

The background features a dark blue field on the left and a grey field on the right, separated by a diagonal line. In the upper right, there are white, stylized waveforms. In the lower right, there is a 3D wireframe mesh with a color gradient from yellow to blue, and a faint blue circuit board pattern.

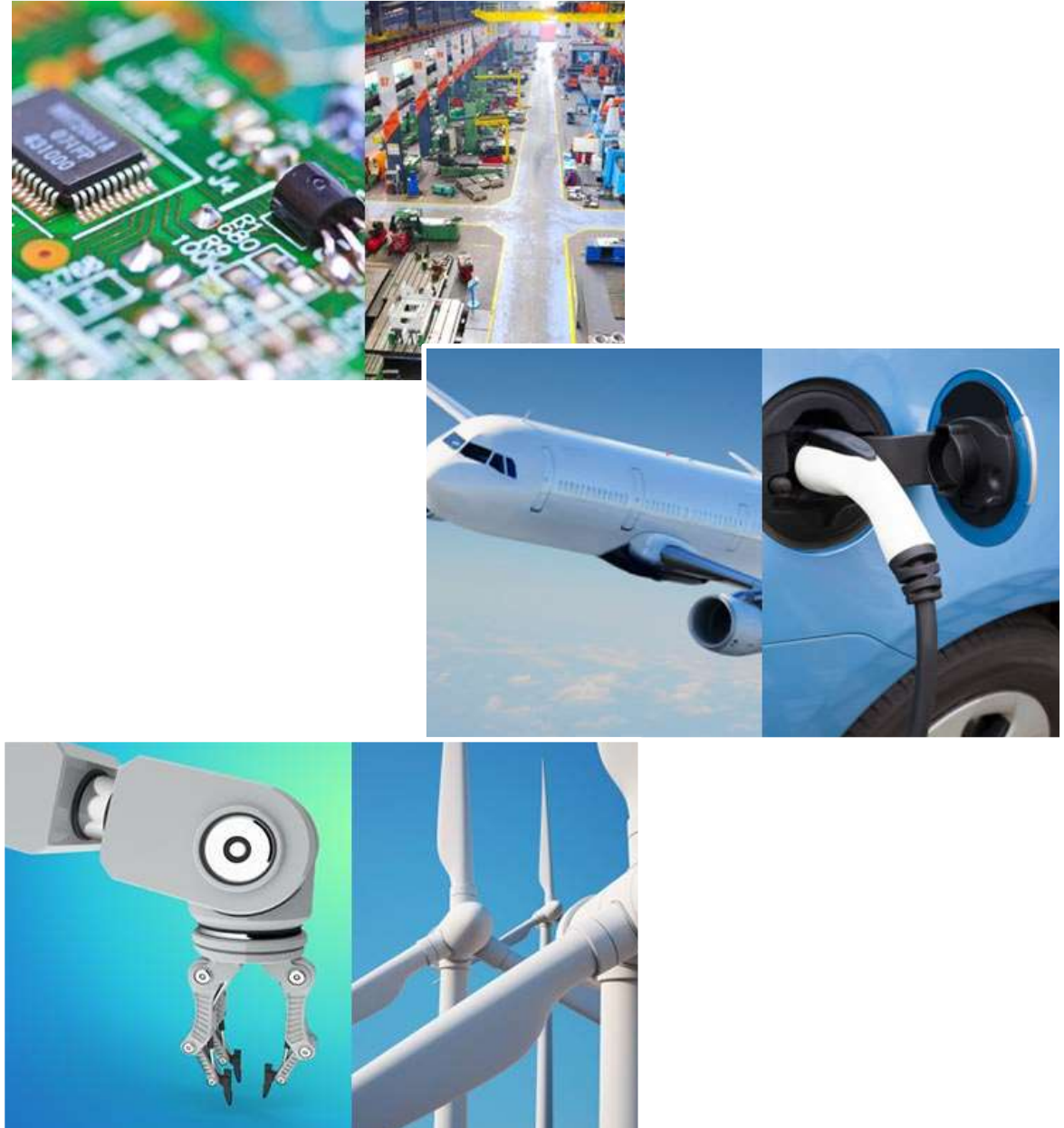
MATLAB EXPO 2017

Simulink as Your Enterprise Simulation Platform

Prasanna Deshpande & Naga Pemmaraju

Enterprise Simulation Platform

- Enterprise - Any size business or project
- Simulation – Evaluating system behavior through computation
- Platform – Scalable environment for multi-disciplinary collaboration



Challenges faced by teams working at enterprise level

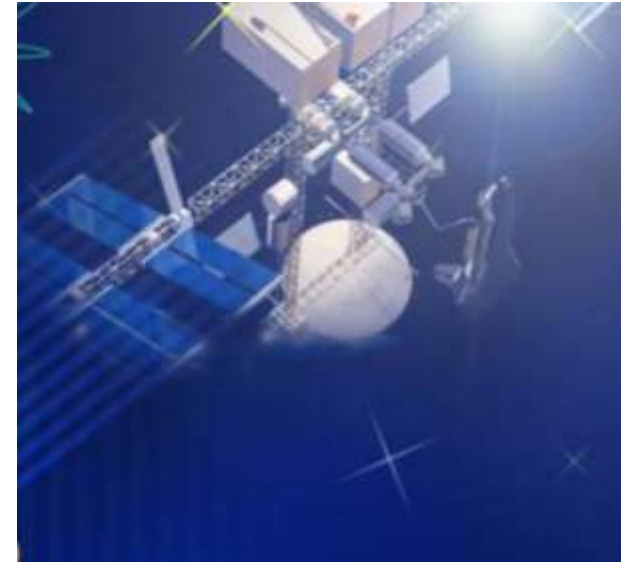
- Products / projects involve multiple engineering and non-engineering domains
- Systems are complex; require many teams to work together on different components and share available resources
- Many different tools may require to work together to achieve the bigger goal

Simulink as an Enterprise Simulation Platform

Simulating Spacecraft Communications for Deep-Space Missions

Dr. Deepak Mishra, Scientist/Engineer (SF)

Indian Space Research Organization



Challenge

- Integrating large multi-faceted project
- Simulation at multiple stages and in multiple domains to explore the problem

Solution

- Leverage Simulink as a platform

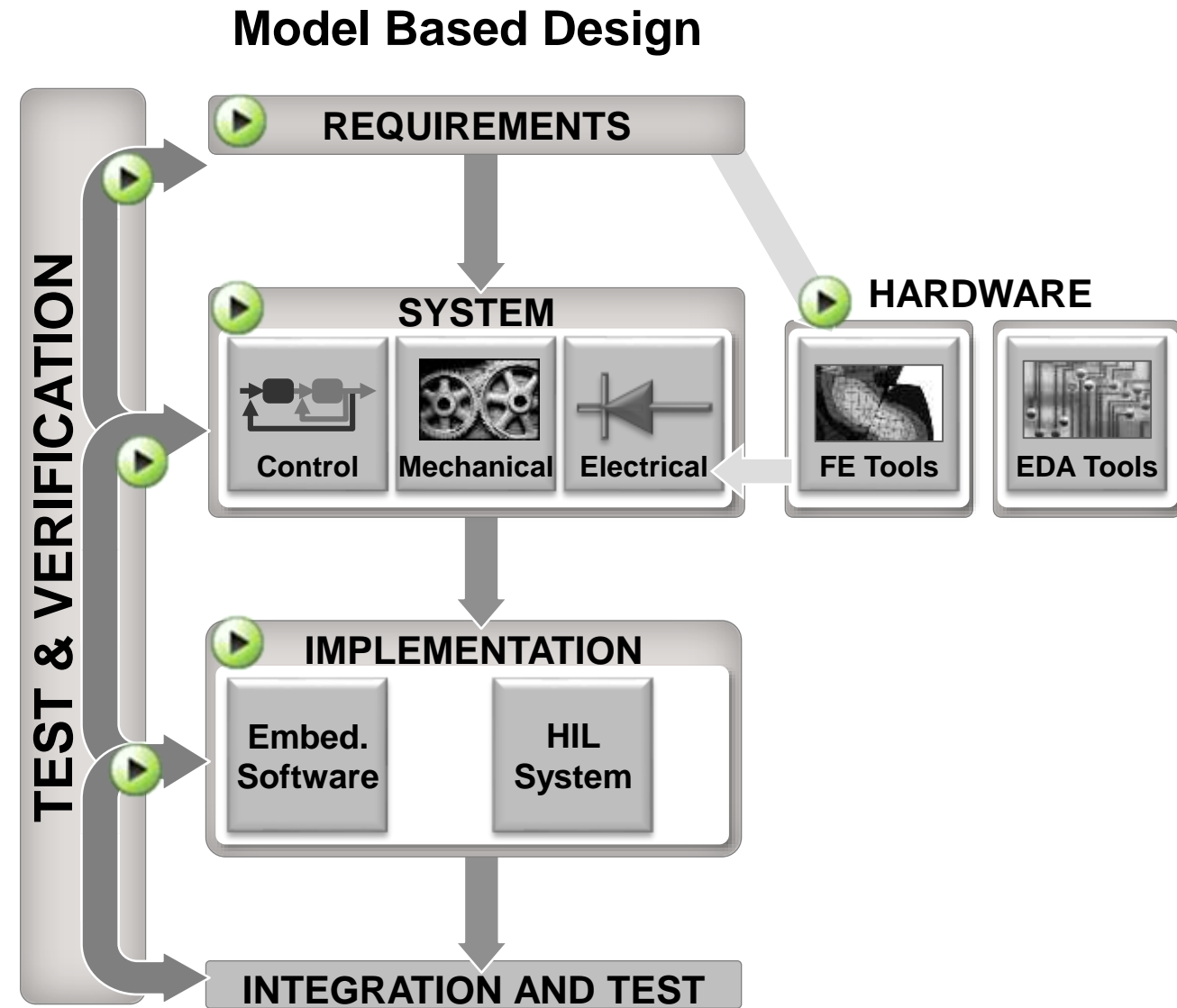
Simulink as an Enterprise Simulation Platform



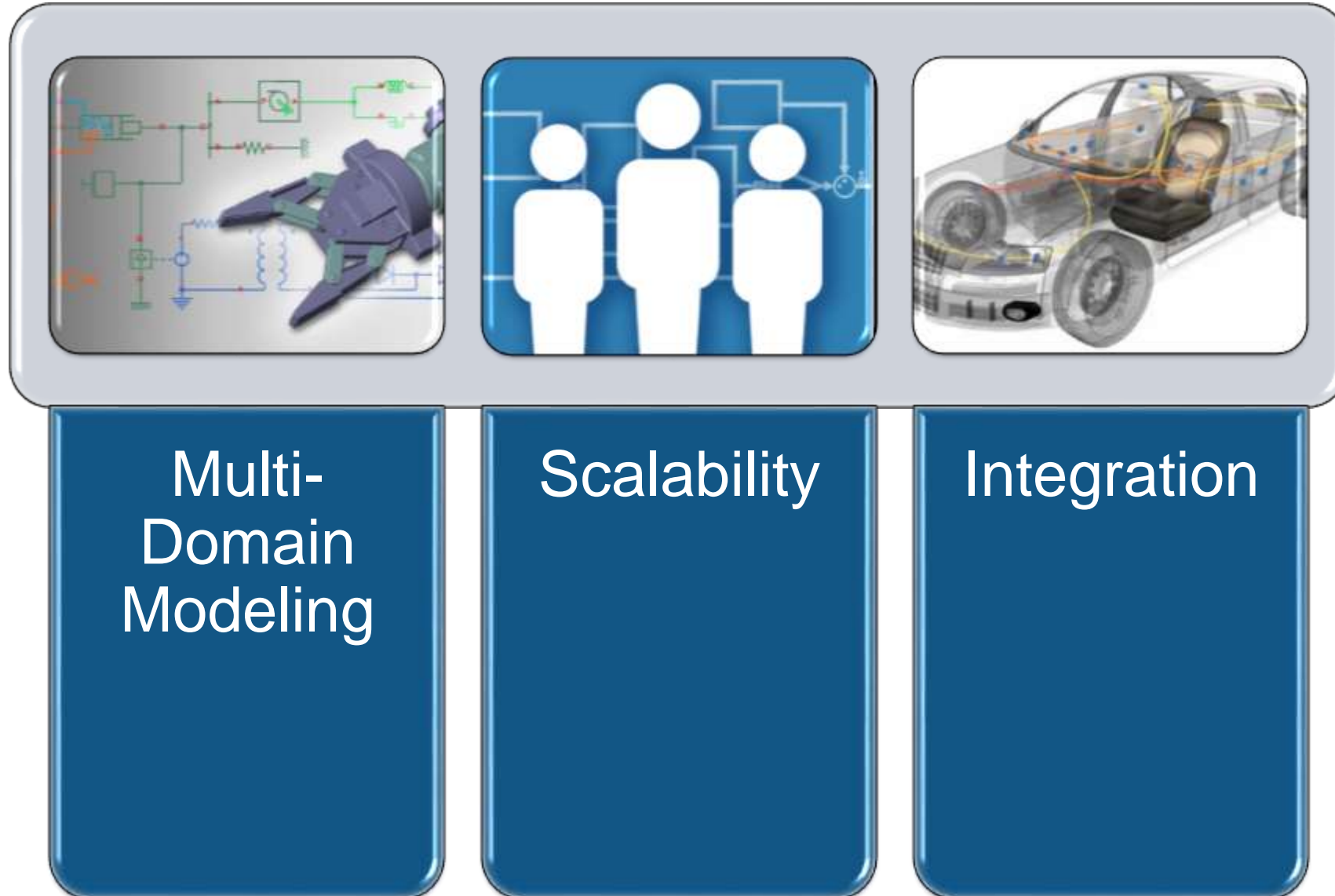
Simulink as Enterprise Simulation Platform

- Enterprise - Any size business or project
- Simulation – Evaluating system behavior through computation
- Platform – Scalable environment for multi-disciplinary collaboration

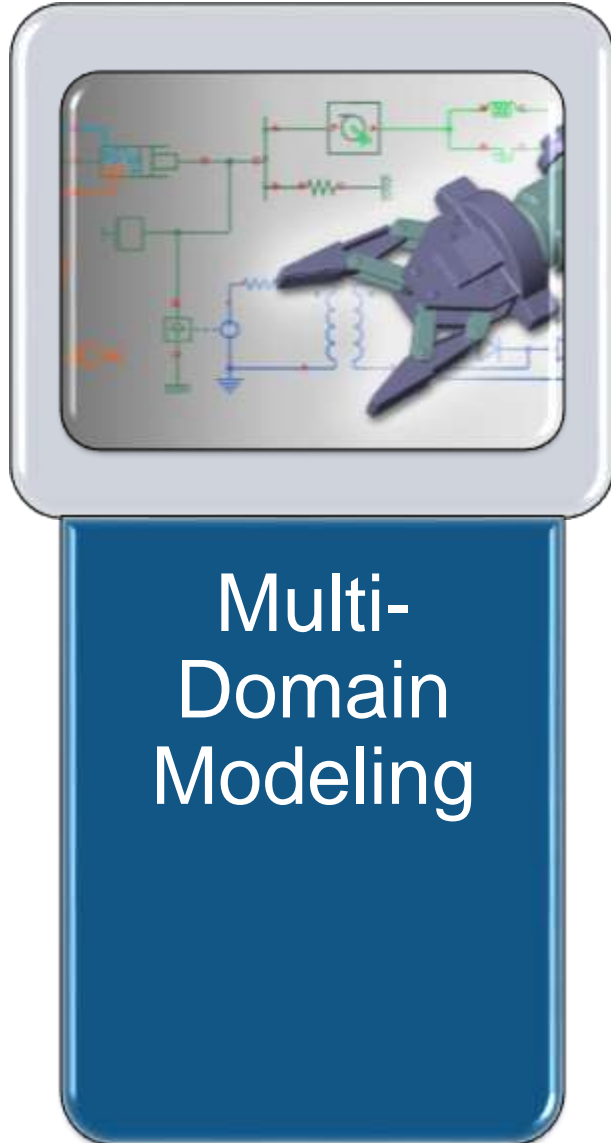
Simulation



Enterprise Simulation Platform Enablers



Enterprise Simulation Platform Enablers



Multi-Domain Modeling in Simulink



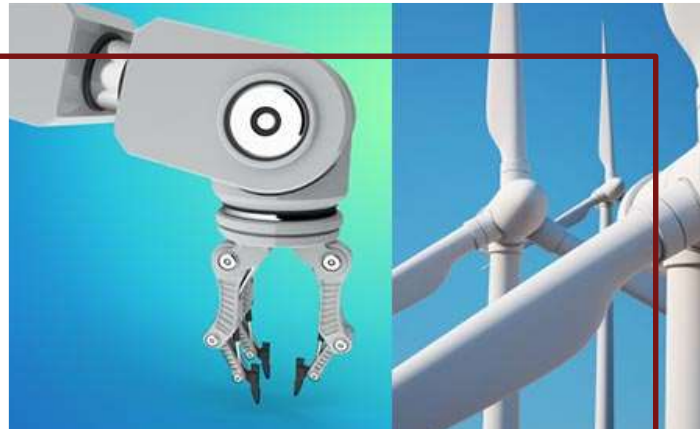
Dynamic Systems



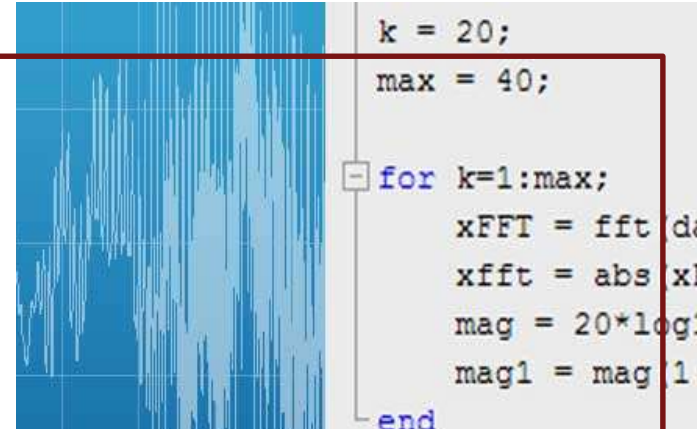
State Machines



Discrete-Event Systems



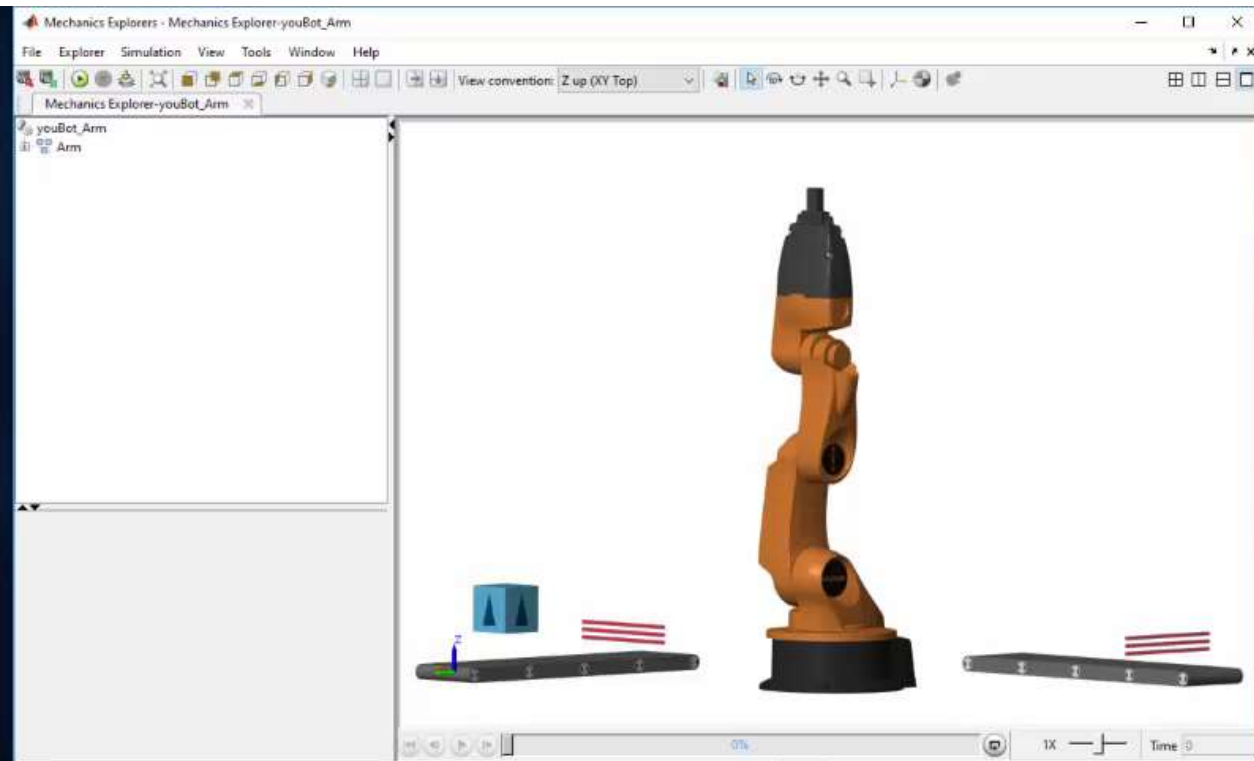
Physical Modeling



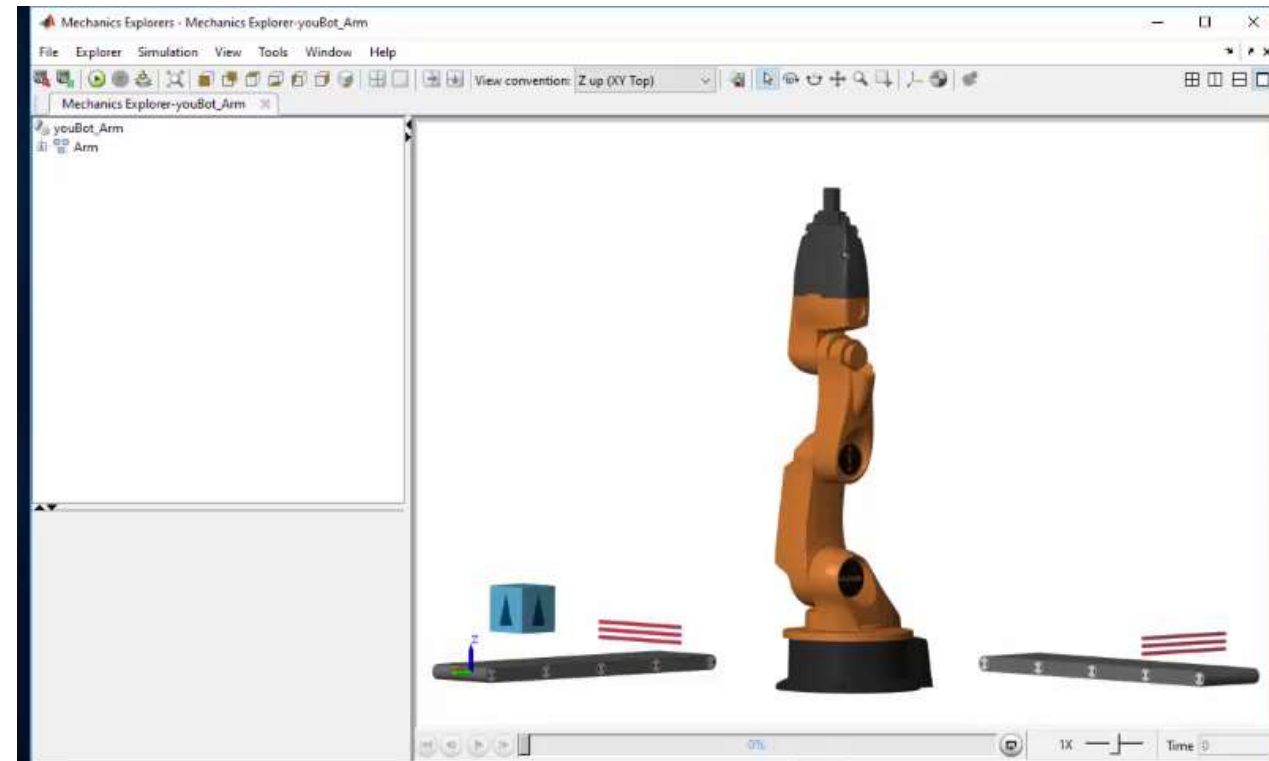
Object-Oriented

Robot Arm Multi-Domain Simulation

Mechatronic System Model



Mechatronic system model with communication latencies



Multi-Domain Model

youBot Arm

1. Plot motor [currents \(code\)](#) and [torques \(code\)](#)
2. Plot joint [angles \(code\)](#) and [forces \(code\)](#)
3. Plot box [trajectory \(code\)](#)
4. [Explore simulation results](#) using [sscexplore](#)
5. Plot optimization results: [Friction](#), [No Friction \(code\)](#)
6. [Compare](#) optimization results ([code](#))
7. Load [model parameters \(code\)](#)
8. [Learn more](#) about this example

Configure Test : [Default \(code\)](#)
 Box Transfer only: [Linear](#); Splines: [Manual](#), [Optim \(friction\)](#), [Optim \(no friction\)](#)
 Joint Tests: [Pivot](#), [Bicep](#), [Forearm](#), [Wrist](#), [Max Torque](#), [All 35](#)

Run optimization: [Friction](#), [No Friction \(code\)](#)

Ready 123% ode15s

Physical Modeling

youBot_Arm - Simulink

File Edit View Display Diagram Simulation Analysis Code Tools Help

youBot_Arm

youBot_Arm

Input Control

Network (CAN Bus)

Arm

World

Environment

Finger A

Finger B

Gripper

Actuation

Configure Actuation: [Matlab](#), [Motor](#)

In1

Sensors

Configure Test : [Default](#) ([code](#))
 Box Transfer only: [Linear](#); Splines: [Manual](#), [Optim \(friction\)](#), [Optim \(no friction\)](#)
 Joint Tests: [Pivot](#), [Bicep](#), [Forearm](#), [Wrist](#), [Max Torque](#), [All 35](#)

Run optimization: [Friction](#), [No Friction](#) ([code](#))

123%

ode15s

Multi-Domain Model

The screenshot shows the Simulink interface for a 'youBot_Arm' model. The main workspace contains three interconnected blocks: 'Input Control', 'Arm', and 'Network (CAN Bus)'. The 'Input Control' block includes sub-blocks for 'Home', 'Belt In', and 'Belt Out', with 'Sensors' inputs. The 'Arm' block features a 3D model of an orange robotic arm and its own 'Sensors'. The 'Network (CAN Bus)' block acts as a communication hub, with 'Out1' connected to the 'Input Control' and 'In1' connected to the 'Arm'. Signal lines for 'Arm' and 'Belt' connect the control logic to the physical arm model.

youBot Arm

1. Plot motor [currents \(code\)](#) and [torques \(code\)](#)
2. Plot joint [angles \(code\)](#) and [forces \(code\)](#)
3. Plot box [trajectory \(code\)](#)
4. [Explore simulation results](#) using [sscexplore](#)
5. Plot optimization results: [Friction](#), [No Friction \(code\)](#)
6. [Compare](#) optimization results ([code](#))
7. Load [model parameters \(code\)](#)
8. [Learn more](#) about this example

Configure Test : [Default \(code\)](#)
 Box Transfer only: [Linear](#); Splines: [Manual](#), [Optim \(friction\)](#), [Optim \(no friction\)](#)
 Joint Tests: [Pivot](#), [Bicep](#), [Forearm](#), [Wrist](#), [Max Torque](#), [All 35](#)

Run optimization: [Friction](#), [No Friction \(code\)](#)

Ready 123% ode15s

State Charts and System Dynamics

youBot Arm

1. Plot motor [currents \(code\)](#) and [torques \(code\)](#)
2. Plot joint [angles \(code\)](#) and [forces \(code\)](#)
3. Plot box [trajectory \(code\)](#)
4. [Explore simulation results](#) using [sscexplore](#)
5. Plot optimization results: [Friction, No Friction \(code\)](#)
6. [Compare](#) optimization results ([code](#))
7. Load [model parameters \(code\)](#)
8. [Learn more](#) about this example

Configure Test (code)
 Box Transfer only: [Manual, Or](#)
 Joint Tests: [Pivot, Biceps, Wrist, Max](#)
 Run optimization: [Friction, No Friction](#)

Joint Commands

Multi-Domain Model

The screenshot shows the Simulink interface for a 'youBot_Arm' model. The main workspace contains three interconnected blocks: 'Input Control', 'Arm', and 'Network (CAN Bus)'. The 'Input Control' block includes sub-blocks for 'Home', 'Belt In', and 'Belt Out', with 'Sensors' inputs. The 'Arm' block features a 3D model of an orange robotic arm and its own 'Sensors'. The 'Network (CAN Bus)' block acts as a communication hub with 'Out1' and 'In1' ports. Signal lines connect 'Input Control' to 'Arm' (labeled 'Arm' and 'Belt') and 'Network (CAN Bus)' to both 'Input Control' and 'Arm'.

youBot Arm

1. Plot motor [currents \(code\)](#) and [torques \(code\)](#)
2. Plot joint [angles \(code\)](#) and [forces \(code\)](#)
3. Plot box [trajectory \(code\)](#)
4. [Explore simulation results](#) using [sscexplore](#)
5. Plot optimization results: [Friction](#), [No Friction \(code\)](#)
6. [Compare](#) optimization results ([code](#))
7. Load [model parameters \(code\)](#)
8. [Learn more](#) about this example

Configure Test : [Default \(code\)](#)
 Box Transfer only: [Linear](#); Splines: [Manual](#), [Optim \(friction\)](#), [Optim \(no friction\)](#)
 Joint Tests: [Pivot](#), [Bicep](#), [Forearm](#), [Wrist](#), [Max Torque](#), [All 35](#)
 Run optimization: [Friction](#), [No Friction \(code\)](#)

Discrete-Event Modeling

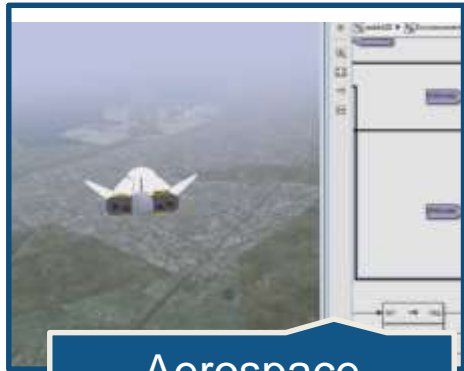
youBot Arm

1. Plot motor [currents](#) ([code](#)) and [torques](#) ([code](#))
2. Plot joint [angles](#) ([code](#)) and [forces](#) ([code](#))
3. Plot box [trajectory](#) ([code](#))
4. [Explore simulation results](#) using [sscexplore](#)
5. Plot optimization results: [Friction](#), [No Friction](#) ([code](#))
6. [Compare](#) optimization results ([code](#))
7. Load [model parameters](#) ([code](#))
8. [Learn more](#) about this example

Configure Test : [Default](#) ([code](#))
 Box Transfer only: [Linear](#); Splines:
 Joint Tests: [Pivot](#), [Bicep](#), [Forearm](#),
 Run optimization: [Friction](#), [No Friction](#)

Domain-Specific Blocksets and Toolboxes

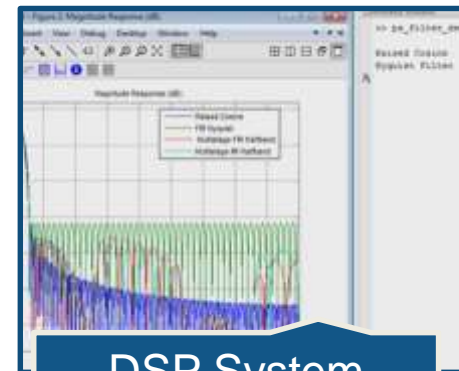
Simulink has numerous domain-specific tools, for example:



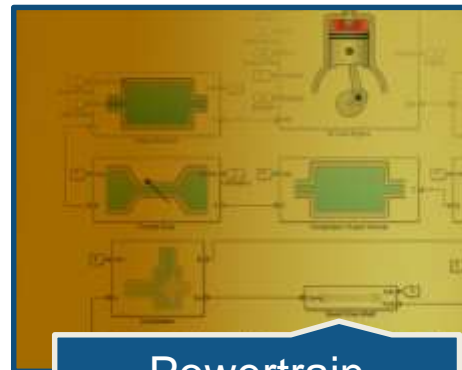
Aerospace
Blockset



Computer Vision
System Toolbox



DSP System
Toolbox



Powertrain
Blockset



Robotics System
Toolbox

Customer Success in Multidomain Modeling

ABB, Deltamarin, and VTT Simulate and Optimize Ship Energy Flows

Challenge

- Increase the energy efficiency of large vessels

Solution

- Use Simulink and Simscape to model, simulate, and optimize ship energy flow

Results

- Cost- and fuel-saving design improvements
- Testing costs reduced by tens of thousands of euros



Customer Success in Multidomain Modeling

“Simulink and Simscape enabled us to create a dynamic model of a complex energy system that spans several physical domains. By simulating this model, we can see how a new energy subsystem will perform before it is built, and provide customers with an accurate estimate of their return on investment.”

Juha Orivuori, ABB



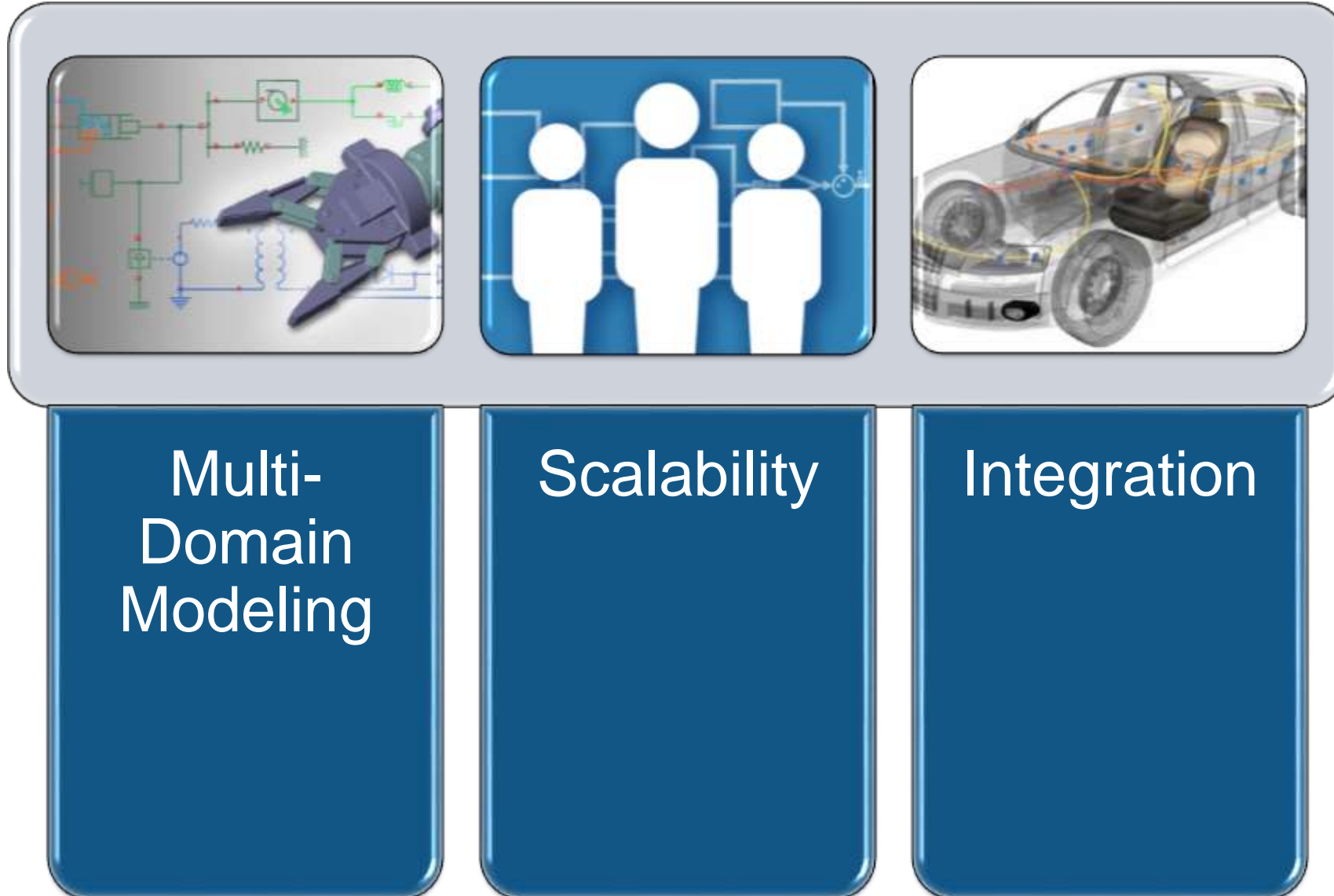
Solution

- Use Simulink and Simscape to model, simulate, and optimize ship energy flow

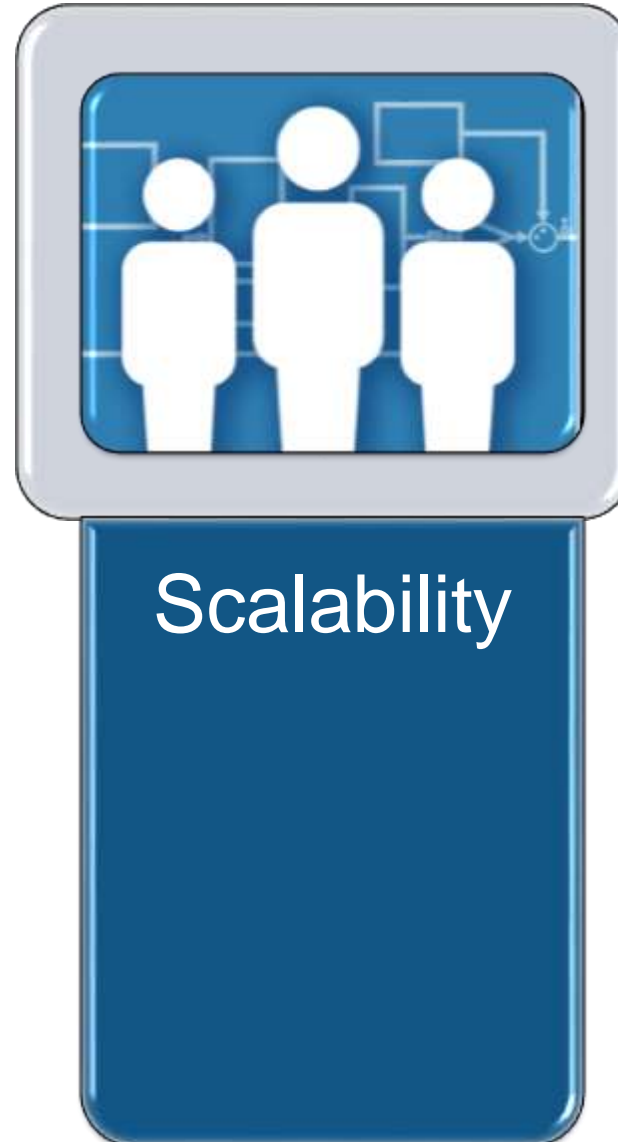
Results

- Cost- and fuel-saving design improvements
- Testing costs reduced by tens of thousands of euros

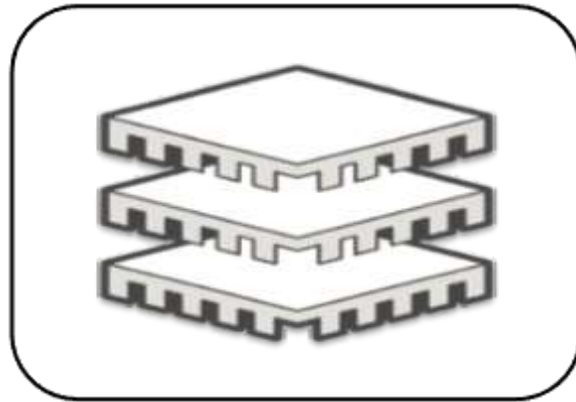
Enterprise Simulation Platform Enablers



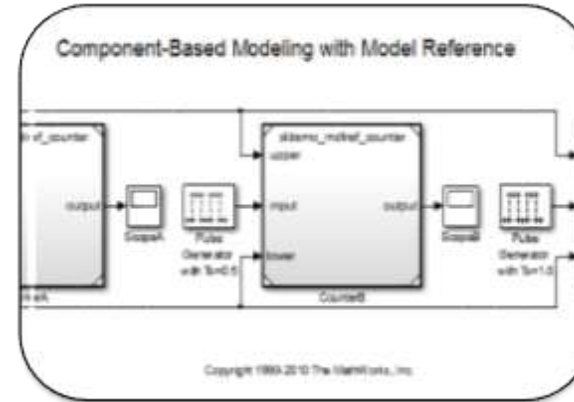
Enterprise Simulation Platform Enablers



Scalability Challenges



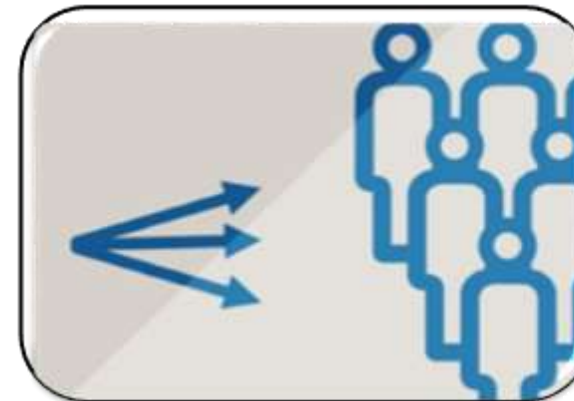
Performance



Componentization

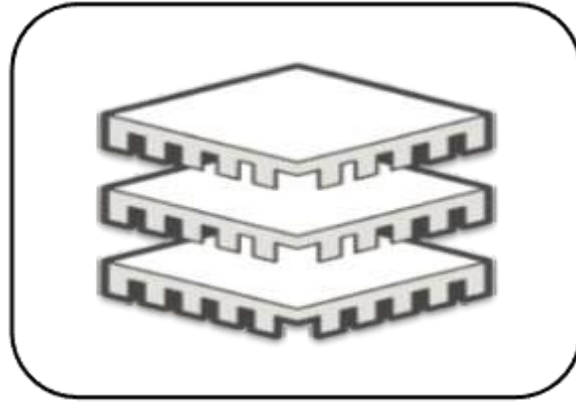


Team Workflows



Sharing

Scalability Challenges



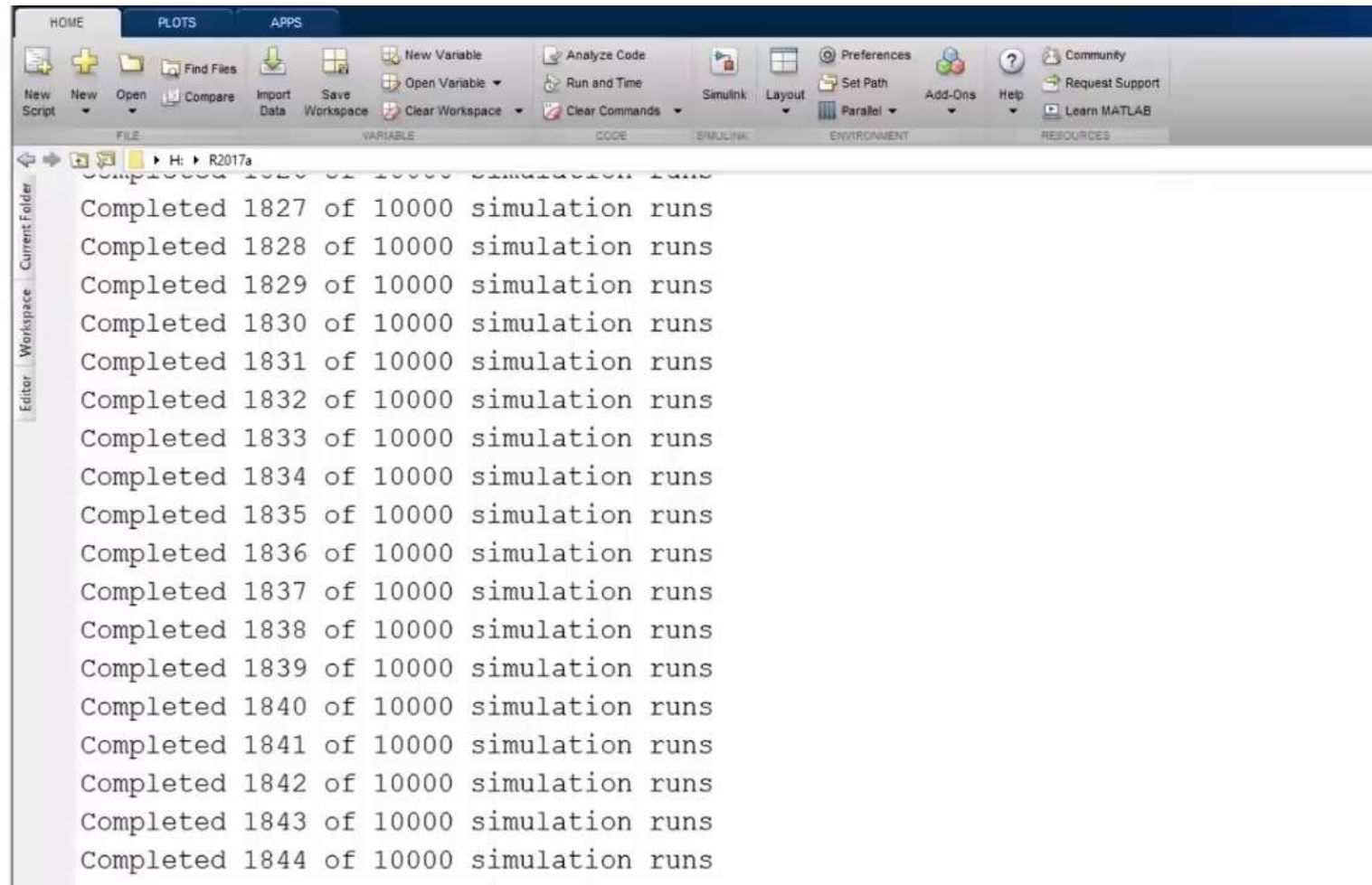
Performance

Tools and Techniques for Speeding Up Simulations

- Choosing the right solver – Automatic Solver Selection
- Examine model dynamics with Solver Profiler
- Using simulation acceleration modes
- Using Performance Advisor

Performance Scalability

Easy scalability to multicore or cluster/cloud computation environment



The screenshot displays the MATLAB Parallel Computing Toolbox interface. The top menu bar includes options like HOME, PLOTS, and APPS. Below the menu, there are various tool icons for file operations, workspace management, and simulation control. The main workspace area shows a list of simulation runs, each marked as 'Completed'.

Run ID	Status	Progress	Description
1827	Completed	1827 of 10000	simulation runs
1828	Completed	1828 of 10000	simulation runs
1829	Completed	1829 of 10000	simulation runs
1830	Completed	1830 of 10000	simulation runs
1831	Completed	1831 of 10000	simulation runs
1832	Completed	1832 of 10000	simulation runs
1833	Completed	1833 of 10000	simulation runs
1834	Completed	1834 of 10000	simulation runs
1835	Completed	1835 of 10000	simulation runs
1836	Completed	1836 of 10000	simulation runs
1837	Completed	1837 of 10000	simulation runs
1838	Completed	1838 of 10000	simulation runs
1839	Completed	1839 of 10000	simulation runs
1840	Completed	1840 of 10000	simulation runs
1841	Completed	1841 of 10000	simulation runs
1842	Completed	1842 of 10000	simulation runs
1843	Completed	1843 of 10000	simulation runs
1844	Completed	1844 of 10000	simulation runs

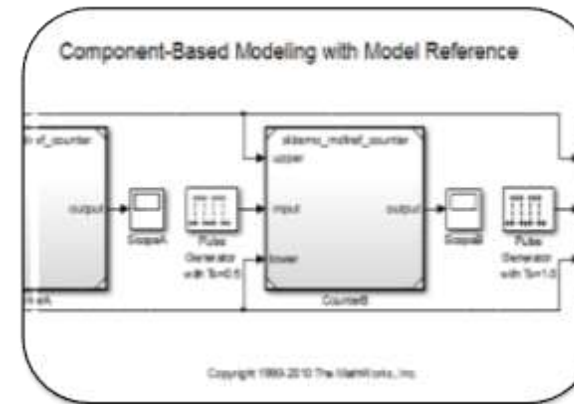
Performance Scalability

Big data workflow

- Processing large amount of simulation inputs / outputs

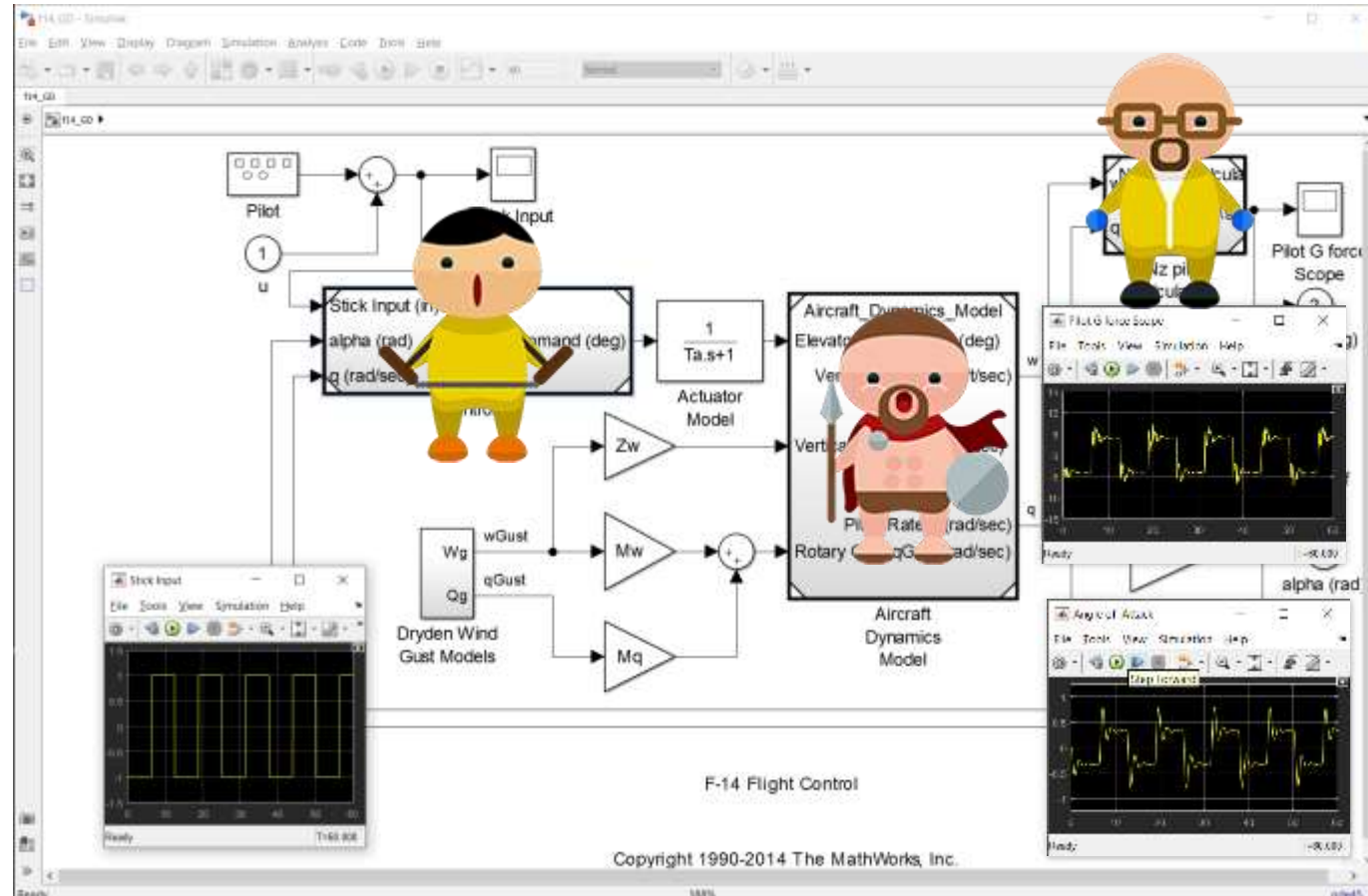


Scalability Challenges

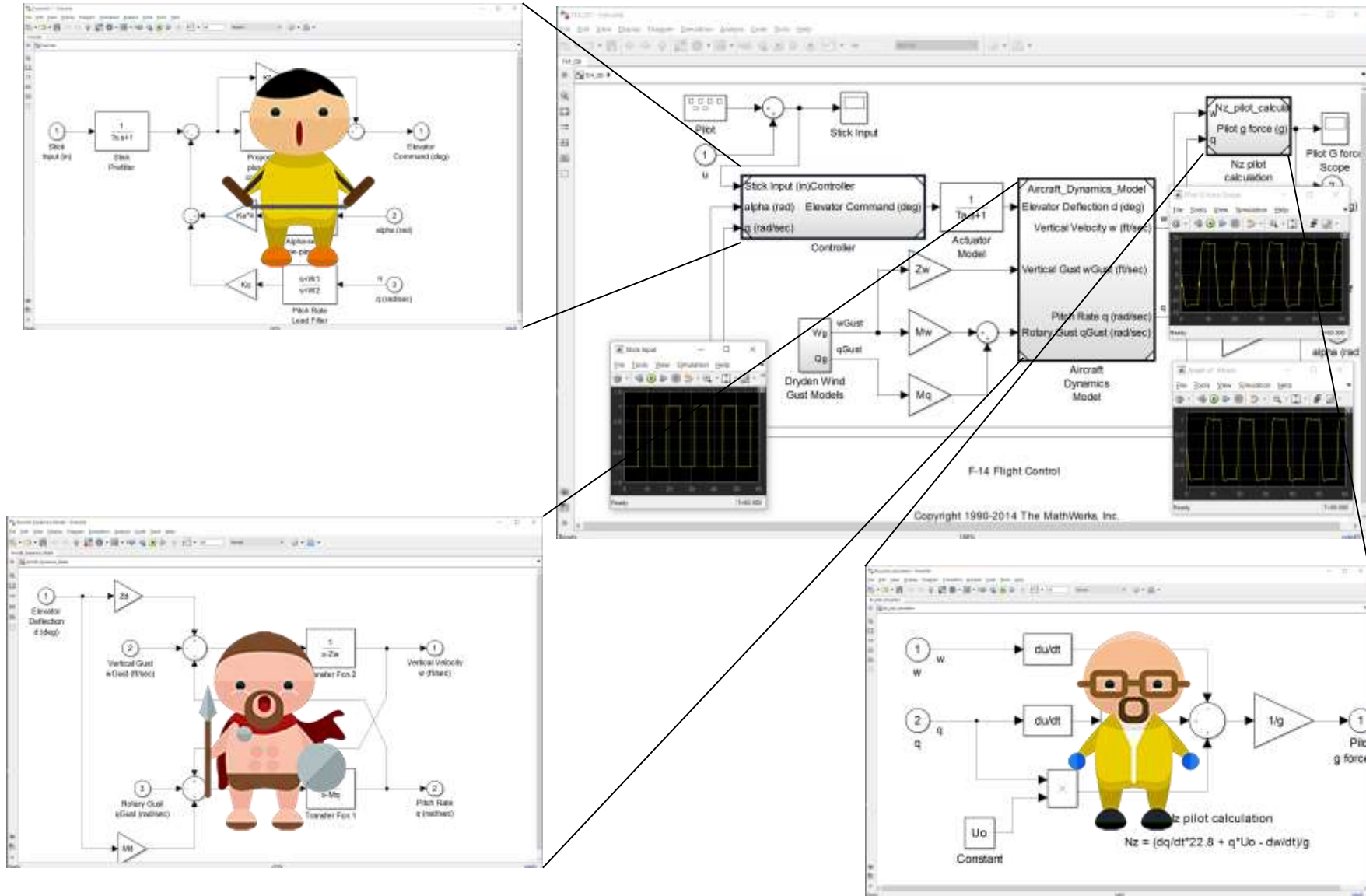


Componentization

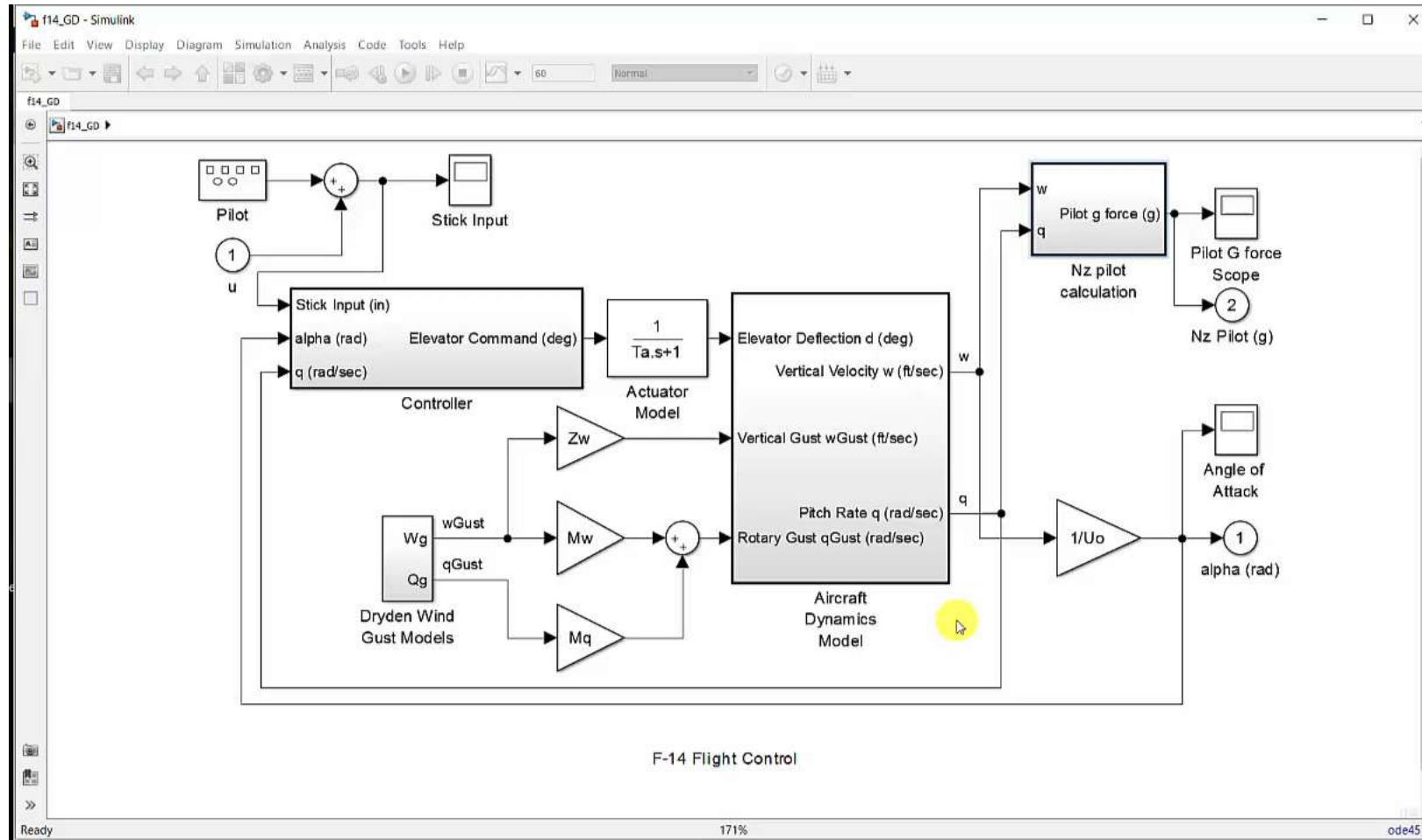
Complex Design Development through Componentization



Partitioning a Model using Model Referencing Technique

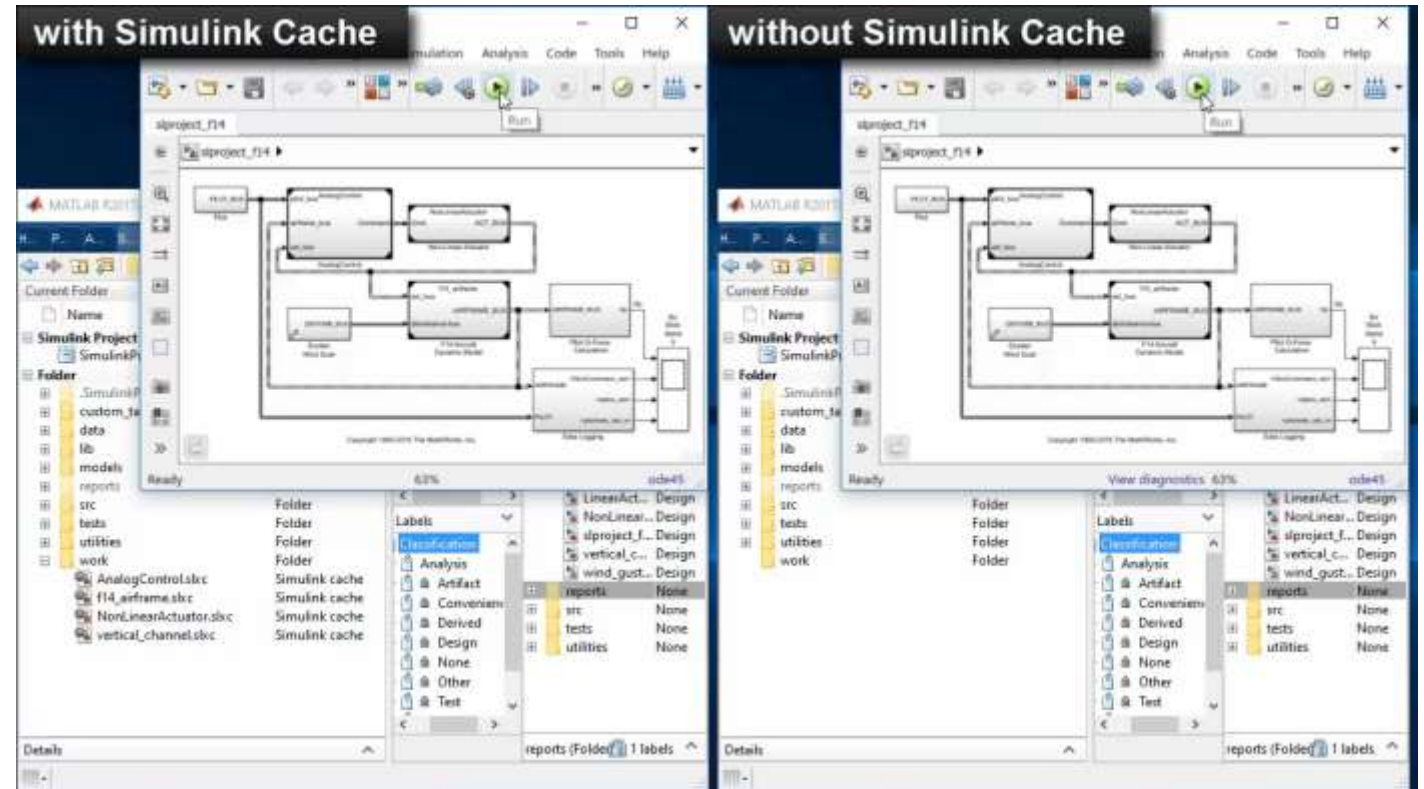


Partitioning a Model using Model Referencing Technique



Improve Performance by Team Sharing and Reusing of Model Artifacts – Simulink Cache

- Get simulation results faster by using pre-built model artifacts
- Share Simulink Cache easily with your team members
- Reduce unnecessary builds



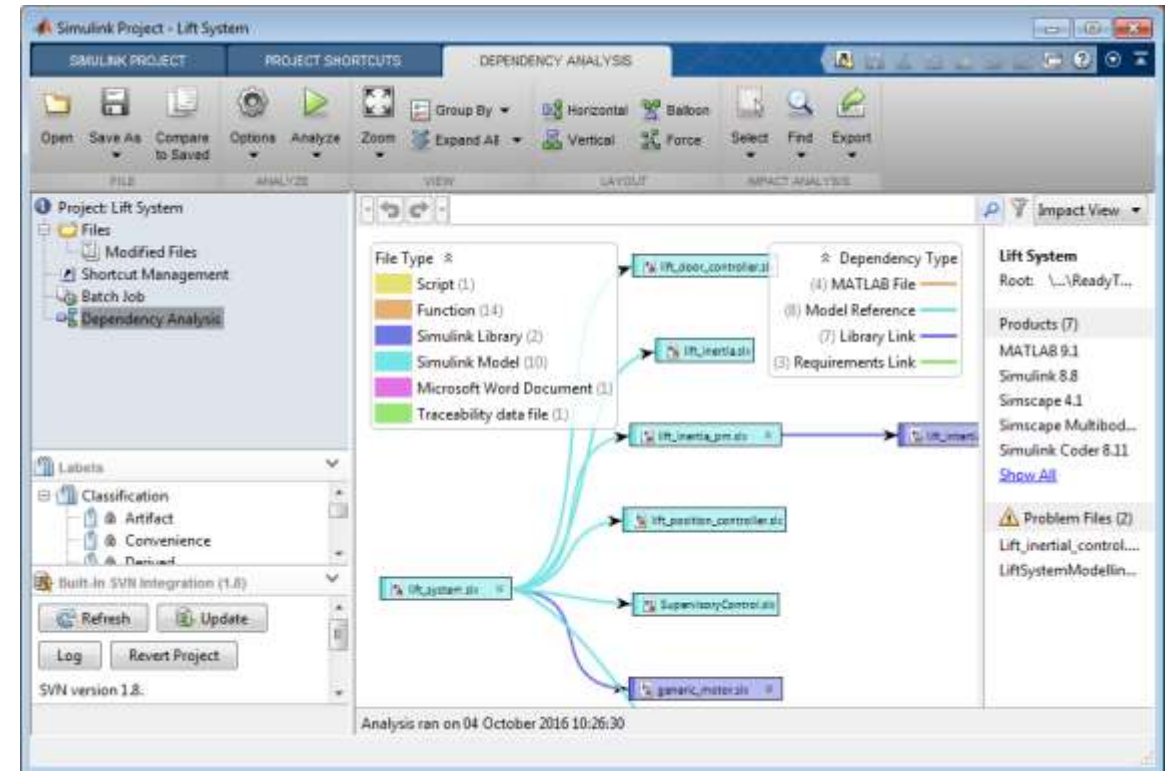
Scalability Challenges



Team Workflows

Capabilities Enabling Team Workflows

- Source control
- Design comparison and merging
- Dependency analysis
- Task automation



Source Control Integrations

Microsoft Team Foundation Server (TFS) integration available now from MathWorks File Exchange



Products Solutions Academia Support **Community** Events

File Exchange

TFS Version Control Integration

by [Jasper Schneider](#)

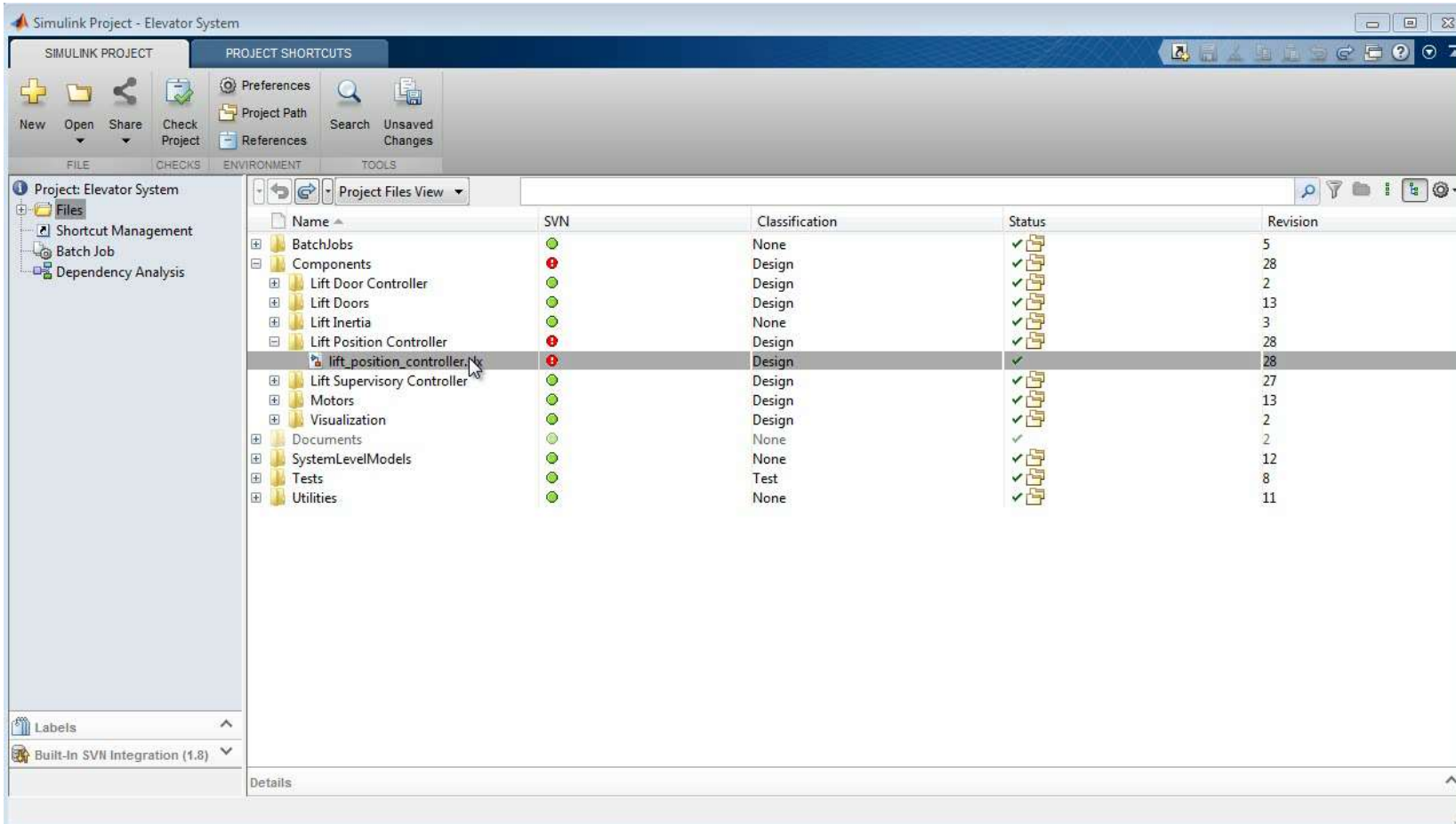
17 May 2016 (Updated 26 May 2016)

TFS Version Control integration in MATLAB and Simulink

[Watching this File](#)



Integrating Work from Different Engineers via Merge



The screenshot displays the Simulink Project - Elevator System interface. The main window shows the Project Files View, which is a table listing project files and their status. The table has columns for Name, SVN, Classification, Status, and Revision. The 'lift_position_controller' file is highlighted, indicating it is the current focus of the merge operation.

Name	SVN	Classification	Status	Revision
BatchJobs	✓	None	✓	5
Components	✓	Design	✓	28
Lift Door Controller	✓	Design	✓	2
Lift Doors	✓	Design	✓	13
Lift Inertia	✓	None	✓	3
Lift Position Controller	✓	Design	✓	28
lift_position_controller	✓	Design	✓	28
Lift Supervisory Controller	✓	Design	✓	27
Motors	✓	Design	✓	13
Visualization	✓	Design	✓	2
Documents	✓	None	✓	2
SystemLevelModels	✓	None	✓	12
Tests	✓	Test	✓	8
Utilities	✓	None	✓	11

- Supports concurrent engineering
- Lets you concentrate on design

Dependency Analysis – Modular Development

Simulink Project - Elevator System

SIMULINK PROJECT PROJECT SHORTCUTS

Lift History
 Lift System Modelling

Reset Default Model Template Set Default Model Template
 Reset Team Prefs Set Team Prefs
 Reset slprj Set slprj

Lift System Generate ICD

MANAGE DOCUMENTATION ENVIRONMENT TOP LEVEL MODELS UTILITIES

Project: Elevator System

Files

- Shortcut Management
- Batch Job
- Dependency Analysis

Project Files View

Name	Path	Status	Classification
Lift	\$\Tests	✓	Test
Lift Door Controller	\$\Components	✓	Design
Lift Door Controller	\$\Tests	✓	Test
Lift Doors	\$\Components	✓	Design
Lift Doors	\$\Tests	✓	Test
Lift Inertia	\$\Components	✓	None
Lift Motor	\$\Tests	✓	Test
Lift Position Controller	\$\Components	✓	Design
Lift Position Controller	\$\Tests	✓	Test
Lift Supervisory Controller	\$\Components	✓	Design
Lift Supervisory Controller	\$\Tests	✓	Test
Motors	\$\Components	✓	Design
SystemLevelModels	\$\	✓	None
Tests	\$\	✓	Test
Utilities	\$\	✓	None
Visualization	\$\Components	✓	Design
Visualization	\$\Tests	✓	Test
basic_animation.slx	\$\Components\Visualization	✓	Design
ElevatorTemplate.slx	\$\Utilities	✓	Other
exportToR2016a.m	\$\BatchJobs	✓	Design
generateBillOfMaterials.m	\$\BatchJobs	✓	Design
generateICD.m	\$\Utilities	✓	Design
generic_motor.slx	\$\Components\Motors	✓	Design
history.m	\$\Utilities	✓	Design
lift_door.req	\$\Components\Lift Doors	✓	Design
lift_door.slx	\$\Components\Lift Doors	✓	Design
lift_door_controller.slx	\$\Components\Lift Door Controller	✓	Design

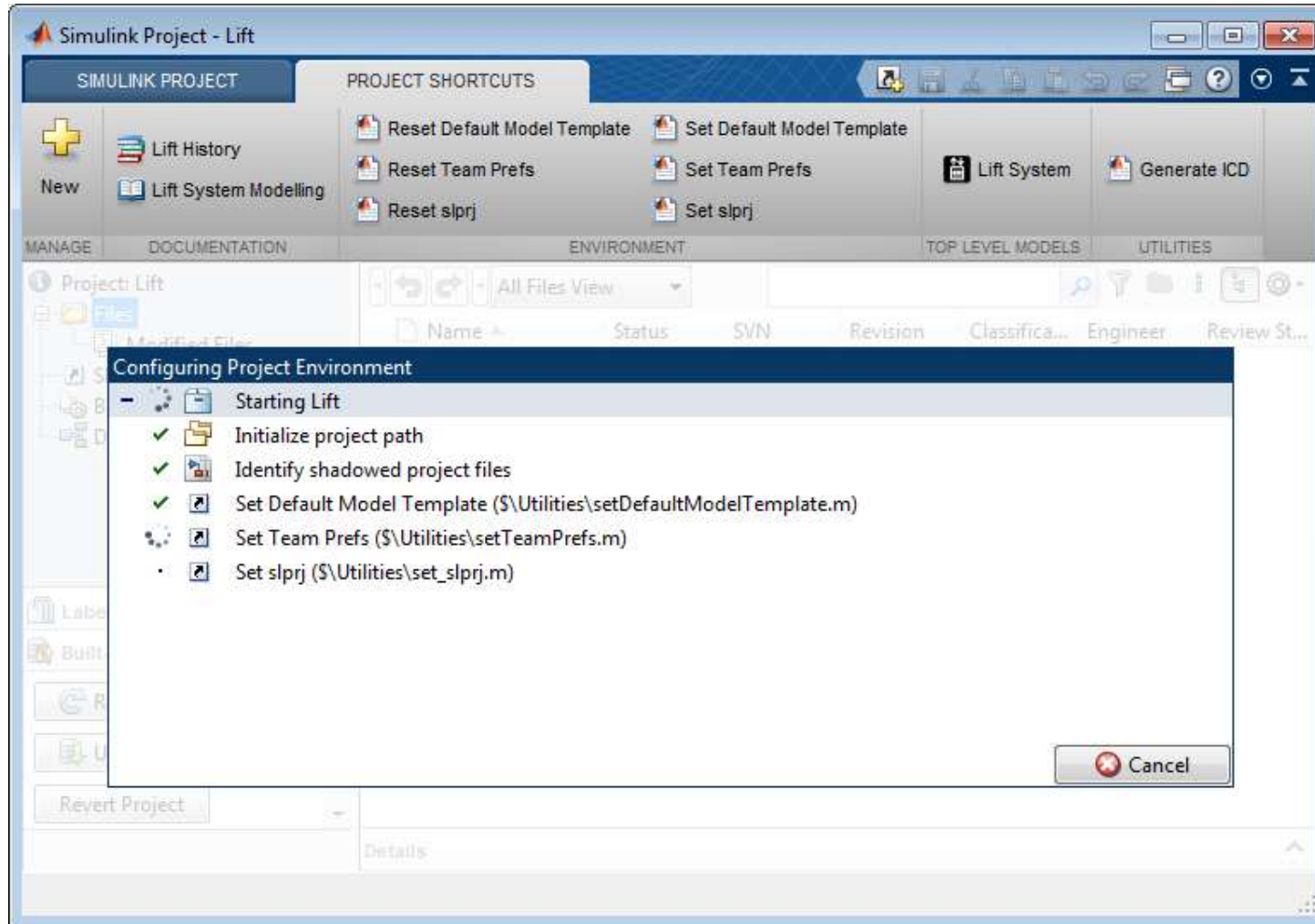
Labels

Details

Dependency Analysis – Modular Development

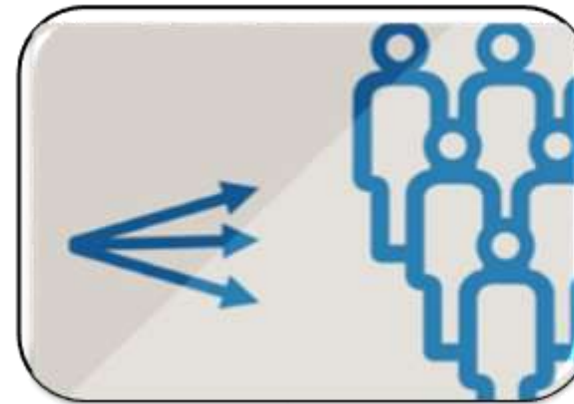
The screenshot displays the Simulink Dependency Analysis interface. The main workspace shows a dependency graph for the project 'lift_system_orig'. The root node is 'lift_system.slx', which has arrows pointing to several sub-modules: 'generic_motor.slx', 'lift_door_controller.slx', 'lift_inertia.slx', 'lift_inertia_pm.slx', and 'lift_position_controller.slx'. The 'lift_inertia_pm.slx' node has an arrow pointing to 'lift_inertia_utils.slx'. A separate dependency graph for 'LiftSystemModelling.docx' is also visible, with arrows pointing to 'lift_door.req', 'lift_system_Harness1.slx', and 'lift_system.slx'. The right-hand pane shows a list of products (MATLAB 9.1, Simulink 8.8, Simscape 4.1, Simscape Multibody 4.9, Simulink Coder 8.11) and a section for 'Problem Files (1)' containing 'LiftSystemModelling.docx'. Three callout boxes are overlaid on the image: 'List products required' points to the product list; 'Show model structure' points to the dependency graph; and 'Highlight issues' points to the 'Problem Files' section.

Task Automation – Configuring Project Environment



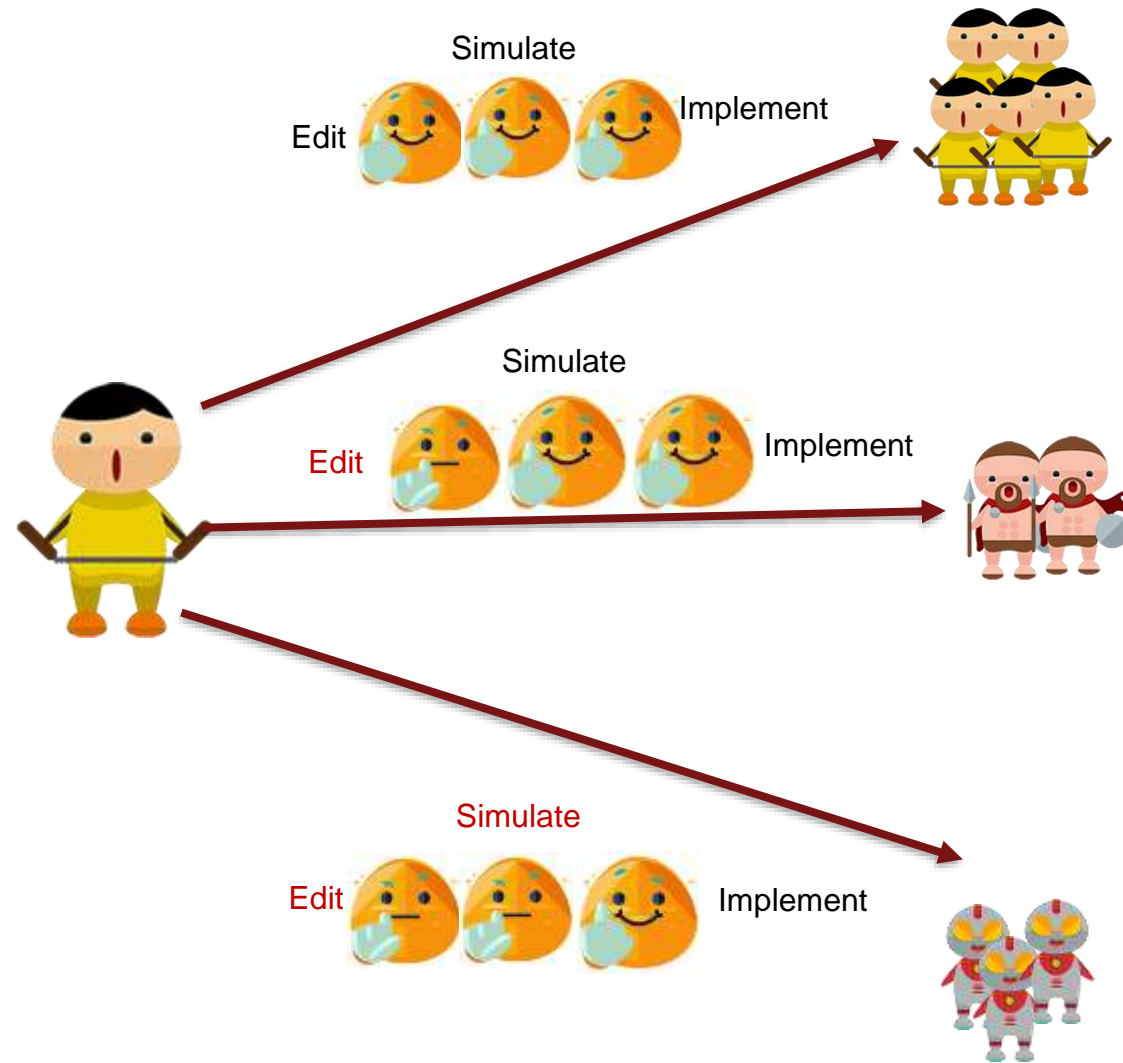
- Robustly configure the team environment
- For everyone
- Automatically

Scalability Challenges

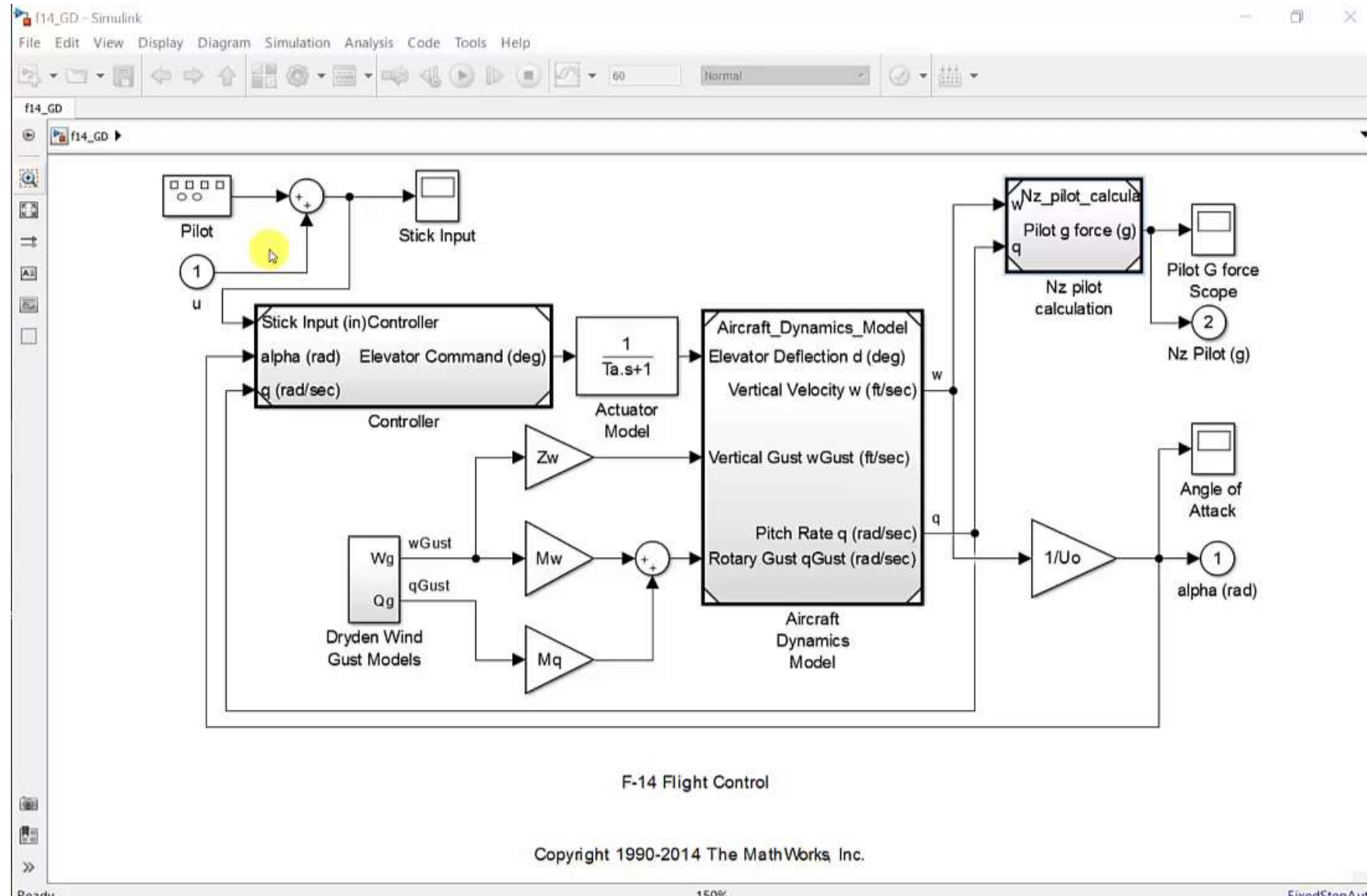


Sharing

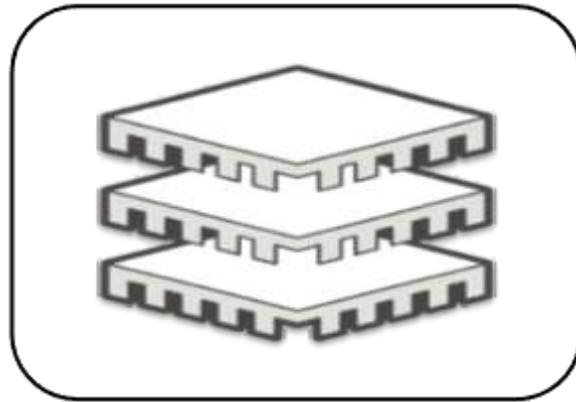
Sharing models with access control



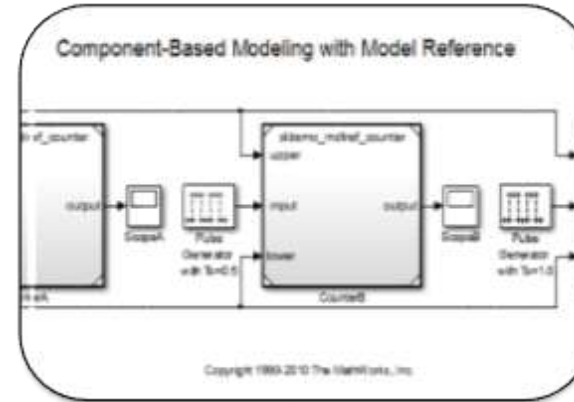
Protecting your Intellectual Property (IP)



Simulink Addressing Scalability Challenges



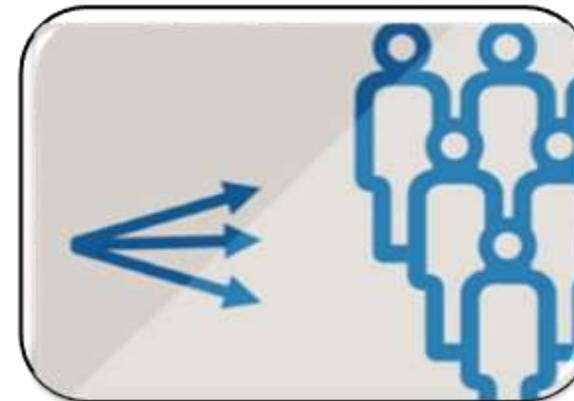
Performance



Componentization

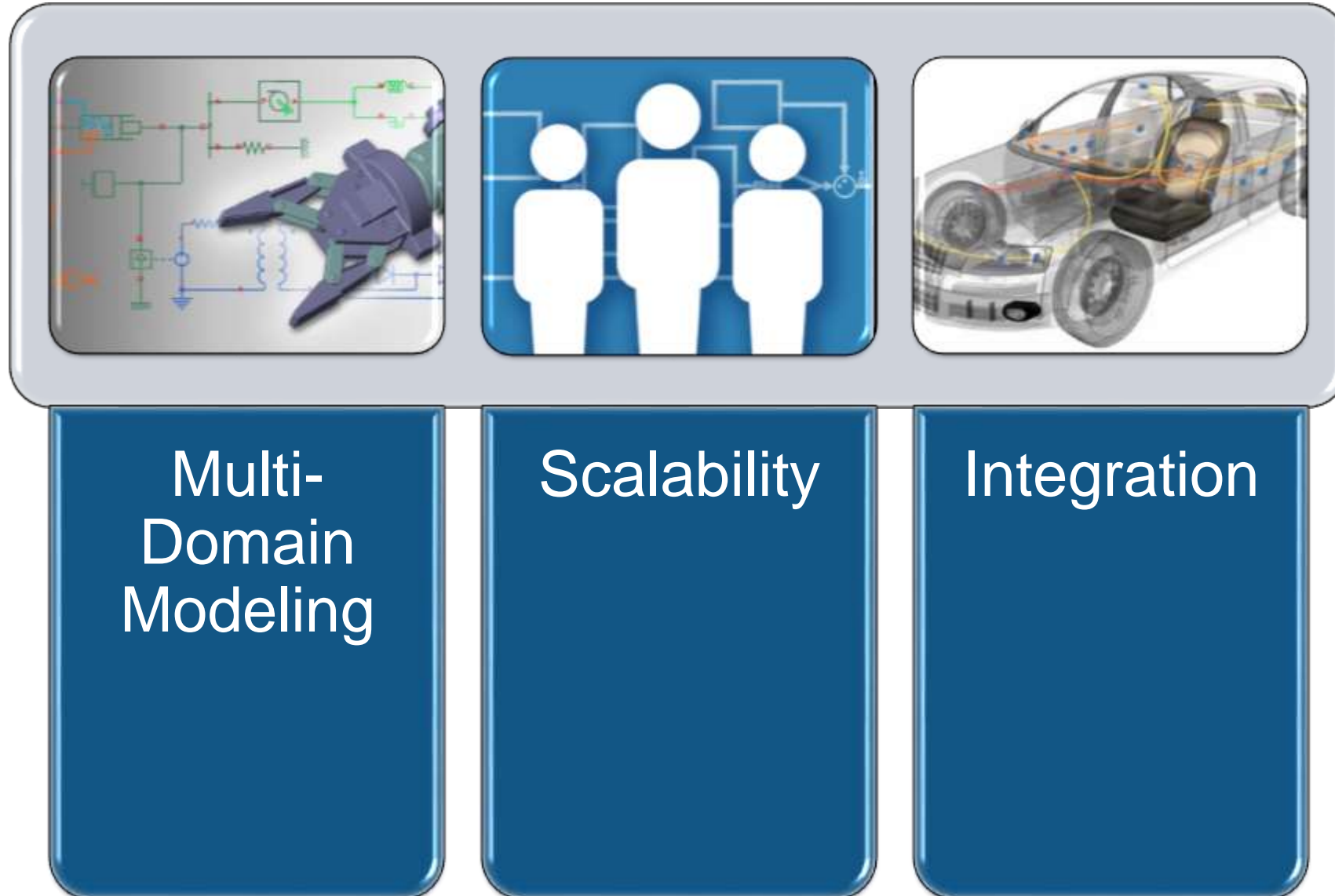


Team Workflows

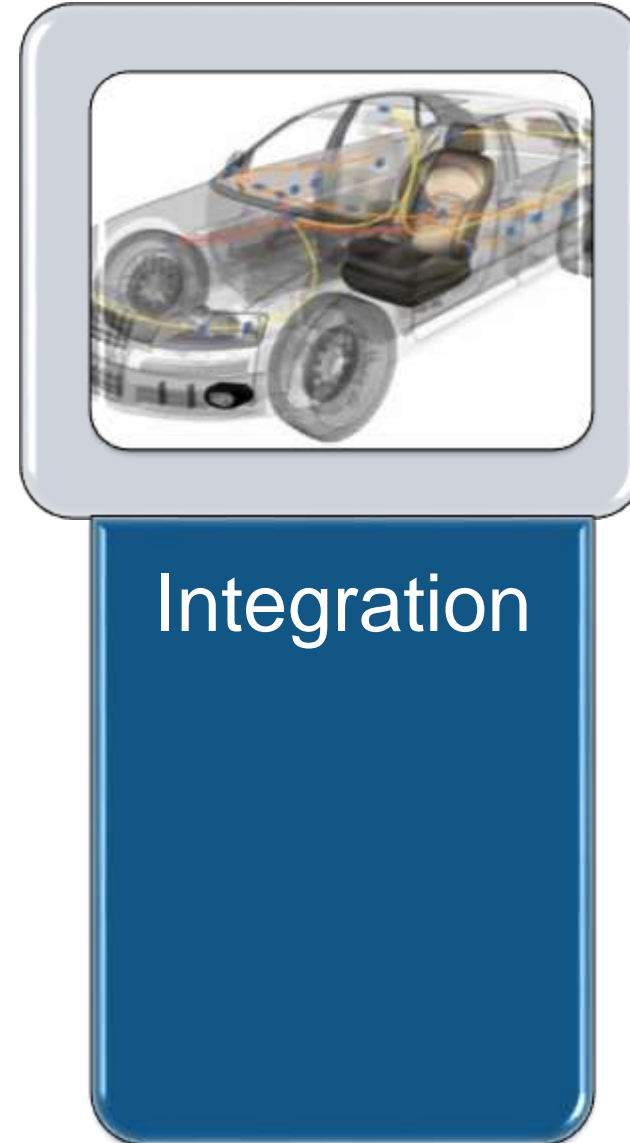


Sharing

Enterprise Simulation Platform Enablers

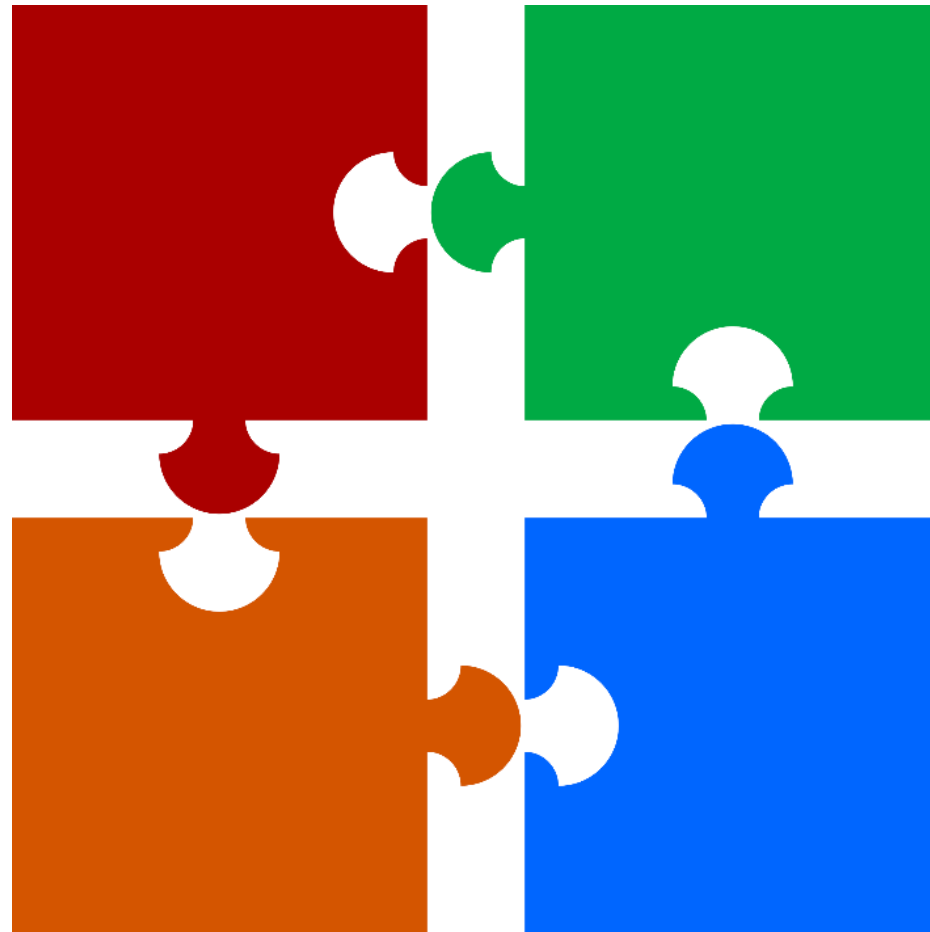


Enterprise Simulation Platform Enablers



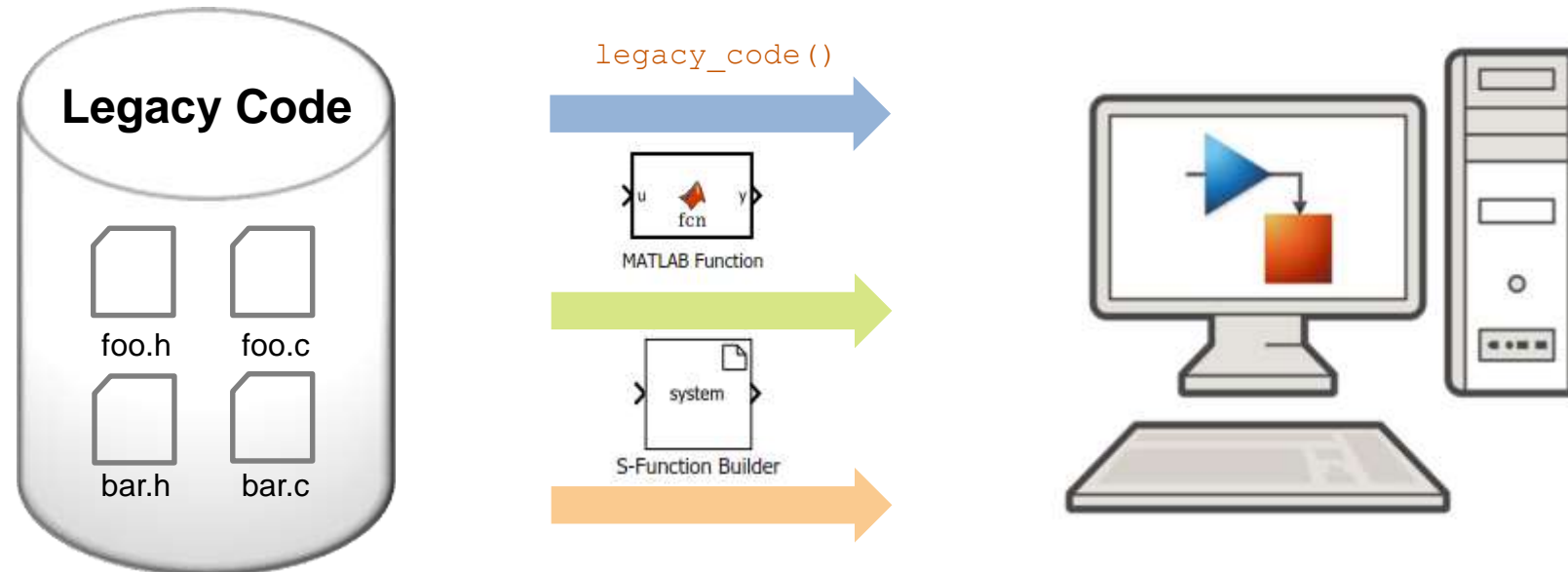
Disconnected Component Intellectual Property (IP)

Your IP exists in many forms and in many locations, making integration difficult



Integrating Your Code

Multiple ways to reuse your legacy code with Simulink



Legacy Code Tool

- Legacy Code Tool automates creation of S-Function block
- Call existing, external functions as part of a Simulink simulation
- Code generation is allowed with Legacy Code Tool blocks

```
1 #include "SimpleTable.h"
2
3
4 double SimpleTable(double xIn, double xAxis[], double yAxis[], int axisLength)
5 {
6     /* This routine assumes monotonically increasing values of X */
7     double outValue;
8     int axisLoc;
9     axisLoc = (int) (axisLength * 0.5); /* Start at mid point of table */
10
11     if (xIn >= xAxis[axisLength - 1])
12     {
```

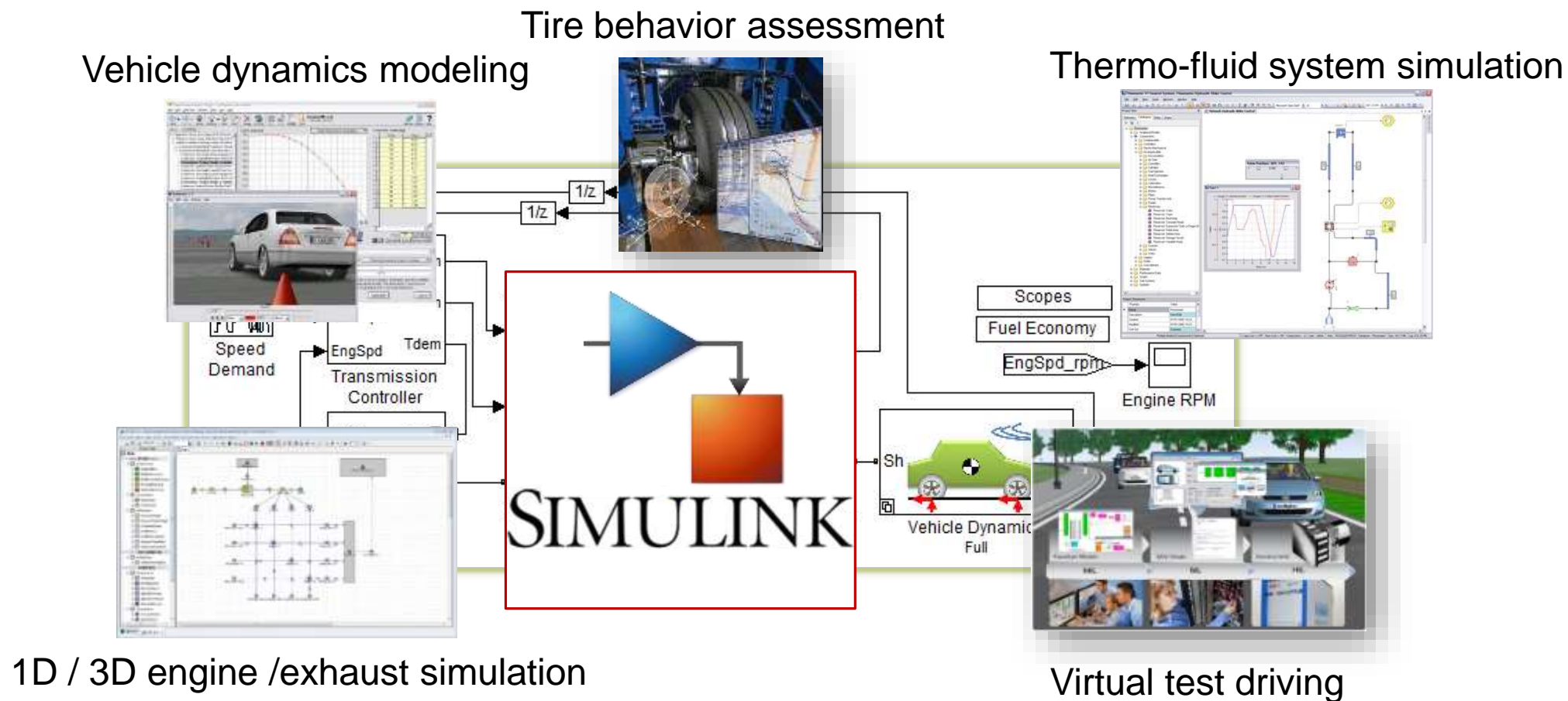
Command Window
fx >>

Legacy Code Tool Example:
Generate a 1D lookup table block from an existing C code

C / CPP source or header file | Ln: 1 | Col: 1

Integrating Third-Party Simulation Tools

Mature and extensive APIs for third-party tool integration



Tool Integration Made Easy

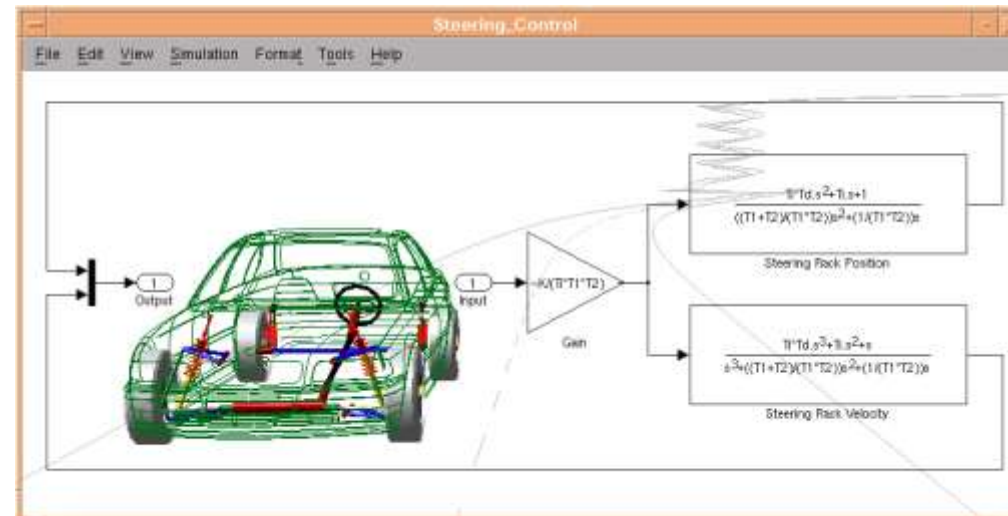
- Numerous tool integration interfaces with Simulink are maintained by our partners for you
- Typical interface can be one or all of the following:
 - Export of linear matrices from partner tool to Simulink
 - Export of non-linear partner tool model and solver to Simulink
 - Co-simulation of partner and Simulink

SIMPACK

Complete multibody simulation in combination with MATLAB

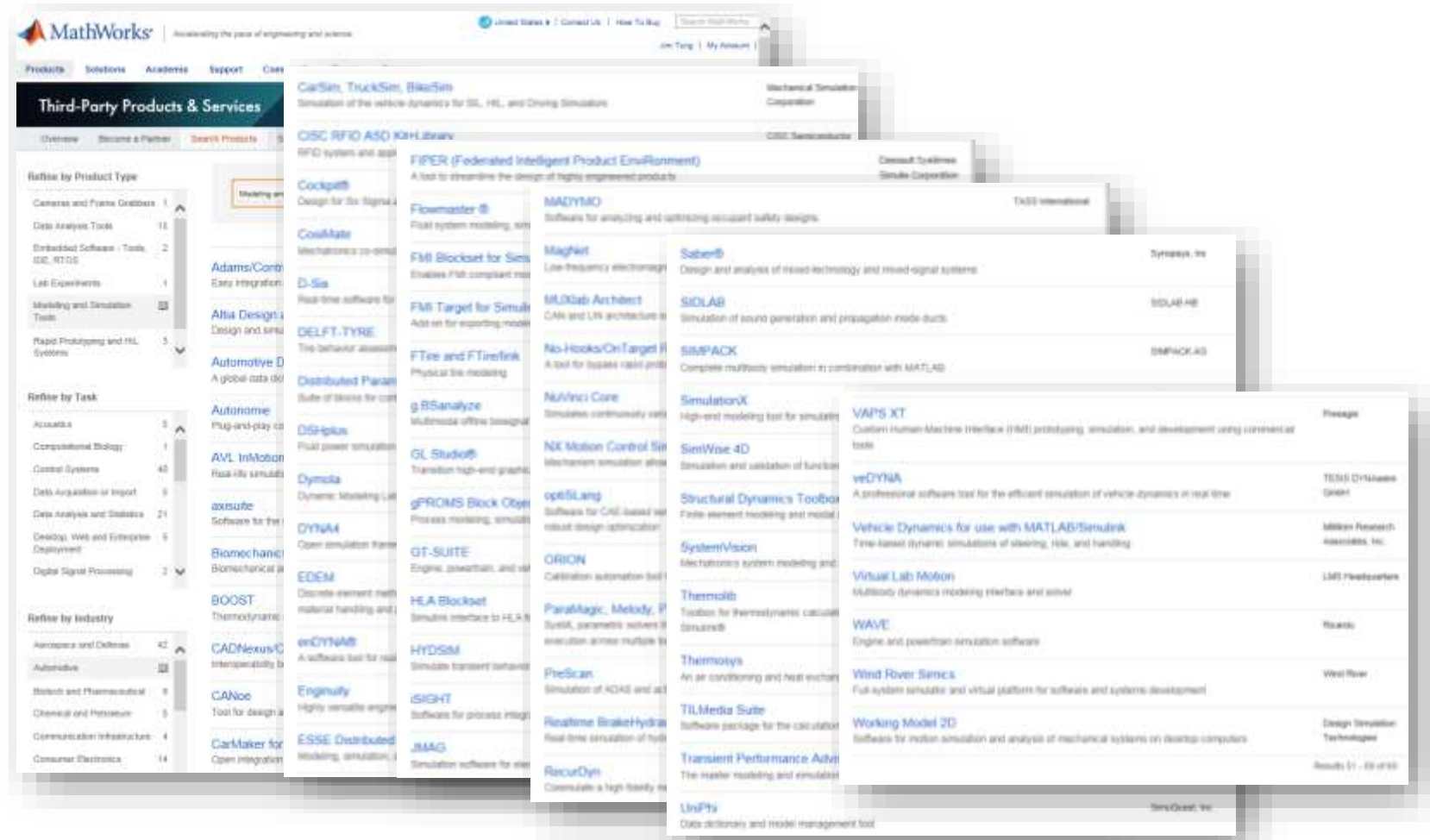
Highlights

- General 3D multibody simulation
- Export of ABCD matrices from SIMPACK to Simulink[®]
- Cosimulation of SIMPACK and Simulink
- Import of Simulink model to SIMPACK
- Export of nonlinear SIMPACK model and solver to Simulink



Partner Ecosystem

Numerous partners provide interface to Simulink



Customer Success in Simulation Integration

Develop Integrated Vehicle Safety Applications

Siddharth D'Silva, Principal Engineer

Autoliv



Challenge

- Design and validate safety-critical algorithms before implementation

Solution

- Leverage Simulink as a platform by integrating third-party software

Customer Success In Simulation Integration

“Seamless integration with third party software solutions enables rigorous development in a safe environment. For application engineers or system engineers, it is very useful that you can export these complex third-party tool functionalities in the form of S-functions and run co-simulation.”

Siddharth D’Silva, Autoliv



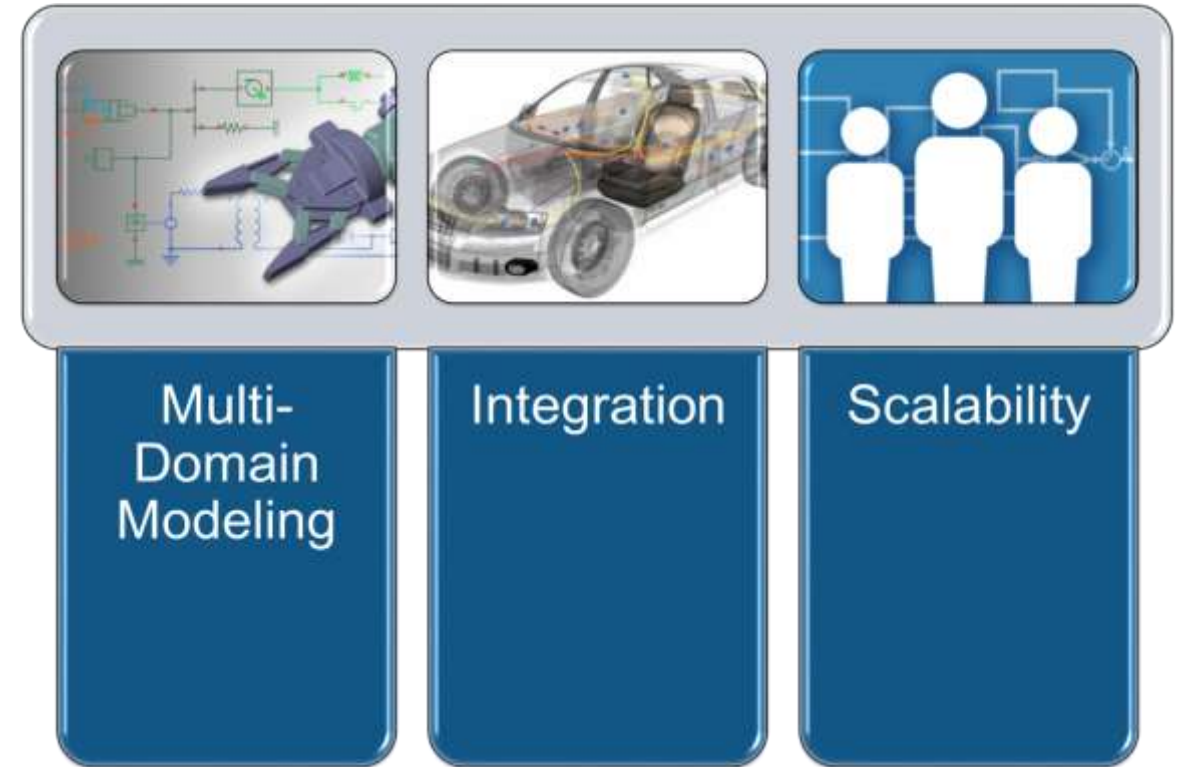
Results

- Industry first integration of stability control inertial sensor into airbag control unit
- Restraint control module software development time reduced by 30%

Simulink as Enterprise Simulation Platform

*“There is no such tool, which gives the simulation environment as well as the hardware verification and validation. In a single environment, I am getting these together. **That is why I use MATLAB and Simulink.**”*

Dr. Deepak Mishra,
Indian Space Research Organization



Training Services

Exploit the full potential of MathWorks products

Flexible delivery options:

- Public training available in several cities
- Onsite training with standard or customized courses
- Web-based training with live, interactive instructor-led courses



More than 48 course offerings:

- Introductory and intermediate training on MATLAB, Simulink, Stateflow, code generation, and Polyspace products
- Specialized courses in control design, signal processing, parallel computing, code generation, communications, financial analysis, and other areas

Simulink as Your Enterprise Simulation Platform

- **Simulink for System and Algorithm Modeling**
 - This two-day course is for engineers who are new to system and algorithm modeling and design validation in Simulink. The course demonstrates how to apply basic modeling techniques and tools to develop Simulink block diagrams
- **Stateflow for Logic-Driven System Modeling**
 - This two-day course shows how to implement complex decision flows and finite-state machines using Stateflow®. The course focuses on how to employ flow charts, state machines, truth tables, and state transition tables in Simulink designs
- **Simulink Model Management and Architecture**
 - This two-day course describes techniques for applying Model-Based Design in a common design workflow. It provides guidance on managing and sharing Simulink models when working in a large-scale project environment



Speaker Details

Email: Prasanna.Deshpande@mathworks.in

LinkedIn: <https://in.linkedin.com/in/deshprasan>

Twitter: [@InfPrasanna](https://twitter.com/InfPrasanna)

Contact MathWorks India

Products/Training Enquiry Booth

Call: 080-6632-6000

Email: info@mathworks.in

Your feedback is valued.

Please complete the feedback form provided to you.