
  - button, slider, check box
  - Edit field (text), text area, list box, ...
- Menu and toolbars
- Predefined dialog boxes – file, color
- Grid layout
- Customize components
  - size
  - color
  - default values
- Keyboard Shortcuts





Object-Oriented Programming

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

Code

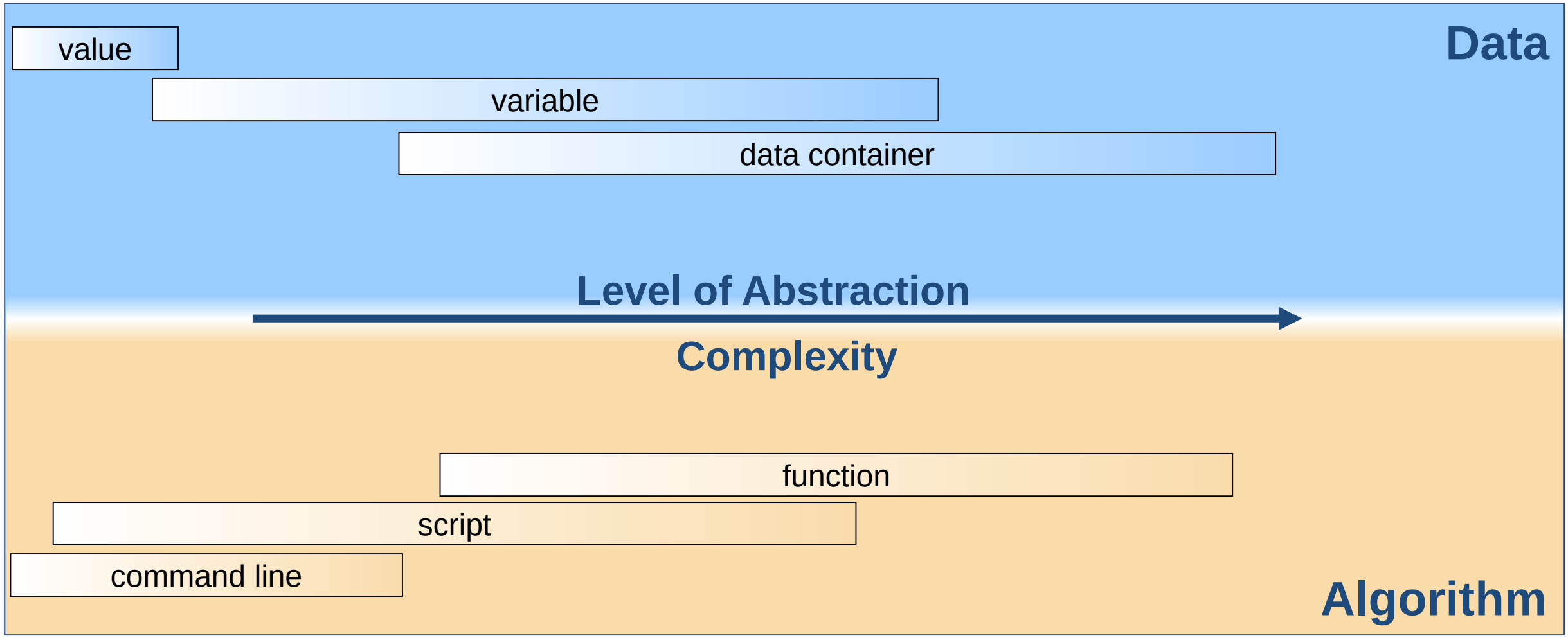
Data

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

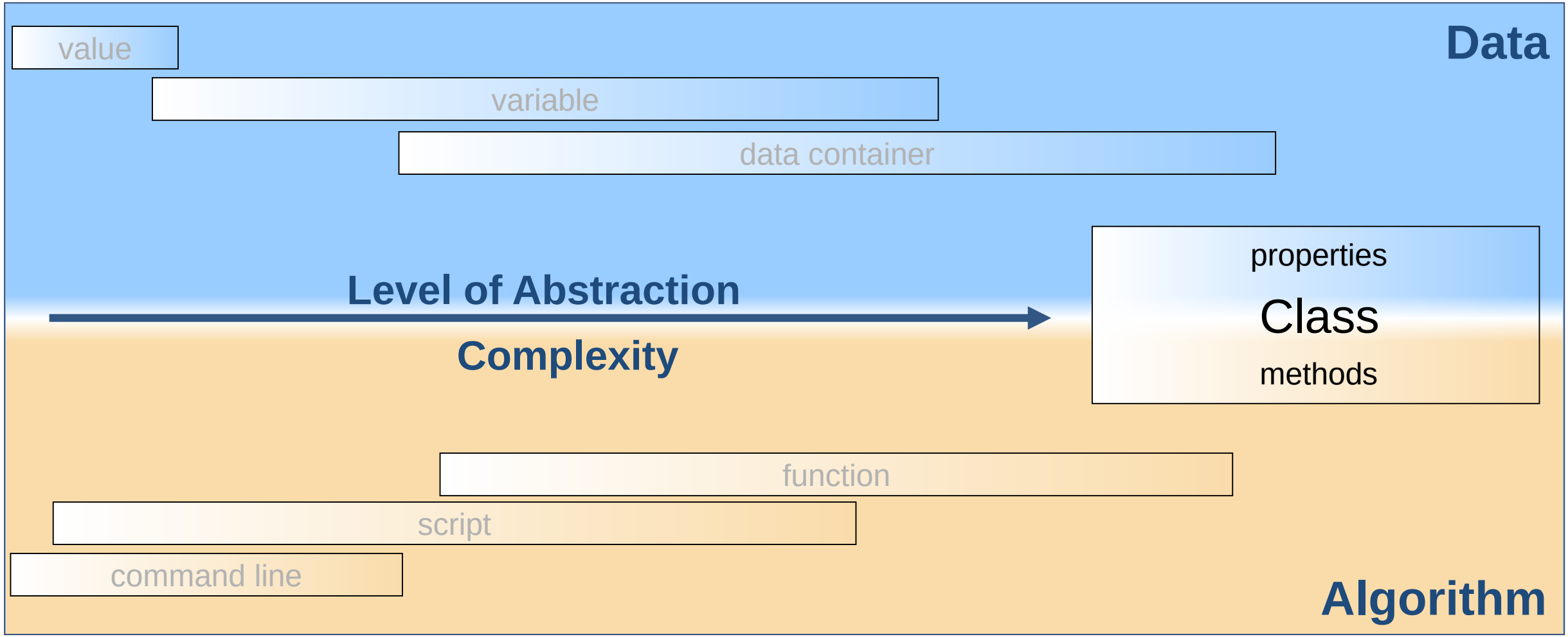
```
Assignment
Loop condition
  Increment
Action condition
  Action
End
End
```

Algorithm

Object-Oriented Programming



Object-Oriented Programming



Object-Oriented Programming

- **Class**
  - Object group with similar properties
  - Defines behavior of objects
  - filename *class_name.m*
- **Object**
  - instance of class
  - specific values for properties
  - *object = class_name*
- **Class includes:**
  - Data ... *properties*
  - Algorithm... *methods*
- **Access control – attributes**

```
classdef class_name
    properties
        variables;
    end
    methods
        function f1
            commands;
        end
        function f2
            commands;
        end
    end
end
```

GUI building – Code view

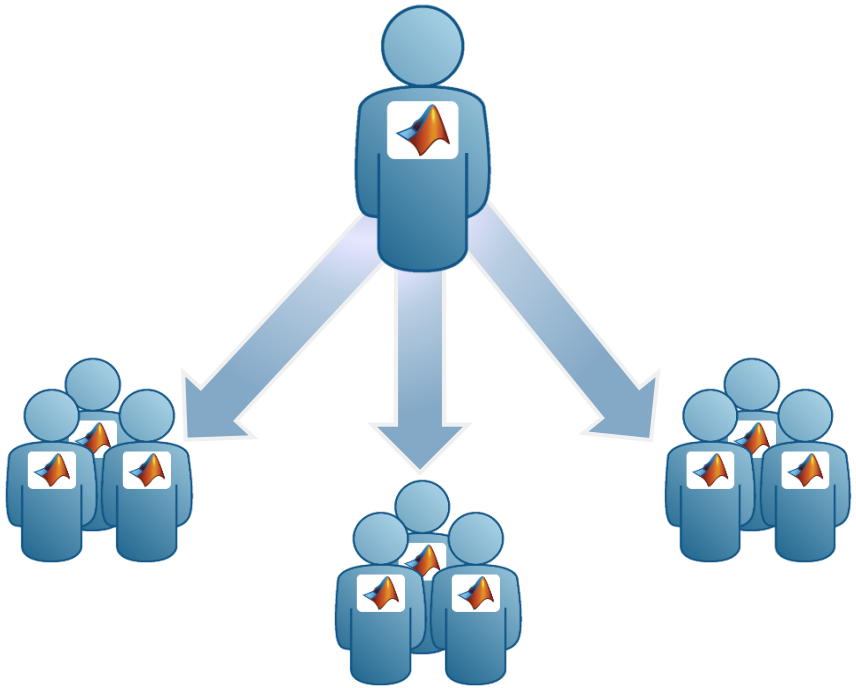
- App programming
- Code View
  - OOP – App is class
  - Graphical components – properties
  - Callbacks – methods
  - Other functions
- Code Browser
  - Properties and methods list
- Data sharing
  - Properties
- Special methods
  - Startup, CloseRequest

Design View | Code View

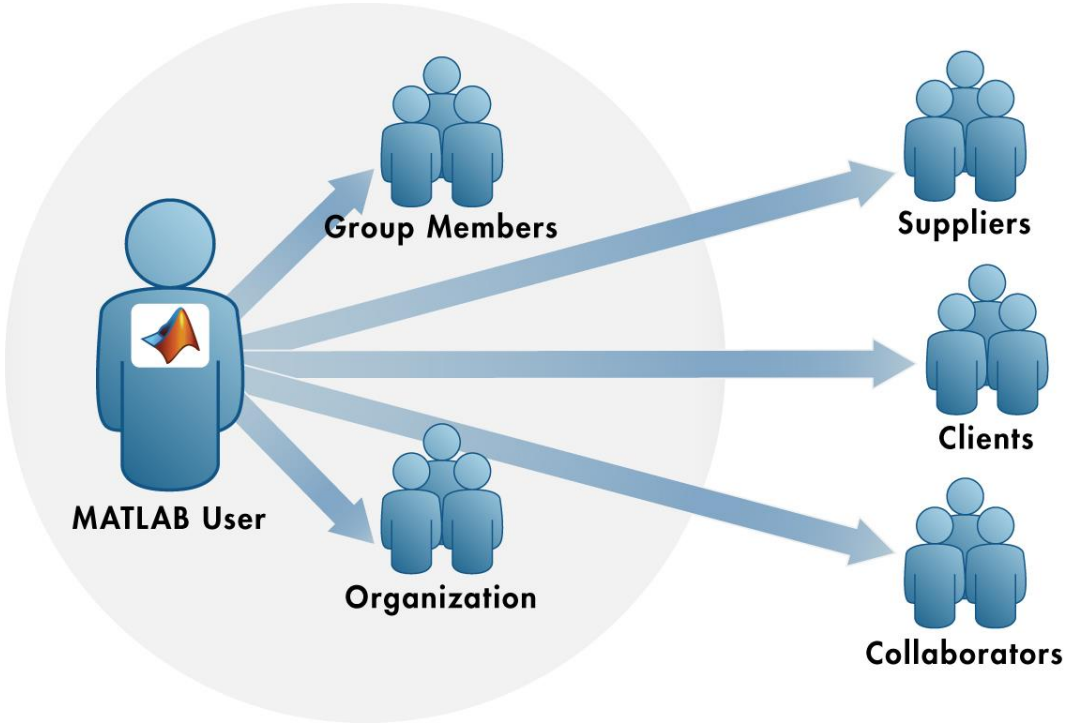
```
properties (Access = private)
    dataDir;
    selectedYears = [];
    lastData;
end

methods (Access = private)
    function updateYearList(app)

        files = dir(fullfile(app.dataDir, '*.xlsx'));
        if isempty(files)
            years = {};
        else
            years = {files.name};
            years = cellfun(@(x) strtok(x, 'd'), years, 'UniformOutput'
        end
        app.yearListBox.Items = years;
        app.yearListBox.Value = years{1};

        app.selectedYears = [];
    end

    function [cf, gof] = MPG(app, carData, CarTruck, CityHighway, CI)
            % MPG(carData, CARTRUCK, CITYHIGHWAY, CI)
            %
            %   carData      : car data (table object)
            %   CARTRUCK     : 'car' or 'truck' or 'all'
            %   CITYHIGHWAY  : 'city' or 'highway' or 'all'
```

MATLAB Programs Can be Shared With Anyone

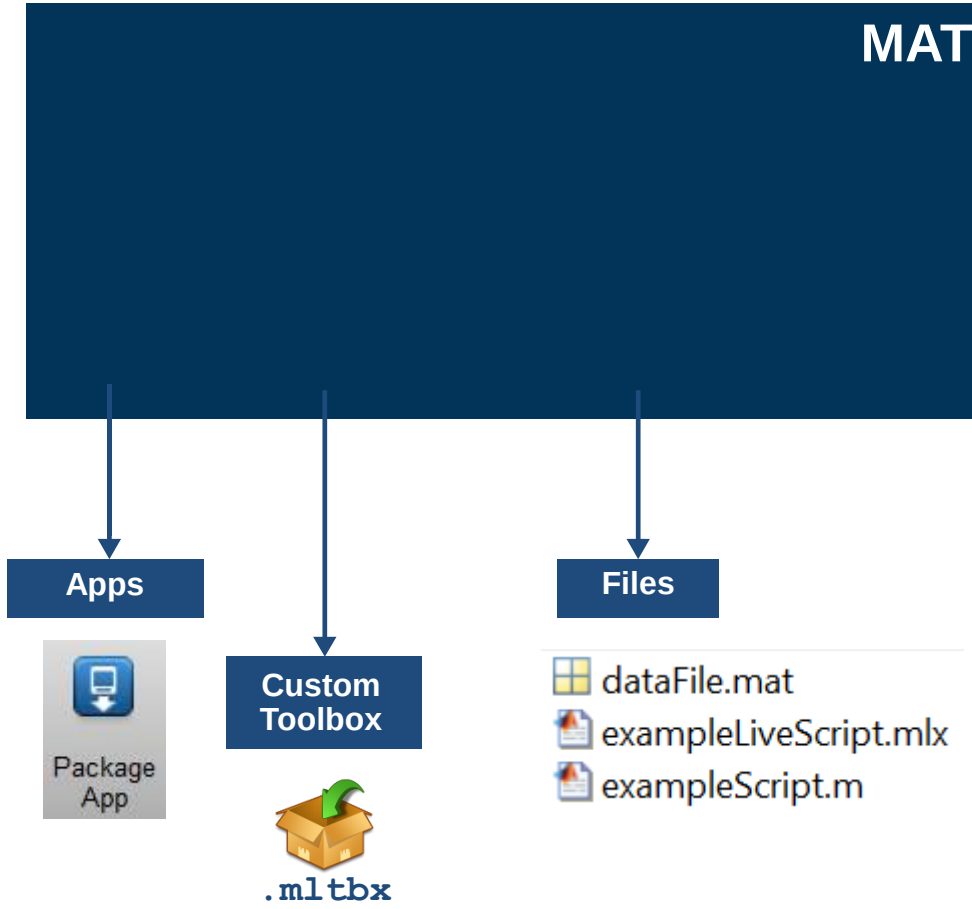
Share With Other MATLAB Users



Share With People Who do Not Have MATLAB



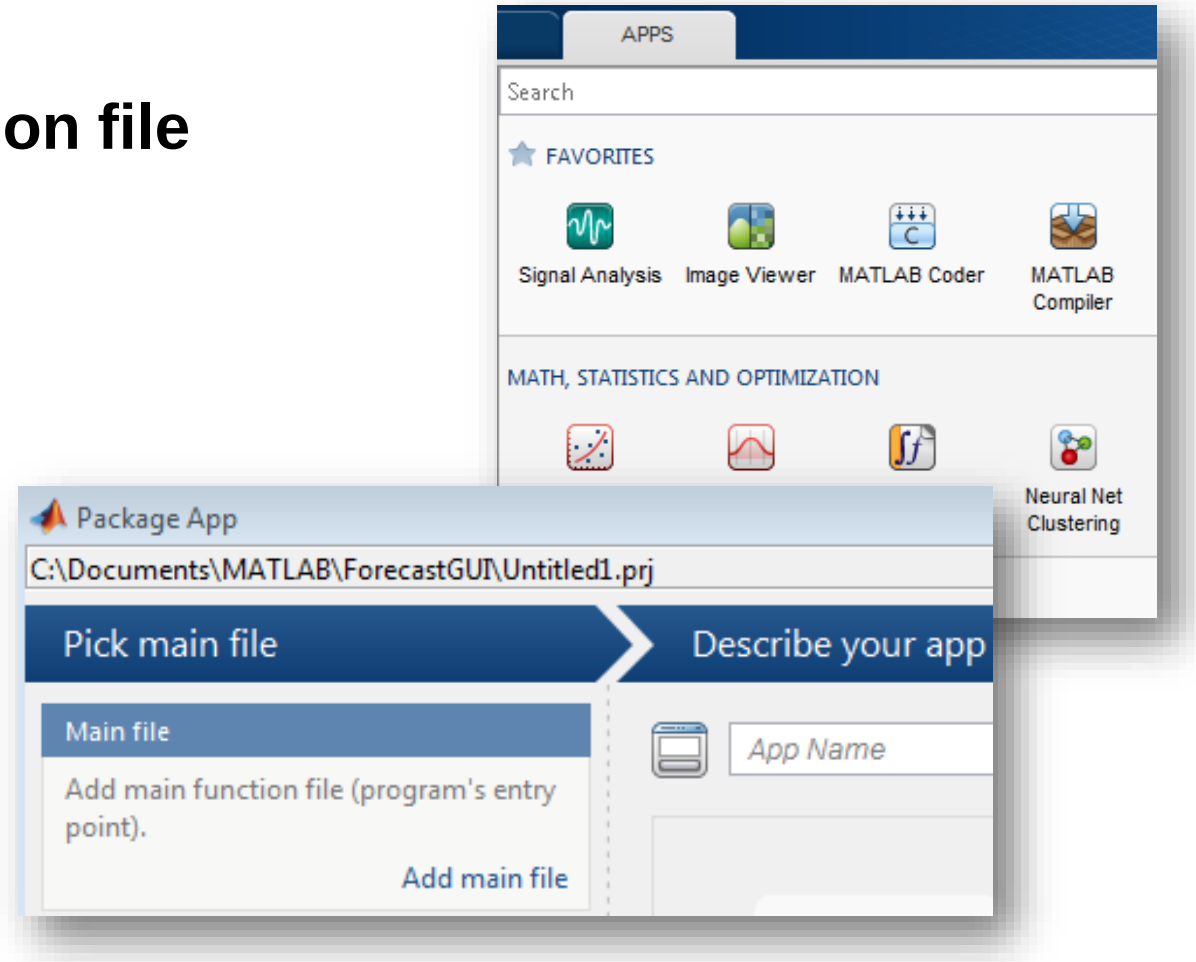
Share with MATLAB Users



- **Package an App**
- **Package Entire Toolboxes**
- **Directly share MATLAB files**

App Packaging

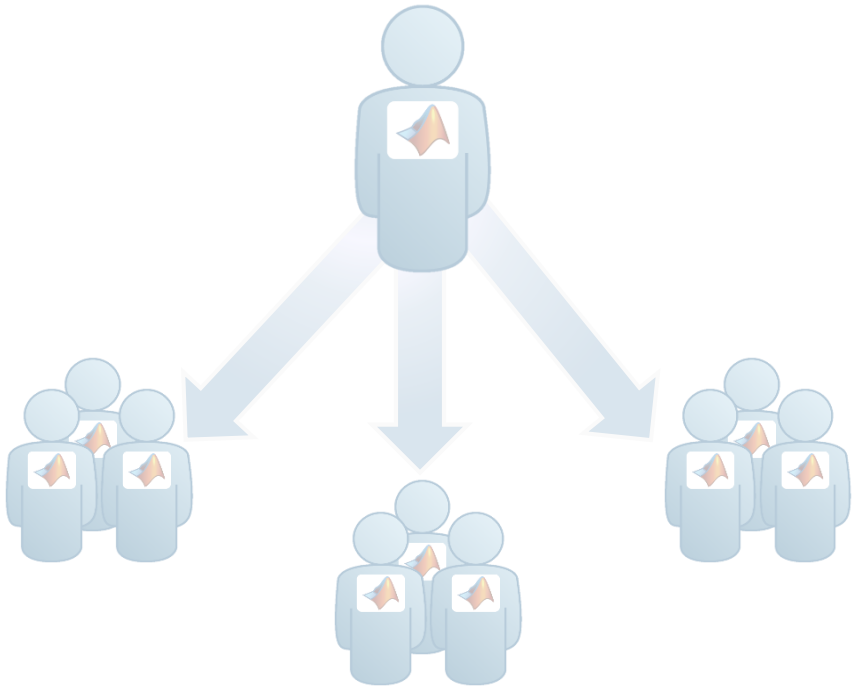
- Package your app as single installation file
- Workflow
  1. Create app
  2. Package app
  3. Select main file
  4. Files included through analysis
  5. Describe your app



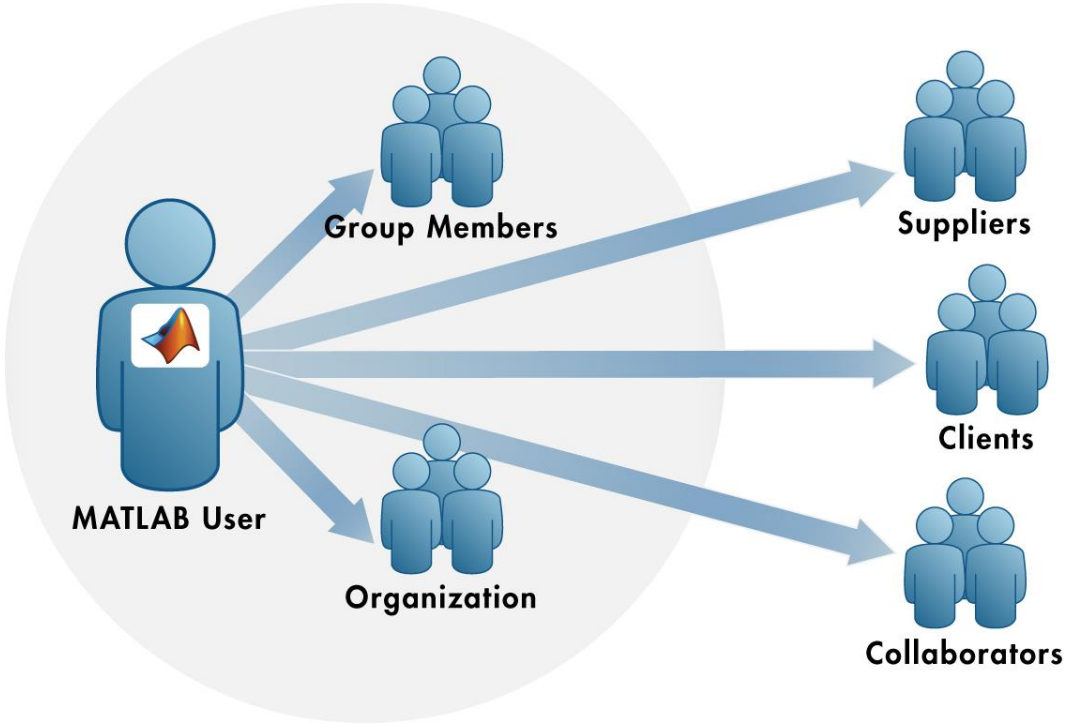
- Easy distribution
- Installation into the apps gallery
- Automatically includes all necessary files
- Documents required products

MATLAB Programs Can be Shared With Anyone

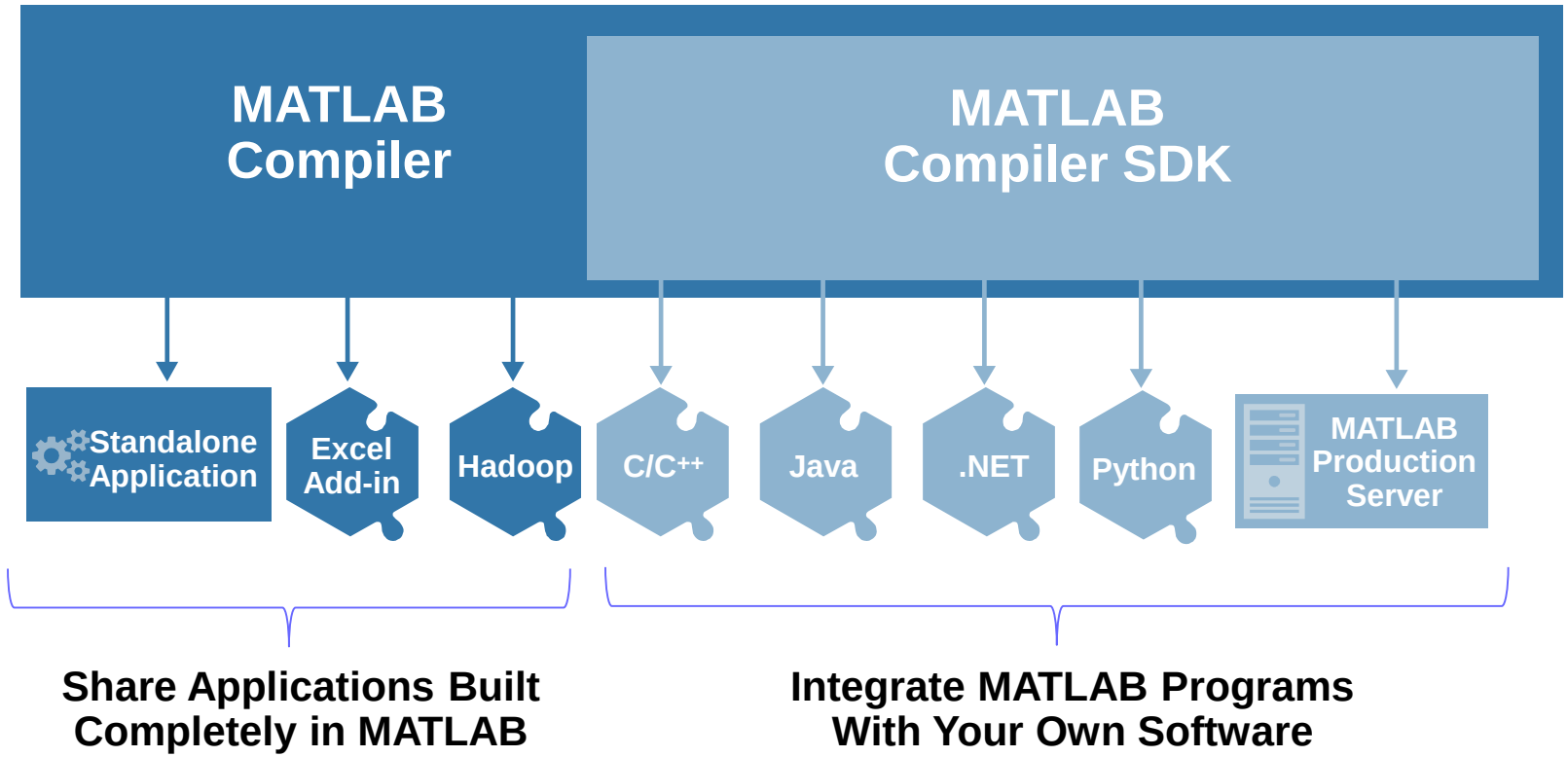
Share With Other MATLAB Users



Share With People Who do Not Have MATLAB



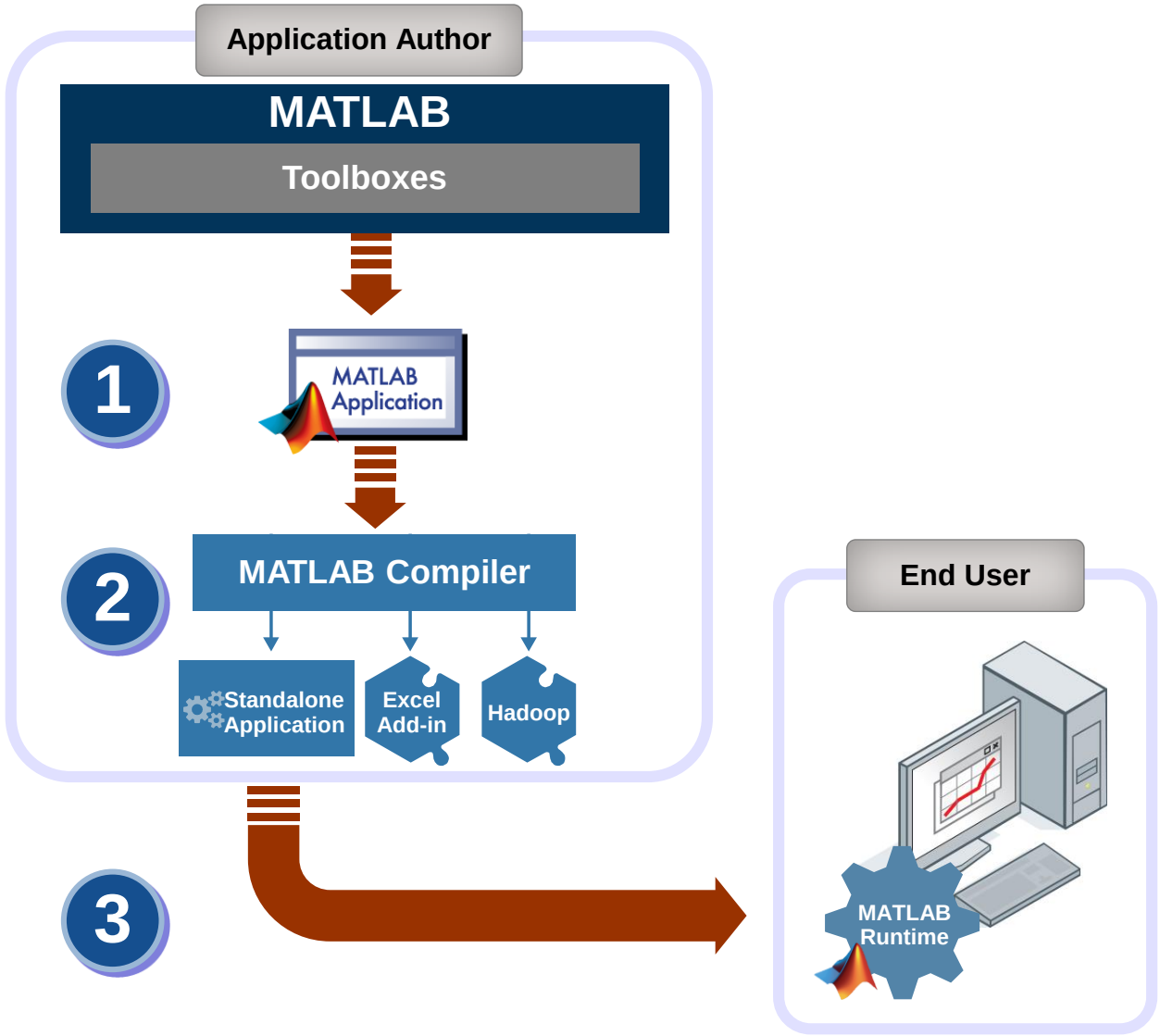
Share with People Who Do Not Have MATLAB



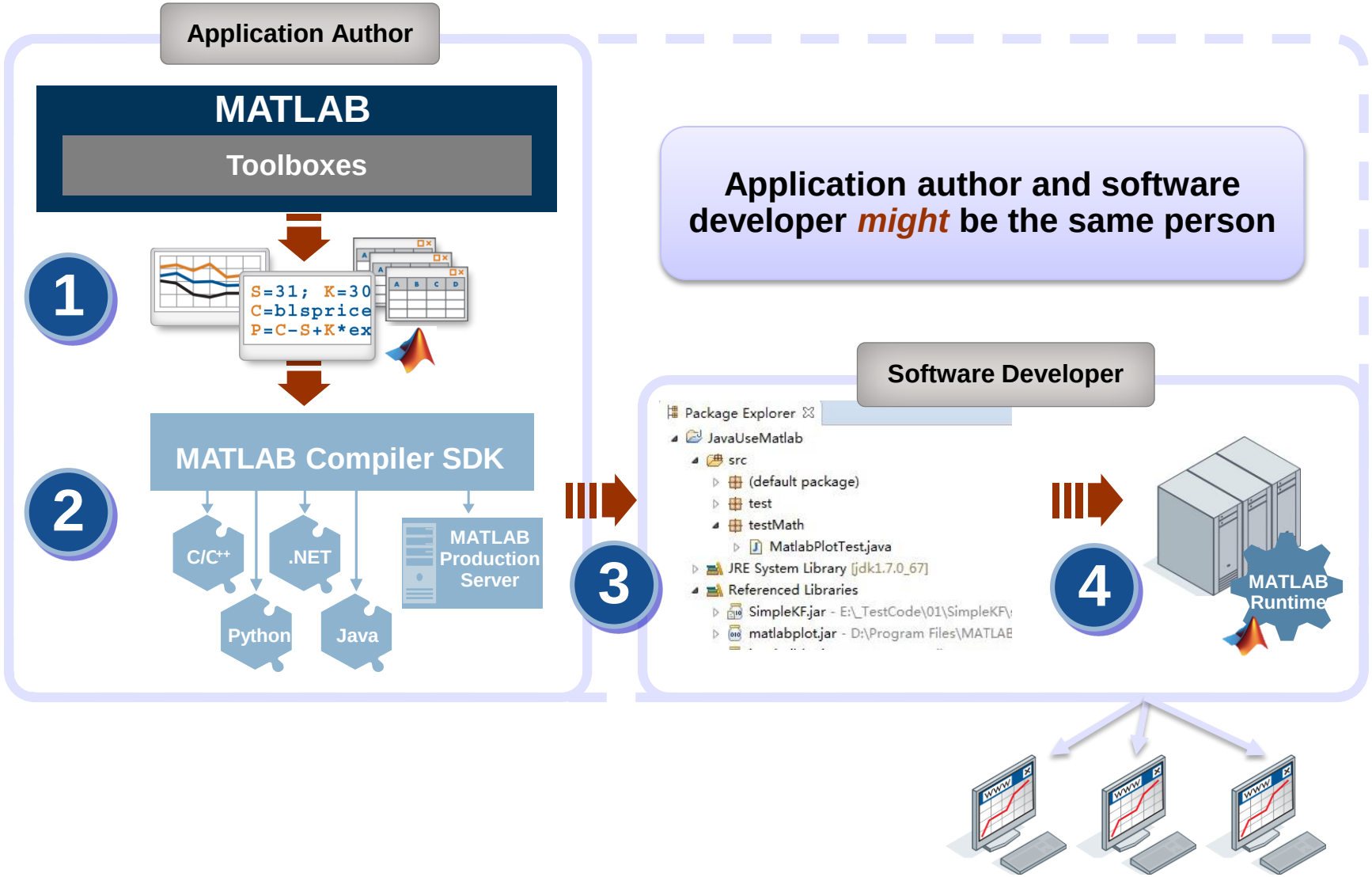
Share Applications Built Completely in MATLAB

- **MATLAB programs as standalone executables**
  - Users don’t need MATLAB license
- **Royalty-free distribution**
  - Creator needs only MATLAB Compiler
- **MATLAB Compiler encrypts your MATLAB code**
- **MATLAB Runtime**
  - free to download – MathWorks website
  - can be added to application
- **Interactively**
  - App – Application Compiler
- **Programmatically**
  - Command line – mcc

Share Applications Built Completely in MATLAB

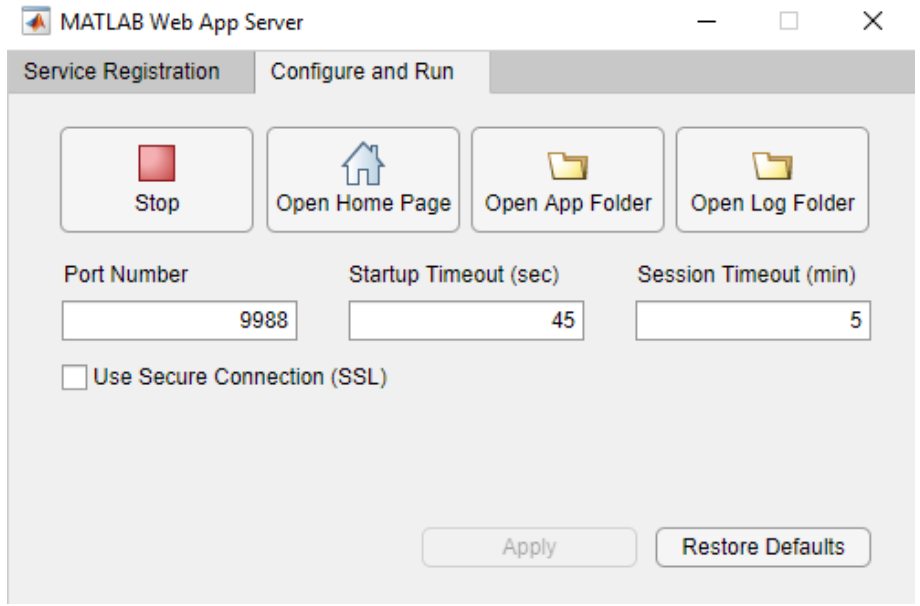


Integrate MATLAB Programs With Your Own Software



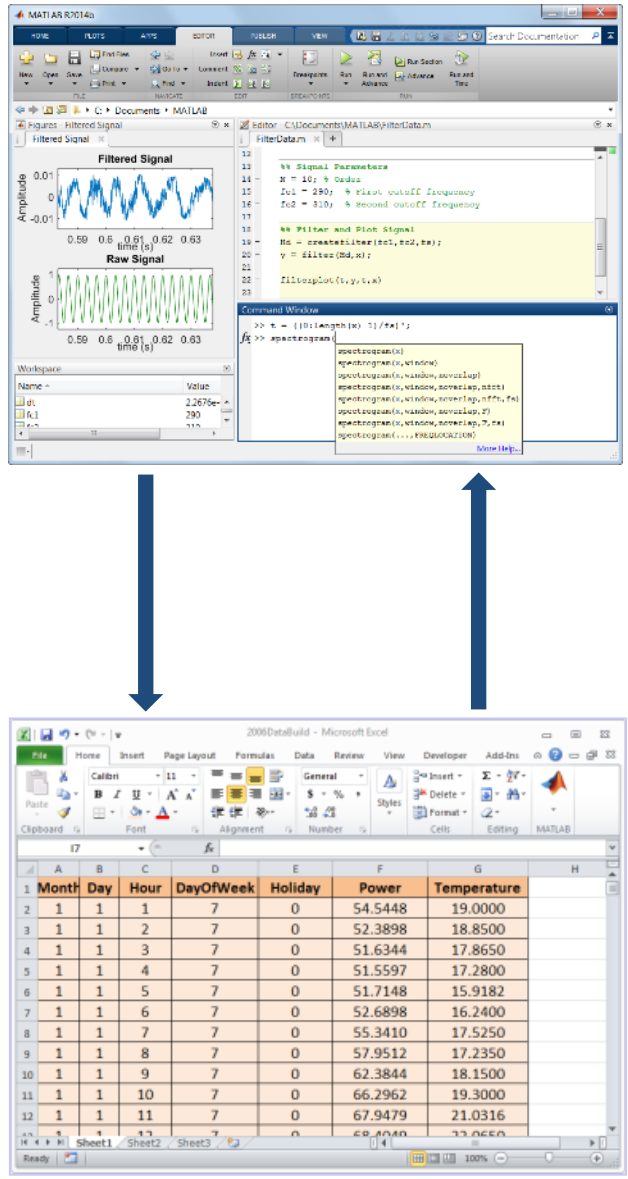
Small web apps

- **Web App Compiler**
- **Apps to web apps transformation**
  - **Only apps created via App Designer**
  - **Not all input functions supported**
- **Integrated WebAppServer**
- **MATLAB Runtime**
- **Security limitations/recommendations**
  - **Restrict network access**
  - **Server software only on dedicated hardware**
  - **Data or code injection – eval and disk operation usage**
- **Maximum number of sessions is limited to 32**
- **Licensed users who can upload and run web apps is limited to 10**



Using MATLAB with Excel

- **Passing data between MATLAB and Excel**
  - MATLAB
  - table, timetable, tall
  - *readtable, writetable*
- **Accessing MATLAB from an Excel spreadsheet**
  - MATLAB
  - Spreadsheet Link (for Microsoft Excel)
- **Deploying MATLAB as an Excel add-in**
  - MATLAB
  - MATLAB Compiler



Excel add-in

- **Sharing with Excel users**
  - analysis
  - graphical outputs
- **Library Compiler**
  - Excel Add-in
- **Output**
  - Add-ins
  - Macros
- **Workflow**
  1. Create function
  2. Package with Library Compiler
  3. Deploy to user/Excel
  4. Create Macro

Resources

- **Web pages**
  - www.humusoft.cz
  - www.mathworks.com
- **MATLAB Central**
  - Community for MATLAB and Simulink users
  - www.mathworks.com/matlabcentral/
- **Social networks**
  - Public Facebook group – MATLAB a Simulink (SK CZ)
  - www.facebook.com/groups/matlab4students/
  - Facebook
  - www.facebook.com/humusoft

Resources

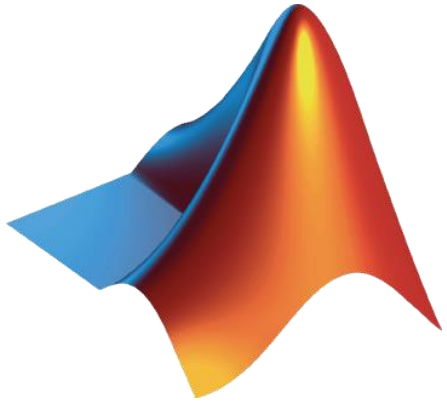
- **Webinars**
  - Free on-line seminars (EN, CZ, SK), access to recorded webinars
  - www.humusoft.cz/wwwseminare
- **Workshops**
  - Practical introduction to MATLAB & Simulink
  - www.humusoft.cz/workshop/
- **Training**
  - MATLAB, Simulink, dSPACE, COMSOL Multiphysics
  - www.humusoft.cz/skoleni
- **MATLAB Trial**
  - MATLAB for testing, time restricted – 30 days
  - request at info@humusoft.cz

End of Part 1

Application development workflow: From algorithm to Production systems

Part 2

Jorge Paloschi
MathWorks Consulting Services, Spain



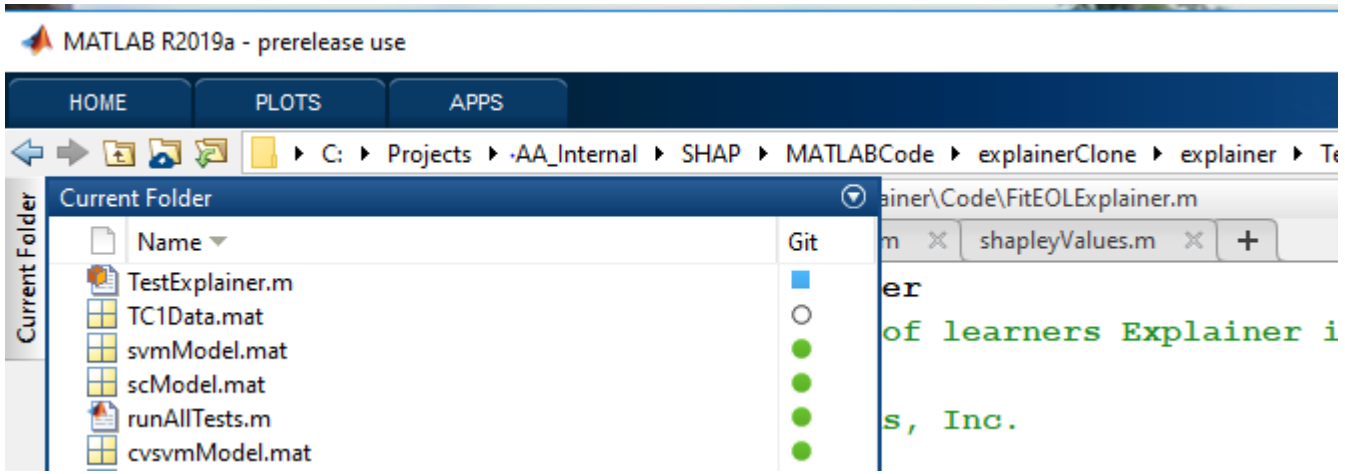
Agenda

- Version control
- Testing
- Deploying to Production

Source control (version control)

Source control

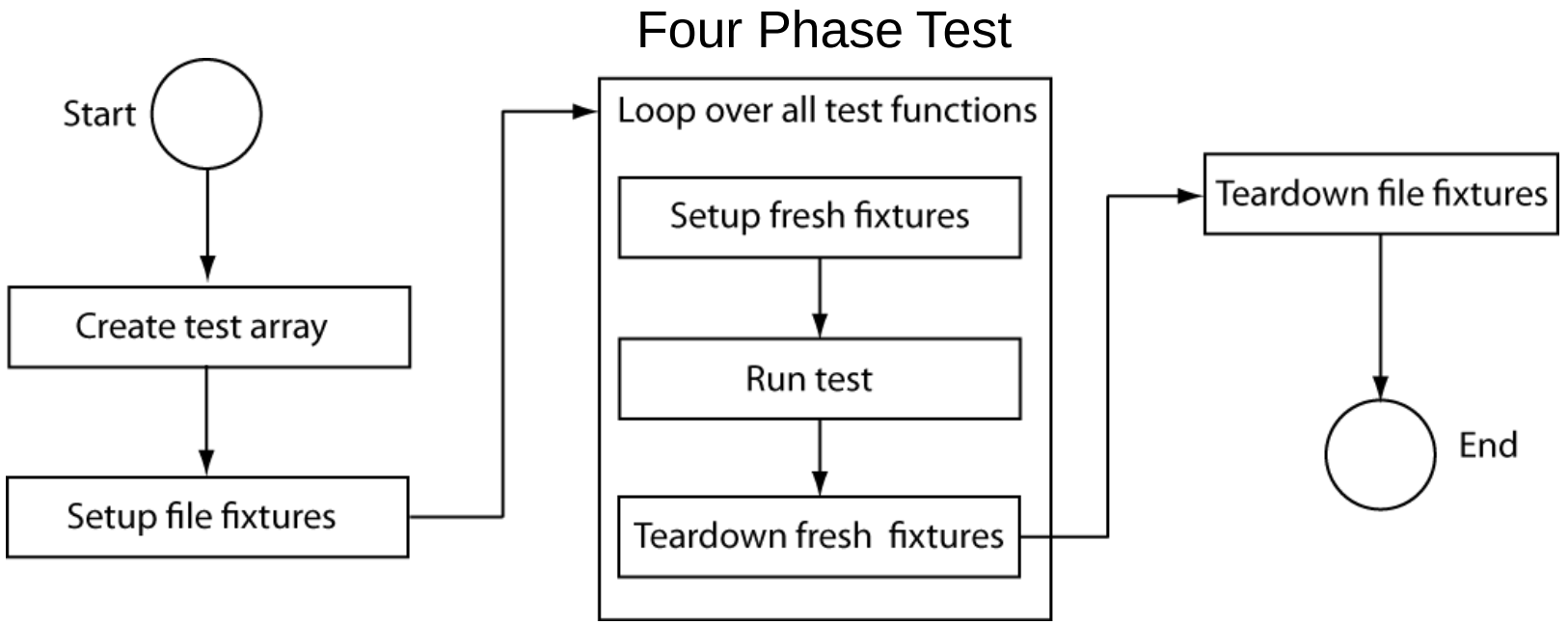
- MATLAB supports
  - Subversion (SVN)
  - GIT



Automated Testing

Why use Automated Testing?

- Improve quality
- Understand and document code
- Reduce and not introduce risk
- Catch bugs early and often



Authoring Tests

- Script, function, and class based tests
- Setup and teardown for four phase testing
- Rich qualifications and diagnostics
- Parameterization to build combinations of inputs

```
methods(TestMethodSetup)
    %Run before each test method
    function makeFigure(testCase)
        testCase.Figure = figure;
        plot(sin(1:100))
        xlabel('X')
        ylabel('Hello World','FontAngle','italic')
        patch([0 10 10 0],[0 0 1 1],'b','FaceAlpha',0.3)
    end
end
```

```
methods(TestMethodTeardown)
    %Run after each test method
    function closeFigure(testCase)
        delete(testCase.Figure)
    end
end
```

Type of Test	Verification	Assumption	Assertion	Fatal Assertion
Value is true.	verifyTrue	assumeTrue	assertTrue	fatalAssertTrue
Value is false.	verifyFalse	assumeFalse	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	verifyNotSameHandle	assumeNotSameHandle	assertNotSameHandle	fatalAssertNotSameHandle

Testing Infrastructure

- Run suites of tests from many files, directories, and packages
- Select tests based on tags or parameters
- Shared fixtures (e.g. path management, temporary folders)

```
Command Window
>> TS = TestSuite.fromPackage('Test.Visualization', 'IncludingSubPackages', true)
TS =
  1×34 Test array with properties:

    Name
    BaseFolder
    ProcedureName
    SharedTestFixtures
    Parameterization
    Tags
Tests Include:
   6 Unique Parameterizations, 0 Shared Test Fixture Classes, 0 Tags.
```

```
import matlab.unittest.fixtures.TemporaryFolderFixture
tdir = testCase.applyFixture(TemporaryFolderFixture);
```

```
properties (TestParameter)
    RasterImageTypes = {'bmp', 'png', 'jpg', 'gif'} % Different raster image types
    StreamOutputTypes = {'uint8', 'int8'} % Different byte stream data types
end
```

Test Outputs

- Coverage report

Coverage results

Show coverage for parent directory

Total lines in function	28
Non-code lines (comments, blank lines)	21
Code lines (lines that can run)	7
Code lines that did run	4
Code lines that did not run	3
Coverage (did run/can run)	57.14 %

	Total coverage: 88.2%
Model.TrainedModel	Coverage: 100.0% Total time: 0.0 seconds Total lines: 6
Model.predict	Coverage: 100.0% Total time: 10.6 seconds Total lines: 2
Model.set.ModelName	Coverage: 100.0% Total time: 0.0 seconds Total lines: 3
Model.set.ModelType	Coverage: 100.0% Total time: 0.0 seconds Total lines: 4
Model.predictImpl	
	Total coverage: 81.3%
egressionComparer.RegressionComparer	Coverage: 100.0% Total time: 0.1 seconds Total lines: 9
RegressionComparer>RegressionComparer.compareImpl	Coverage: 57.1% Total time: 10.8 seconds Total lines: 7
CurrentModel.m	Total coverage: 75.0%
CurrentModel>CurrentModel.CurrentModel	Coverage: 92.3% Total time: 0.0 seconds Total lines: 13
CurrentModel>CurrentModel.predict	

Function listing

Color highlight code according to noncoverage

```
  time    Calls   line
                    46          function [bestmodel, modelvalues] = compareImpl(obj,
                    63              % Verify everything is a Regression Model
< 0.01          3   64              mt = cellfun(@(x)isa(x, 'LoadForecaster.Predictio
< 0.01          3   65              if ~all(mt)
                    66                  me = MException('LoadForecaster:Prediction:Re
                    67                  throwAsCaller(me);
                    68              end
                    69
                    70              % Use superclass compareImpl
 10.40          3   71              [bestmodel, modelvalues] = compareImpl@LoadForecaster.Prediction.ModelComparer(obj, models, modelpvpairs);
                    72
< 0.01                          end
```

Test Outputs

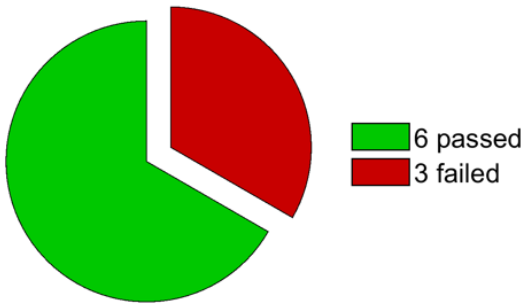
- Test results report

MATLAB® Test Report

Timestamp: 22-Sep-2017 12:16:21
Host: AH-SDEWOLSK
Platform: win64
MATLAB Version: 9.3.0.701794 (R2017b)

Number of Tests: 9
Testing Time: 4.7620 seconds

Overall Result: FAILED



Failure Summary

3 tests failed.

Name of Failing Test	Failure Reasons	
Test.DataManagement.TestDatabase/testGetData	Failed by verification.	(Details)
Test.DataManagement.TestDatabase/testAllNansInGetSynchronize	Failed by verification.	(Details)
Test.DataManagement.TestDatabase/testGetDataOneSource	Failed by verification.	(Details)

Overview

C:\Documents\MATLAB\DataAnalytics\LoadForecastingApplication\Source\

Test.DataManagement.TestDatabase 4.7620 seconds

testClosedConnection

The test passed.

Duration: 0.0932 seconds

(Overview)

testGetData

The test failed.

Duration: 2.1368 seconds

Event:

Verification failed.

Framework Diagnostic:

```
verifyEqual failed.
--> The objects are not equal using "isequal".

Actual Value:
  0×2 empty timetable
Expected Value:
  4×3 timetable
            Time                N_Y_C_          TemperatureF_KLGA     Dewpoint_KLGA
    ____________________    _____________    _____________________    _____________
    01-May-2007 00:00:00            4854.4                    58                30
    01-May-2007 00:05:00            4802.3    58.0833333333333     29.8333333333333
    01-May-2007 00:10:00            4740.3    58.1666666666667     29.6666666666667
    01-May-2007 00:15:00    4700.34155844156               58.25                29.5
```

Event Location: Test.DataManagement.TestDatabase/testGetData

Stack:

```
In C:\Documents\MATLAB\DataAnalytics\LoadForecastingApplication\Source\+Test\+DataManagement\TestDatabase.m
(TestDatabase.testGetData) at 98
```

Continuous Integration

- Runs tests after code changes to track development progress
- Support for TAP, jUnit XML

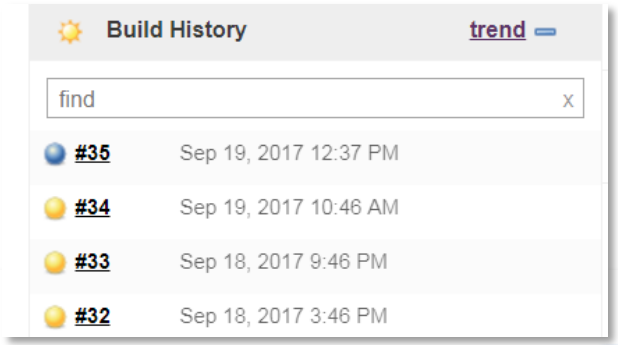
	59	- Test.DataManagement.TestWUAPI/testDefaultTimeRange
	60	- Test.DataManagement.TestWUAPI/testWUWebsiteAccess
	61	- Test.DataManagement.TestWUAPI/testOfflineIsConnected
	62	- Test.DataManagement.TestWUAPI/testGetDataOffline
	63	- Test.DataManagement.TestWUAPI/testSevenDaysAgoKLGA

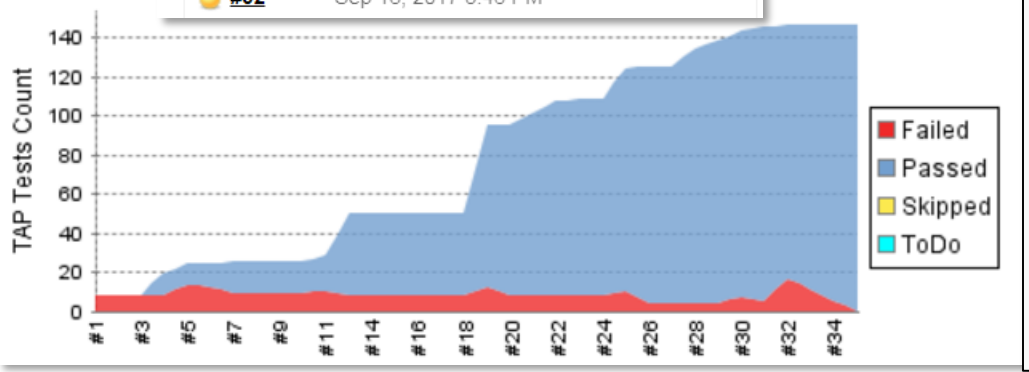
Event

Event Name	ExceptionThrown
Event Location	Test.DataManagement.TestWUAPI/testSevenDaysAgoKLGA
Error Identifier	MATLAB:datetime:InvalidTimeZone

Error Report	Error using datetime (line 515)
	A time zone must be specified as a character vector.
	
	Error in LoadForecaster.DataManagement.WUAPI/getWUHistoryByStation (line 167)
	Time = datetime(double(wudate.string_year), double(wudate.string_mon), ...
	
	Error in LoadForecaster.DataManagement.WUAPI/getWUHistory (line 124)
	data{ii} = getWUHistoryByStation(obj, stations{ii}, dates, key);
	
	Error in LoadForecaster.DataManagement.WUAPI/getDataImpl (line 38)
	datacell{1} = getWUHistory(obj, stations, dates);
	
	Error in LoadForecaster.DataManagement.DataSource/getData (line 49)
	data = getDataImpl(obj, trange, vars);
	
	Error in Test.DataManagement.TestWUAPI/testSevenDaysAgoKLGA (line 73)
	data = getData(api, 'TimeRange', [daystart dayend], 'SelectedVariables', {'KLGA'});

Stack	In C:\Program Files\MATLAB\R2017a\toolbox\matlab\timefun\@datetime\datetime.m (datetime.datetime) at 515 In C:\Users\sdewolsk\.jenkins\workspace\LoadForecaster\Source\+LoadForecaster\+DataManagement\WUAPI.m (WUAPI.getWUHistoryByStation) at 167 In C:\Users\sdewolsk\.jenkins\workspace\LoadForecaster\Source\+LoadForecaster\+DataManagement\WUAPI.m (WUAPI.getWUHistory) at 124 In C:\Users\sdewolsk\.jenkins\workspace\LoadForecaster\Source\+LoadForecaster\+DataManagement\WUAPI.m (WUAPI.getDataImpl) at 38 In C:\Users\sdewolsk\.jenkins\workspace\LoadForecaster\Source\+LoadForecaster\+DataManagement\DataSource.m (DataSource.getData)

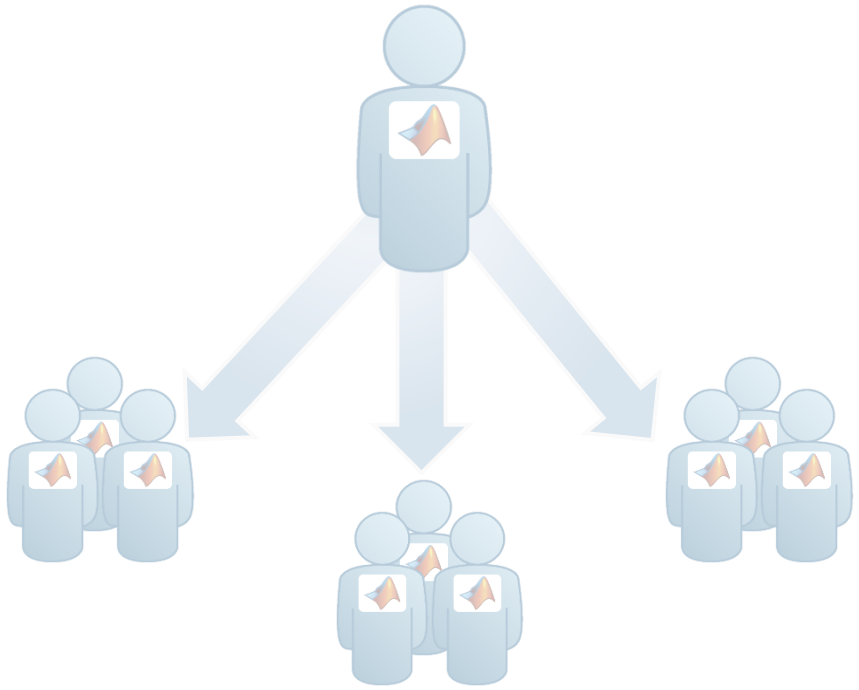




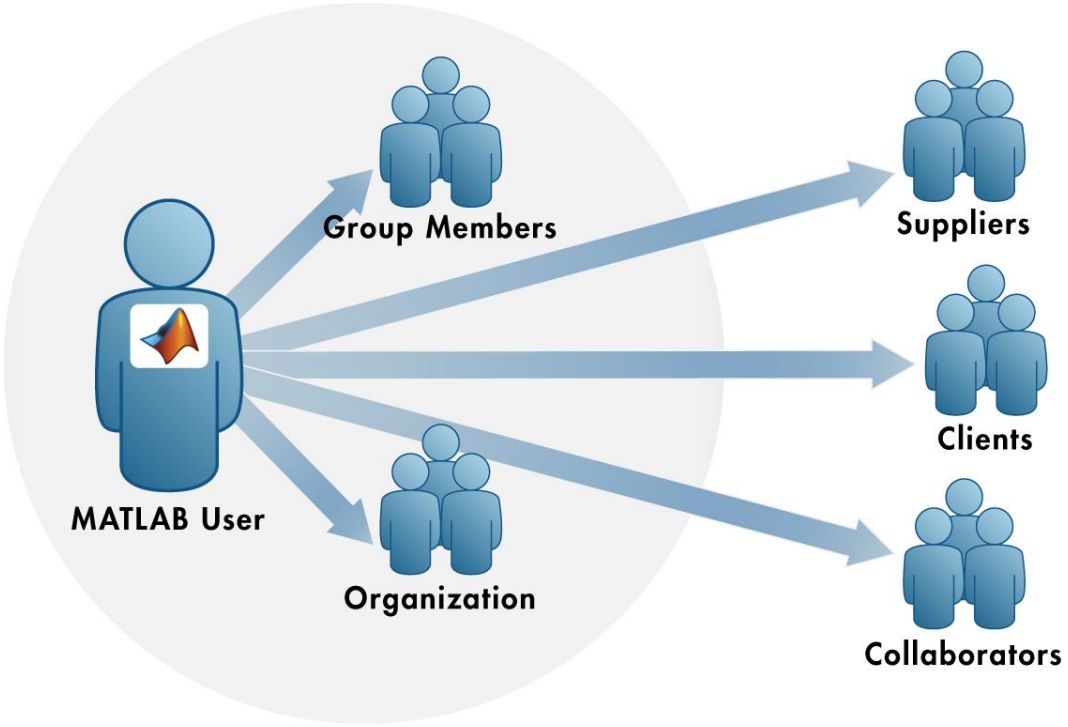
MATLAB code deployment

MATLAB Programs Can be Shared With Anyone

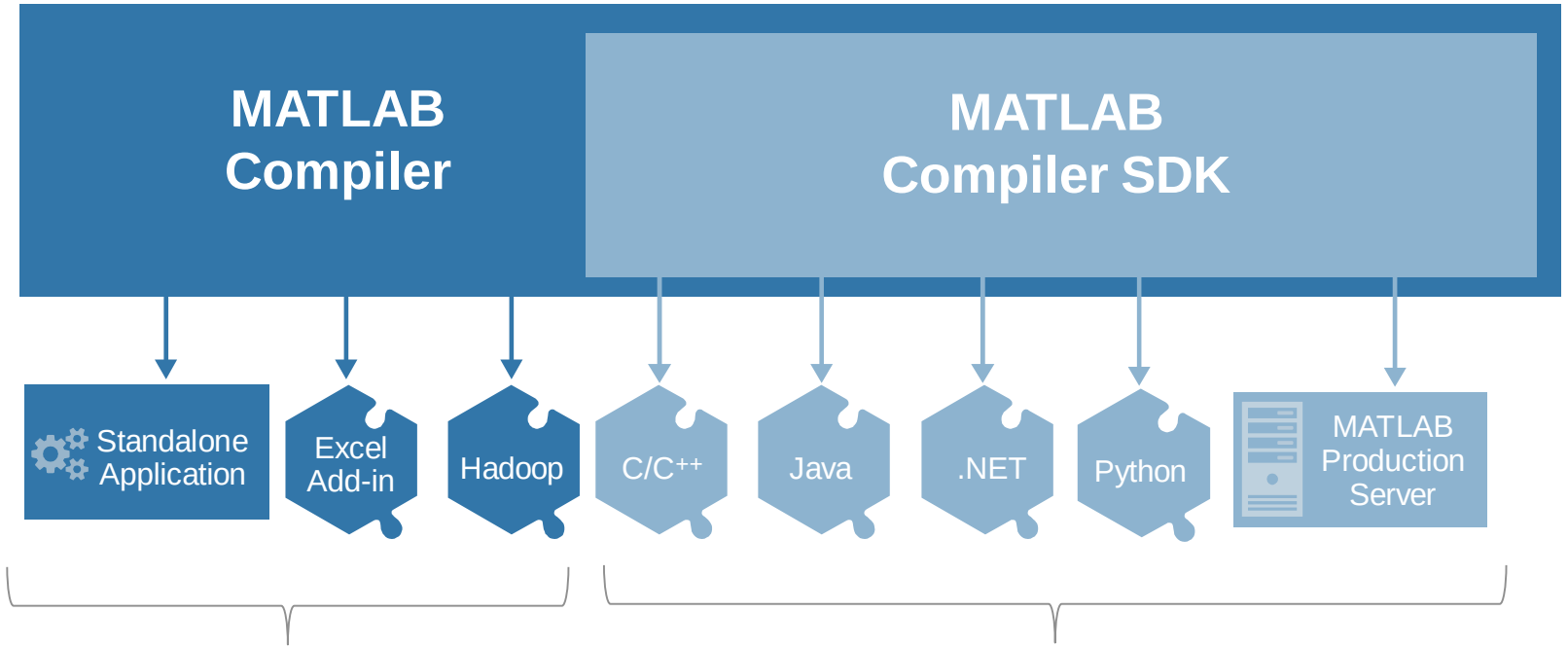
Share With Other MATLAB Users



Share With People Who do Not Have MATLAB

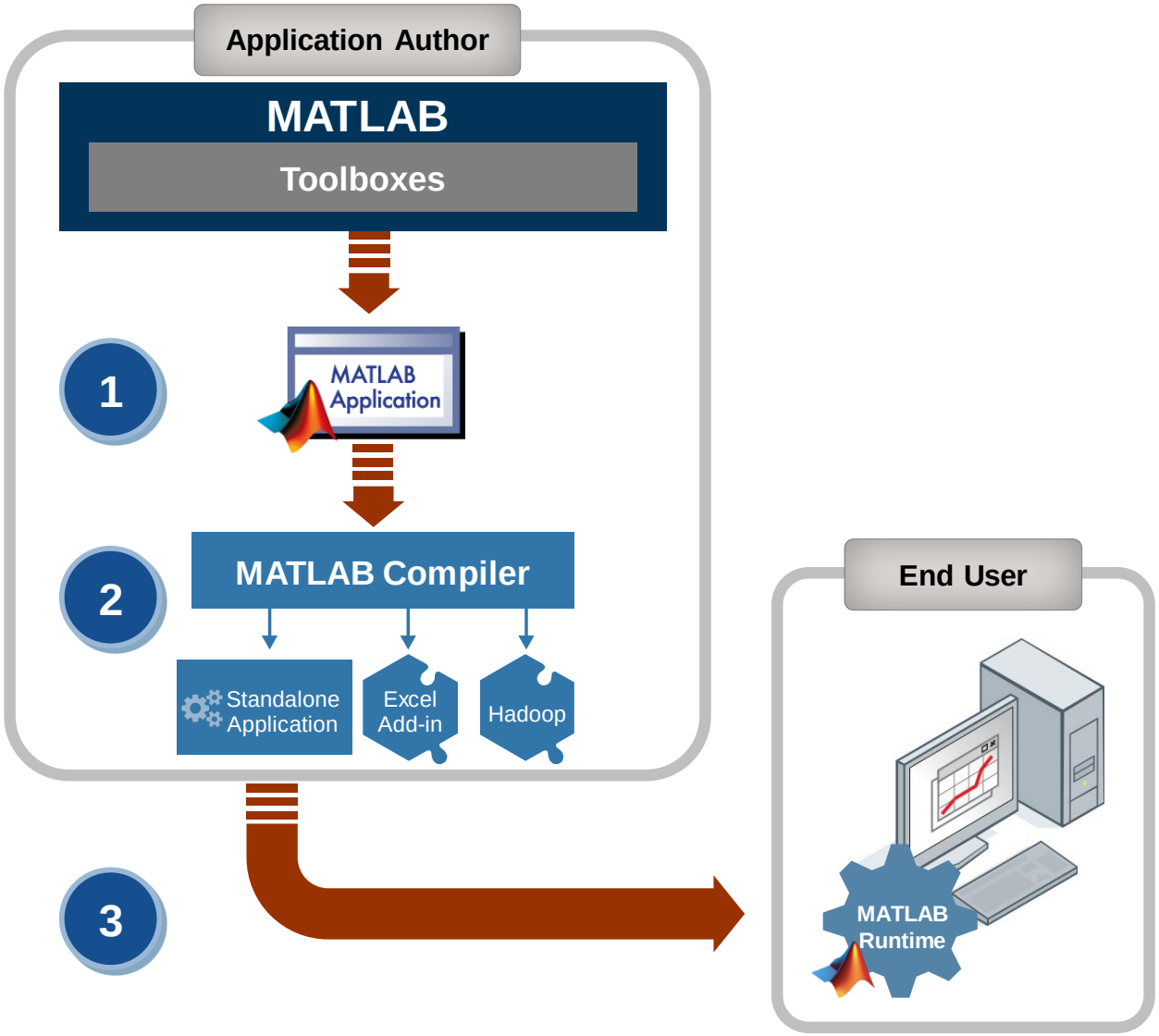


Share with People Who Do Not Have MATLAB



- Royalty-free Sharing
- IP Protection via Encryption

Share Applications Built Completely in MATLAB



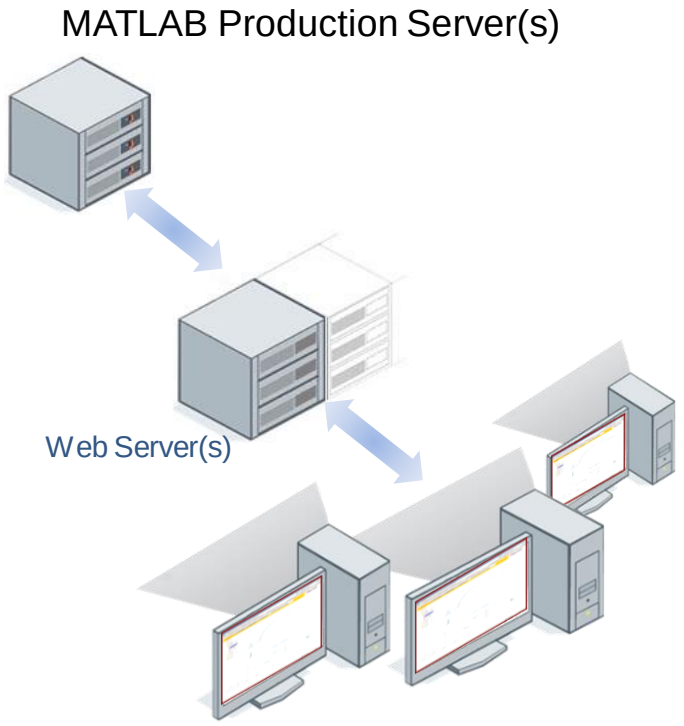
Scale Up with MATLAB Production Server

Most efficient path for creating enterprise applications

Deploy MATLAB programs into production

- Manage multiple MATLAB programs and versions
- Hot Deployment: Update programs without server restarts
- Reliably service concurrent requests

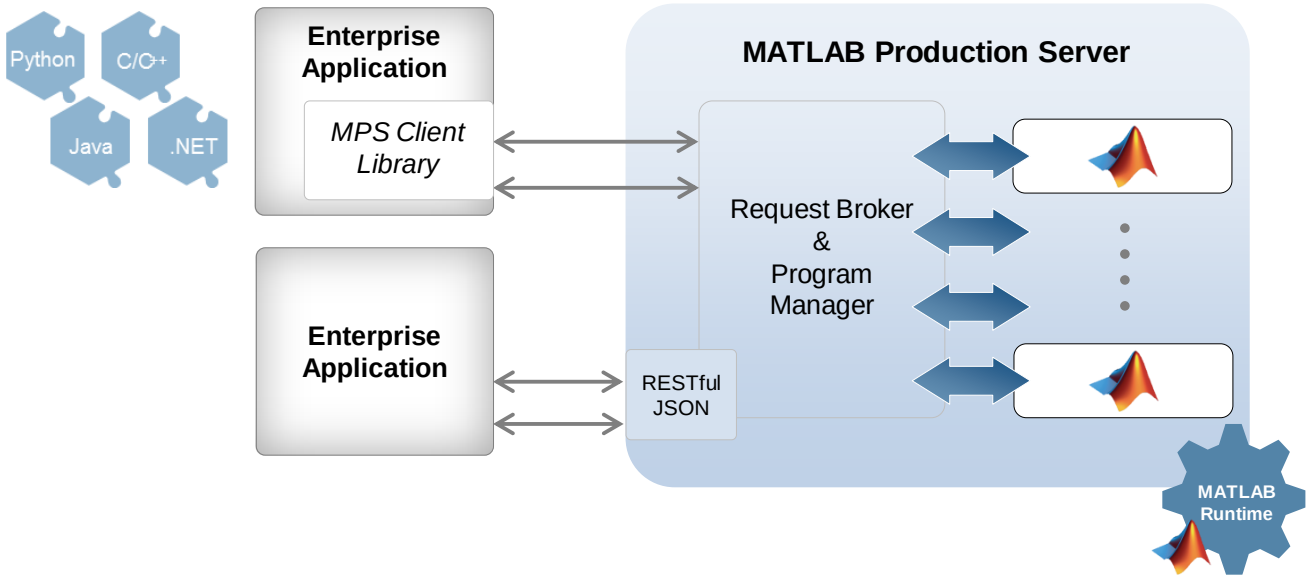
Integrate with web, database, and application servers



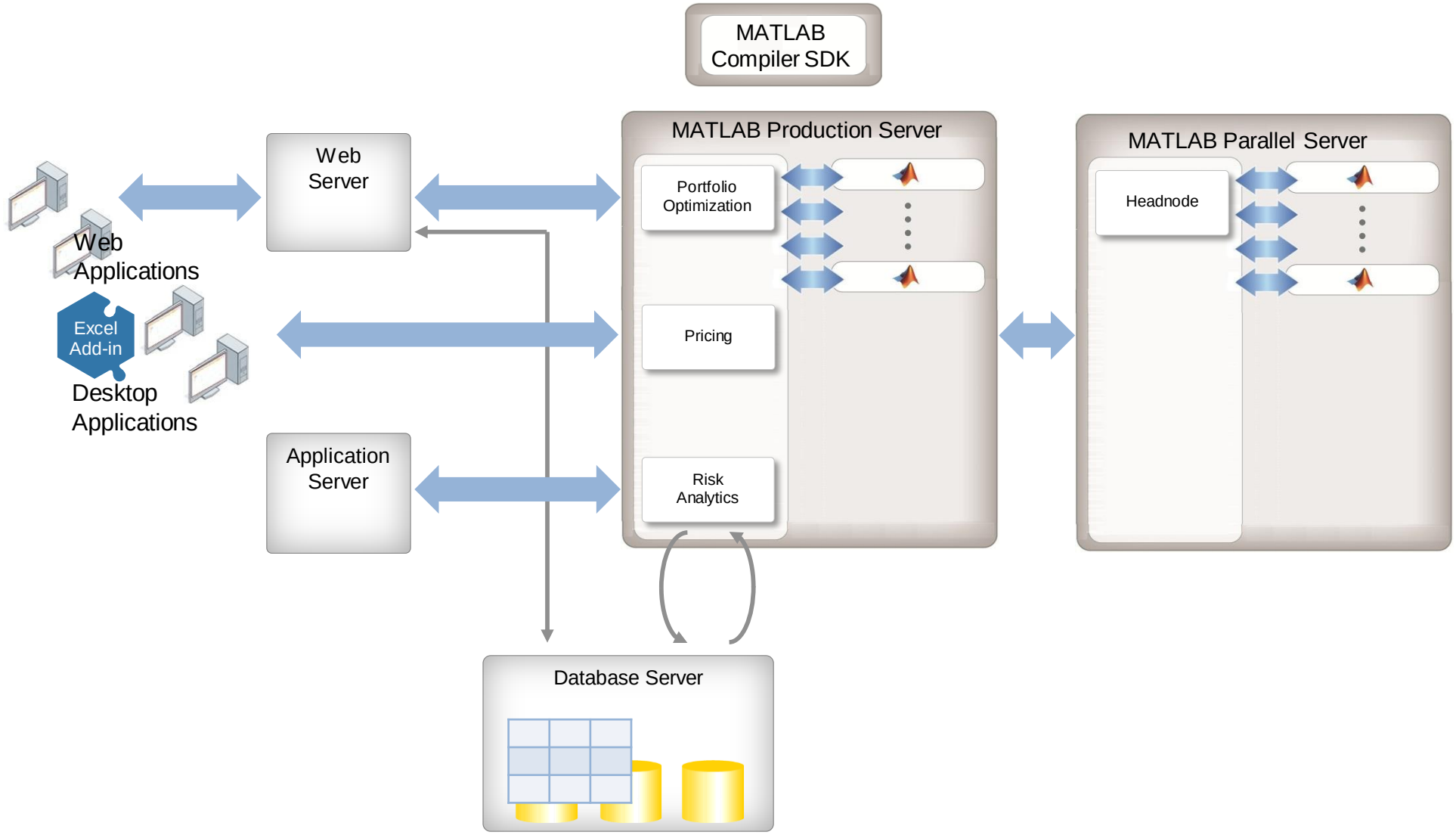
MATLAB Production Server

Enterprise Class Framework For Running Packaged MATLAB Programs

- Server software
  - Manages packaged MATLAB programs and worker pool
- MATLAB Runtime libraries
  - Single server can use runtimes from different releases
- RESTful JSON interface and lightweight client library (C/C++, .NET, Python, and Java)



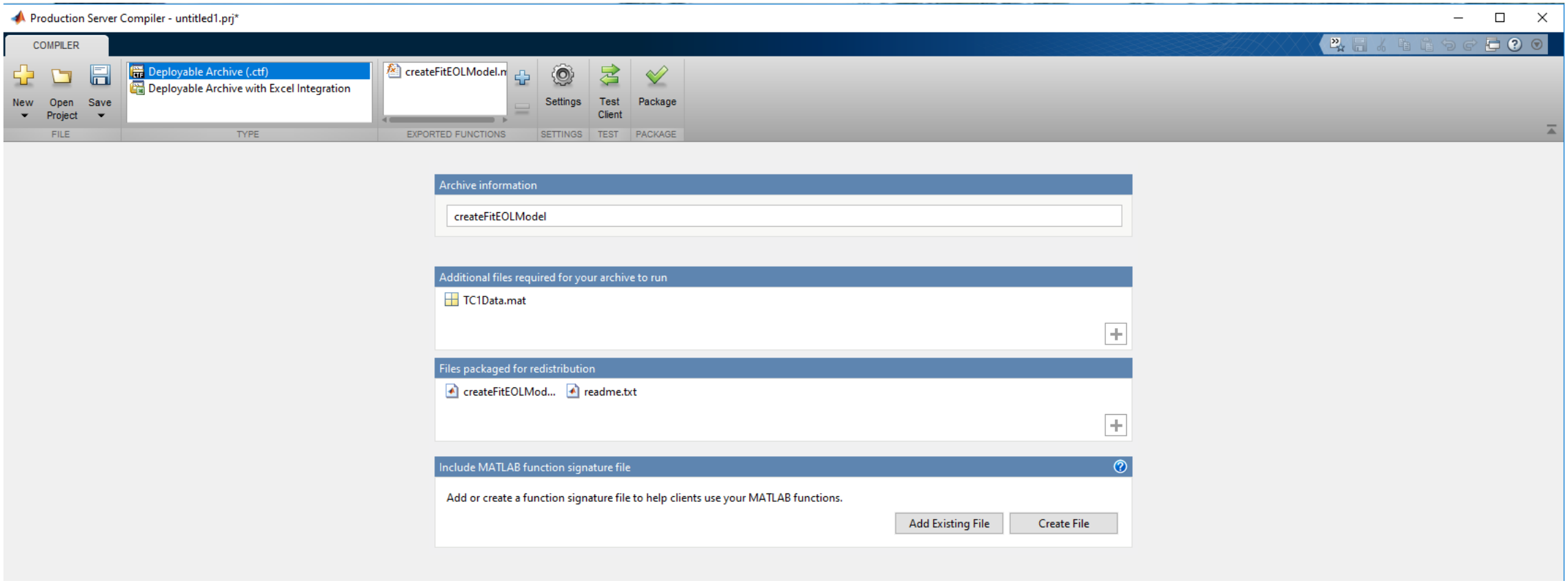
Example - Integrating with IT systems



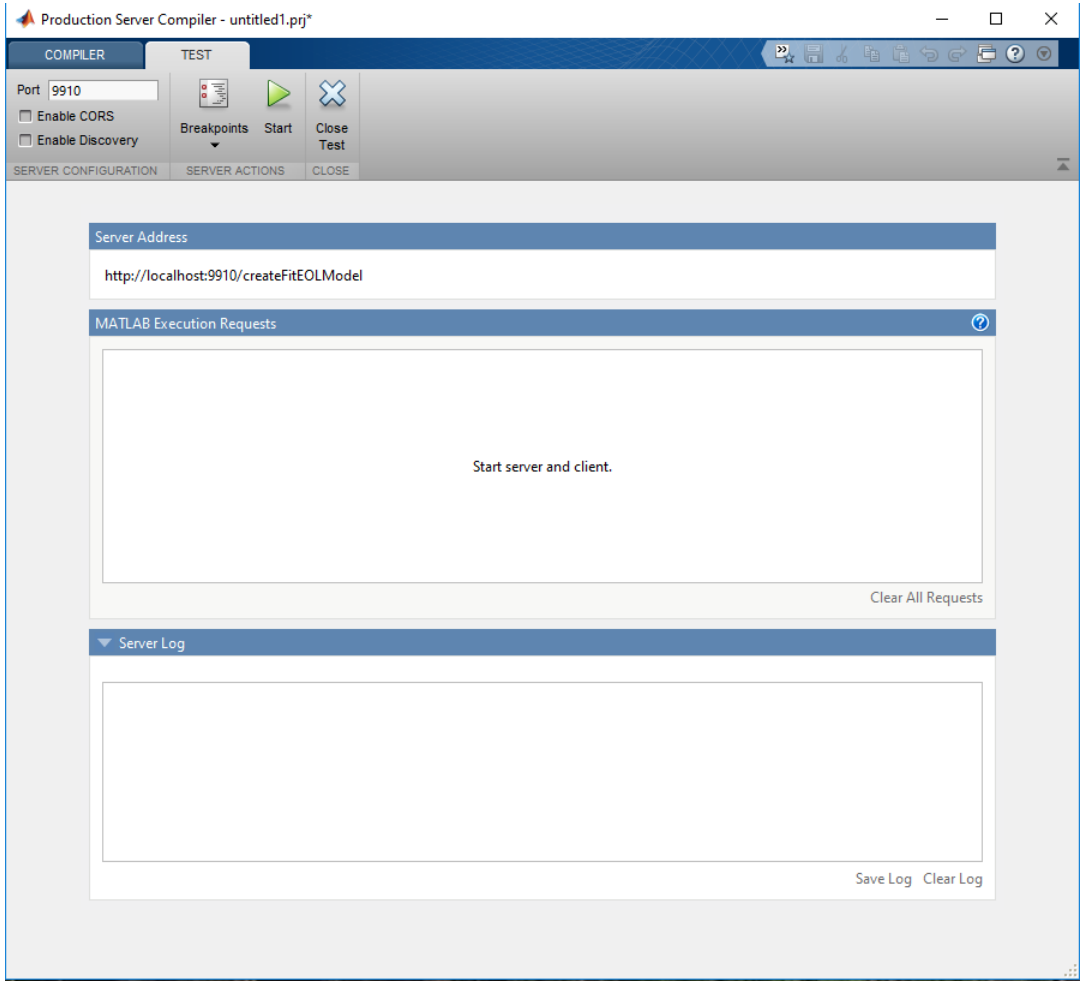
Developing code for MPS

- Test server
- Possible to debug if using test server
- Code needs to be prepared to work under MPS
  - Functionality to be able to reproduce issues

Deploying CTF components with Compiler



Test server


Production Server Compiler - untitled1.prj*
COMPILER
TEST
Port 9910
Enable CORS
Enable Discovery
Breakpoints
Start
Close Test
SERVER CONFIGURATION
SERVER ACTIONS
CLOSE
Server Address
http://localhost:9910/createFitEOLModel
MATLAB Execution Requests
Start server and client.
Clear All Requests
Server Log
Save Log
Clear Log