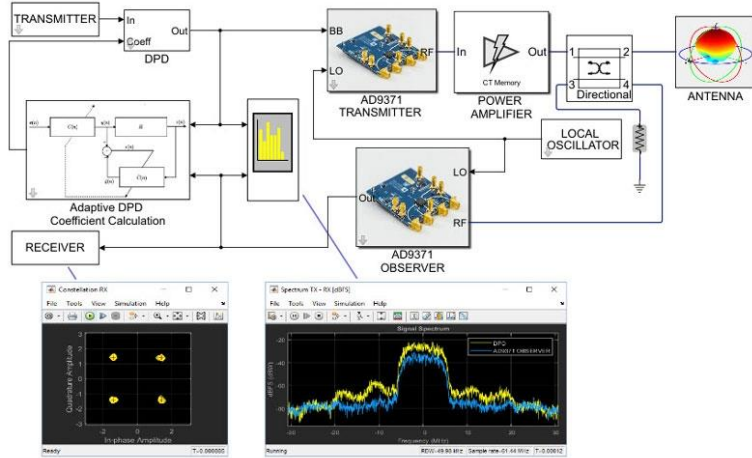
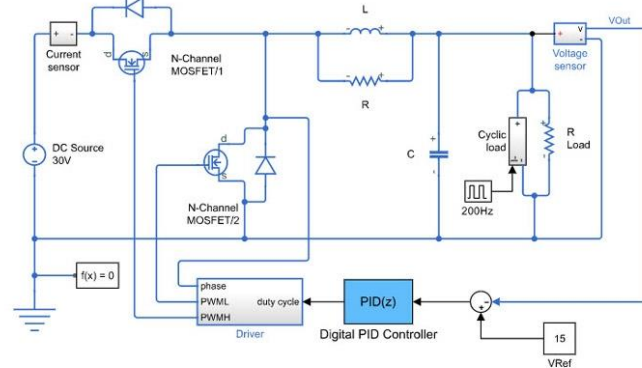




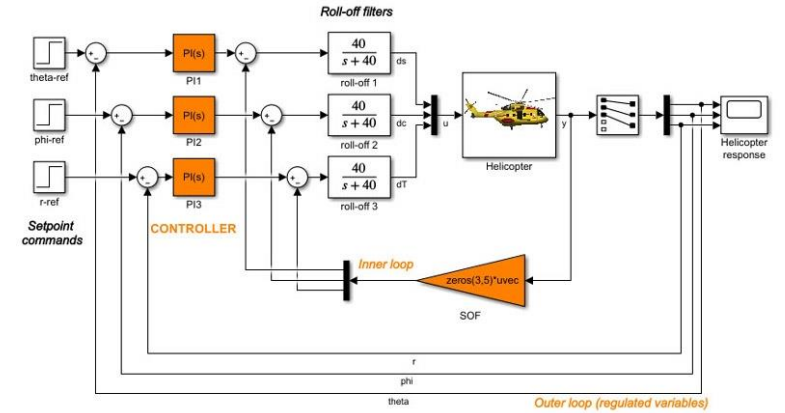
Wireless Communications



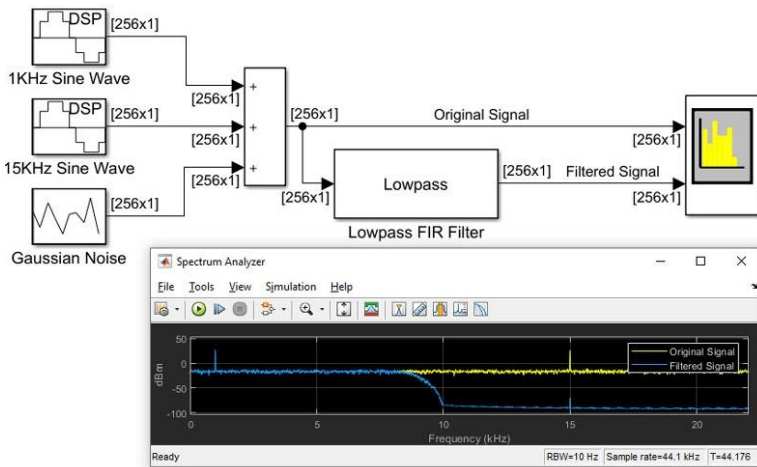
Power Electronics



Control Systems



Signal Processing



Advanced Driver Assistance

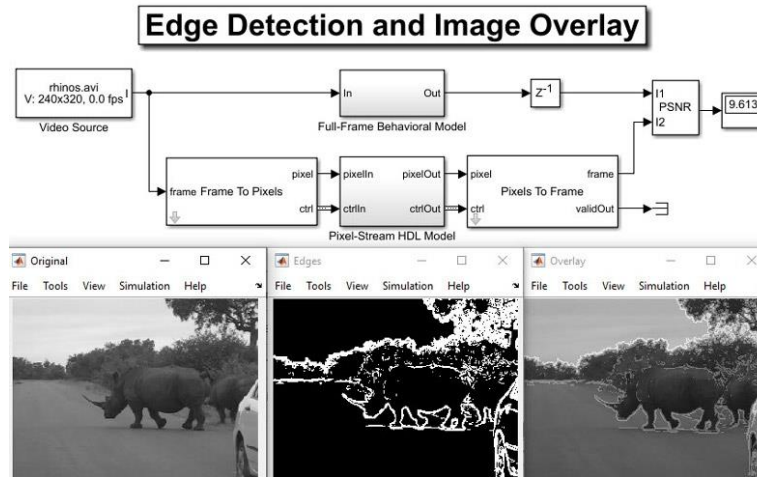
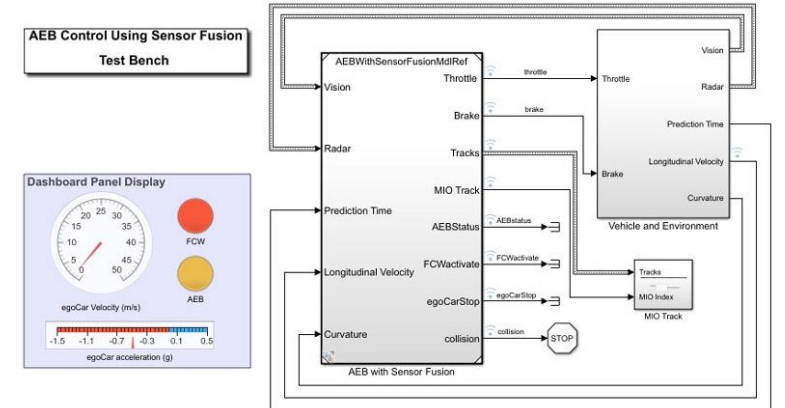
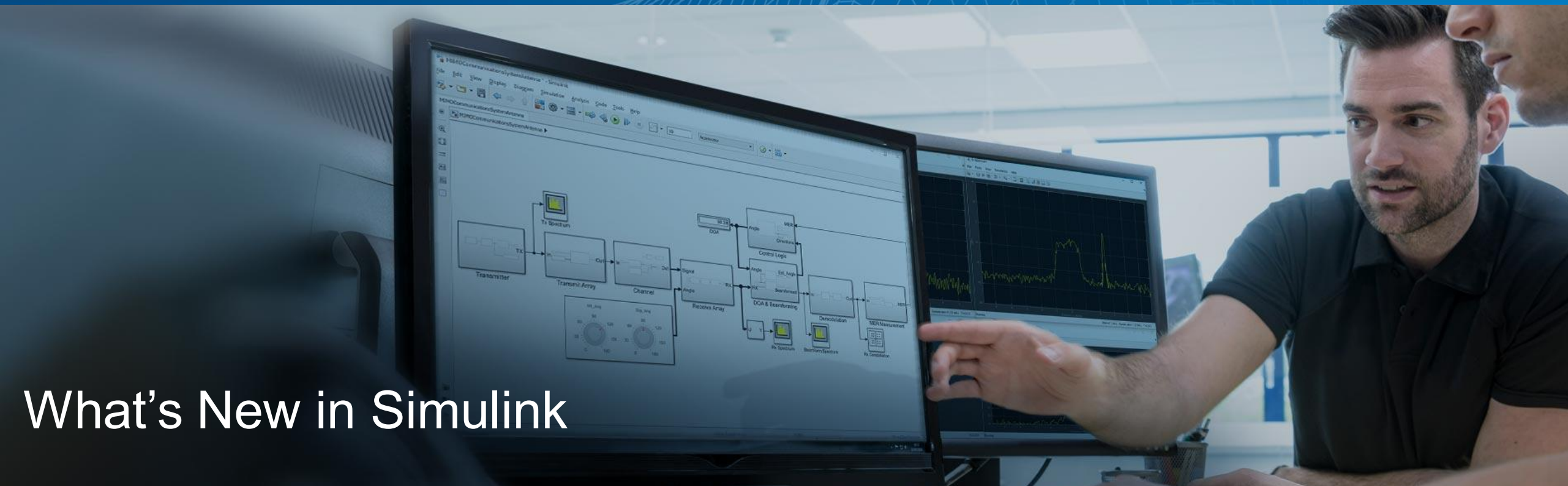


Image Processing





What's New in Simulink

Ed Marquez – Simulink Product Marketing

R2020b

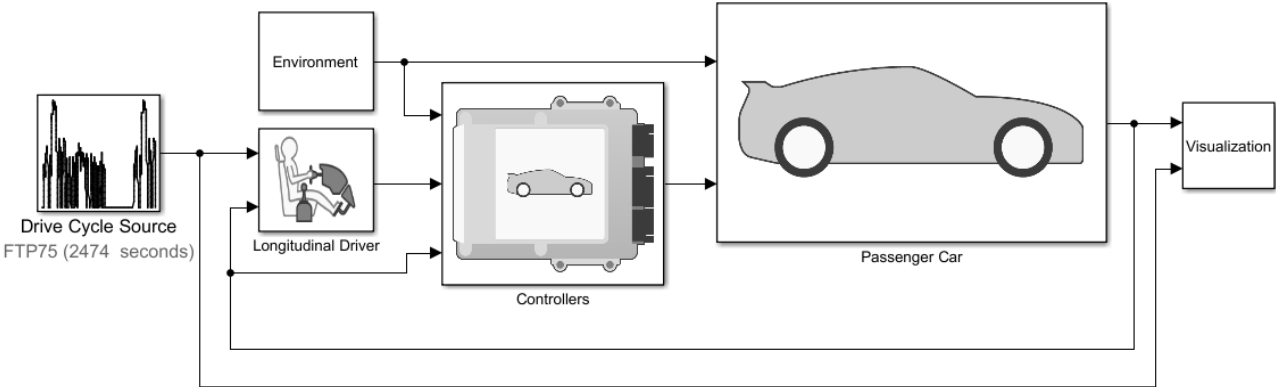
Core MathWorks Products

SIMULINK®

Simulation and Model-Based Design

Model and simulate your system

- Use one multi-domain environment
- Model the system under test and the plant
- Simulate closed-loop system behavior



SIMULINK®

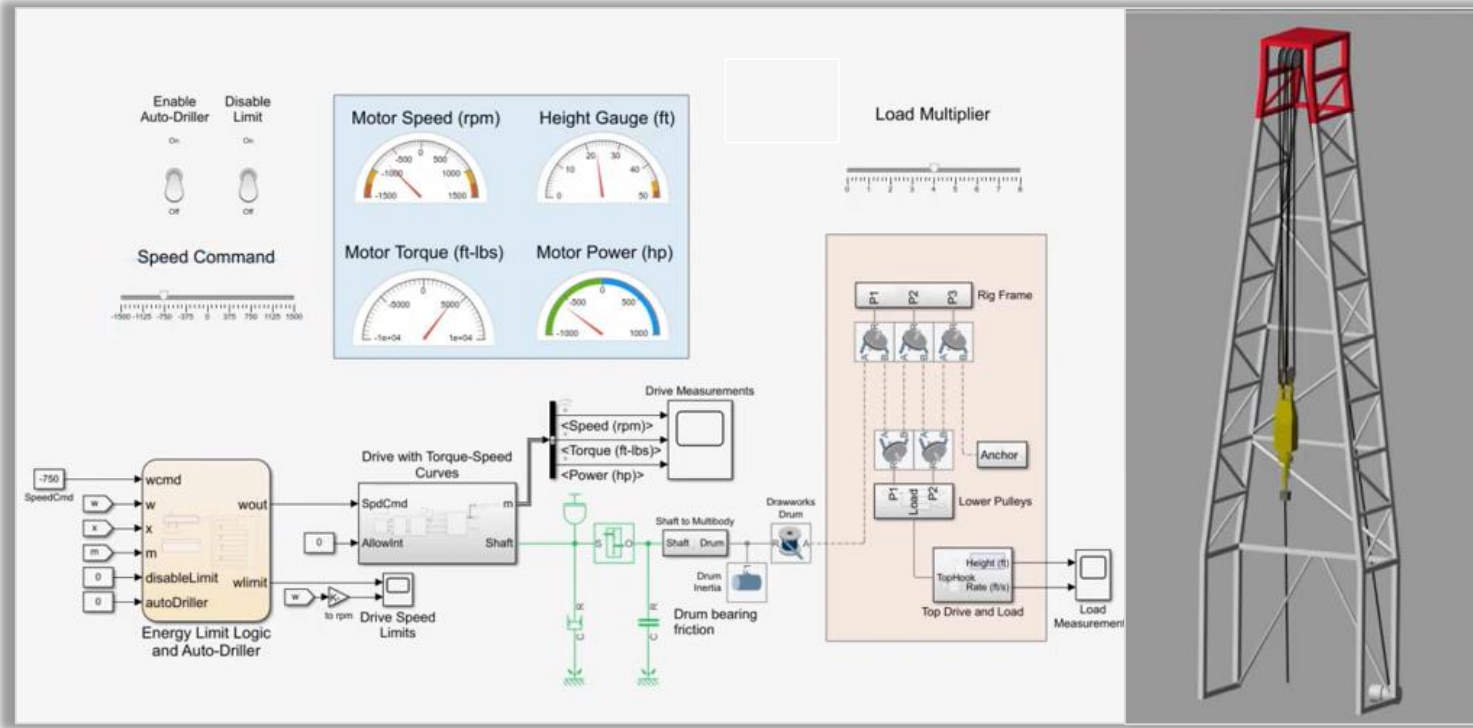
Simulation and Model-Based Design

Model and simulate your system

- Use one multi-domain environment
- Model the system under test and the plant
- Simulate closed-loop system behavior

Test early and often

- Test your system under all conditions
- Validate your design with real-time testing
- Trace from requirements to design to code



SIMULINK®

Simulation and Model-Based Design

Model and simulate your system

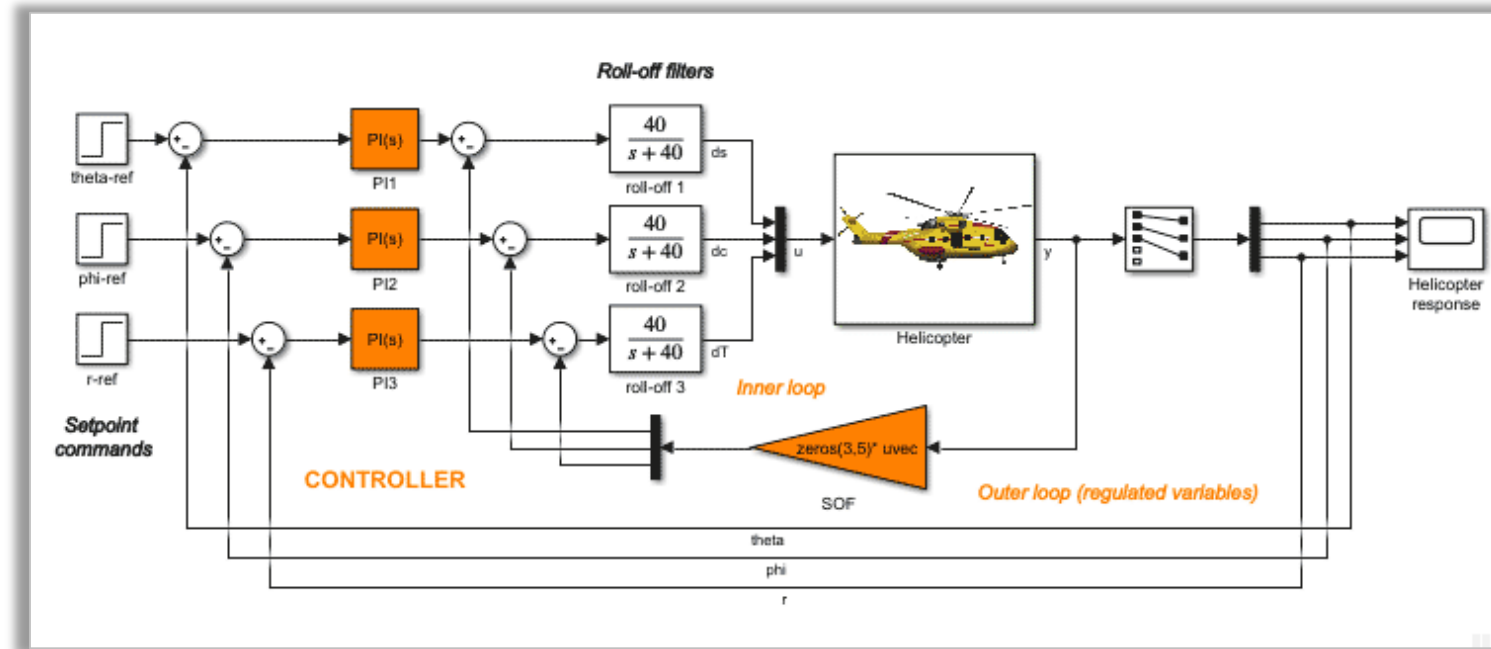
- Use one multi-domain environment
- Model the system under test and the plant
- Simulate closed-loop system behavior

Test early and often

- Test your system under all conditions
- Validate your design with real-time testing
- Trace from requirements to design to code

Automatically generate code

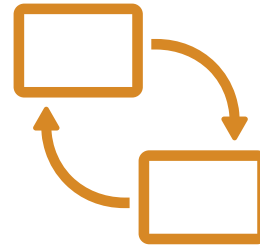
- Generate production-quality C and HDL code
- Deploy directly to embedded processors or FPGA's/ASIC's



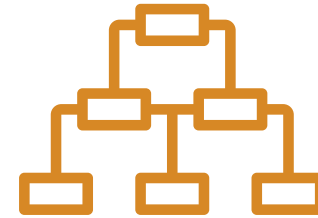
In this session, you will learn about new major Simulink capabilities to **help you work** more **effectively** and **efficiently**



Edit at the Speed of Thought



Model Run-Time Software



Componentize Your Design



Simulink Toolstrip

Access and discover Simulink capabilities when you need them

Can't find what you're looking for? Try [Apps](#) in Simulink or view [Menus to Toolstrip Mapping](#). Do not show again

Intake Airflow Estimation and Closed-Loop Correction

The diagram illustrates a control system for intake airflow estimation and correction. It features the following components and signals:

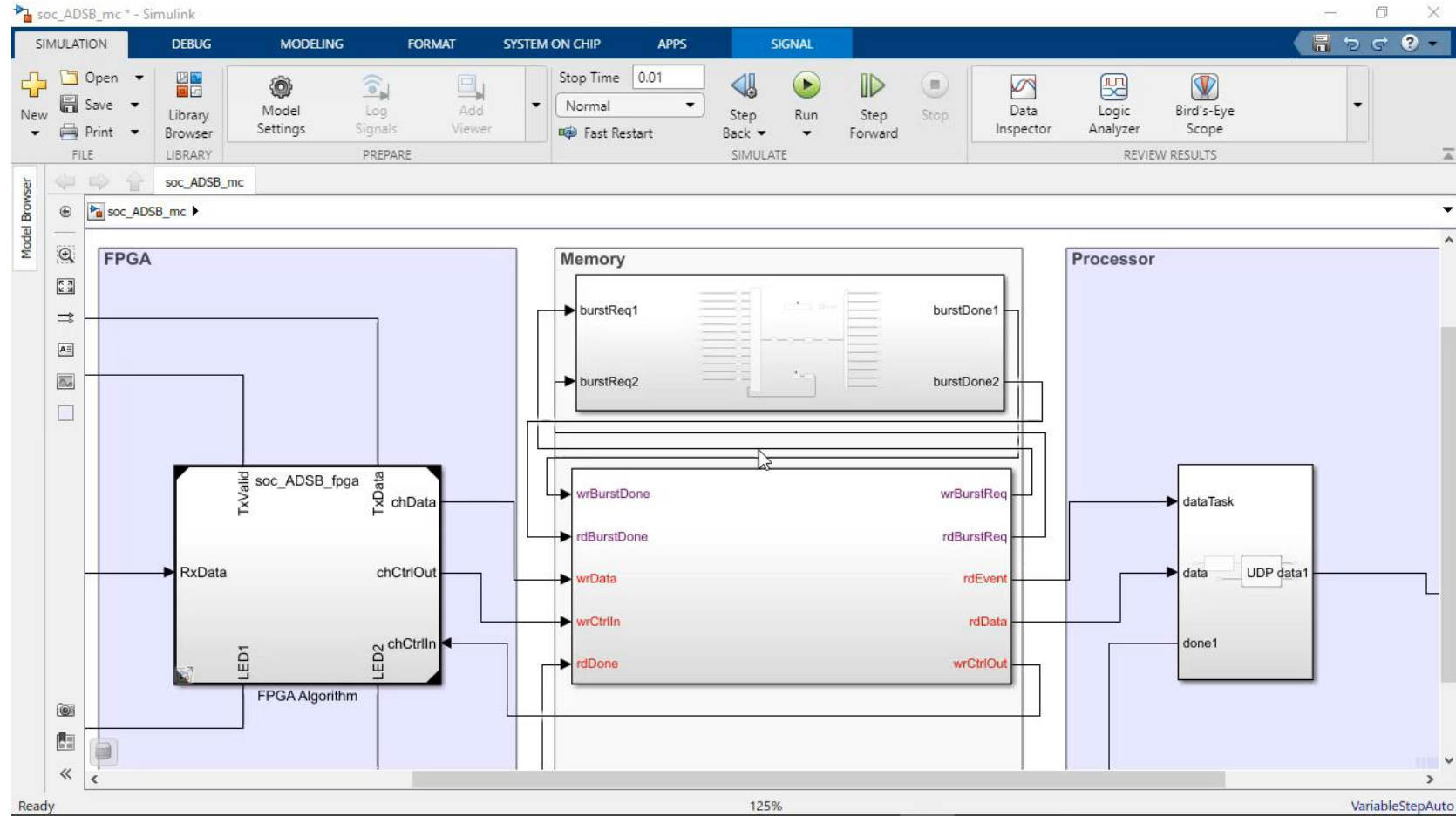
- Inputs:** `<throttle>` (deg), `<speed>` (rad/s), `<map>` (bar), `<ego>` (V), `O2_normal` (2), and `fuel_mode` (3).
- Key Blocks:**
 - Throttle Transient:** A transfer function block with numerator $.01z^{-1}$ and denominator $1-.8z^{-1}$.
 - Pumping Constant:** A 2-D T(u) block.
 - Ramp Rate Ki:** A 2-D T(u) block.
 - Feedback Control:** Includes an integrator (1/s), a gain block $\frac{K T_s}{z-1}$, and a switch.
- Outputs:** `est_airflow` (g/s) and `fb_correction`.
- Control Logic:** The system uses a combination of feedforward control (based on throttle and speed) and feedback control (based on oxygen sensor error `e0` and `e1`).



Edit at the Speed of Thought

Locate ports on any side, in any order, on Subsystems, Subsystem References, Model References and Stateflow charts

Flexible port placement



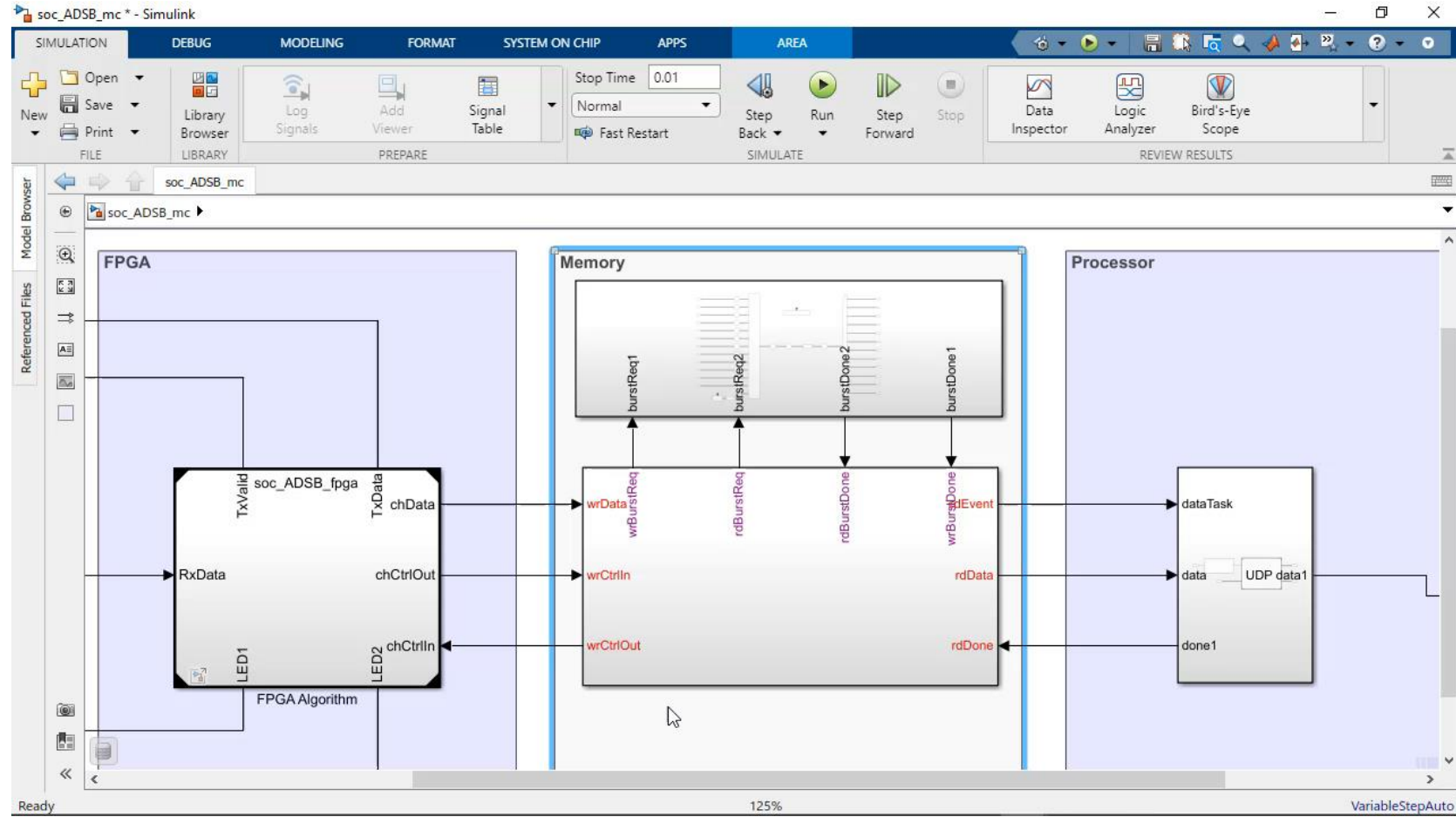


Edit at the Speed of Thought

Use the keyboard to select any combination of blocks and annotations in a diagram

Flexible port placement

Keyboard shortcuts





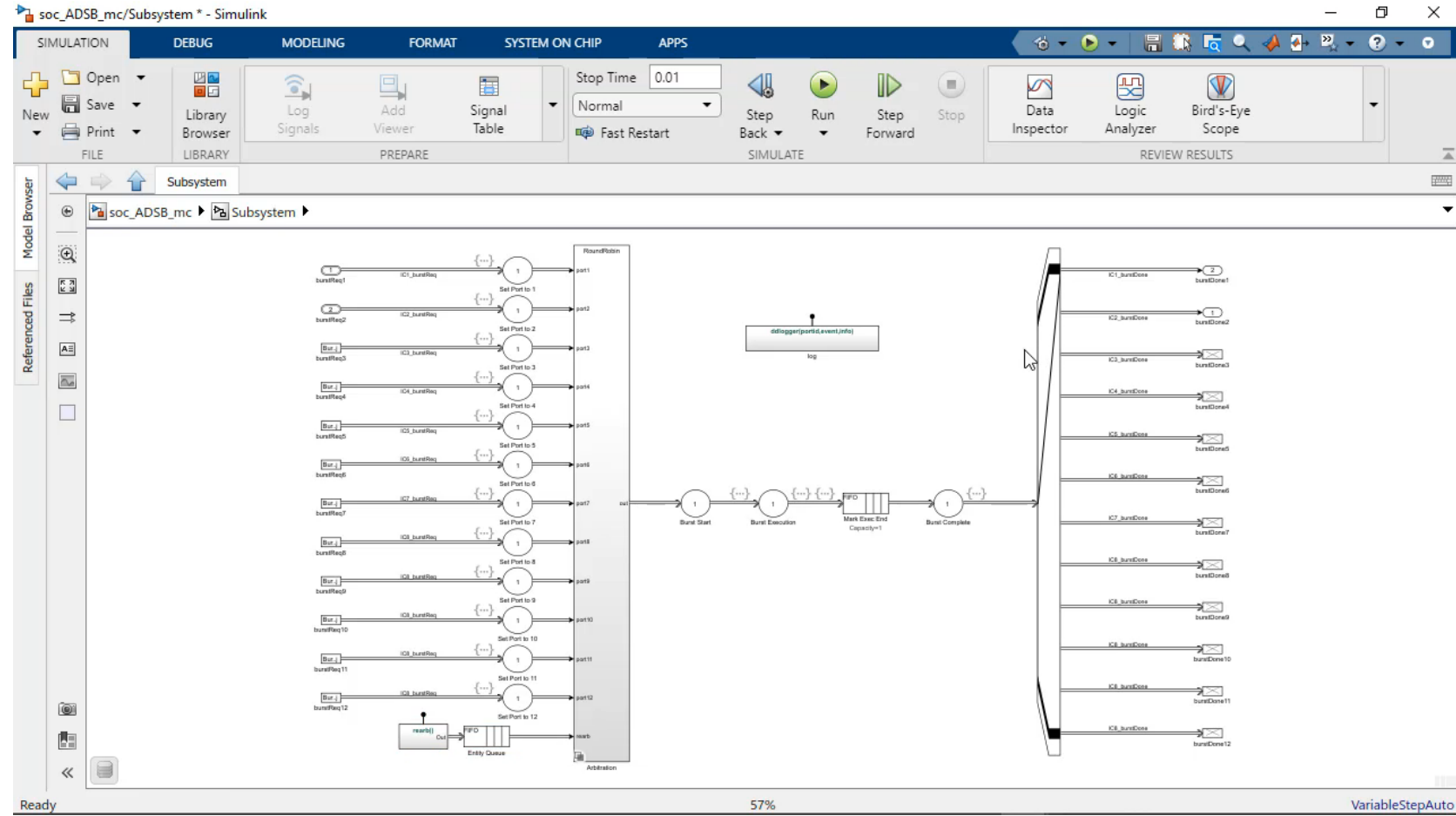
Edit at the Speed of Thought

Drag or paste text into Simulink Editor canvas to create annotations

Flexible port placement

Keyboard shortcuts

Drag text to annotate





Edit at the Speed of Thought

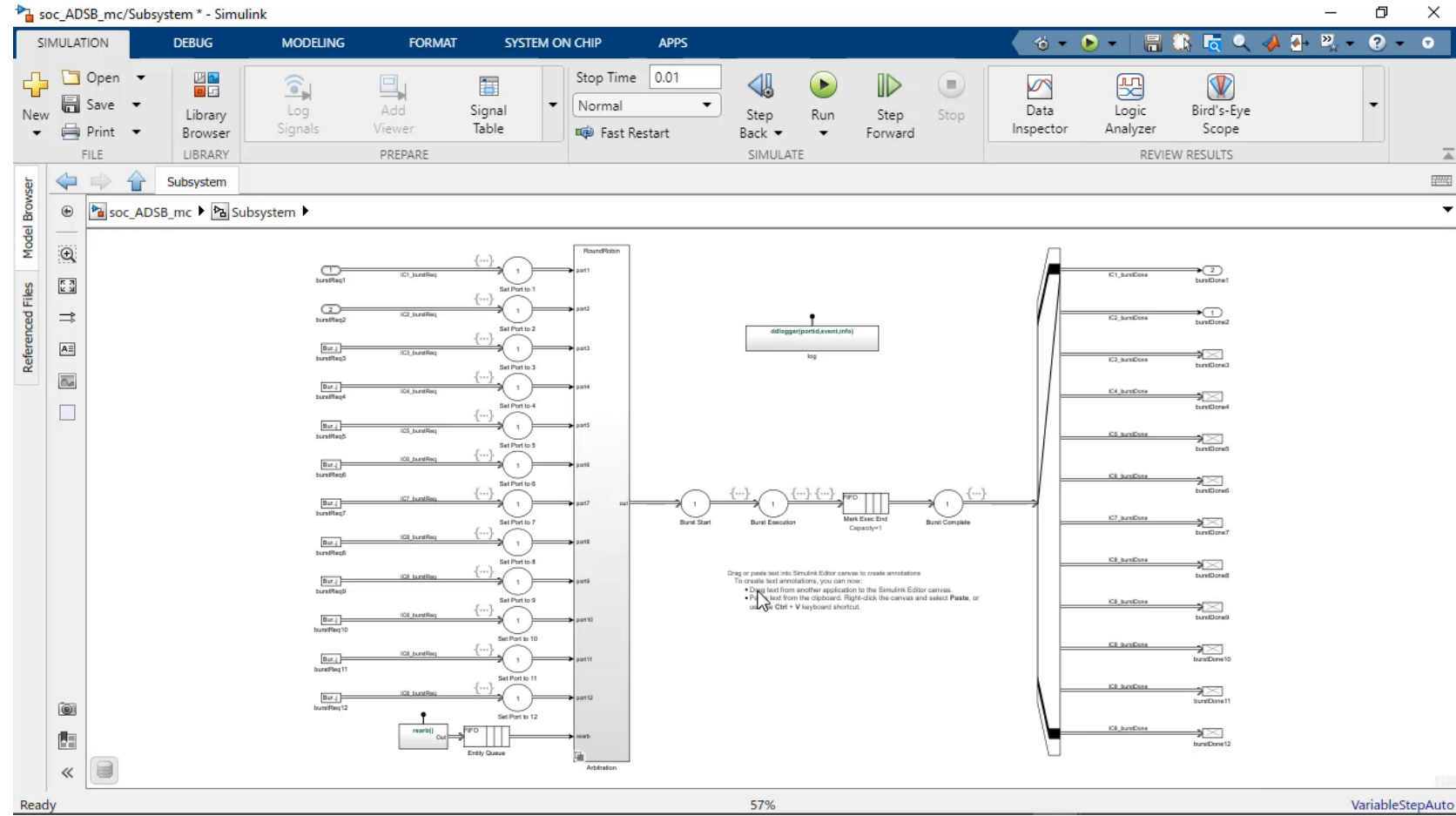
Optionally resize blocks to fit parameter values

Flexible port placement

Keyboard shortcuts

Drag text to annotate

Fit block size to content





Edit at the Speed of Thought

Create a port by clicking or dragging the block outline

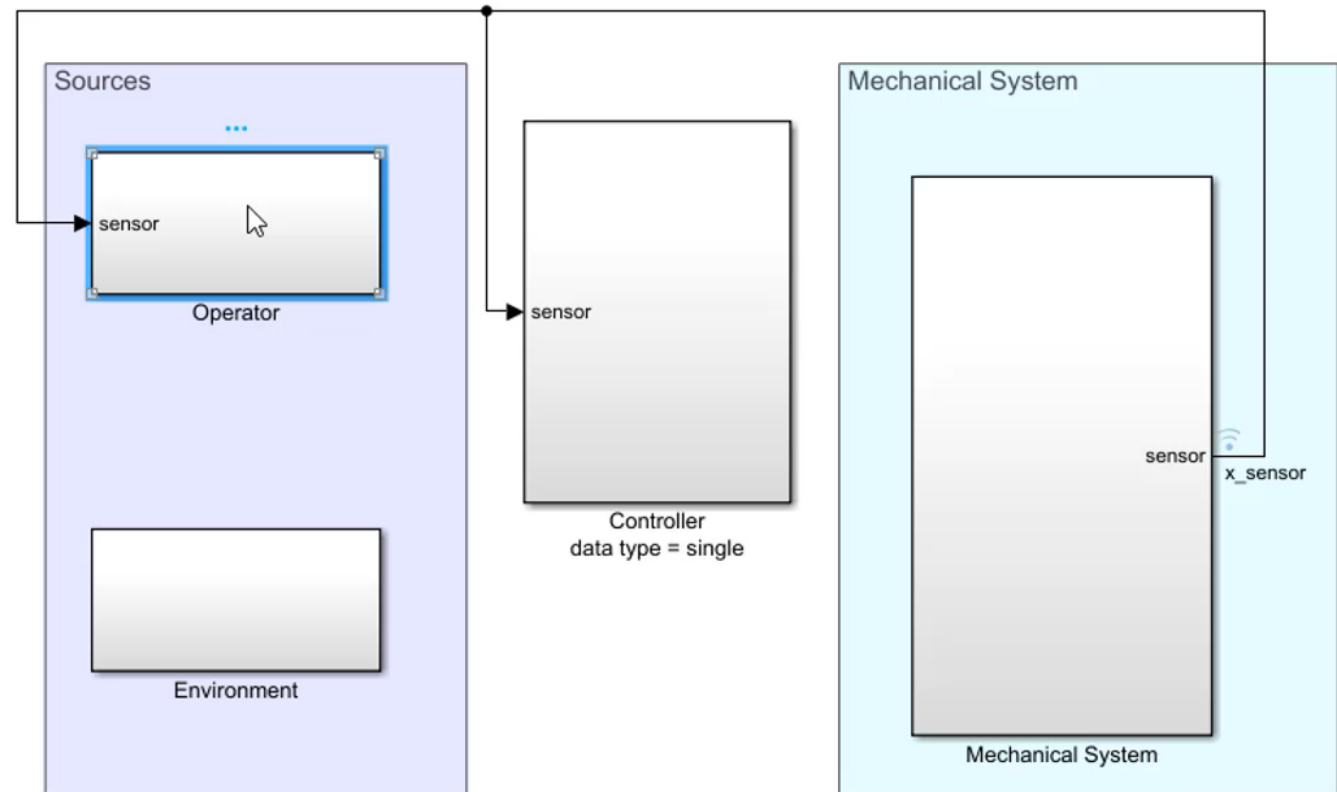
Flexible port placement

Keyboard shortcuts

Drag text to annotate

Fit block size to content

Automatic port creation





Edit at the Speed of Thought

Modify block parameters without opening a dialog box

Flexible port placement

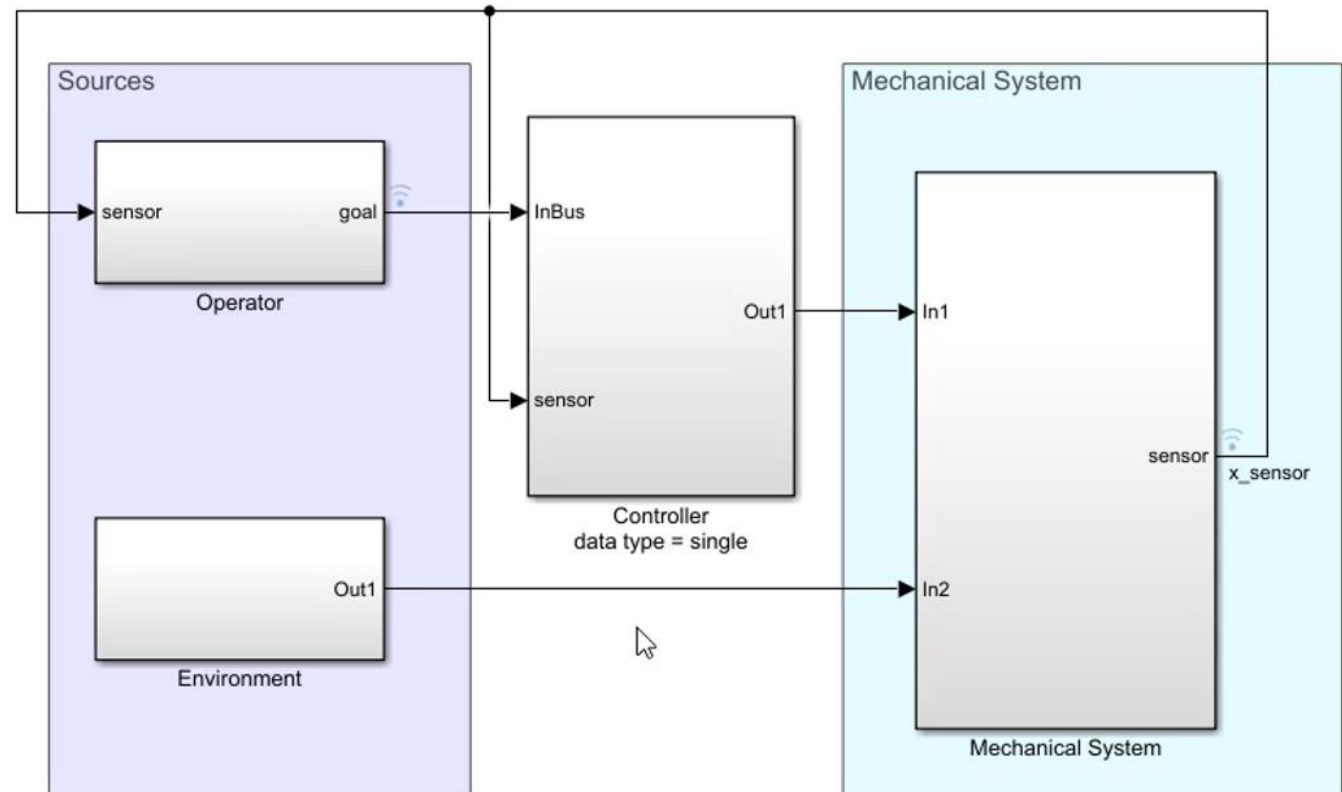
Keyboard shortcuts

Drag text to annotate

Fit block size to content

Automatic port creation

Edit on block icon





Edit at the Speed of Thought

Improve the speed and accuracy of block parameter editing by selecting from variable or function names as you type

Flexible port placement

Keyboard shortcuts

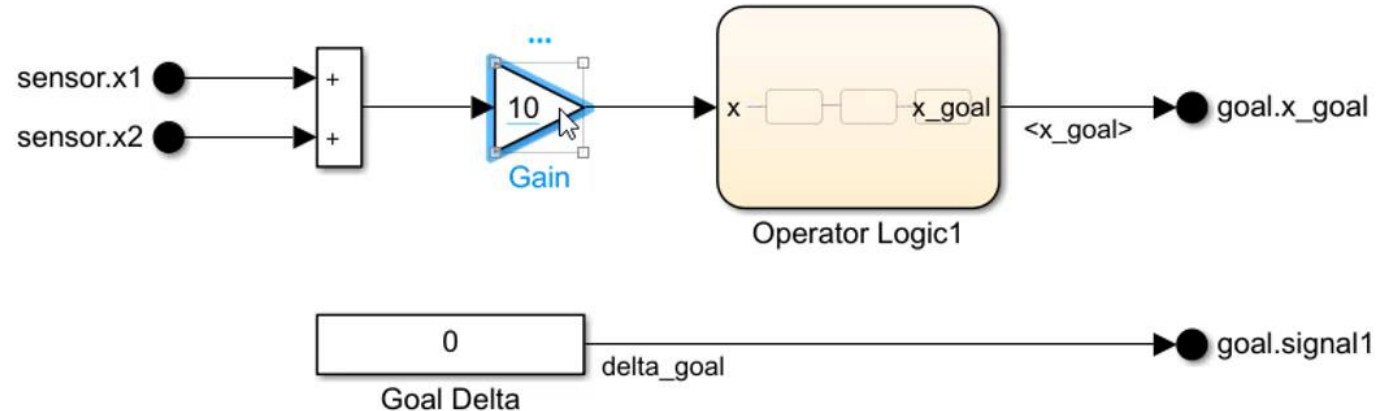
Drag text to annotate

Fit block size to content

Automatic port creation

Edit on block icon

Block parameter autocomplete





Edit at the Speed of Thought

Open referenced models in the same window as their parent models

Flexible port placement

Keyboard shortcuts

Drag text to annotate

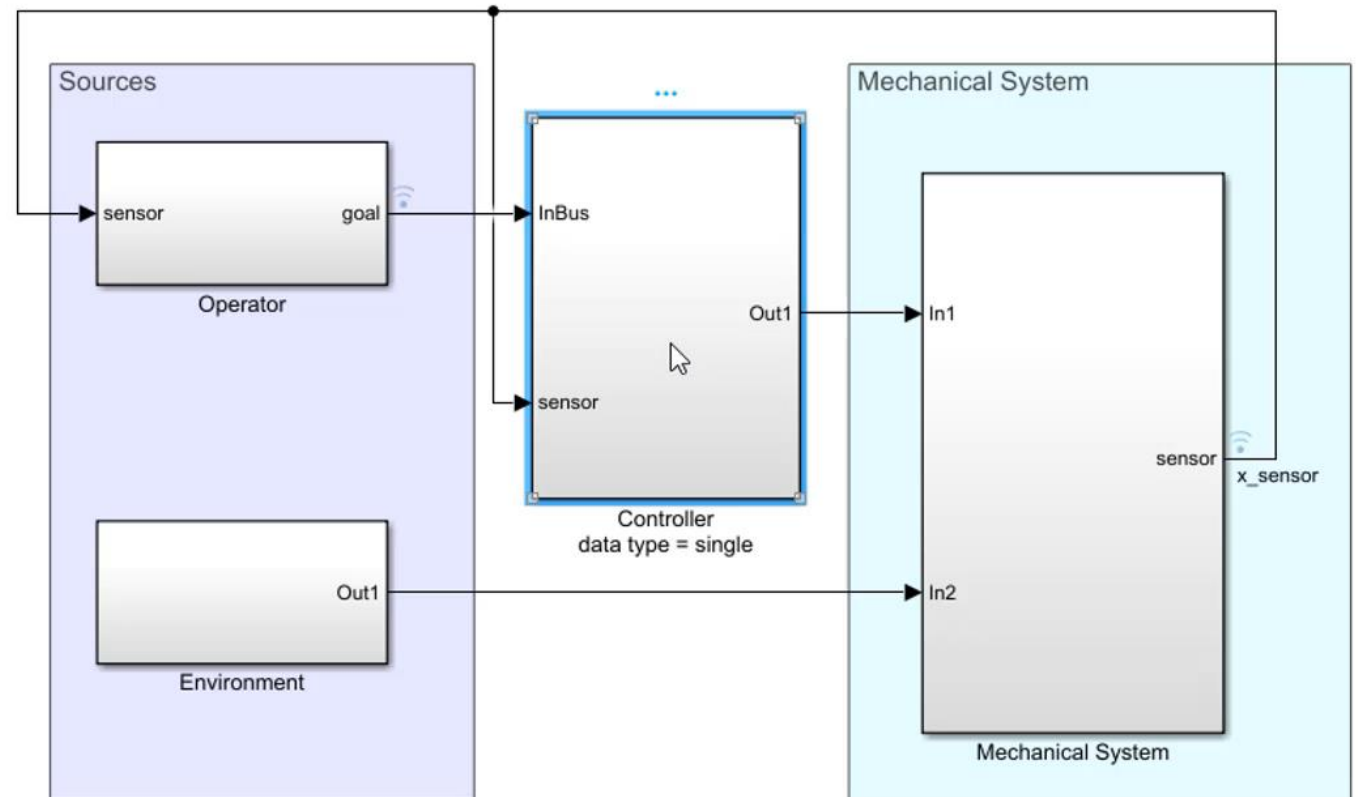
Fit block size to content

Automatic port creation

Edit on block icon

Block parameter autocomplete

Model reference in same window





Edit at the Speed of Thought

Connect a recommended block to an existing block in your model, sorted by frequency of use

Flexible port placement

Keyboard shortcuts

Drag text to annotate

Fit block size to content

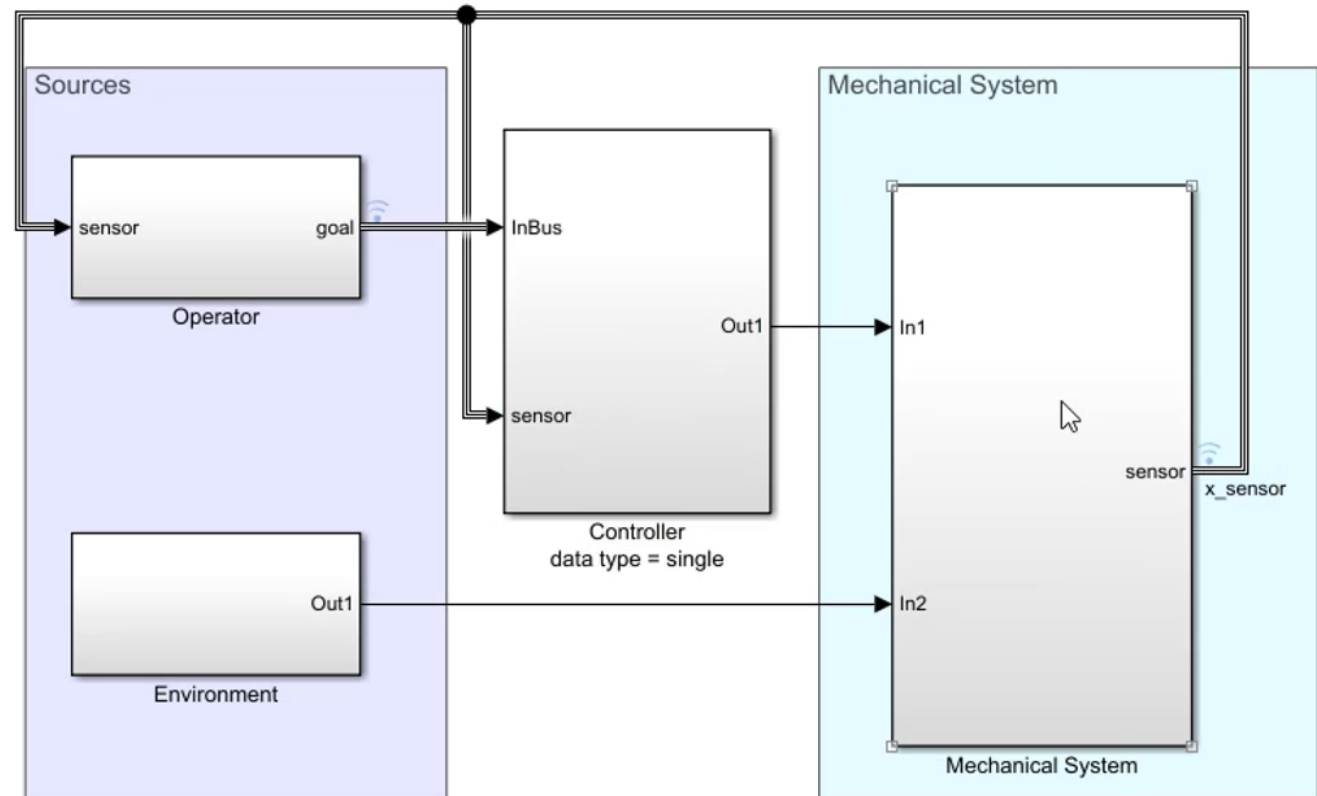
Automatic port creation

Edit on block icon

Block parameter autocomplete

Model reference in same window

Predictive quick insert

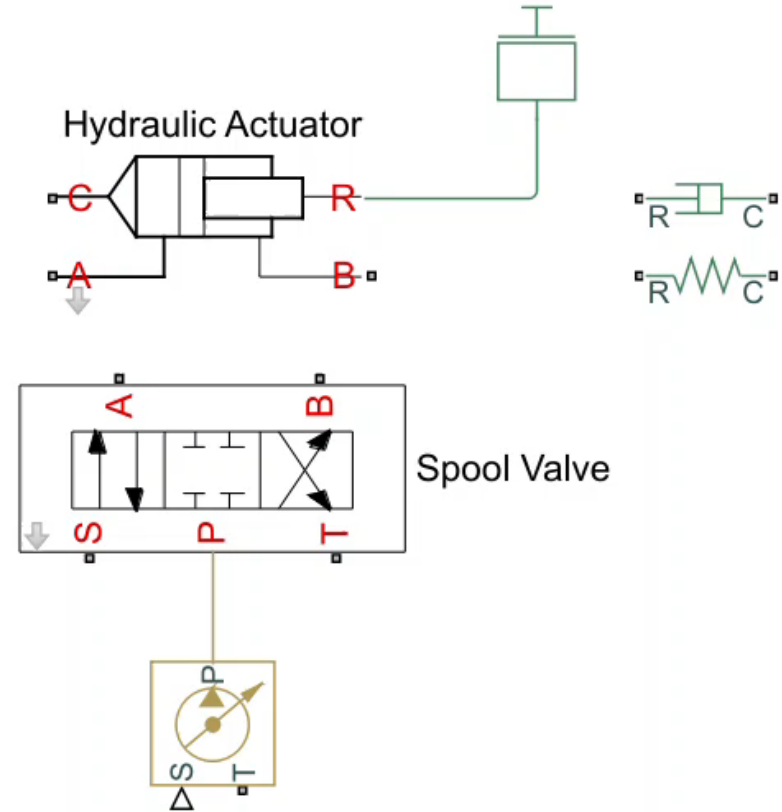


Edit at the Speed of Thought

Highlight compatible ports as you draw a signal line

...

Port connection cues





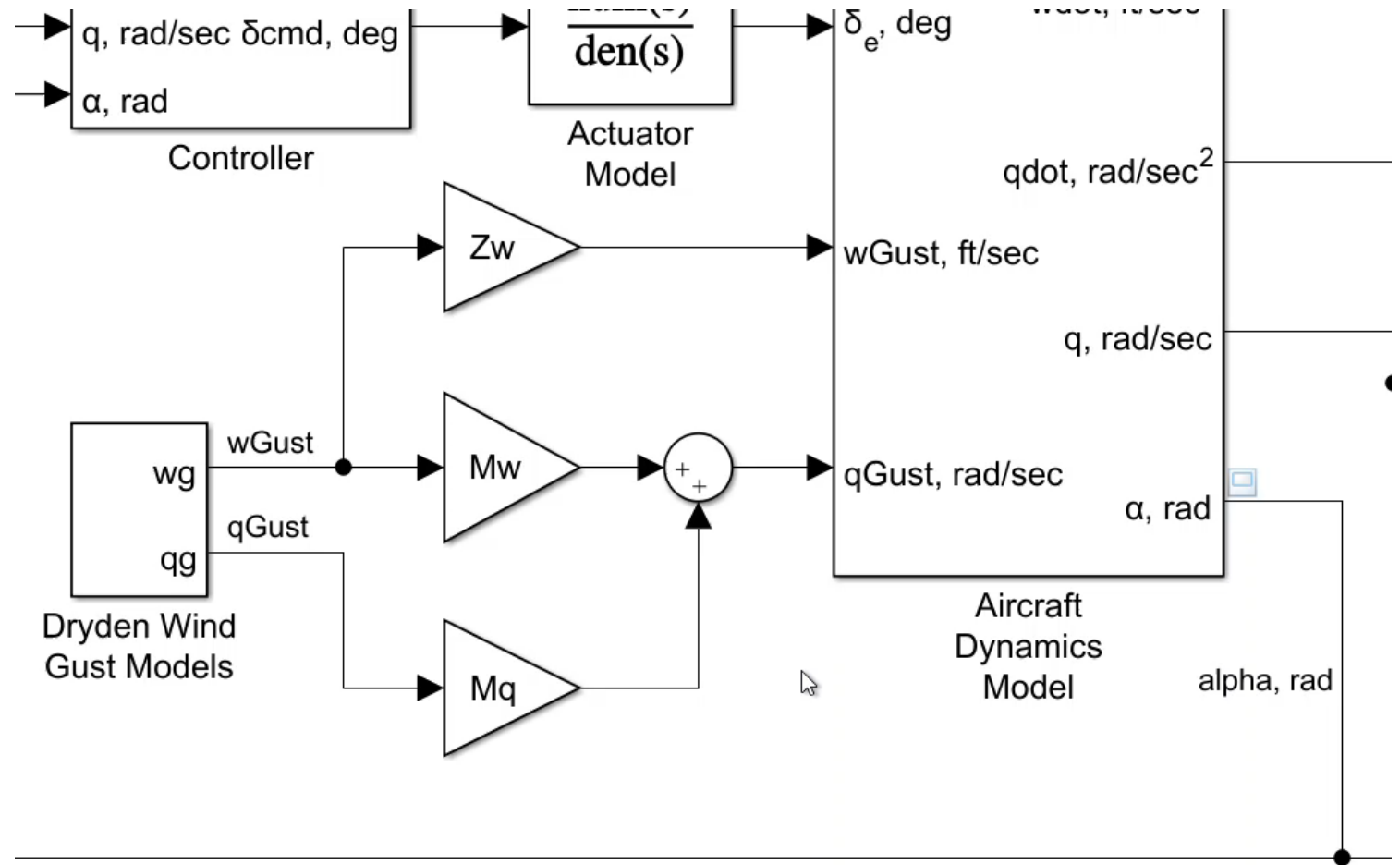
Edit at the Speed of Thought

Get instant notifications about missing variables while editing your model

...

Port connection cues

Missing variable detection





Edit at the Speed of Thought

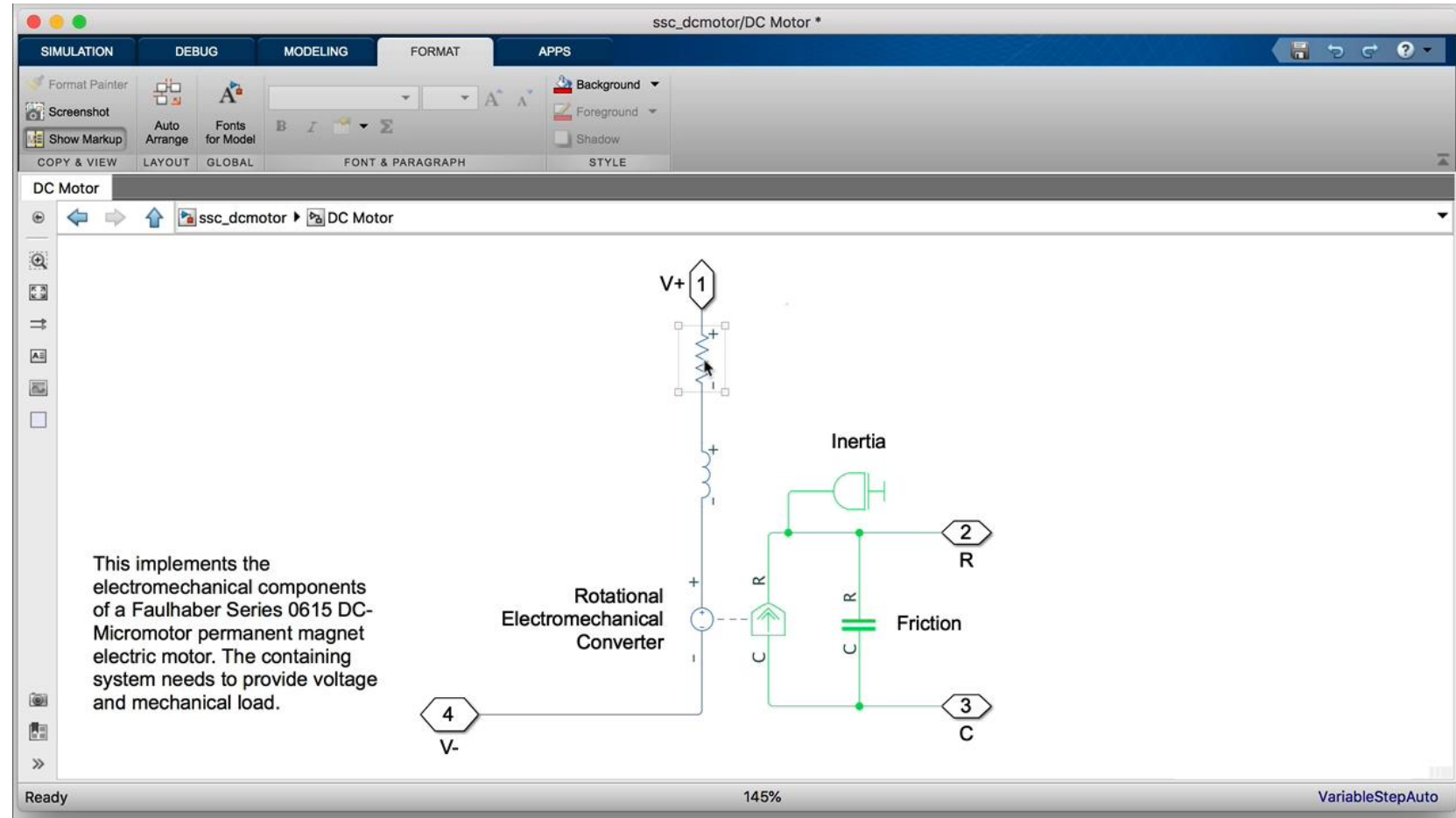
Rotate selected blocks as a group and evenly place blocks in suggested zones

...

Port connection cues

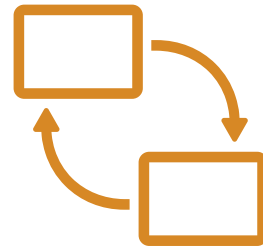
Missing variable detection

Group rotation and smart guides

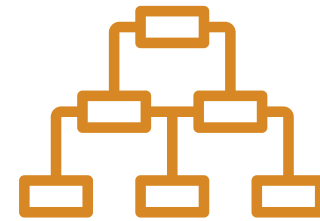




Edit at the Speed of Thought



Model Run-Time Software



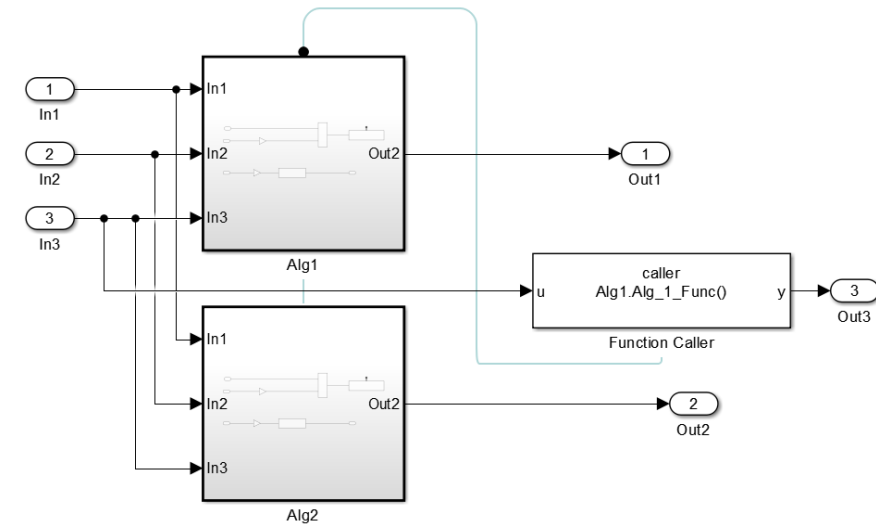
Componentize Your Design



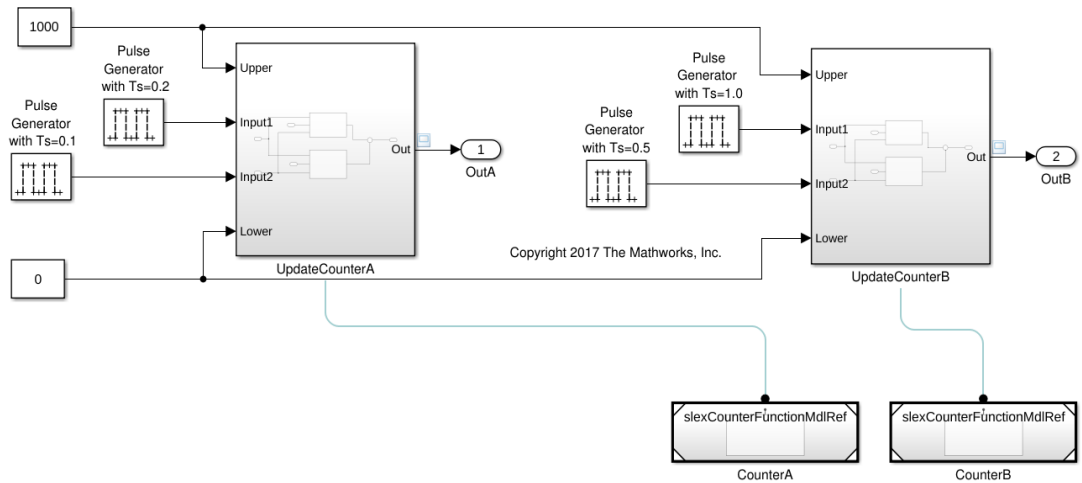
New and enhanced ways to model software components

Scoped Simulink Functions

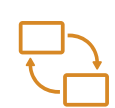
Call Simulink Function blocks within a subsystem hierarchy



Modeling Reusable Components using Multiply Instanced Simulink Functions



Create Simulink Functions that can cross model boundaries for reusable software components



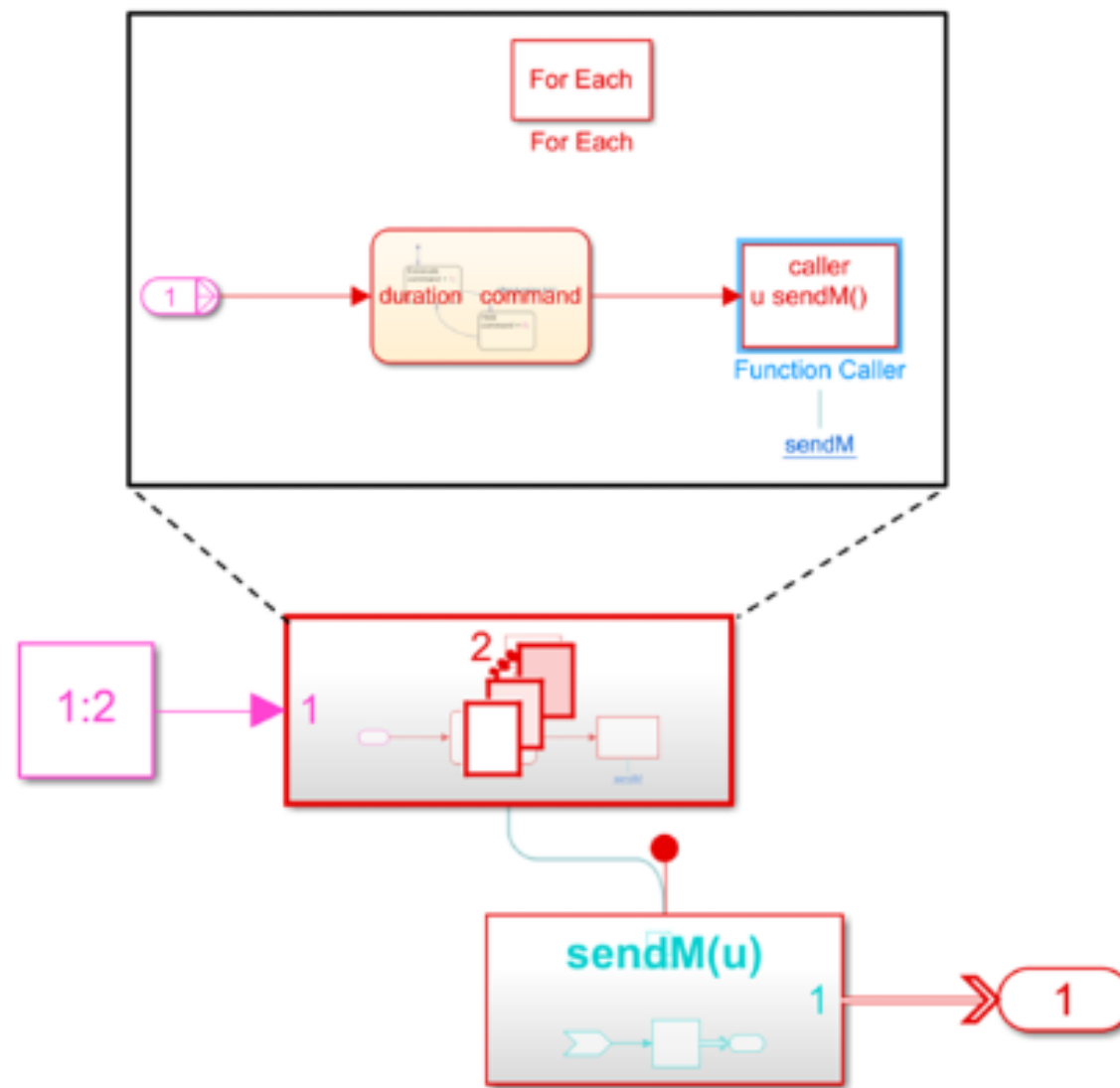
New and enhanced ways to **model software components**

Call a Simulink function from repeated executions (For Each subsystem)

Use Function Caller block inside a For Each subsystem block to call a utility

Utility can be defined in a Simulink function outside the subsystem (parent model)

Enables sending and receiving messages from within a For Each iteration



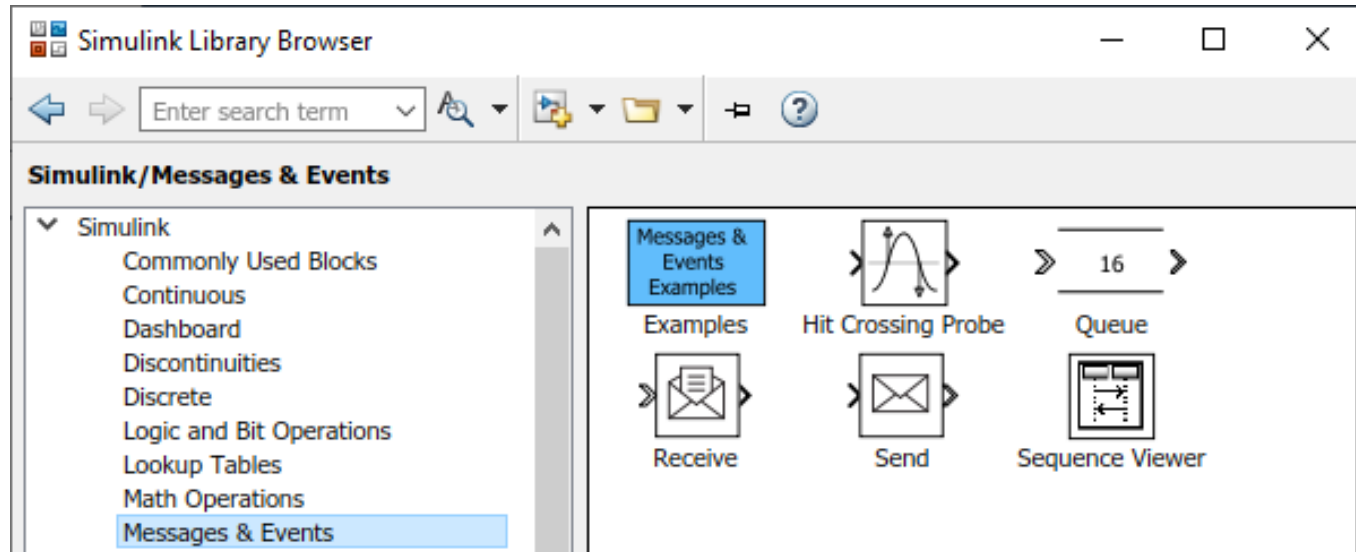
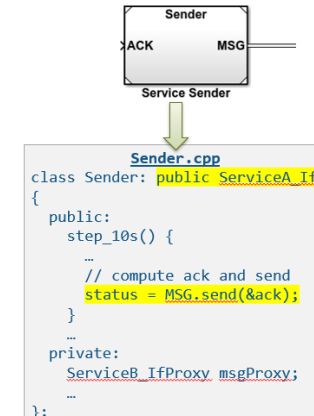


Simulink Messages simplify software component integration

Messages: Model and generate C++ code for software compositions with message-based communication

Integration of software components through middleware is an important and common task

Middleware uses messages (Send/Receive/Async Comm)



New library in Simulink

Messages in Stateflow

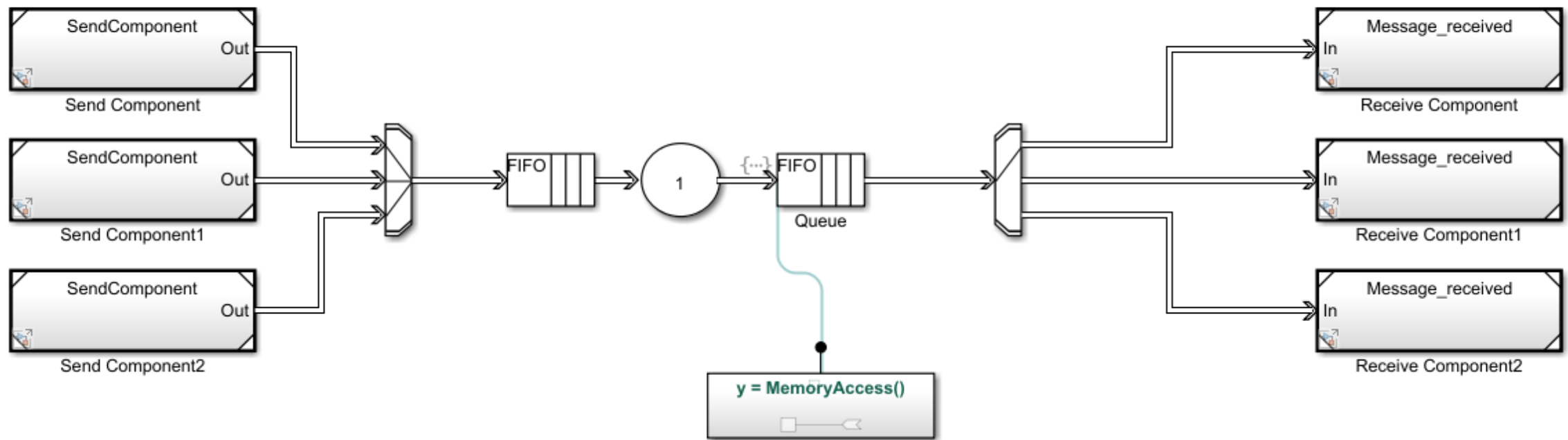
Can send Messages to model root



Simulink Messages simplify software component integration

Messages: Model and generate C++ code for software compositions with message-based communication

Stateflow and SimEvents work seamlessly with Messages



SimEvents used for routing, delaying, broadcasting, replication, gating

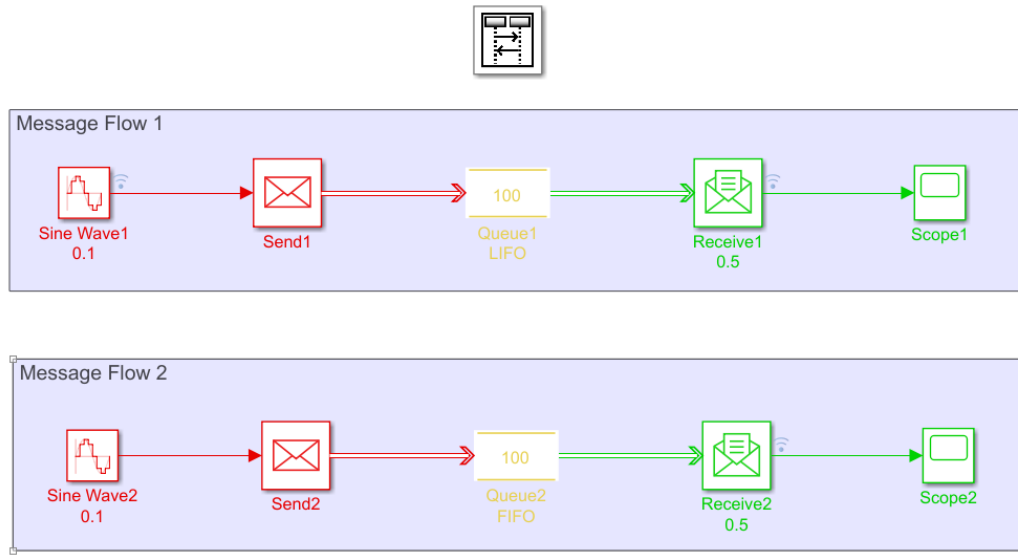
Stateflow used for handshaking, state management, logic



Simulink Messages simplify software component integration

Messages: Model and generate C++ code for software compositions with message-based communication

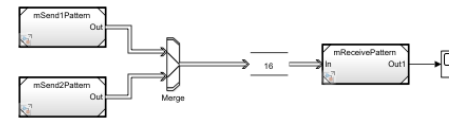
Examples



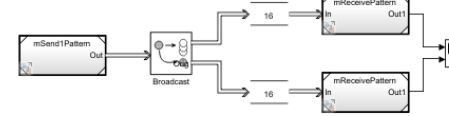
>> QueueSortingPoliciesModel

Modeling Message Communication Patterns with SimEvents

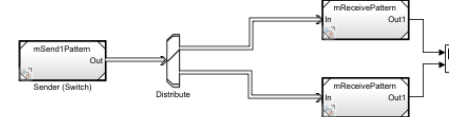
Merge messages from multiple senders



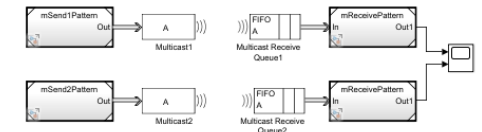
Broadcast messages to multiple receivers



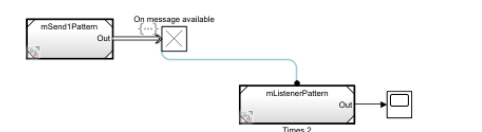
Distribute work to multiple receivers



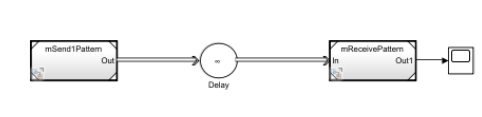
Multicast messages among multiple senders and multiple receivers



Run based on message availability



Delay messages for a set amount of time



>> slxMessagePatterns



New interfaces to **visualize** and **control** component execution and communication

Schedule Editor: Explicitly schedule the execution of your model components

The screenshot displays the Schedule Editor window for a model named 'ThrottlePositionControlTop'. The interface includes a toolbar with various actions like 'Manage Partitions', 'Execution Order', 'Move Up/Down', 'Update Diagram', 'Save Model', 'Highlight', 'Arrange', 'Timing Legend', and 'Layout'. The main workspace shows a graph of components and their dependencies. A central component 'Cont' has two partitions: 'D1' (red) and 'D3' (blue). 'D3' is selected, and its properties are shown in the Property Inspector on the right: Name: D3, Rate: 0.01, Type: Implicit periodic partition. The Execution Order table on the right lists the following components and their rates:

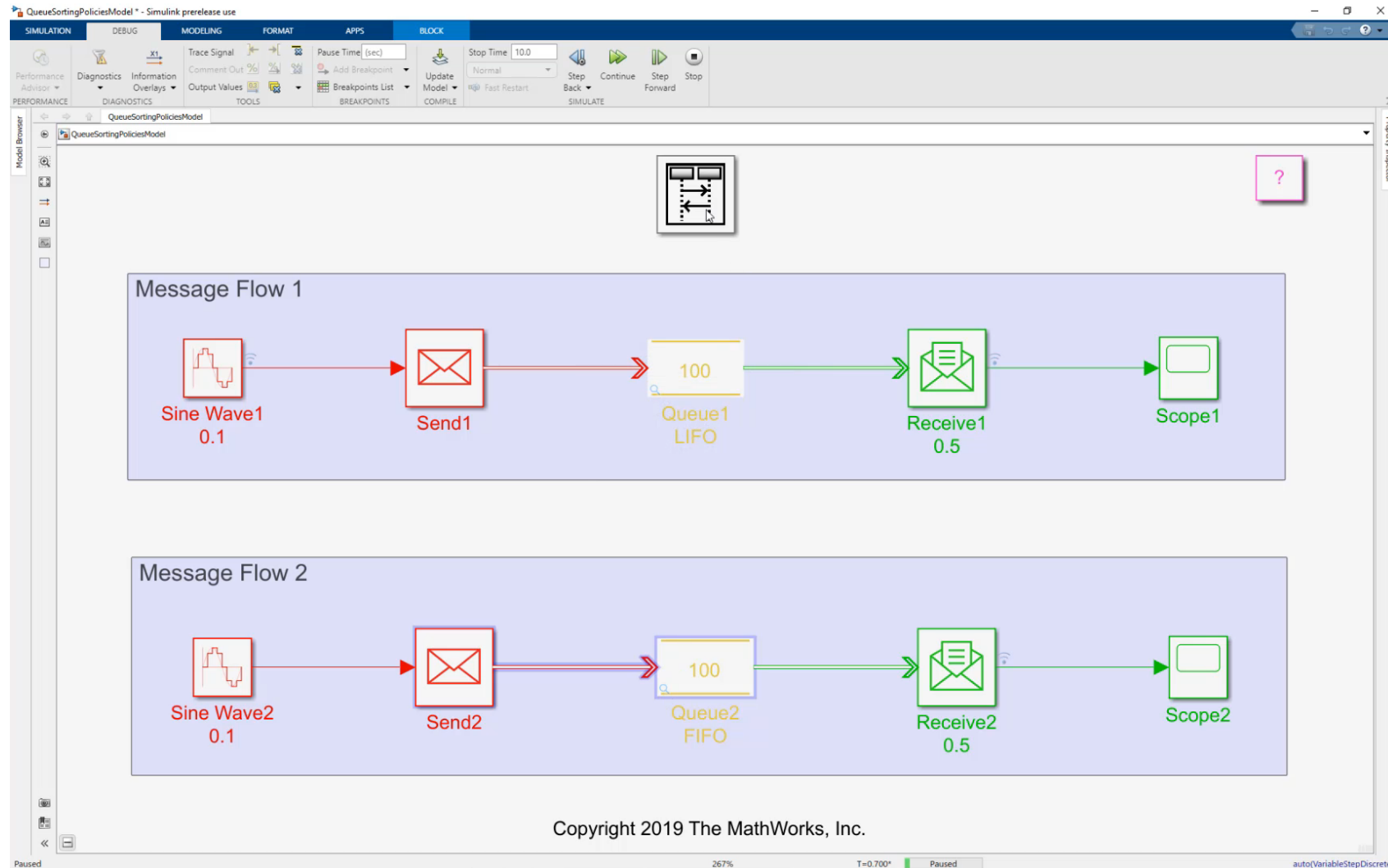
Order	Name	Rate
1	Cont	0
2	D1	0.001
3	D2	0.005
4	ThrottleControl.APPSnsrRun	-1
5	ThrottleControl.ActuatorRun5ms	0.005
6	ThrottleControl.TPSSecondaryRun5ms	0.005
7	ThrottleControl.MonitorRun5ms	0.005
8	D3	0.01
9	ThrottleControl.TPSPrimaryRun10ms	0.01

The graph shows dependencies between components. Solid arrows indicate dependencies, while dashed arrows indicate other relationships. Components include: 'ThrottleControl.ActuatorRun5ms' (0.005), 'ThrottleControl.TPSSecondaryRun5ms' (0.005), 'ThrottleControl.MonitorRun5ms' (0.005), 'ThrottleControl.ControllerRun5ms' (0.005), 'ThrottleControl.TPSPrimaryRun10ms' (0.01), and 'ThrottleControl.APPSnsrRun' (-1). The 'Cont' component has a rate of 0 and is connected to 'D1' (0.001) and 'D3' (0.01).



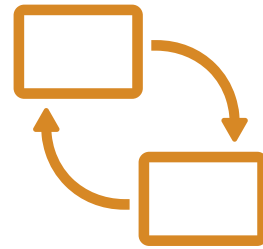
New interfaces to **visualize** and **control** component execution and communication

Sequence Viewer: Visualize function calls, message communication, and Stateflow state changes and event activity

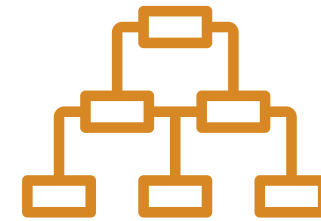




Edit at the Speed of Thought



Model Run-Time Software



Componentize Your Design



Protected Models



Projects

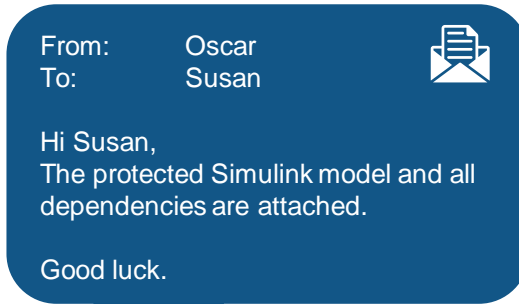
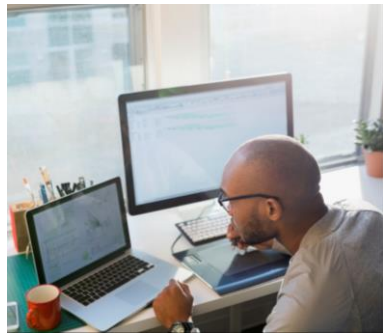


Standalone Apps
Simulink Compiler



Sharing protected models is easy with the latest release

OEM (Oscar)



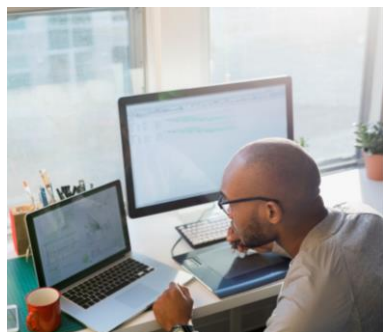
Create an *.mlproj* that contains:

- *.slxp*
- Enumerated types
- Needed variables
- Data dictionaries
- Harness model
- ...



Sharing protected models is easy with the latest release

OEM (Oscar)



From: Oscar
To: Susan

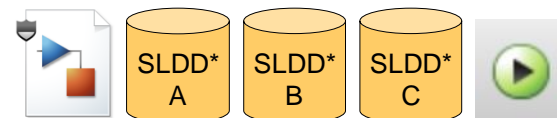
Hi Susan,
The protected Simulink model and all dependencies are attached.

Good luck.

From: Susan
To: Oscar

Awesome!
I'm able to run it.

Supplier (Susan)



Create an `.mlproj` that contains:

- `.slxp`
- Enumerated types
- Needed variables
- Data dictionaries
- Harness model
- ...

The recipient gets one file that contains everything needed to simulate the protected model

HOME PLOTS APPS

Search Documentation Ed

New Script New Live Script New Open Compare Import Data Save Workspace Clear Workspace Analyze Code Run and Time Clear Commands Simulink Layout Preferences Set Path Parallel Add-Ons Help Community Request Support Learn MATLAB RESOURCES

FILE VARIABLE CODE SIMULINK ENVIRONMENT RESOURCES

C: > Work > R2020b > NonlinearActuator

Current Folder

- Name
- buses.sldd
- controller.sldd
- NonLinearActuator.slx
- system_model.sldd

Details

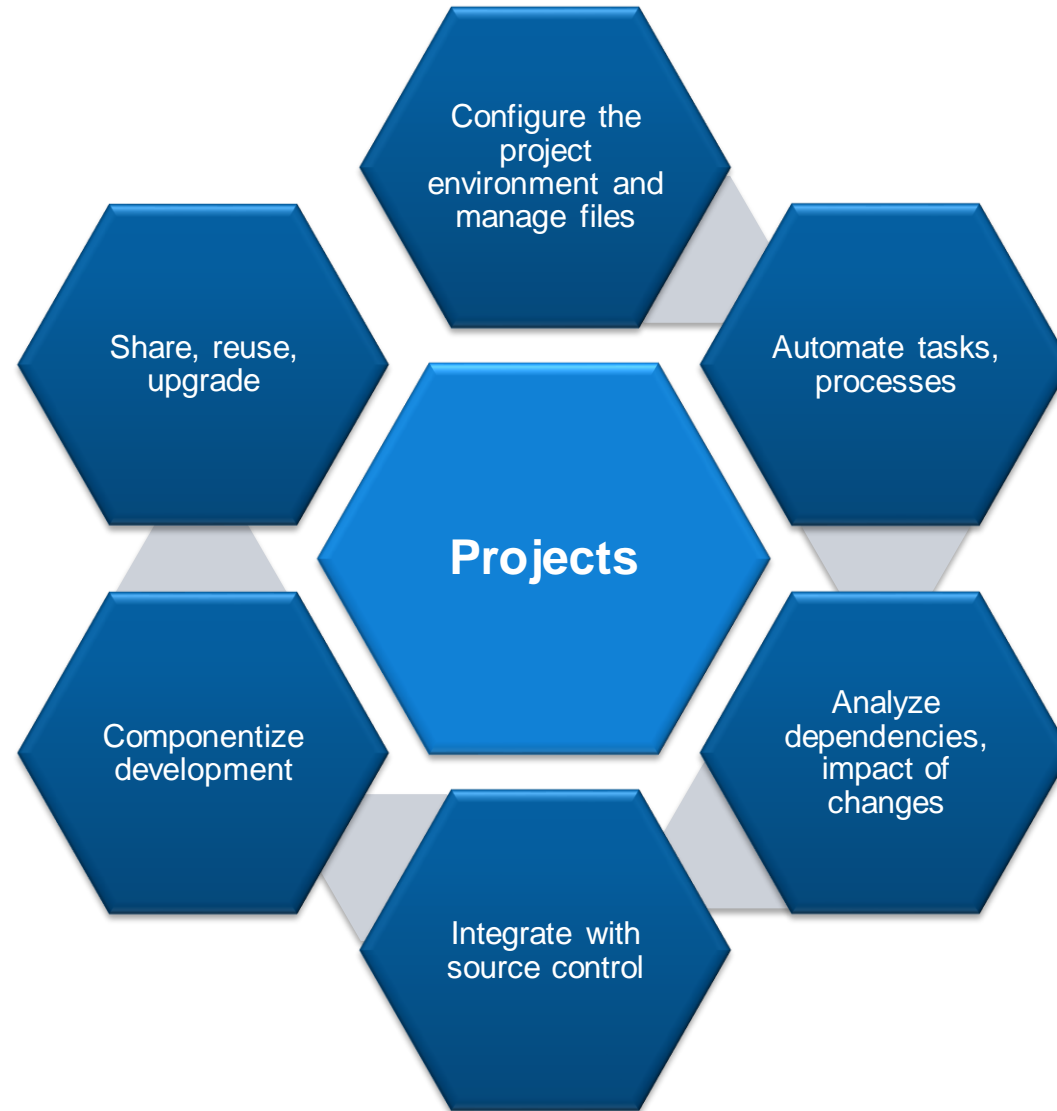
Command Window

```
fx >> |
```




Projects

Organize, manage, and share your work



Share, reuse, and upgrade your project files



Project Compatibility: Export a complete Project to a previous release with just one function

Command Window

```
fx >> Simulink.exportToVersion(currentProject, 'asbhl20_17a.zip', 'R2017a')
```

Additionally converts project metadata for previous versions

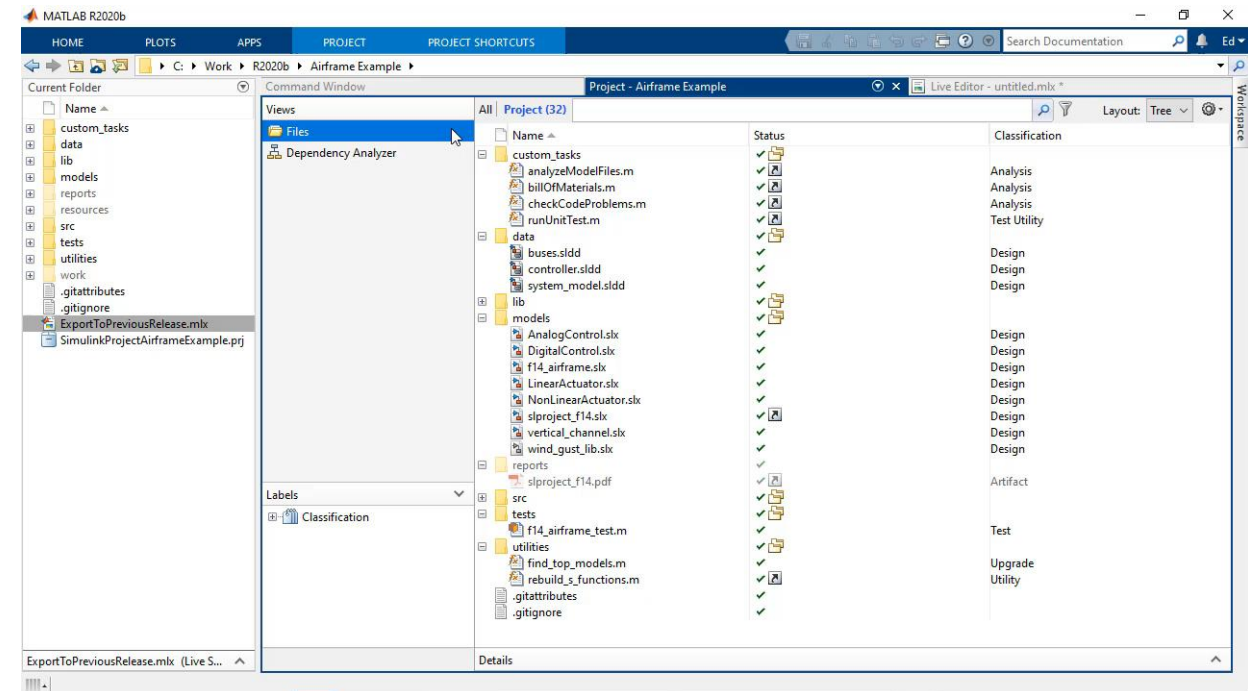
- Including “resources/project” → “.SimulinkProject”

Accessed via same API as Simulink model export

- Clear alignment with Model-Based Design workflows

Supports 7 years of previous releases

Exports the currently loaded top-level project. Does not export referenced projects



Share, reuse, and upgrade your project files

Project Compatibility: Export a complete Project to a previous release with just one function



The screenshot shows the MATLAB R2020b interface with a project named 'Airframe Example'. The 'Current Folder' pane on the left shows a tree view of the project structure, including folders like 'custom_tasks', 'data', 'lib', 'models', 'reports', 'resources', 'src', 'tests', 'utilities', and 'work'. The 'Command Window' pane is active, showing a list of files and their classifications. The 'ExportToPreviousRelease.mlx' file is highlighted in the 'Current Folder' pane.

Name	Status	Classification
analyzeModelFiles.m	✓	Analysis
billOfMaterials.m	✓	Analysis
checkCodeProblems.m	✓	Analysis
runUnitTest.m	✓	Test Utility
buses.sldd	✓	Design
controller.sldd	✓	Design
system_model.sldd	✓	Design
AnalogControl.slx	✓	Design
DigitalControl.slx	✓	Design
f14_airframe.slx	✓	Design
LinearActuator.slx	✓	Design
NonLinearActuator.slx	✓	Design
slproject_f14.slx	✓	Design
vertical_channel.slx	✓	Design
wind_gust_lib.slx	✓	Design
slproject_f14.pdf	✓	Artifact
f14_airframe_test.m	✓	Test
find_top_models.m	✓	Upgrade
rebuild_s_functions.m	✓	Utility



Integrate with source control

Source Control Integration



The screenshot displays the MATLAB IDE interface for a project named 'Project - StreamingPumpDemo'. The 'SOURCE CONTROL' ribbon is active, showing various source control operations. A callout box provides a detailed view of these operations:

- Git Details**: Represented by a database icon.
- Refresh**: Represented by a circular arrow icon.
- Commit**: Represented by a database icon with an upward arrow.
- Fetch**: Represented by a downward arrow pointing to a database icon.
- Push**: Represented by an upward arrow pointing to a database icon.
- Pull**: Represented by a downward arrow pointing to a database icon with an upward arrow.
- Remote**: Represented by a database icon.
- Branches**: Represented by a branching diagram icon.

The main interface shows the 'PROJECT SHORTCUTS' ribbon with options like 'New', 'Open', 'Add Files', 'Unsaved Changes', 'Share', 'Search', 'Custom Tasks', 'Run Checks', 'References', 'Details', 'Project Path', and 'Startup Shutdown'. The 'Views' pane shows 'Files' and 'Dependency Analysis'. The 'Labels' pane shows 'Classification'. The file list includes folders like '+Test', 'ACI', 'Dashboard', 'Documents', 'Elasticsearch', 'MachineLearning', 'MATLAB_Kafka_Producer_Java', 'mps_stream', 'SimExecutable', and 'Simulation', along with files like 'DocExample_MultiClassFaultDetect', 'genPumpData.m', 'javasetup.m', 'Main_ExampleWorkflow.mlx', 'MLModels.mat', 'rawdata.mat', 'README.md', and 'resetFakeInfo.m'.

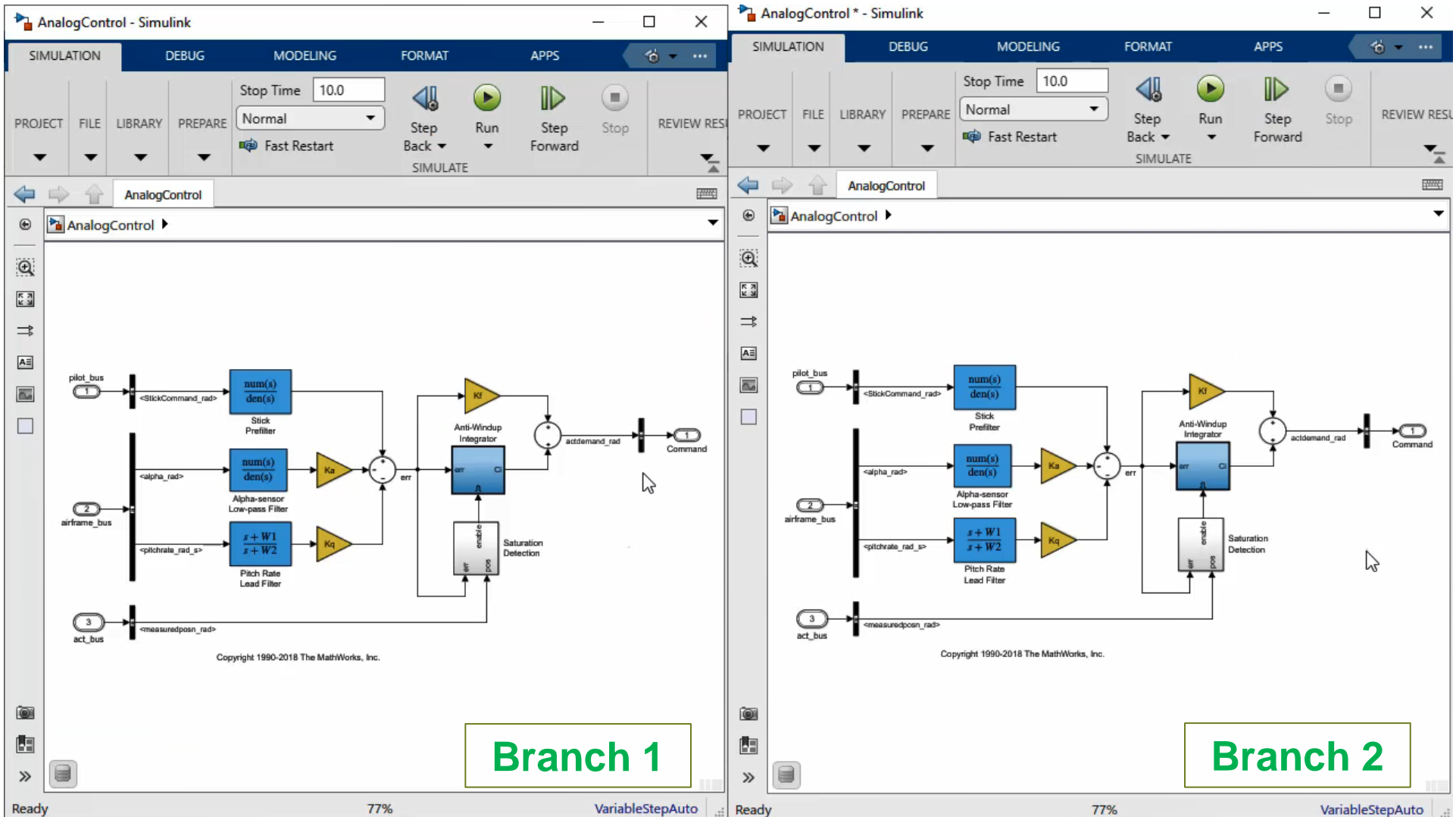


Integrate with source control

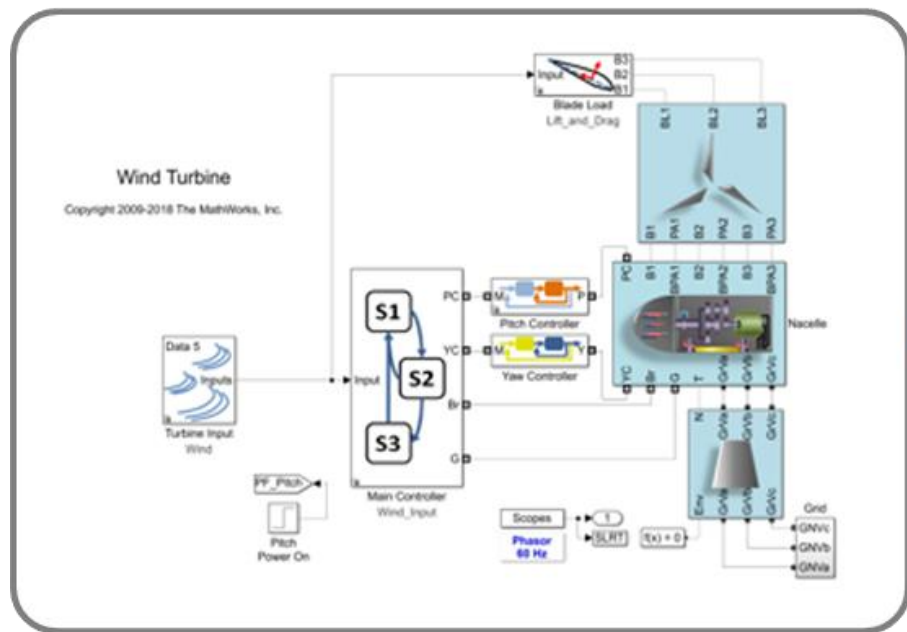
Automerge



Automatically merge branches that contain changes in different subsystems in the same SLX file



The need for simulation deployment



Simulink Simulations



Enabling collaborators, suppliers and clients to access insights from simulation



Leveraging simulations for in-operation usage, such as digital twin



Interchanging with other simulation environments



Simulink Compiler is an out-of-the-box solution to share Simulink simulations

Simulink Compiler
Share simulations as standalone executables, web apps,
and Functional Mockup Units (FMUs)

[Download a free trial](#)

The banner features a background image of a Simulink model for a mass-spring-damper system. The model includes a mass m , a spring with stiffness k , and a damper with damping coefficient c . The system is connected to a controller block labeled 'Cont'. The outputs of the system are labeled v_sc (velocity) and p_sc (position). The parameter list on the right includes Mass (Kg), Stiffness (N/m), Damping (N/m/s), and Initial Position (m).

Support a variety of Simulink simulation features including variable-step solvers

Support flexible simulation inputs / parameter tuning

Royalty-free distribution



Simulink Compiler supports multiple deployment scenarios

Simulink Compiler

Share simulations as standalone executables, web apps, and Functional Mockup Units (FMUs)

[Download a free trial](#)

Mass (Kg)
Stiffness (N/m)
Damping (N/m/s)
Initial Position (m)

v_sc
Cont
p_sc

Standalone Apps

Values

Time, sec

Connection
Port: COM3
Connect Disconnect
Fix and refresh

RED LED GREEN LED BLUE LED

Off On Off On Off On

40 50 60 70 80 90 100

0 10 20 30 40 50 60 70 80 90 100

0 10 20 30 40 50 60 70 80 90 100

Illuminance 93

RGB

Web Apps

File Edit View Navigate Tools Window Help

PieChart Window III

BarChart Window III

Core Window III

Year	2007	2008	2009
Value 1	967	998	1,154
Value 2	1,292	1,661	1,827
Value 3	1,292	2,589	2,774

Service APIs

Zone NYISOP CapIt

Generate Forecast Model Diagnostics Report

Comparison

Zone: CAPITL

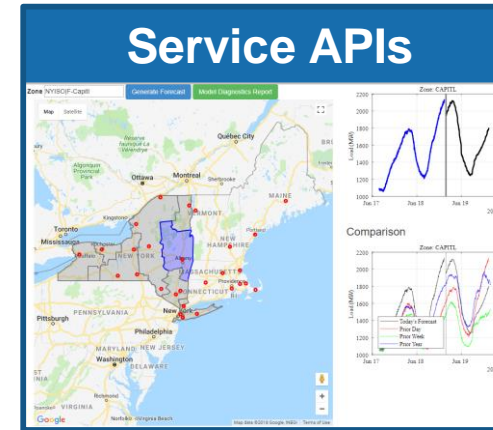
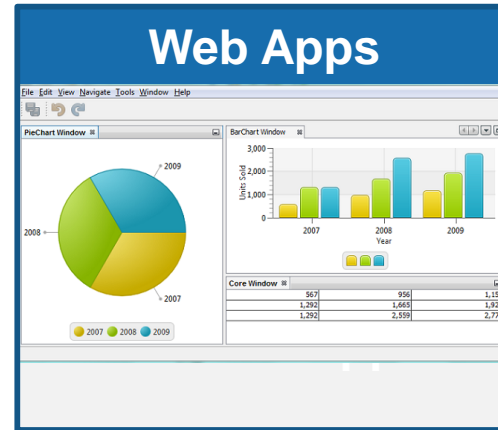
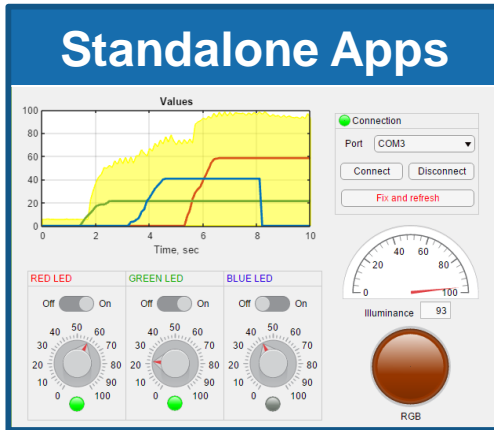
Line Forecast
Post-Dir
Post-File
Post-Tier

Standalone FMUs

fmi FUNCTIONAL MOCK-UP INTERFACE



Simulation deployment users play different roles in different scenarios



Simulation Author: They define, build, edit and *compile* Simulink simulations



Simulation User: They run, tune, and analyze the deployed simulations

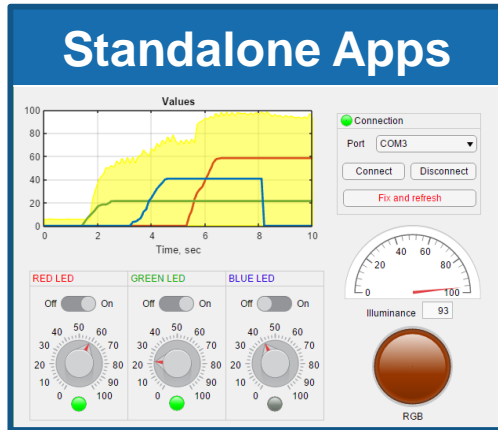


IT: They support integrating deployed simulations with IT systems



Scenario 1

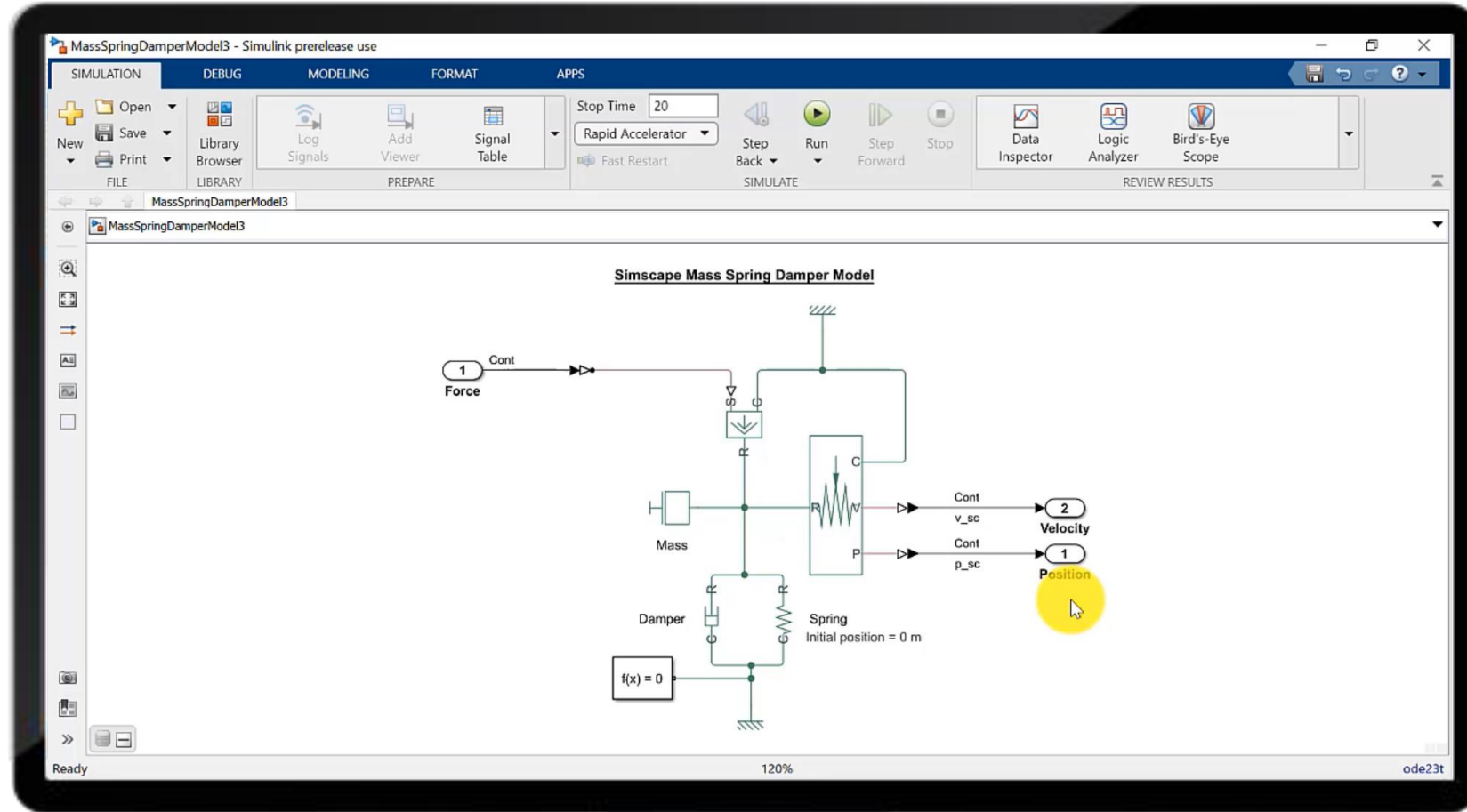
Package simulations as standalone desktop apps



Runs on PC

Can use App Designer GUI

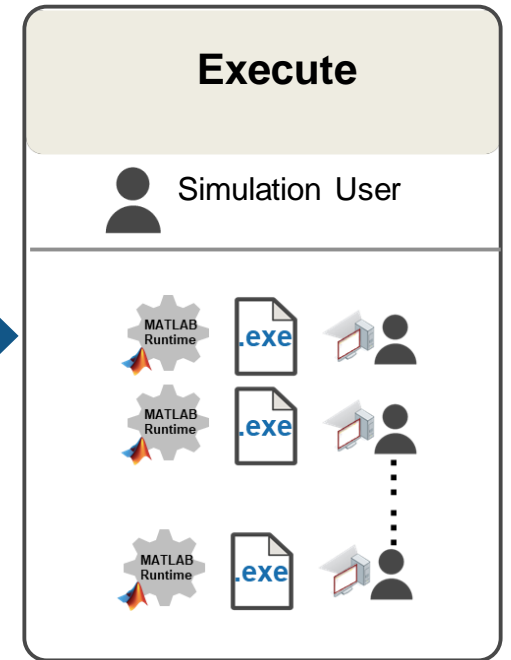
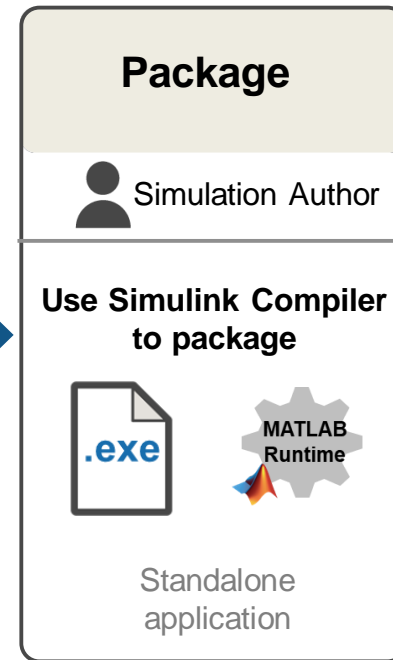
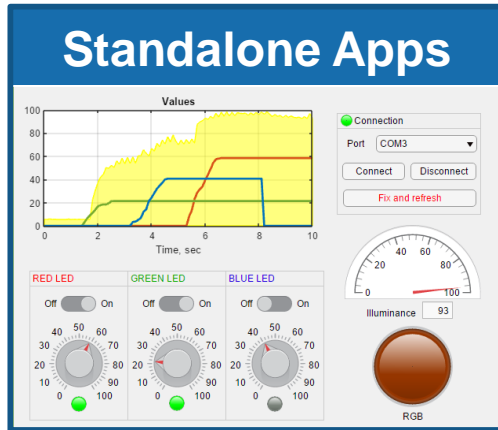
Needs local installation





Scenario 1

Package simulations as standalone desktop apps





Scenario 2

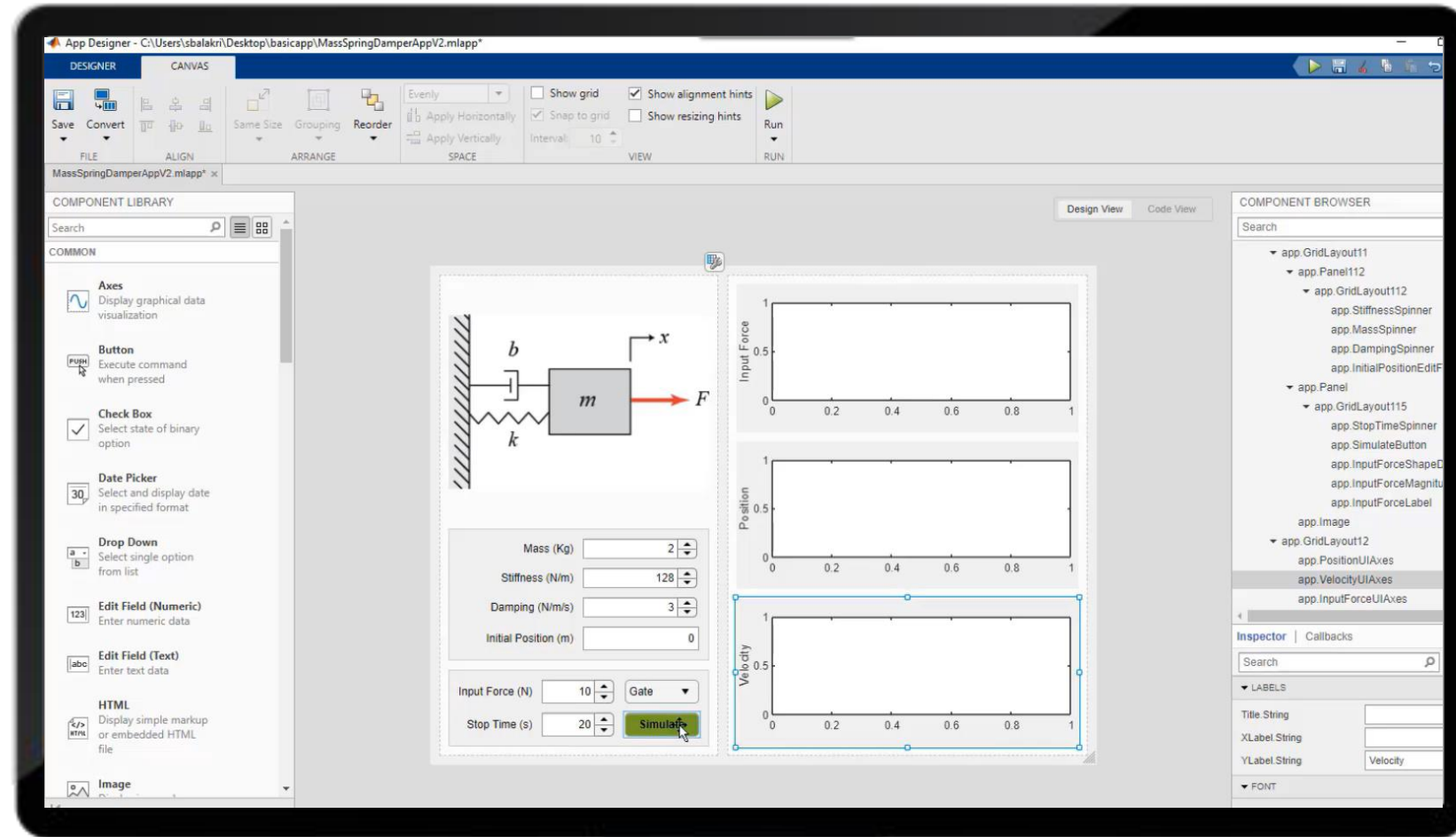
Package simulations as web apps



Runs on MATLAB Web App Server

Uses App Designer GUI

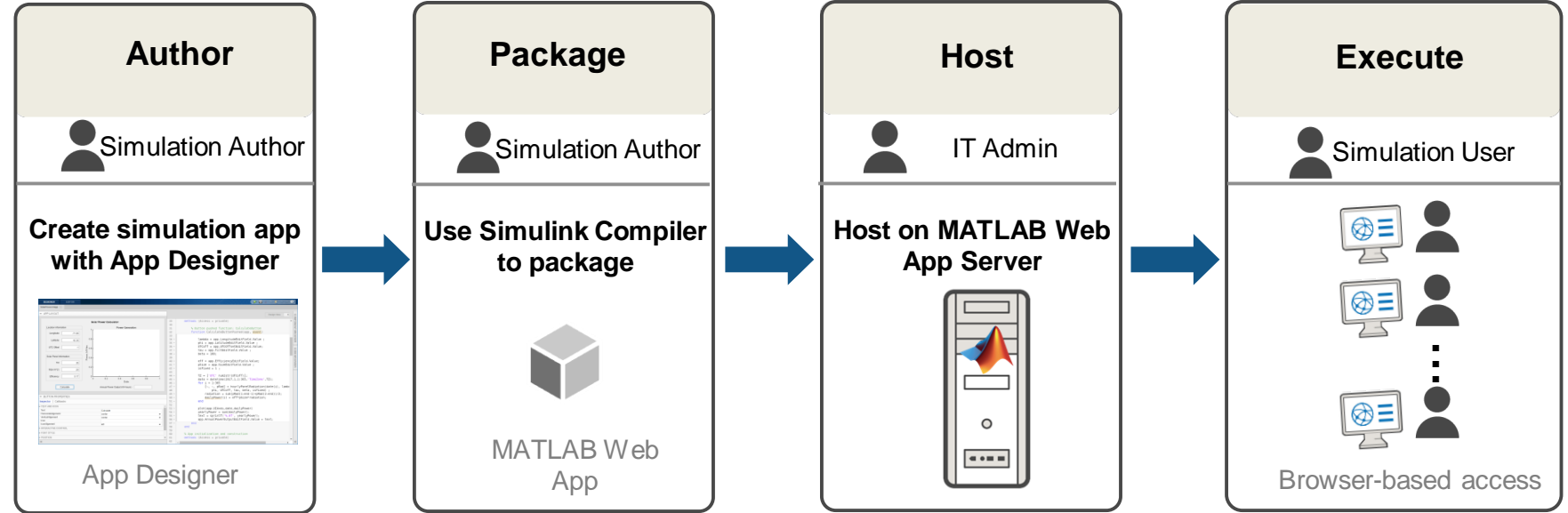
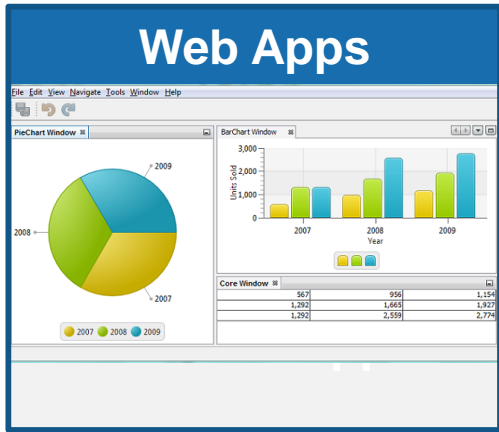
Browser-based access, no local installation needed





Scenario 2

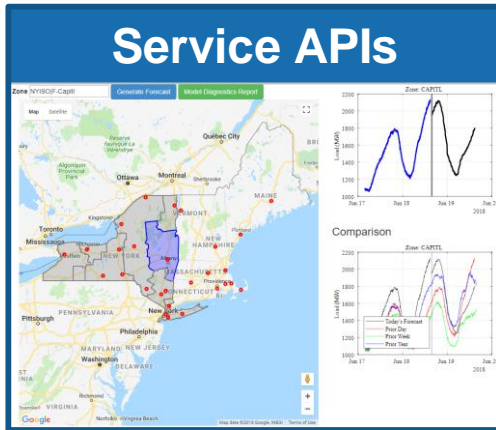
Package simulations as web apps





Scenario 3

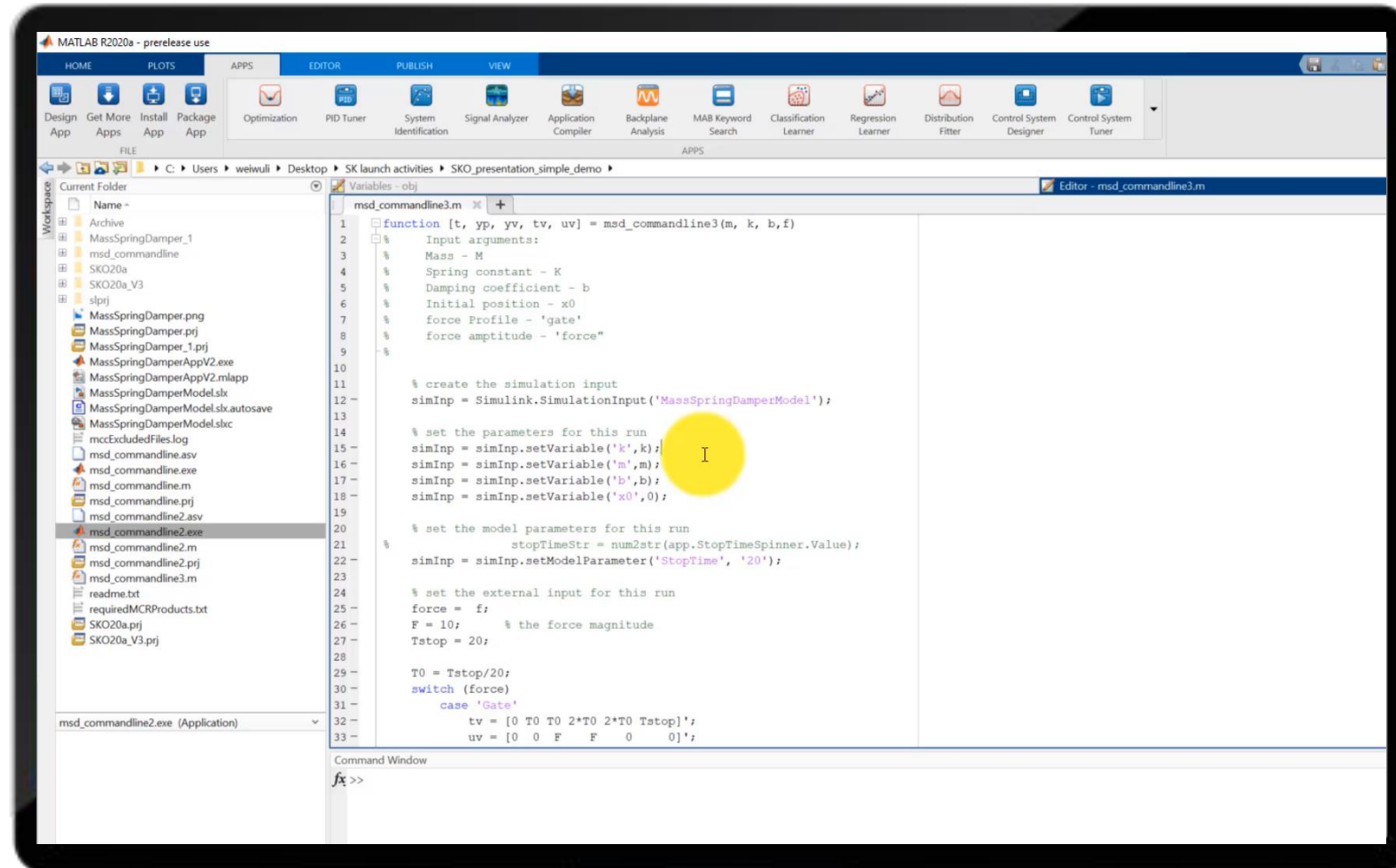
Package simulations as a service API



Runs on MATLAB Production Server*

Supports customer developed client-server App and web apps (e.g. HTML/JavaScript)

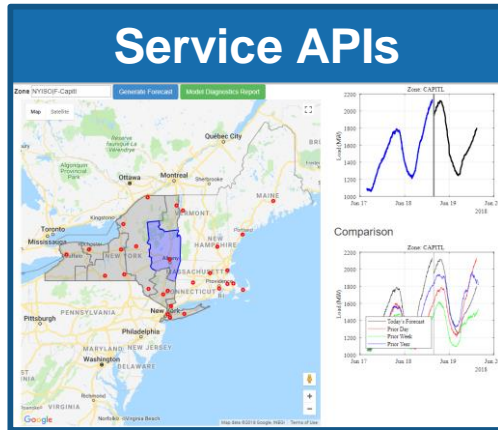
Centrally hosted, no local installation needed





Scenario 3

Package simulations as a service API



Author

Simulation Author

Create simulation function in MATLAB

```

simInp = Simulink.SimulationInput('model1')
simInp = setVariable('mass',mass)
simInp = setVariable('spring',spring)
simInp = setVariable('damping',damping)
simInp = simulink.compiler.configureForDeployment(simInp)
simOut = sim(simInp)
  
```


MATLAB function



Package

Simulation Author

Use Simulink Compiler / Compiler SDK to package




Deployed Archive



Host

IT Admin

Host on MATLAB Production Server

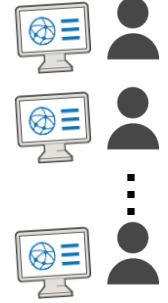


RESTful interface



Execute

Simulation User



Call API via web apps or enterprise applications



Scenario 4

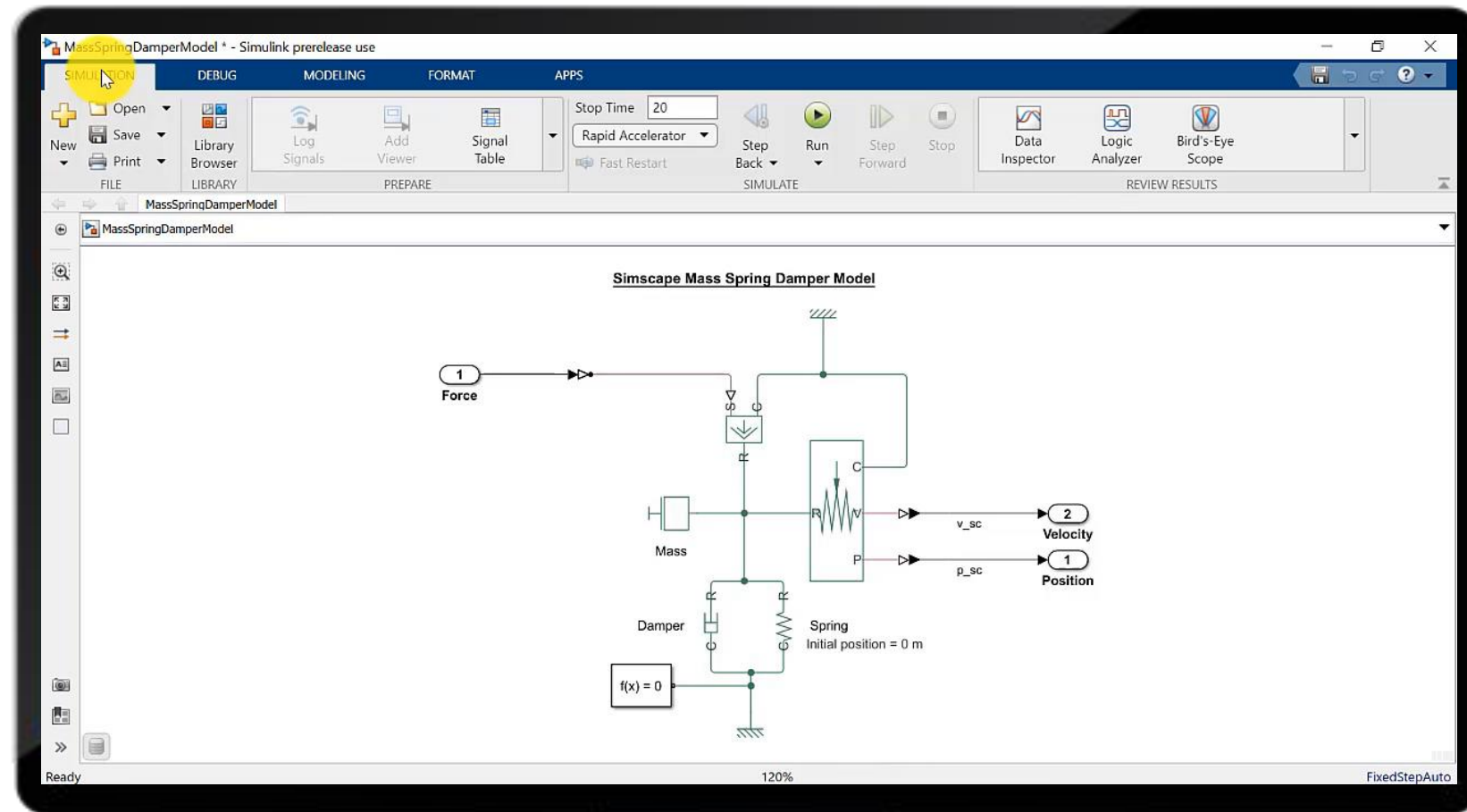
Package simulations as standalone FMU



Runs in 3rd party simulation tools (which support FMI import)

Has no dependency on MATLAB Runtime

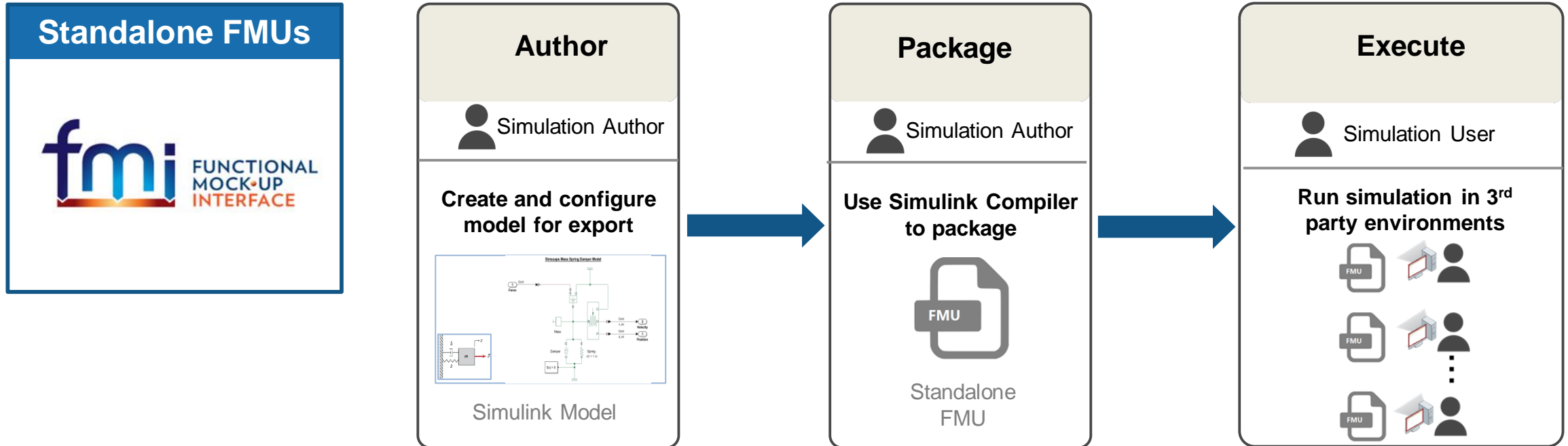
Supports standalone co-simulation FMU with fixed-step solver only





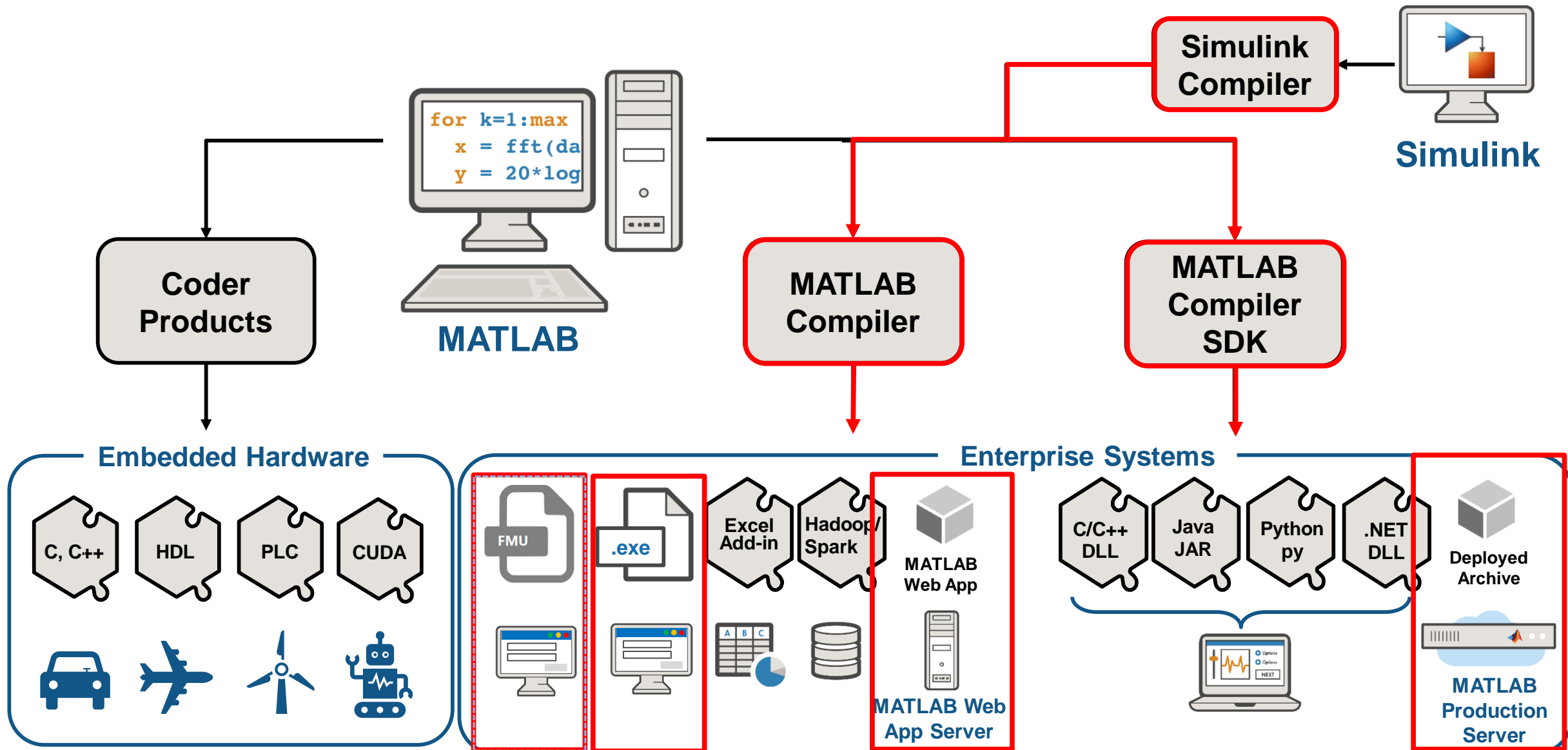
Scenario 4

Package simulations as standalone FMU



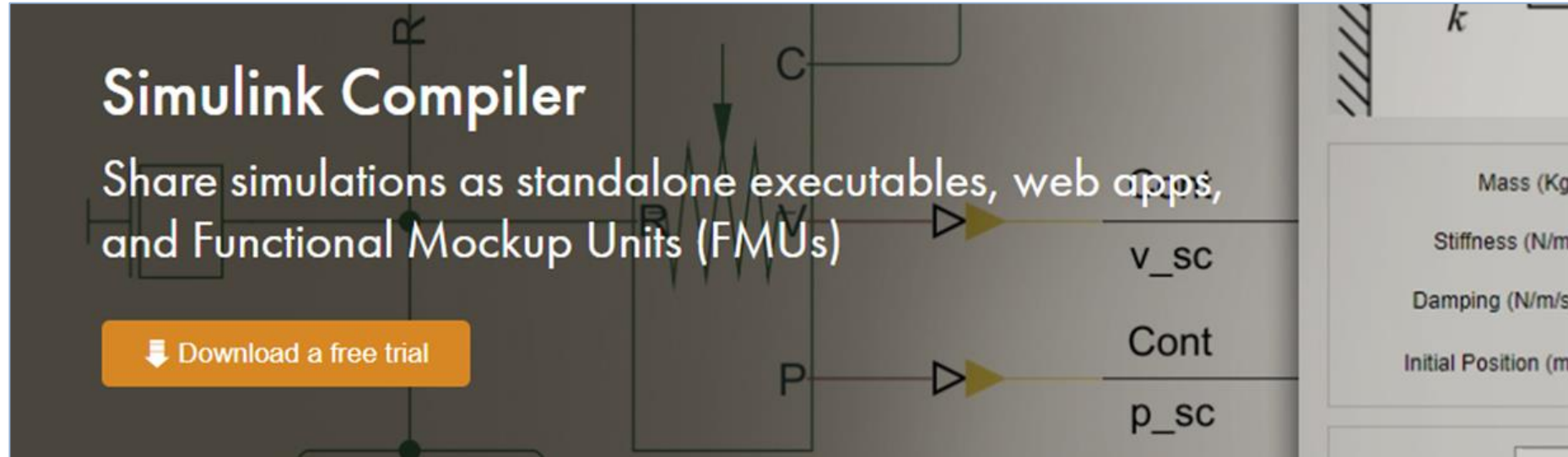


Simulink Compiler is the latest addition to MathWorks deployment options





Simulink Compiler is an out-of-the-box solution to share Simulink simulations

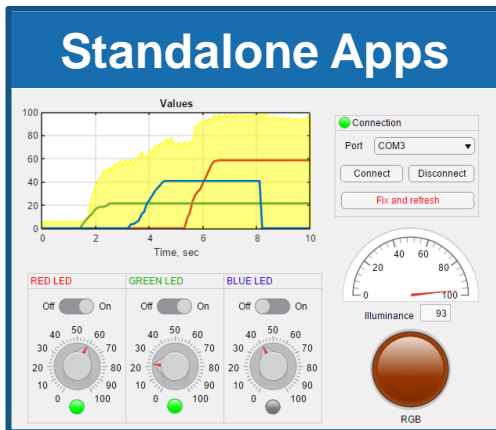


Simulink Compiler

Share simulations as standalone executables, web apps, and Functional Mockup Units (FMUs)

[Download a free trial](#)

Standalone Apps



The interface shows a graph of 'Values' over 'Time, sec' with three data series (red, green, blue). Below the graph are three LED status indicators (RED LED, GREEN LED, BLUE LED) with 'Off' and 'On' toggles and a 'Fix and refresh' button. A 'Connection' section shows 'Port COM3' and 'Connect/Disconnect' buttons. A gauge displays 'Illuminance 93' and an 'RGB' button is at the bottom.

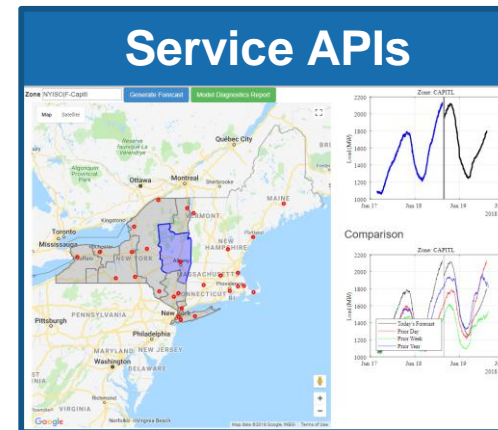
Web Apps



The interface displays a 'PieChart Window II' with a pie chart for years 2007, 2008, and 2009. A 'BarChart Window II' shows a bar chart for the same years. A 'Core Window II' contains a table with data for three years.

Year	Value 1	Value 2	Value 3
2007	967	998	1,154
2008	1,292	1,665	1,827
2009	1,292	2,589	2,774

Service APIs



The interface features a map of the Northeastern United States with various cities marked. On the right, there are two line graphs: 'Zone: CARTEL' and 'Comparison', both showing data over time from June 17 to June 21, 2008.

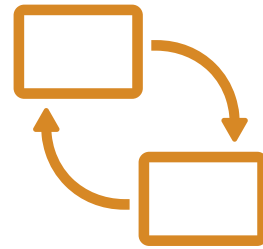
Standalone FMUs



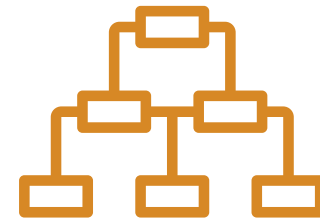
fmi FUNCTIONAL MOCK-UP INTERFACE



Edit at the Speed of Thought



Model Run-Time Software



Componentize Your Design



Protected Models



Projects

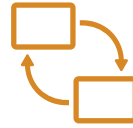


Standalone Apps
Simulink Compiler

Work more **effectively** and **efficiently** with the latest release, and **check out:**



Edit at the Speed of Thought



Model Run-Time Software

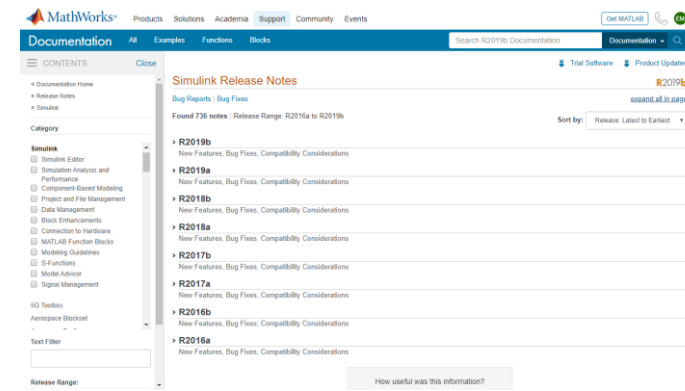


Componentize Your Design

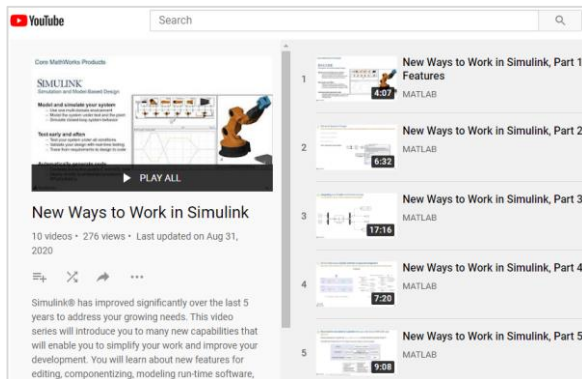
Simulink What's New Page



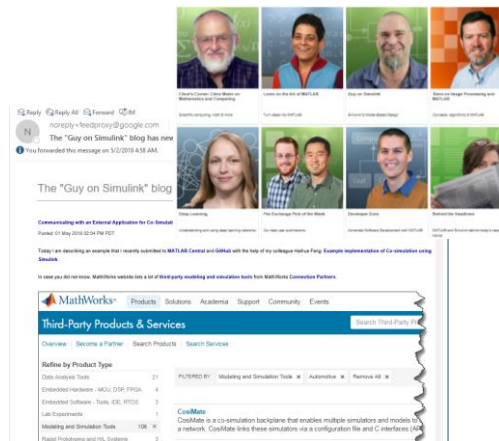
Release Notes



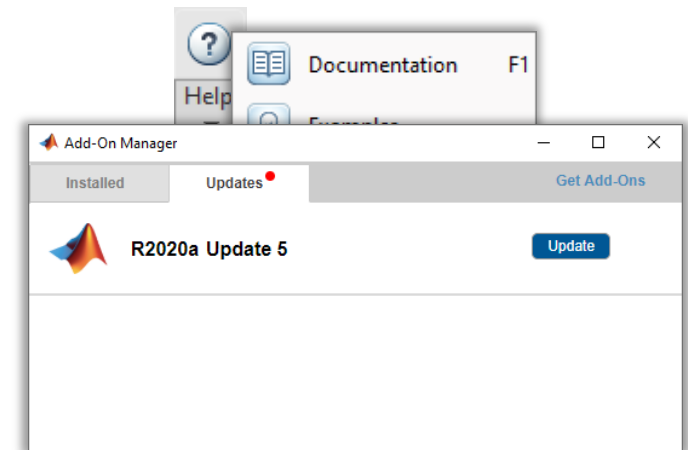
New Ways to Work in Simulink (Video Series)



Blogs



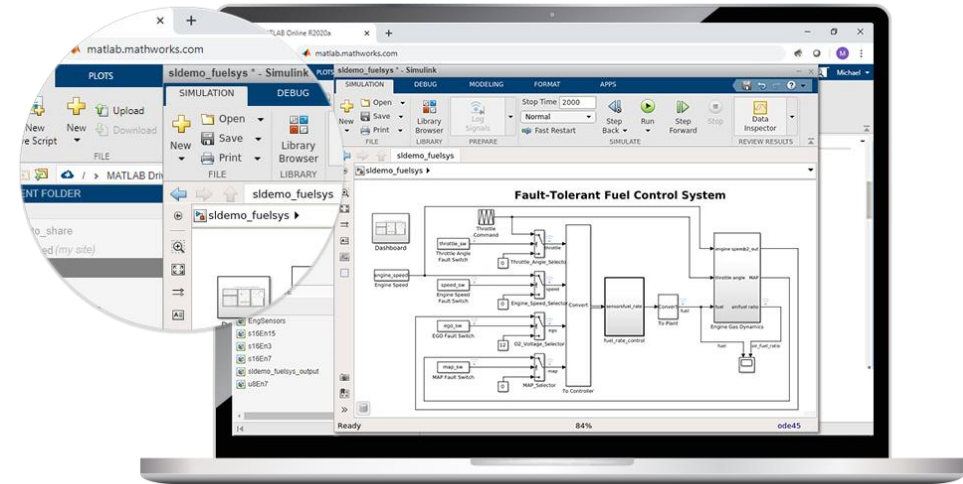
Software Updates



Access and try the latest release with MATLAB and Simulink Online



MATLAB Online

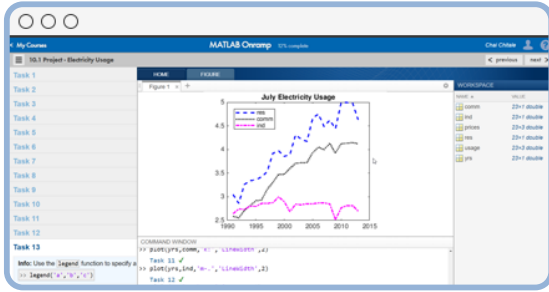


Simulink Online

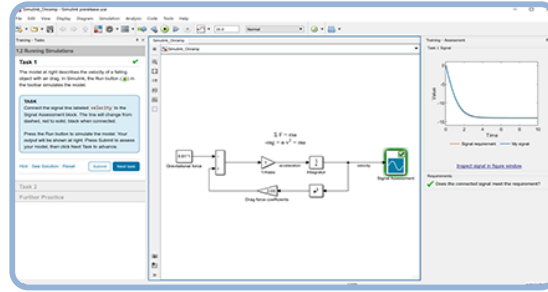
Access and try the latest release with MATLAB and Simulink Online

The screenshot displays the MATLAB Online R2020a web interface. The browser address bar shows `matlab.mathworks.com`. The main workspace contains a Simulink model named `sldemo_fuelsys`. The model is titled **Fault-Tolerant Fuel Control System** and features a **Dashboard** subsystem. The diagram includes several fault switches: **Throttle Angle Fault Switch**, **Engine Speed Fault Switch**, **EGO Fault Switch**, and **MAP Fault Switch**. These switches are connected to selector blocks (`Throttle_Angle_Selector`, `Engine_Speed_Selector`, `O2_Voltage_Selector`, and `MAP_Selector`) which feed into a **To Controller** block. The controller outputs `fuel_rate_control` (g/s), which is converted to `fuel` (g/s) for the **Engine Gas Dynamics** block. The engine block also receives `throttle angle` (deg) and `MAP` (bar) as inputs. The engine outputs `engine speed2_out` (rad/s) and `air_fuel_ratio` (1). A **Data Inspector** window is visible on the right side of the interface. The status bar at the bottom indicates `Ready`, `79%` zoom, and `ode45` solver.

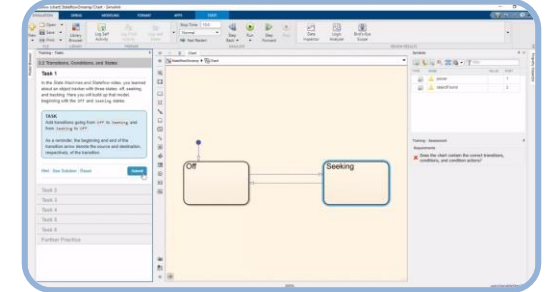
Get started and learn with Onramps



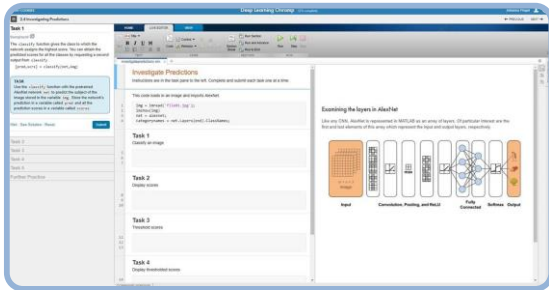
MATLAB Onramp



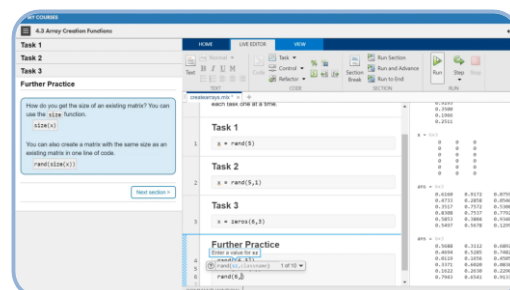
Simulink Onramp



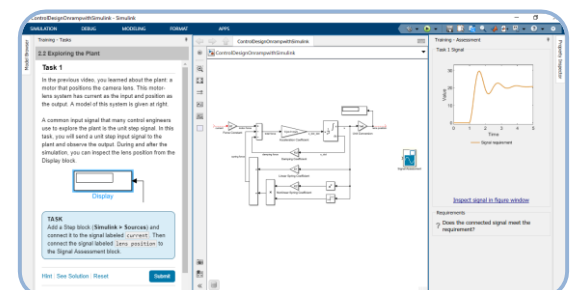
Stateflow Onramp



Deep Learning Onramp



Machine Learning Onramp



Control Design Onramp

Upgrade to the latest release **today**

Get Latest Release

R2020b

» [Learn More](#)



[Download](#)



[Click here to download any MathWorks release](#)