MathWorks
AUTOMOTIVE
CONFERENCE 2023
India

Architecting Software Defined Vehicles through Model-Based Design

Mani Ramamurthy, MathWorks

MathWorks

IEEE Spectrum | This Car Runs on Code

Type to search

Explor

FEATURE | TRANSPORTATION

THIS CAR RUNS ON CODE

It takes dozens of microprocessors running 100 million lines of code to get a premium car out of the driveway, and this software is
get more complex

2009

BY ROBERT N. CHARETTE | 01 FEB 2009 | 7 MIN READ

Search Medium

Write

Sign up

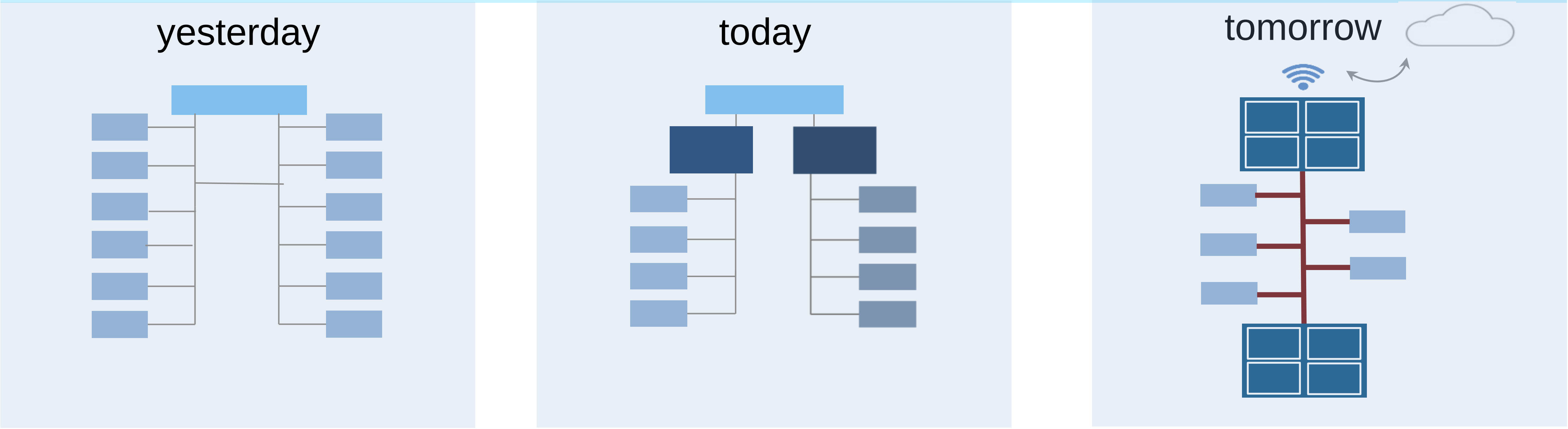
Software isn't just running our vehicles. It's defining them

Taylor Armerding · Follow

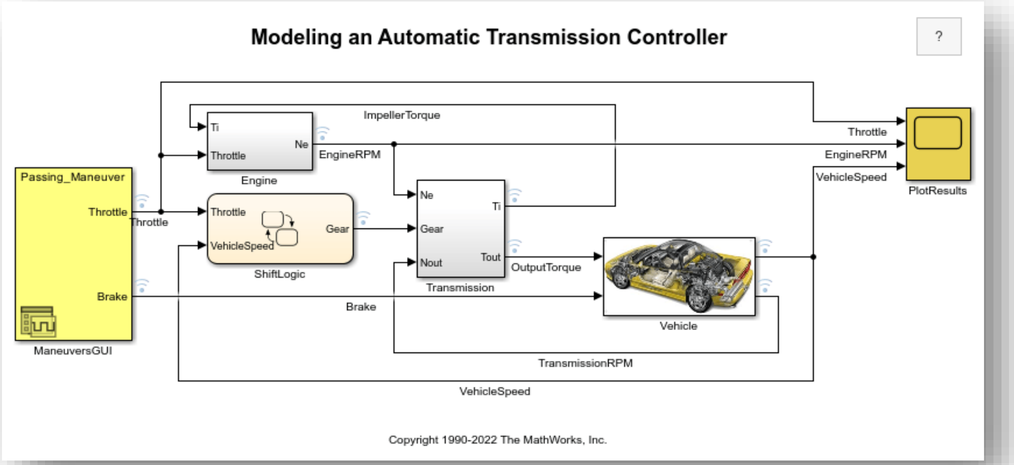
Published in Nerd For Tech · 6 min read · Jun 26

2023

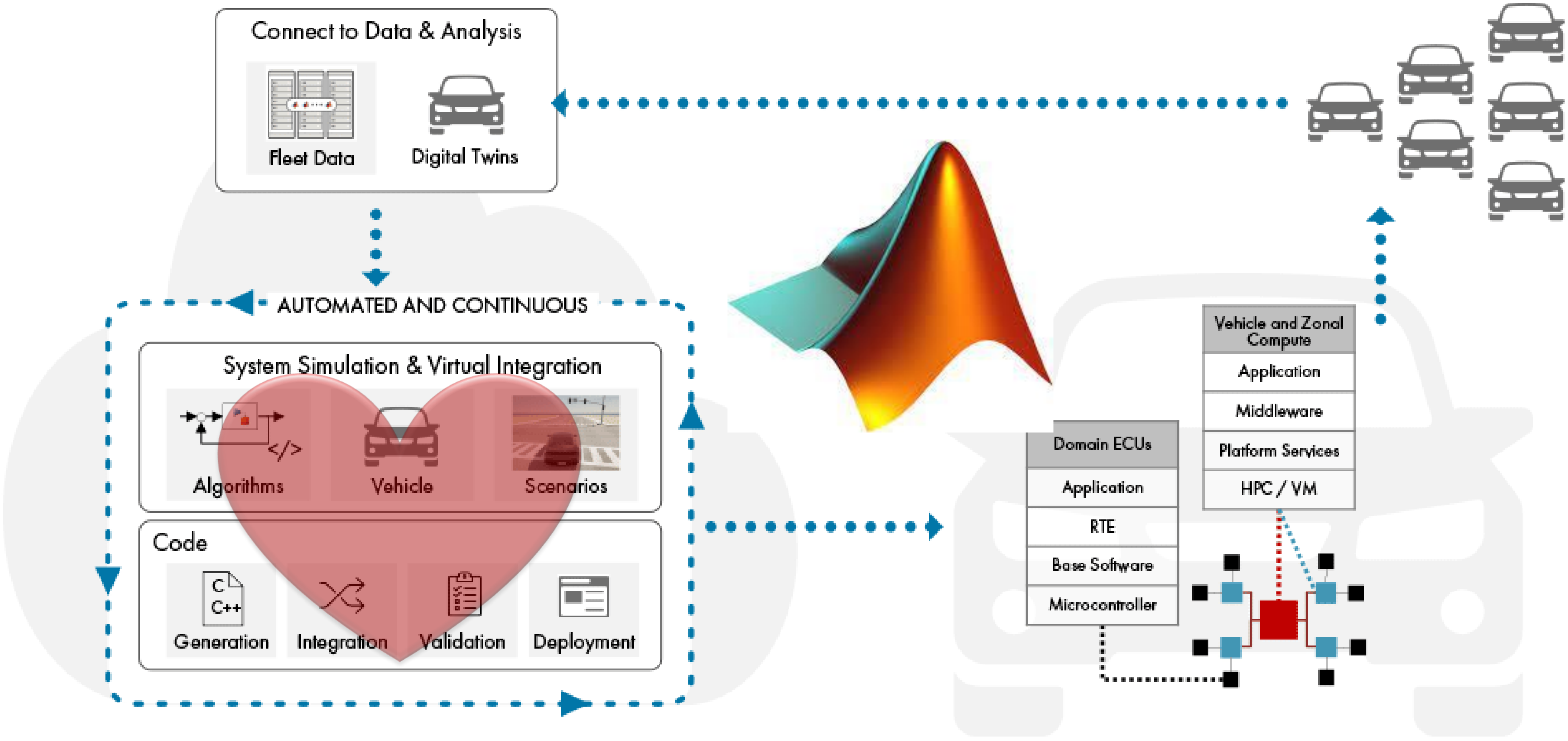
E/E architectures



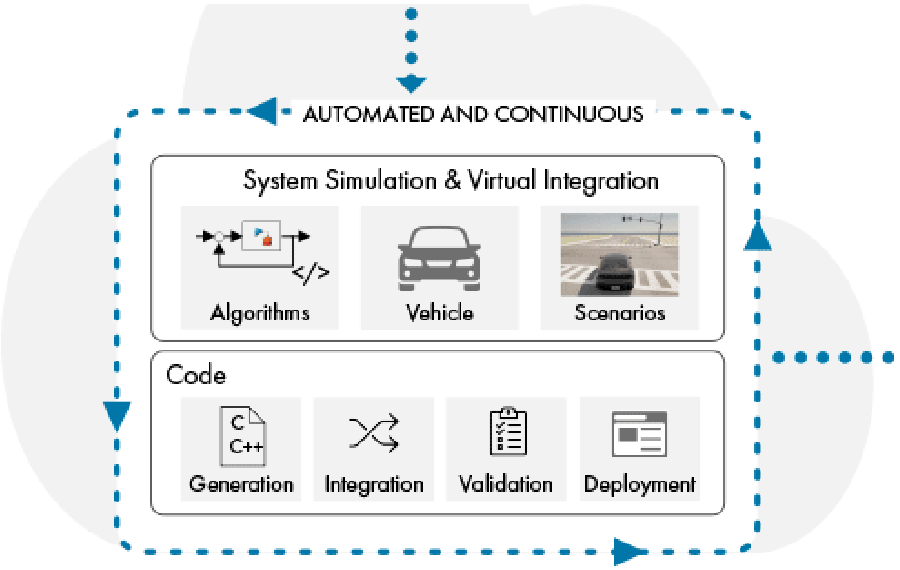
Evolving Modeling & Design Tools



?

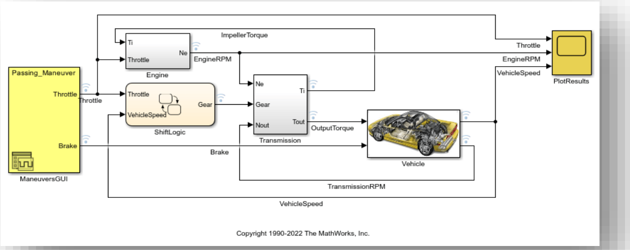

Connect to Data & Analysis
Fleet Data
Digital Twins
AUTOMATED AND CONTINUOUS
System Simulation & Virtual Integration
Algorithms
Vehicle
Scenarios
Code
C
C++
Generation
Integration
Validation
Deployment
Vehicle and Zonal Compute
Application
Middleware
Platform Services
HPC / VM
Domain ECUs
Application
RTE
Base Software
Microcontroller

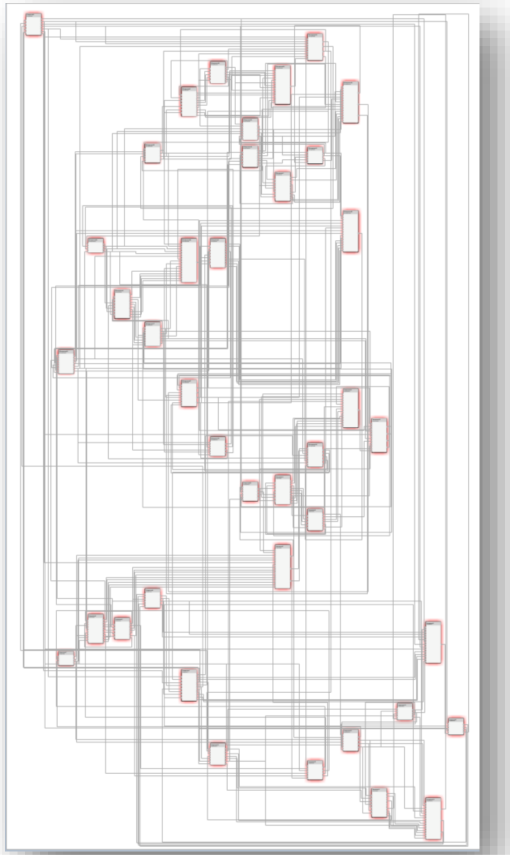
How do we align SDV Development Practices with Model-Based Design?



- Models evolve into Architectures
- Simulation evolves into Scalable Virtual Integration
- Ecosystem evolves to support new workflows

From Models to Architectures



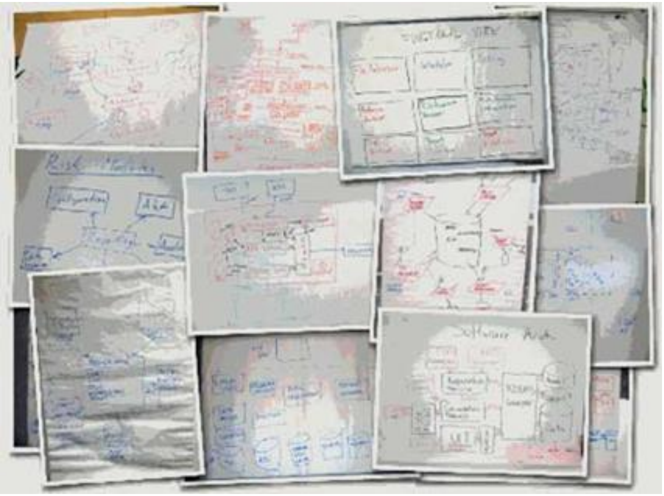


Software architectures are abstractions to get good implementations

AUTOSAR

FACE
Future Airborne Capability Environment

Facilitate the creative design process



Conform to digital engineering

Requirements

allocate

Digital Diagrams

documented by

Code

Implement leveraging Model-based Design

Run

Architectural Model

Algorithm Model

Hand Code

source of

Software System

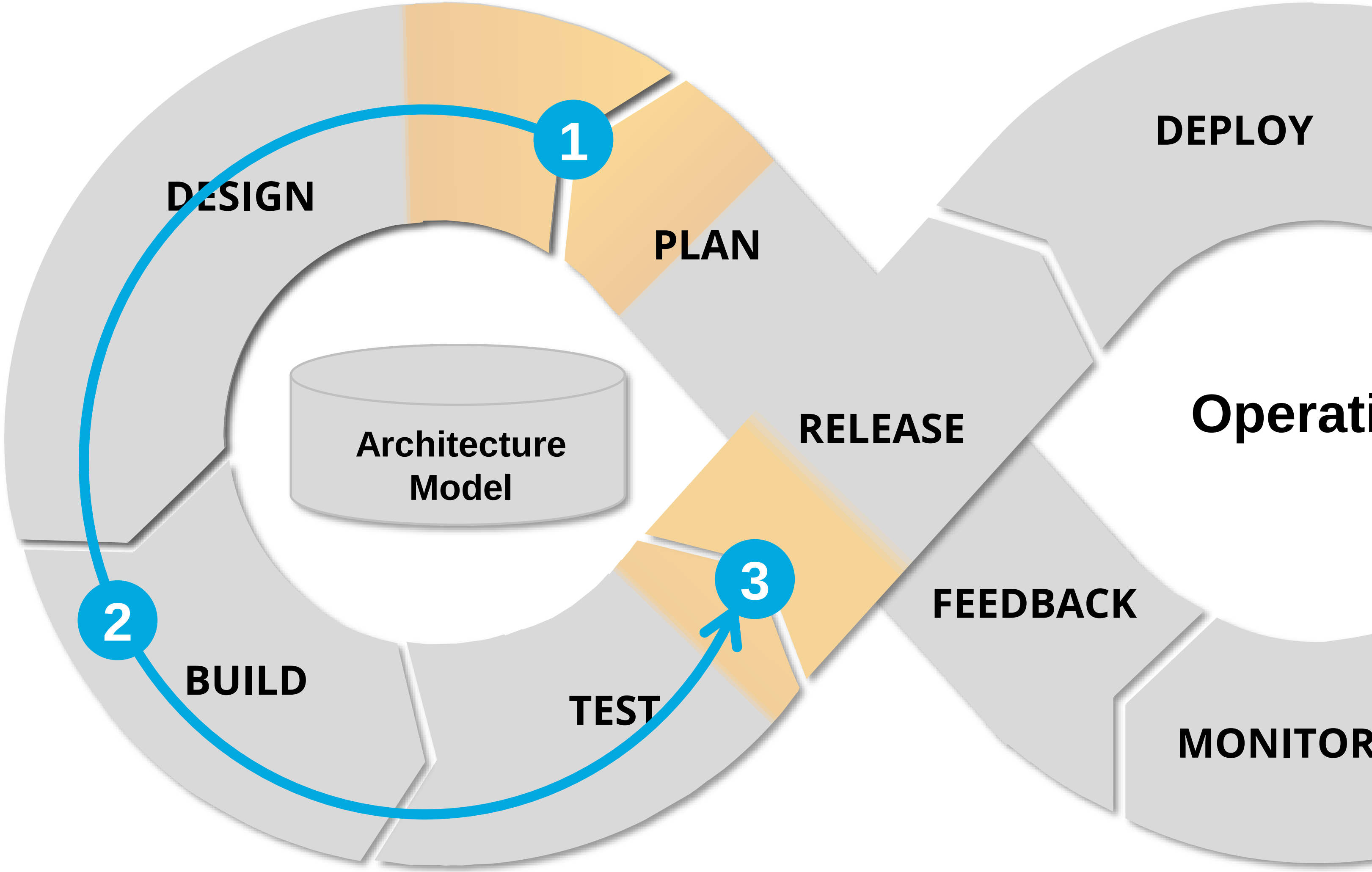
XML
Build files
C/C++

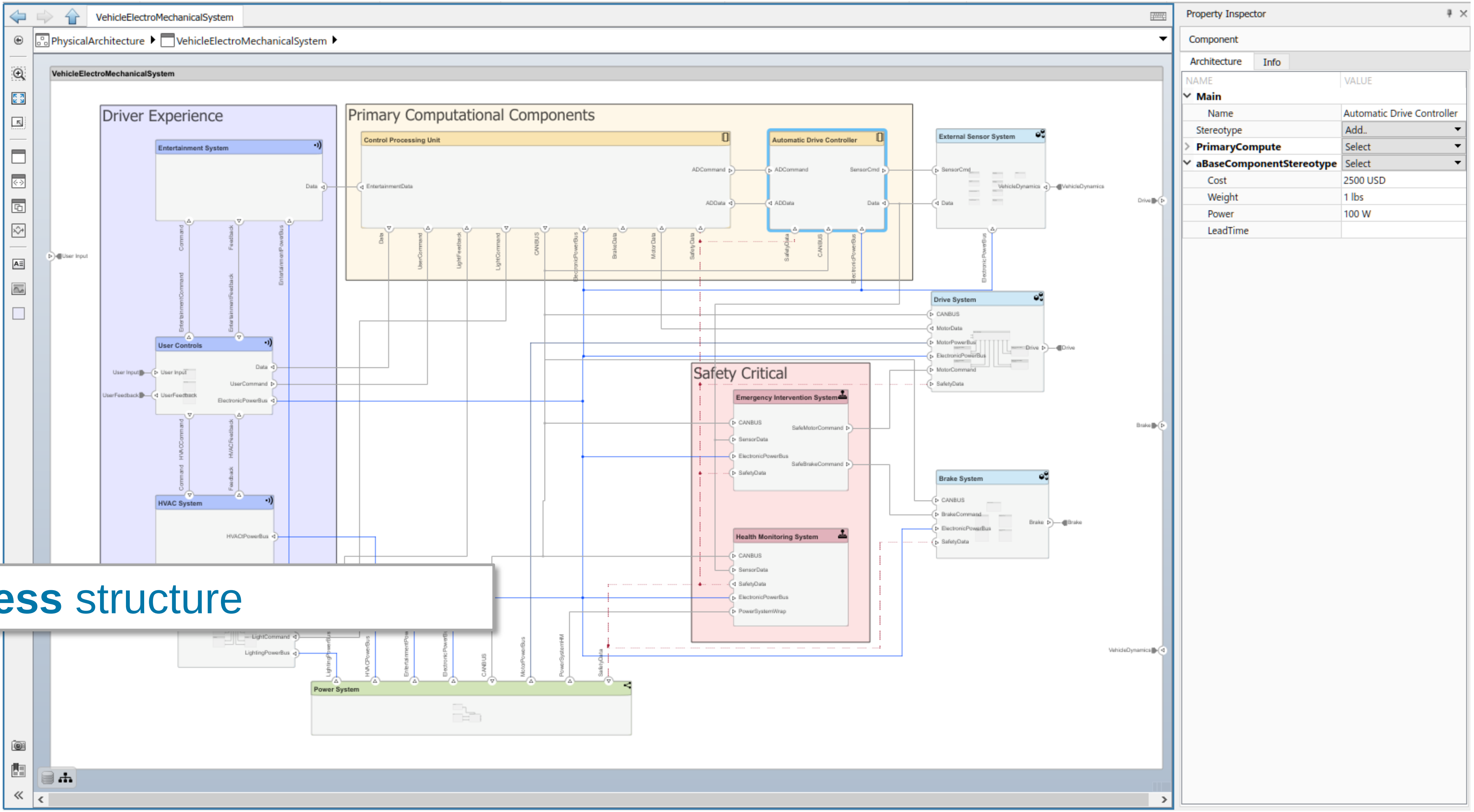
Low barrier of entry

Completes deep workflow

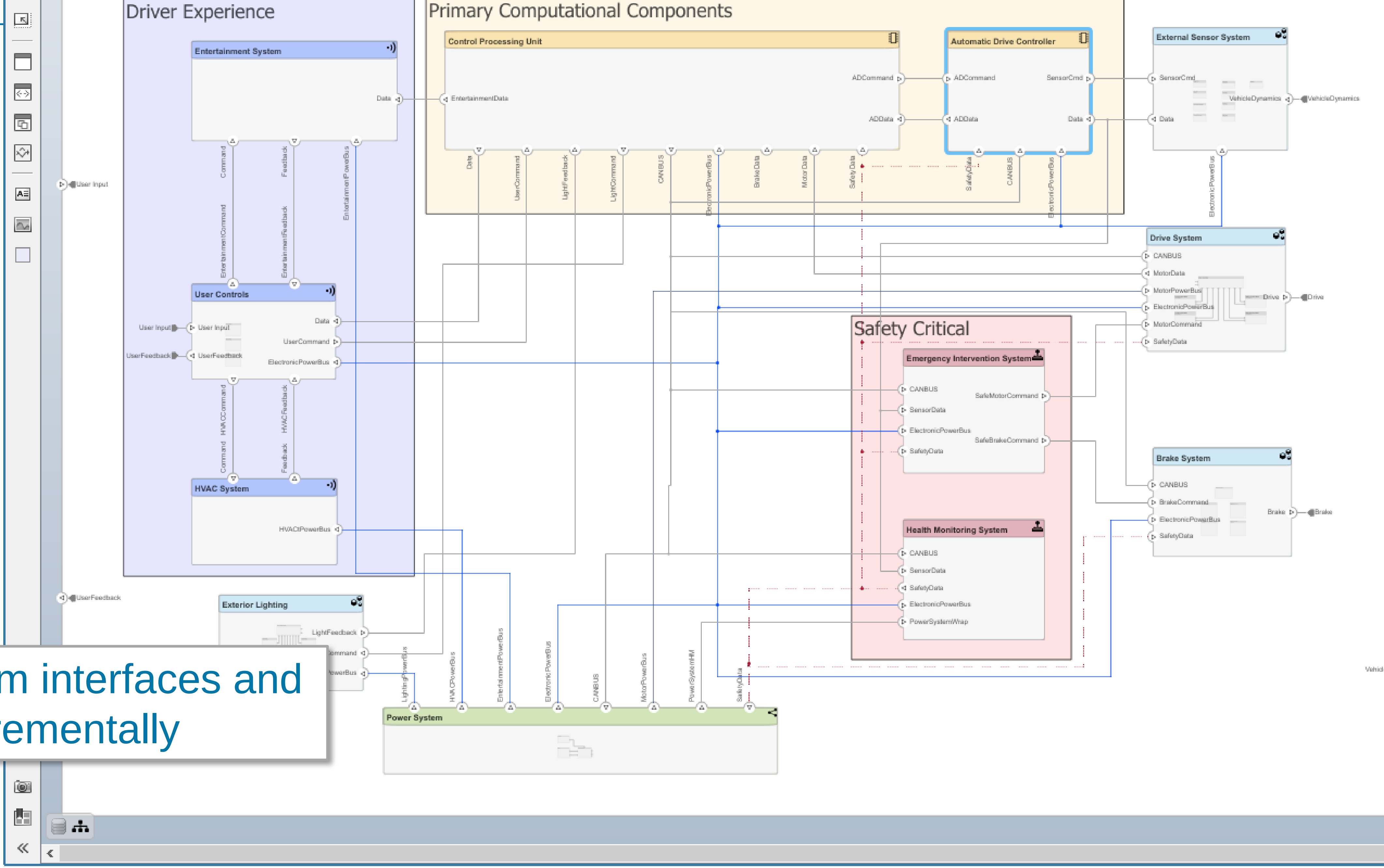
System Composer is our platform for Architecture Modeling

1. Expressive & intuitive language
2. Seamless workflow between MBSE, Software Architecture and MBD
3. Rigor for virtual system integration, test, and implementation

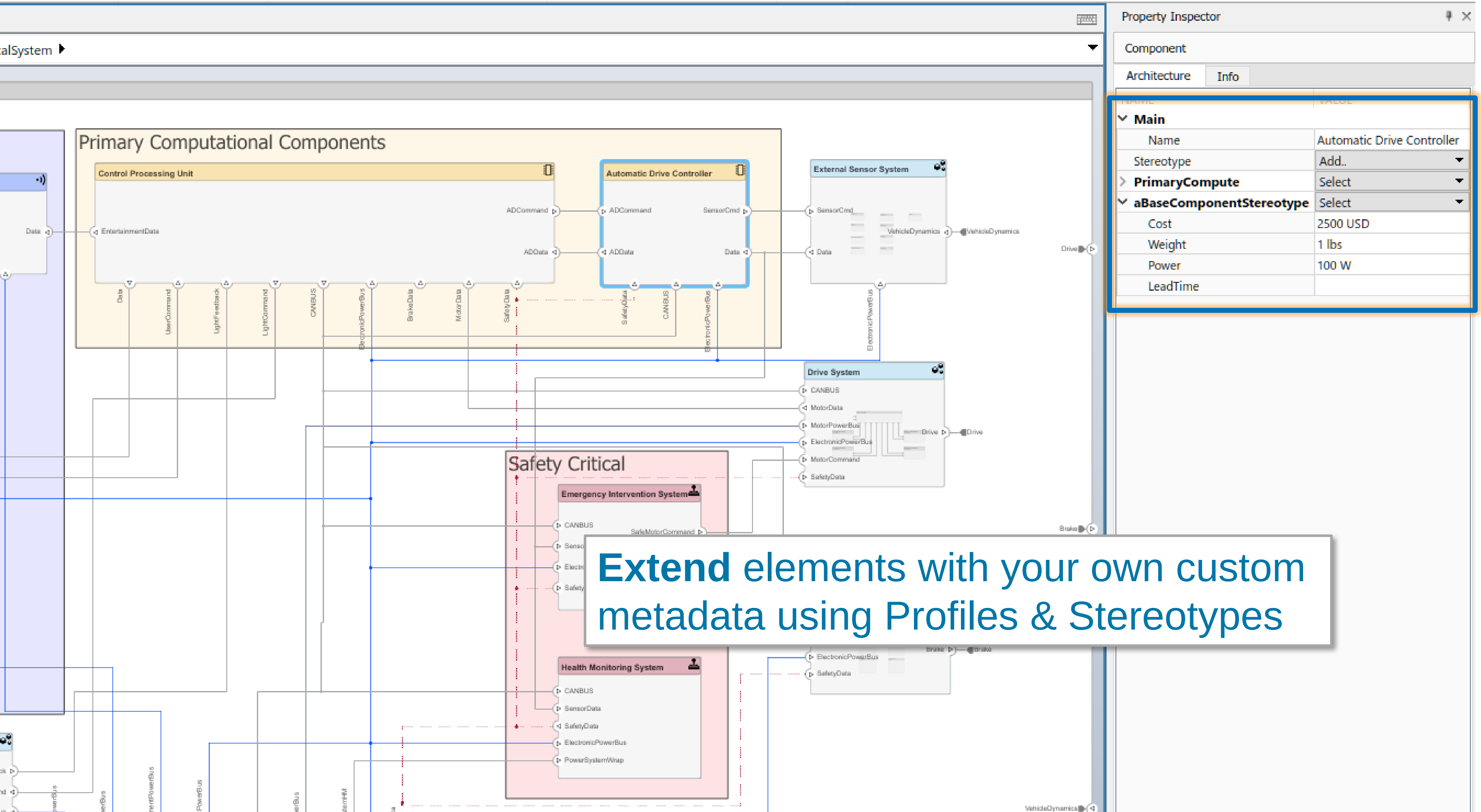




Express structure

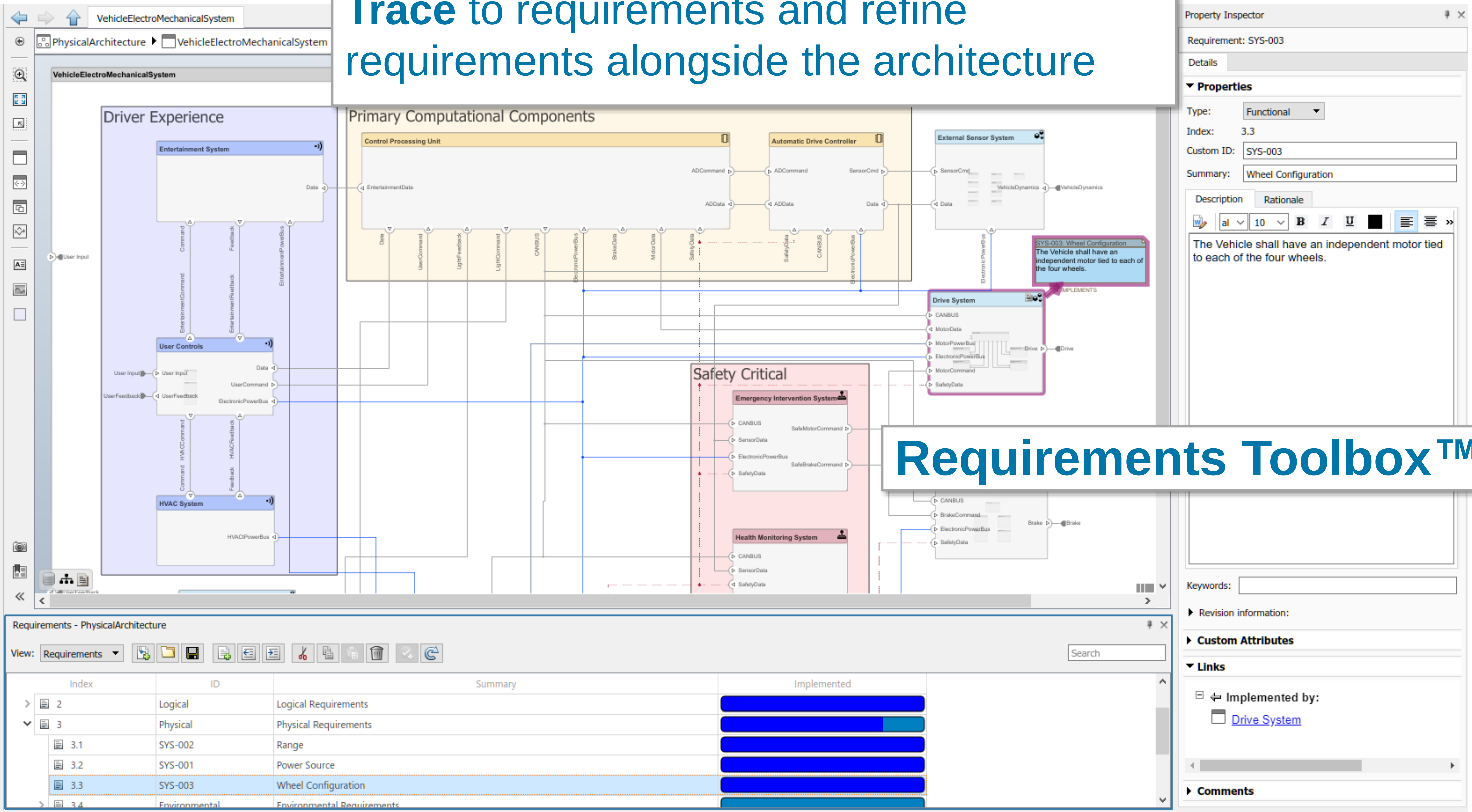


Sketch system interfaces and elaborate incrementally



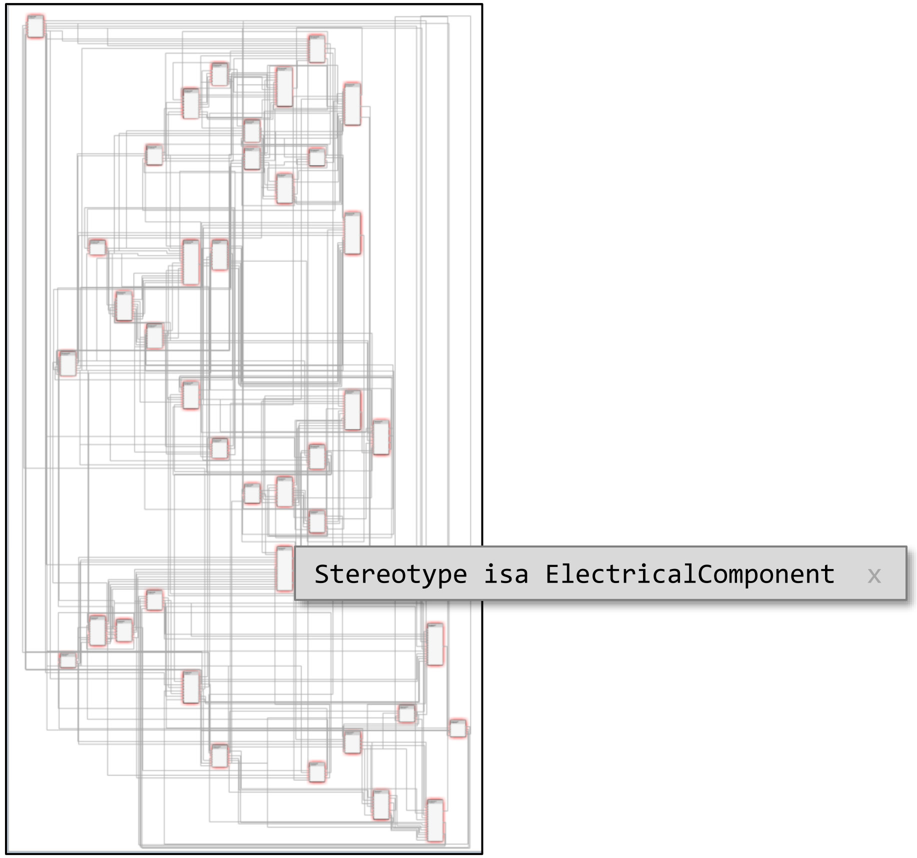
Extend elements with your own custom metadata using Profiles & Stereotypes

Trace to requirements and refine requirements alongside the architecture

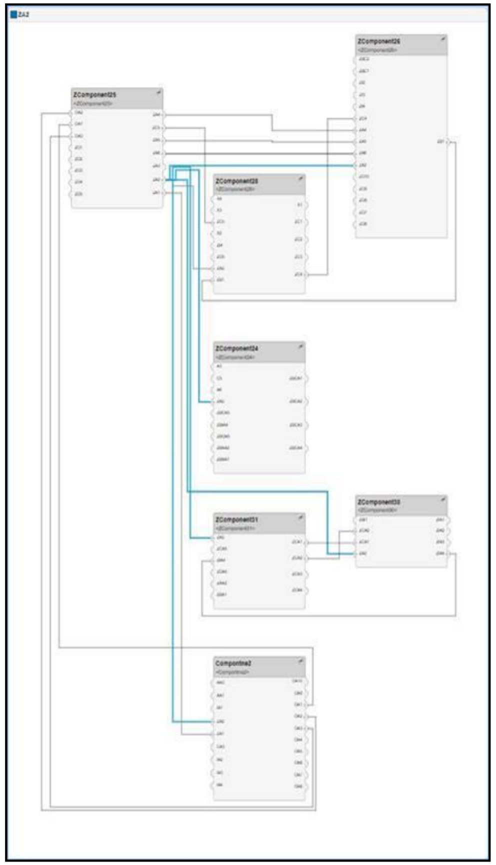


Requirements Toolbox™

Simplify the complex with Filters and autogenerated Views



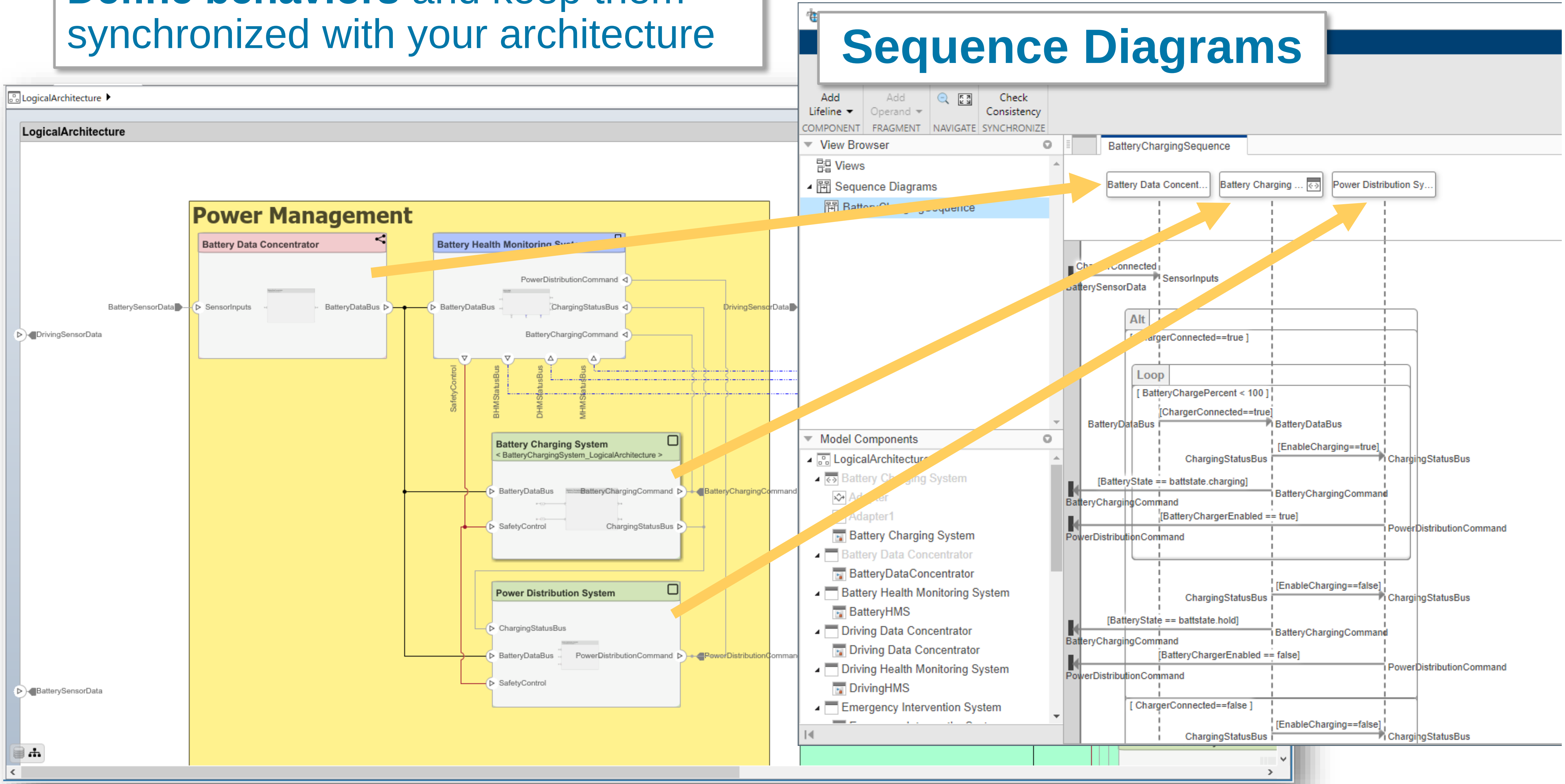
Full system model



Filtered view

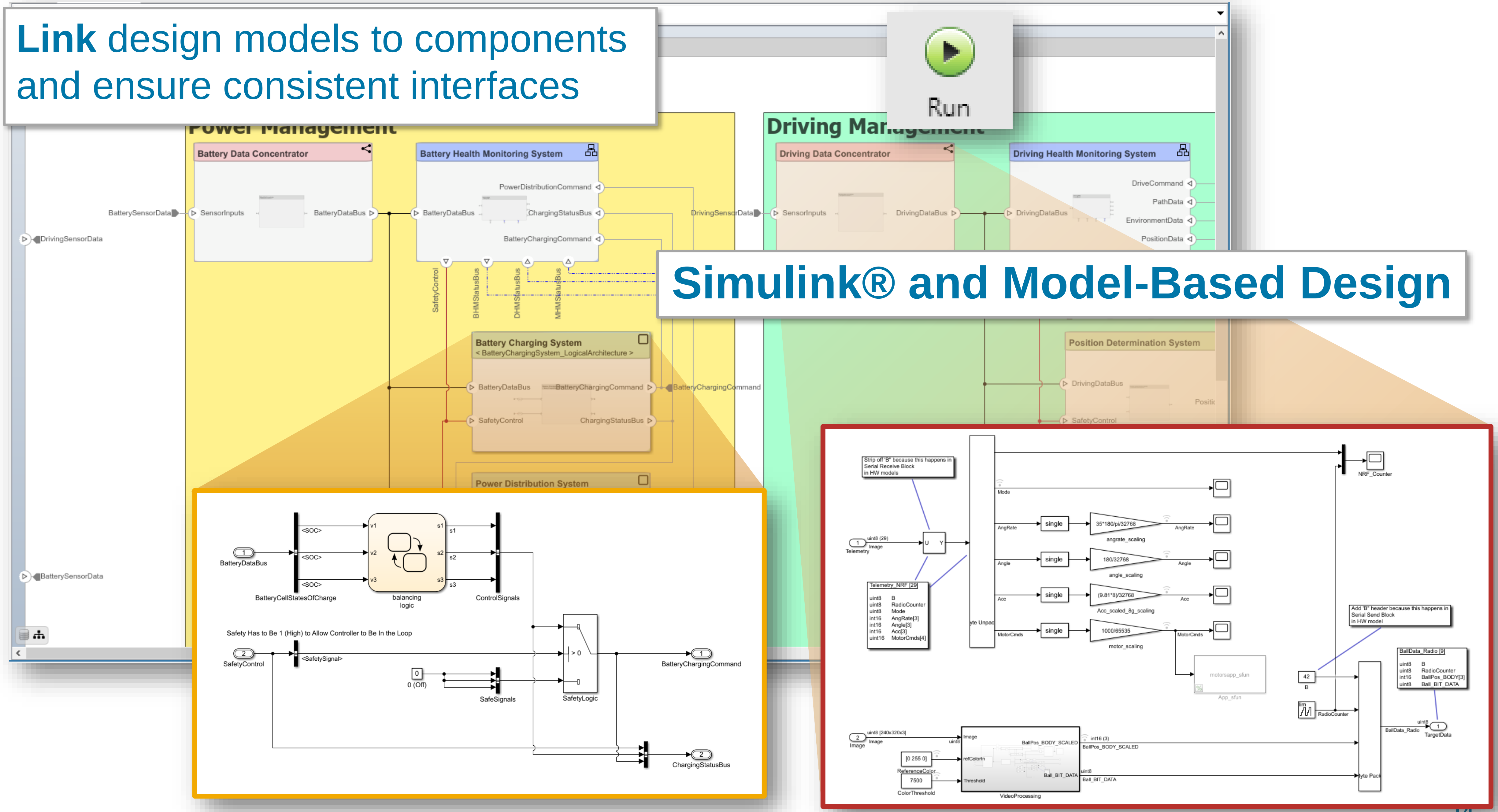
Define behaviors and keep them synchronized with your architecture

Sequence Diagrams

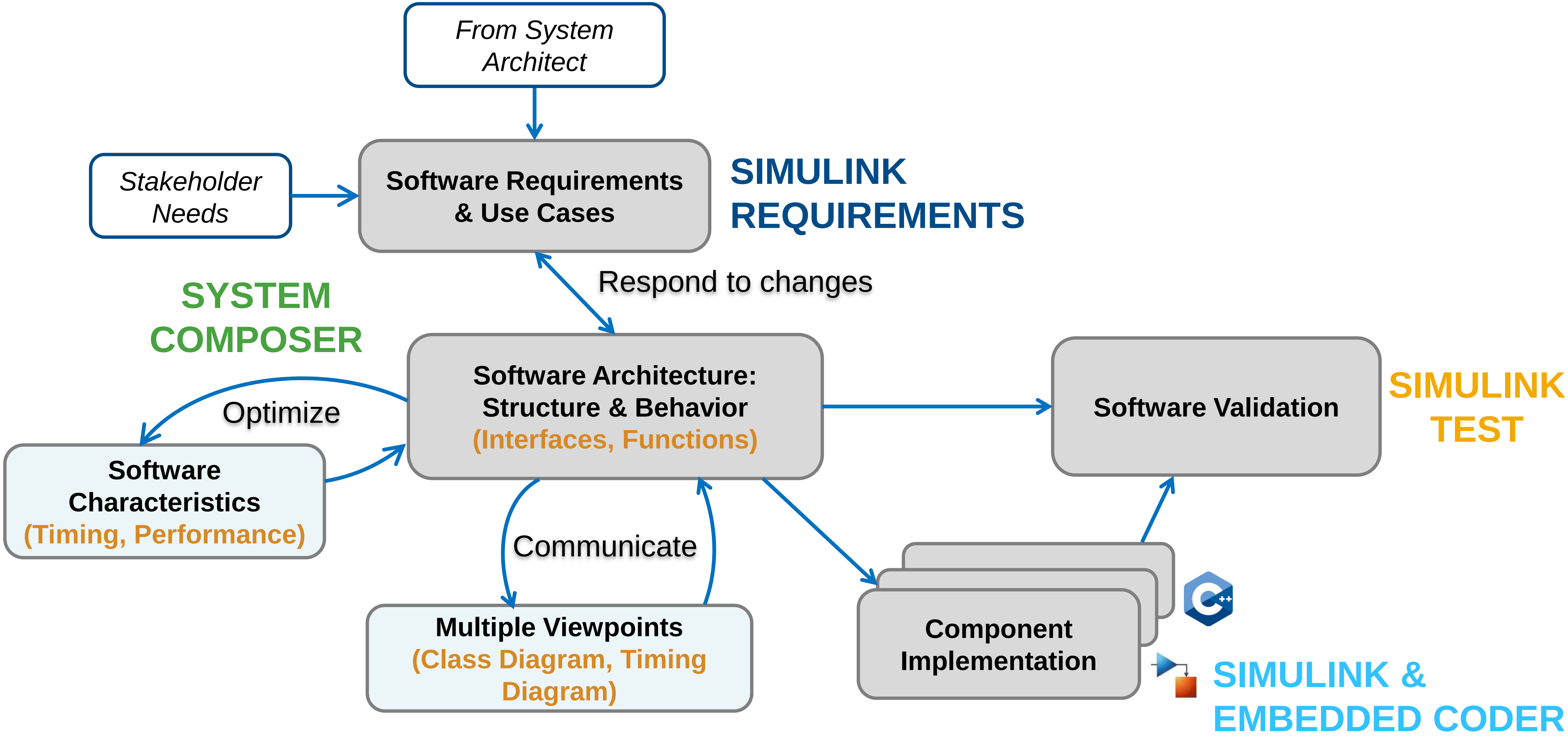


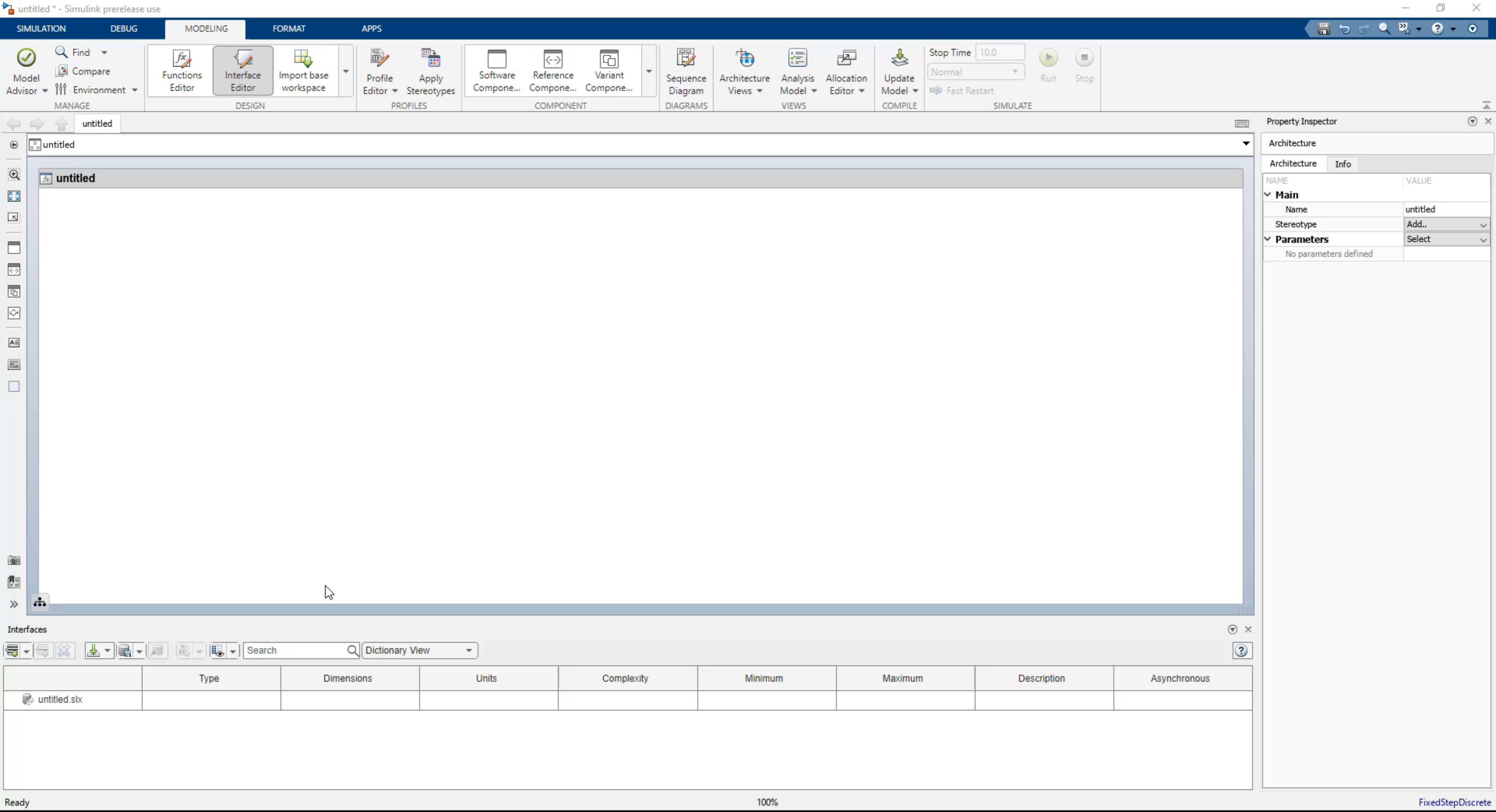
Link design models to components and ensure consistent interfaces

Simulink® and Model-Based Design



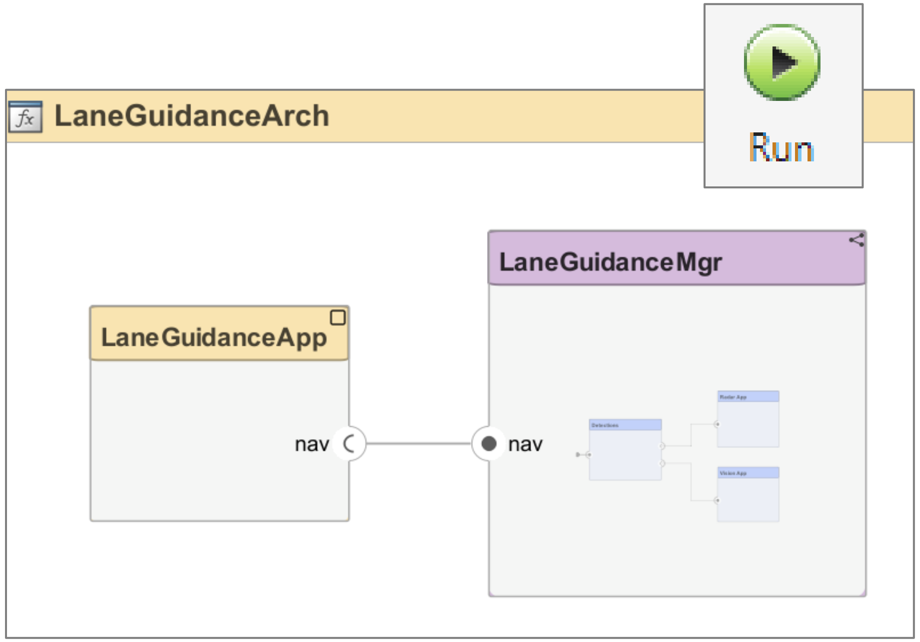
User Workflow for Software Architecture Modeling



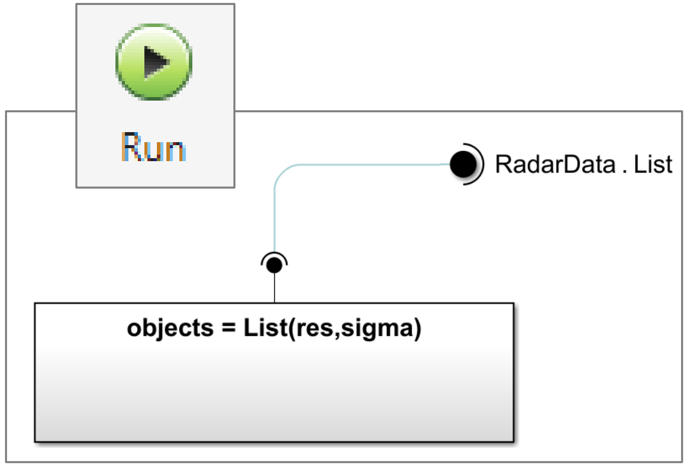

untitled * - Simulink prerelease use
SIMULATION
DEBUG
MODELING
FORMAT
APPS
Model Advisor
Find
Compare
Environment
MANAGE
Functions Editor
Interface Editor
Import base workspace
DESIGN
Profile Editor
Apply Stereotypes
PROFILES
Software Compone...
Reference Compone...
Variant Compone...
COMPONENT
Sequence Diagram
DIAGRAMS
Architecture Views
Analysis Model
Allocation Editor
VIEWS
Update Model
COMPILE
Stop Time
10.0
Normal
Fast Restart
Run
Stop
SIMULATE
untitled
untitled
untitled
Property Inspector
Architecture
Architecture
Info
NAME
VALUE
Main
Name
untitled
Stereotype
Add..
Parameters
Select
No parameters defined
Interfaces
Search
Dictionary View
Type
Dimensions
Units
Complexity
Minimum
Maximum
Description
Asynchronous
untitled.slx
Ready
100%
FixedStepDiscrete

Service-Oriented Architecture (SOA) Design

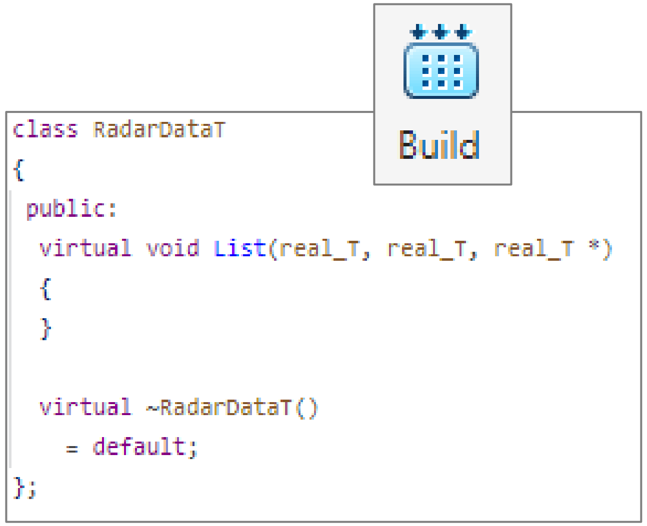
… and much more in terms of workflows (e.g. sequence diagrams, QoS, test harness generation, etc.)



Describe SOA with **System Composer**

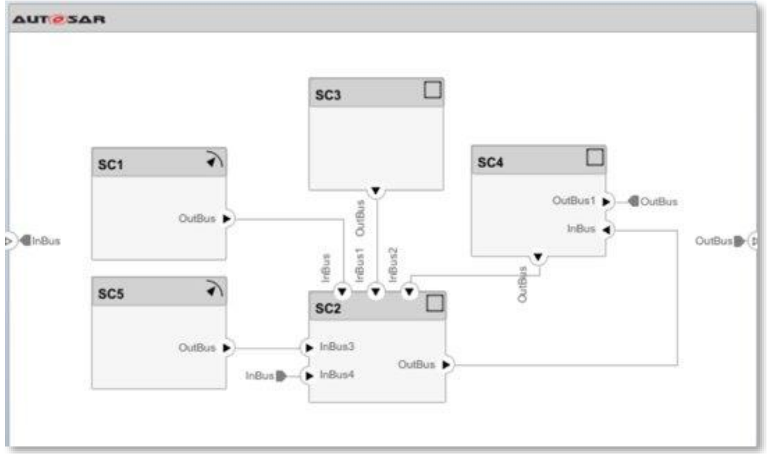


Implement detailed components with **Simulink**



Generate code with **Embedded Coder**

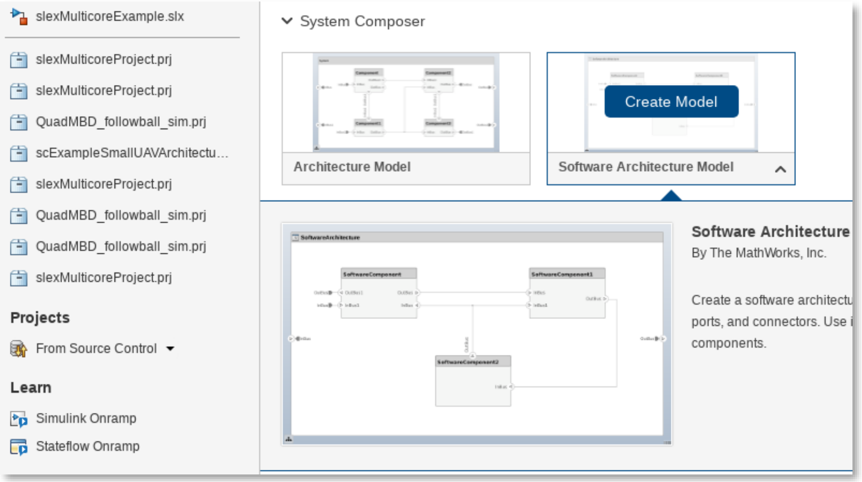
AUTOSAR Architecture



R2019b

- Strong support for Classic
- Growing support for Adaptive

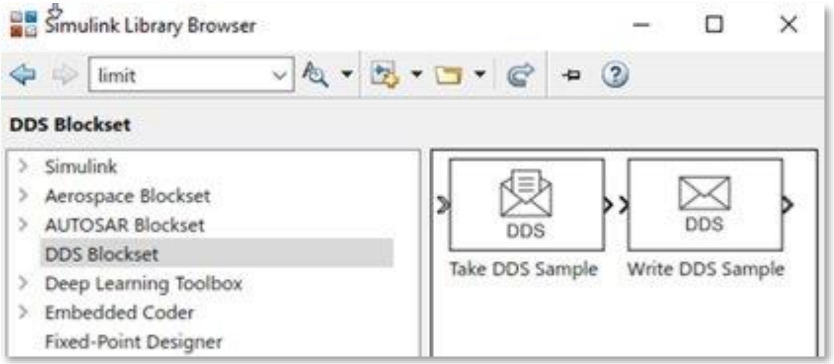
Software Architecture



R2021a

- Embedded coder support package for Linux emerging

DDS Application



R2021a

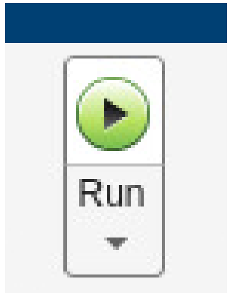
Deep Technical Talk

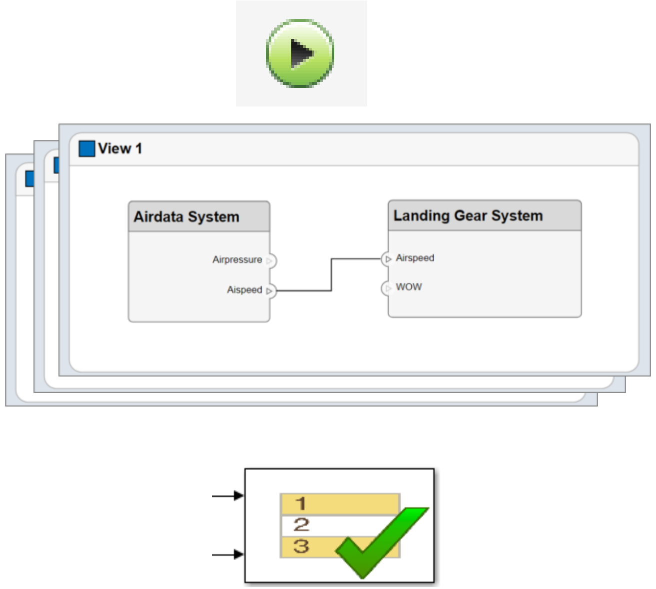
Software-Defined Vehicle Track

The Evolution of Simulink for Service-Oriented Architecture (SOA)

Shwetha Bhadravathi Patil, MathWorks

Towards Virtual Integration





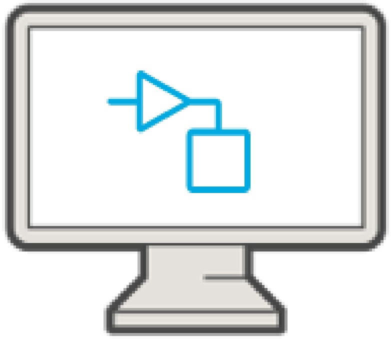
Shifting Left: How far can you go?

Verify that the integration of Application SW components into full Application meets functional requirements

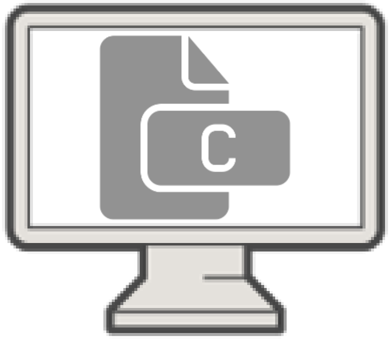
Verify the integration of Application SW with Basic SW

Validate the integration of one/few ECUs with simulated or real sensors, actuators, networks

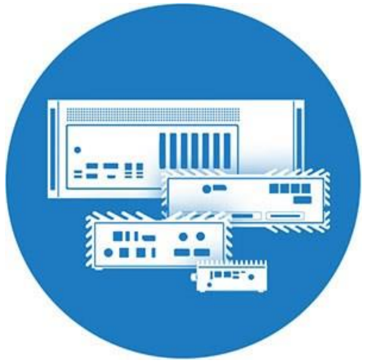
Validate the integration of **ALL** ECUs, Networks, Sensors and Actuators



MIL



SIL

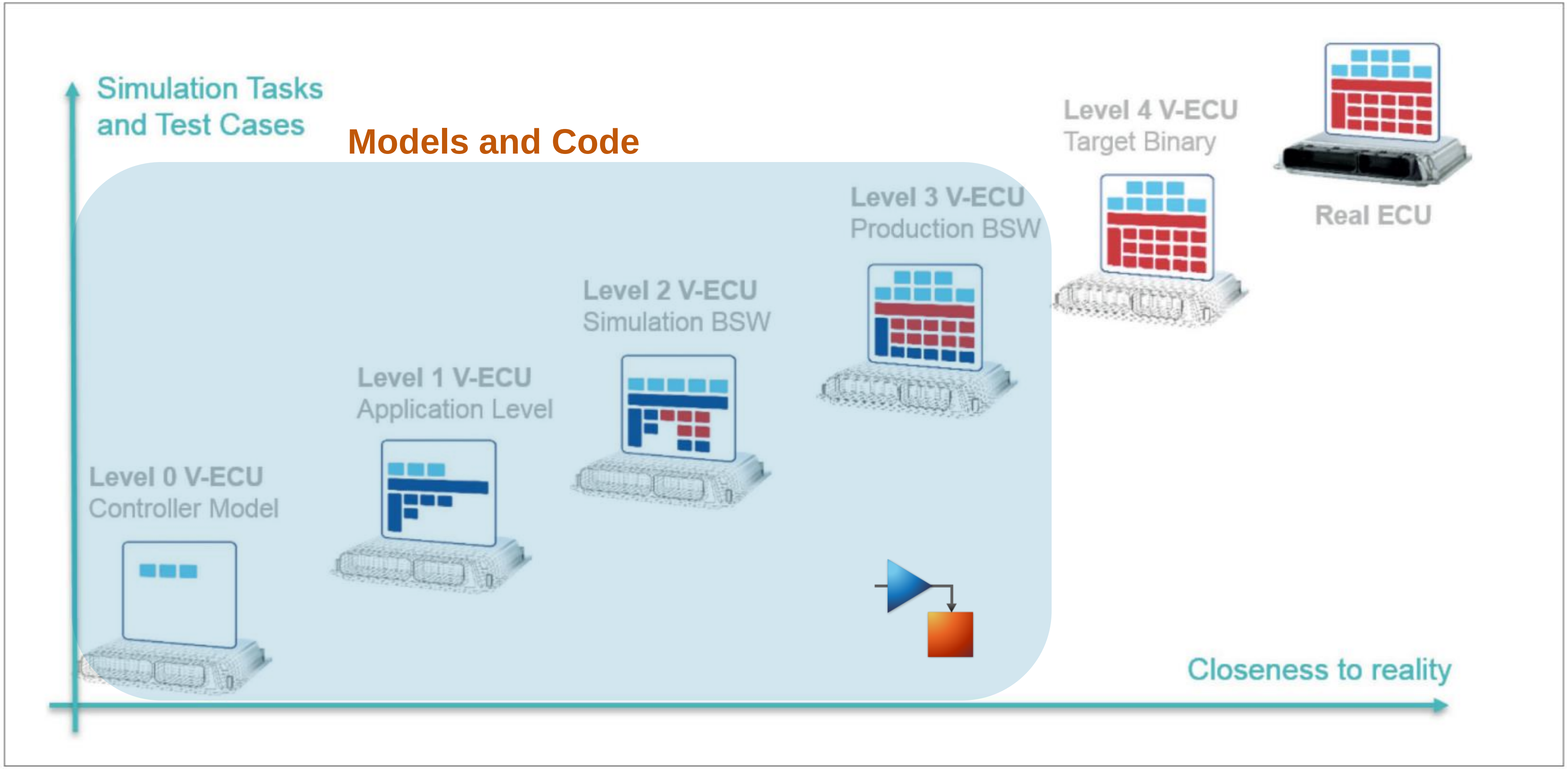


HIL

Vehicle

Key takeaway: use each test facility where it adds value during the process

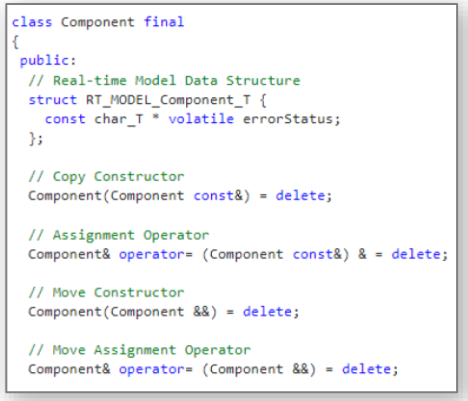
V-ECU with Simulink: Focus on the development process where you first hit complexity



Simulink is a Simulation Integration Platform

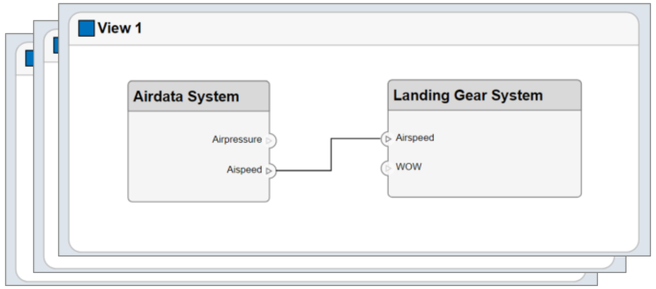
Core capabilities of the Integration Platform

"Ready-to-run"
Model Components



```
class Component final
{
 public:
  // Real-time Model Data Structure
  struct RT_MODEL_Component_T {
    const char_T * volatile errorStatus;
  };

  // Copy Constructor
  Component(Component const&) = delete;

  // Assignment Operator
  Component& operator= (Component const&) & = delete;

  // Move Constructor
  Component(Component &&) = delete;

  // Move Assignment Operator
  Component& operator= (Component &&) = delete;
```

"Ready-to-run"
Code/Other
Components

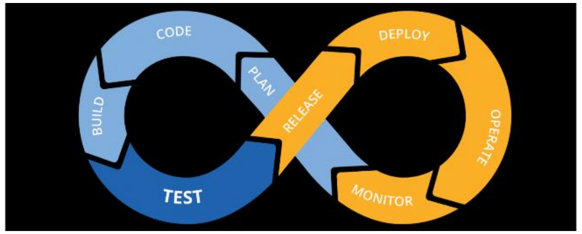


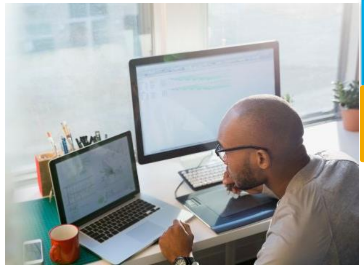
Automated assembly
of models

Performant Simulation!

Protected Models evolve to *Ready-to-run* models for integration

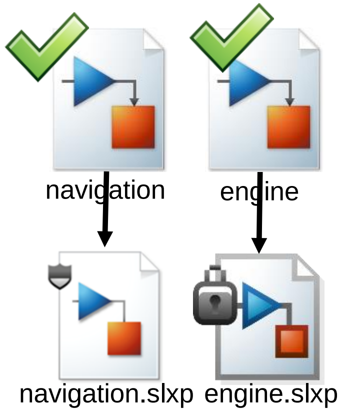
Creator



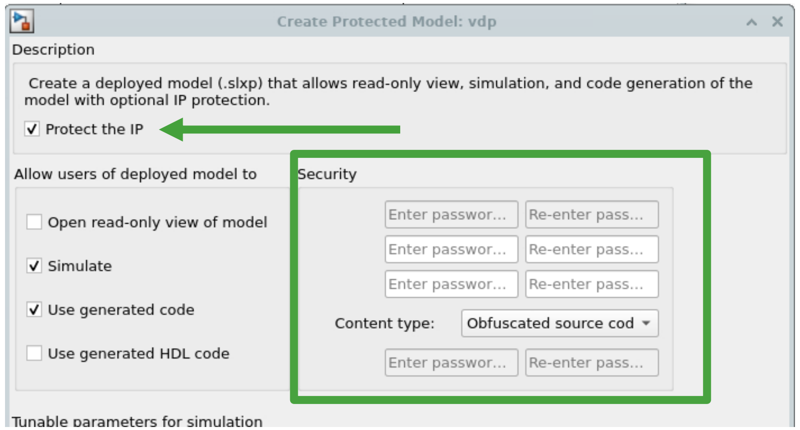


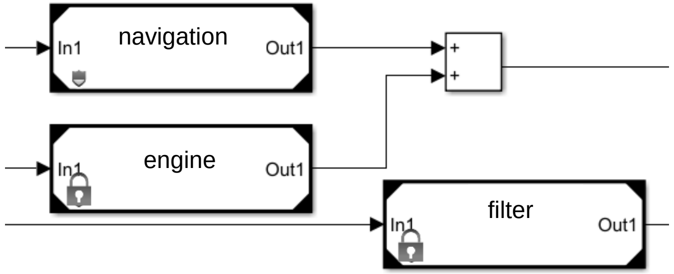
Recipient

Inside/outside the org



Package **verified** component for ready-to-run



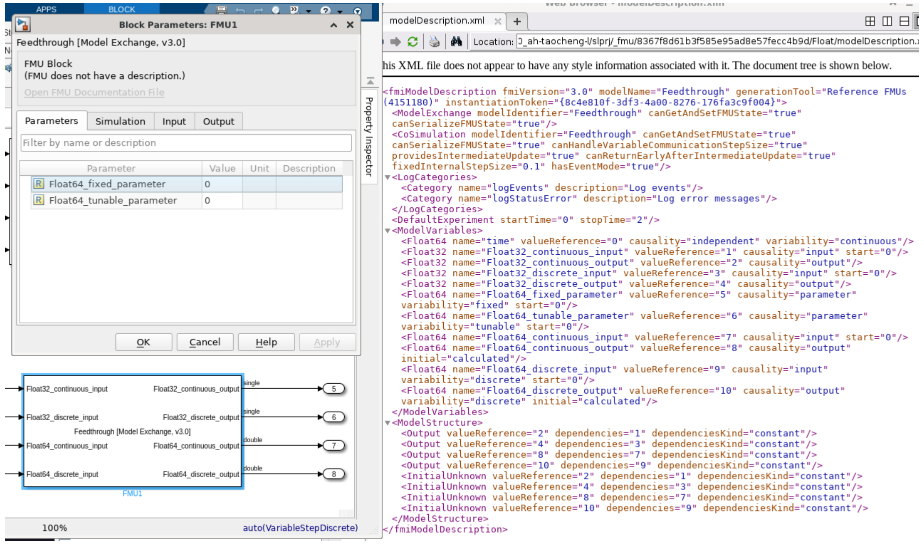


Benefits

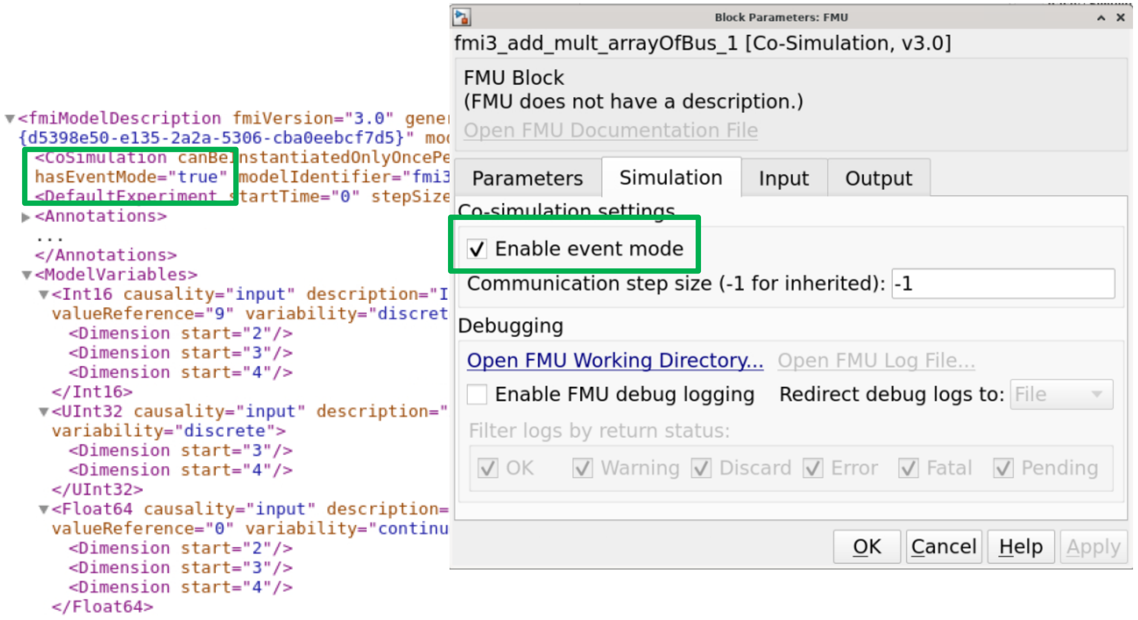
- Flexibility
- Performance
- Encapsulation
- Multi-instantiation
- …

FMUs continue to provide an avenue to make ready-to-run Parts from other tools

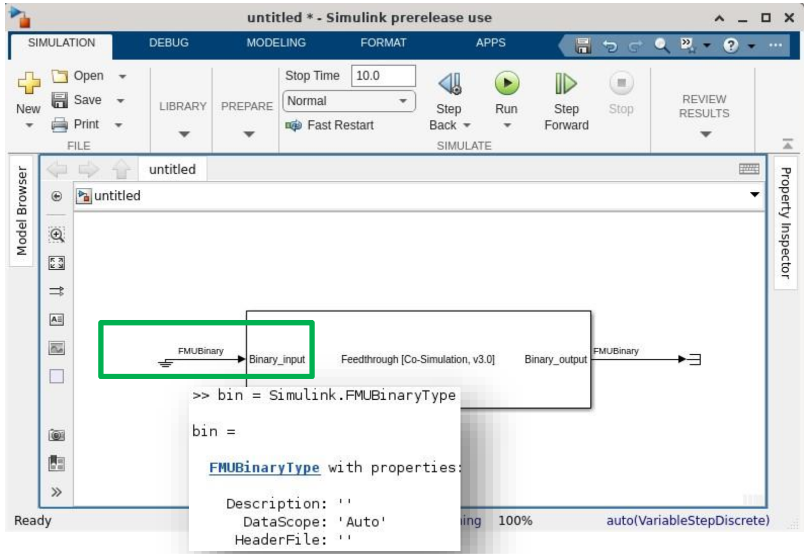
- Simulink supports FMI 3.0 Import in R2023b



FMU Import block loading FMU 3.0 modelDescription file

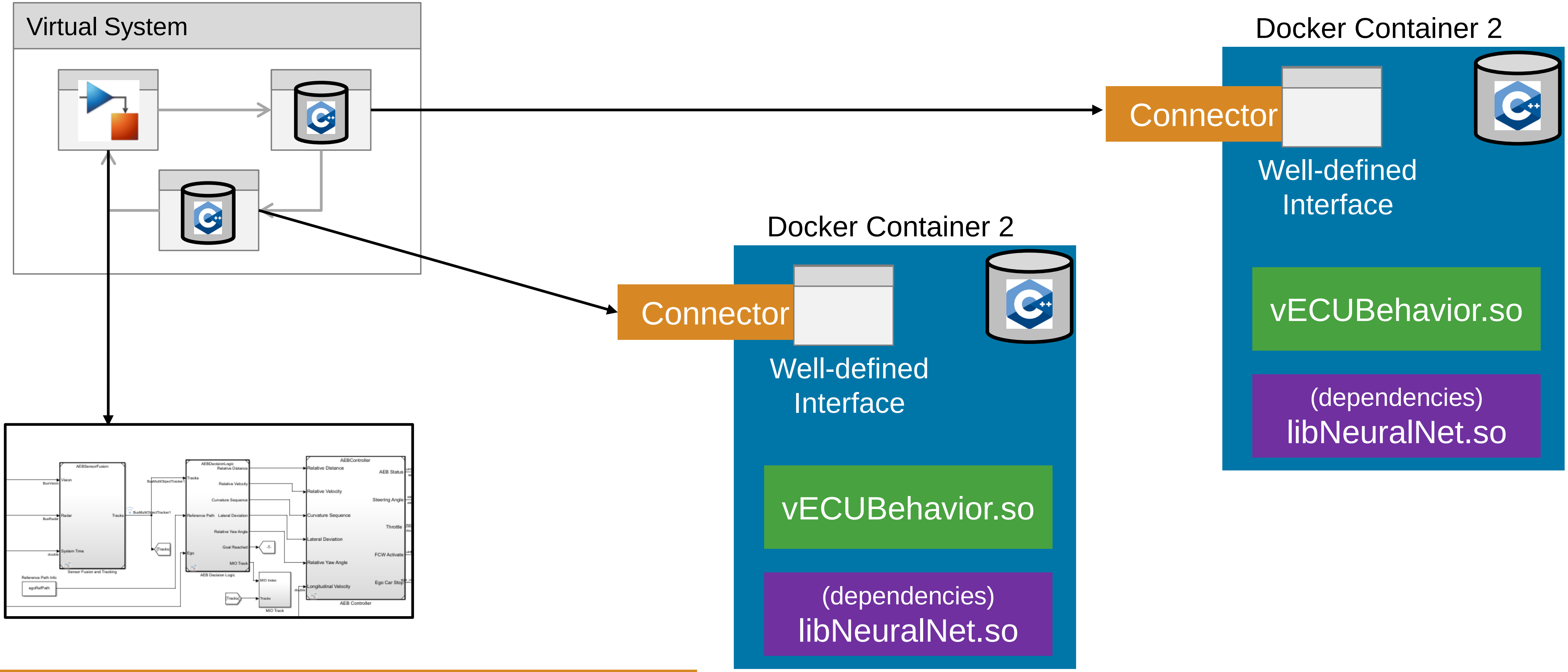


FMU Co-simulation with event mode eliminates one-step delay



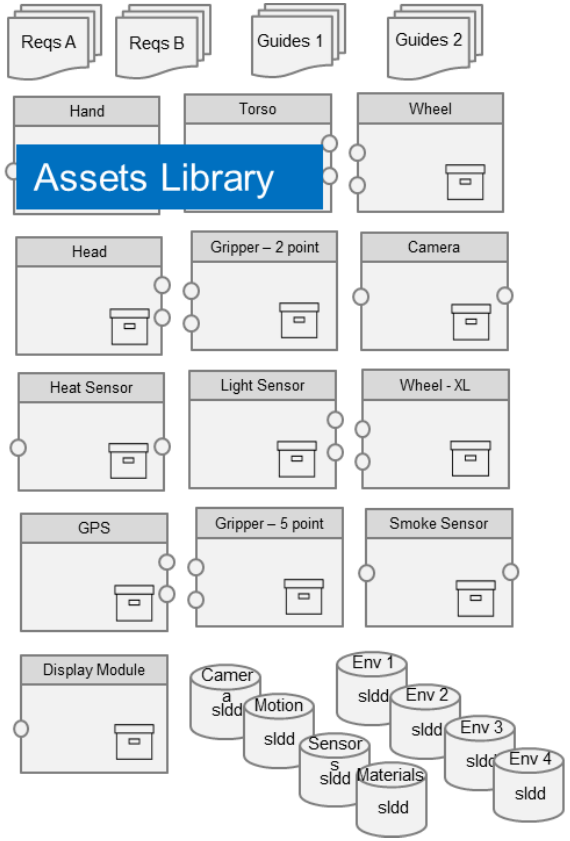
FMUBinary data type

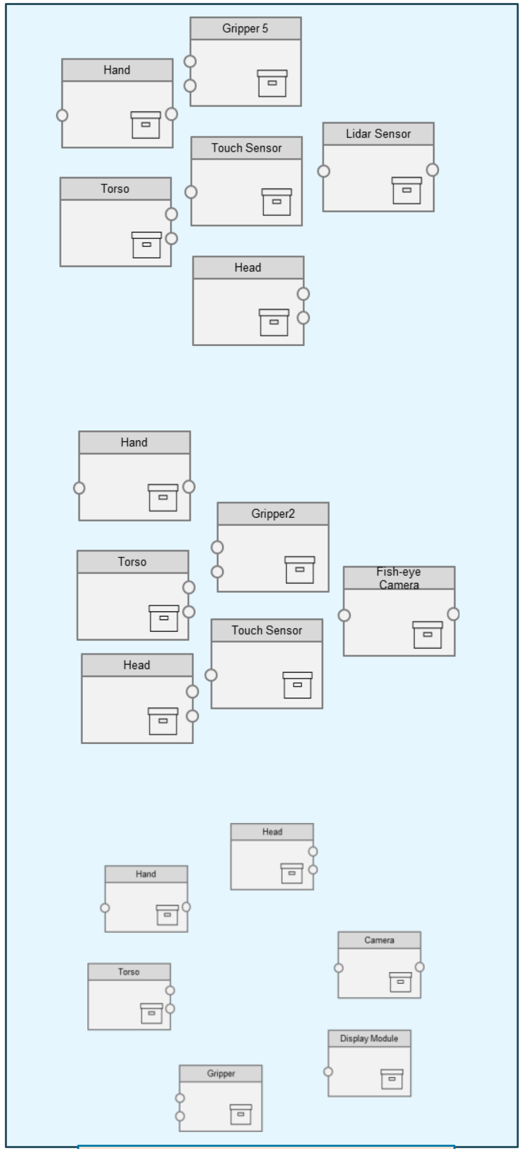
Connector for ready-to-run code components



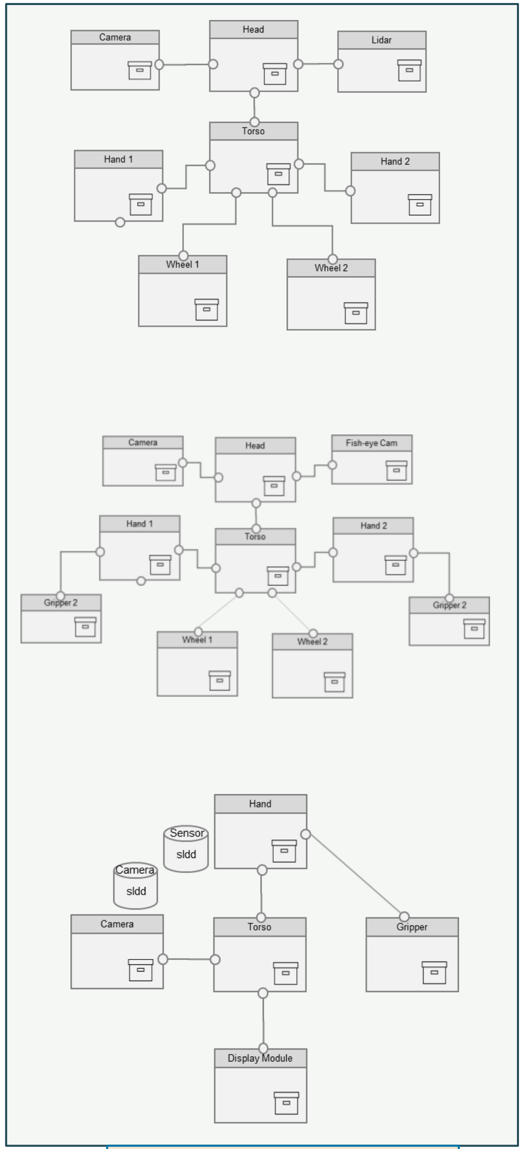
Unit testable, distributable containers

Automated model assembly is being emphasized in many workflows



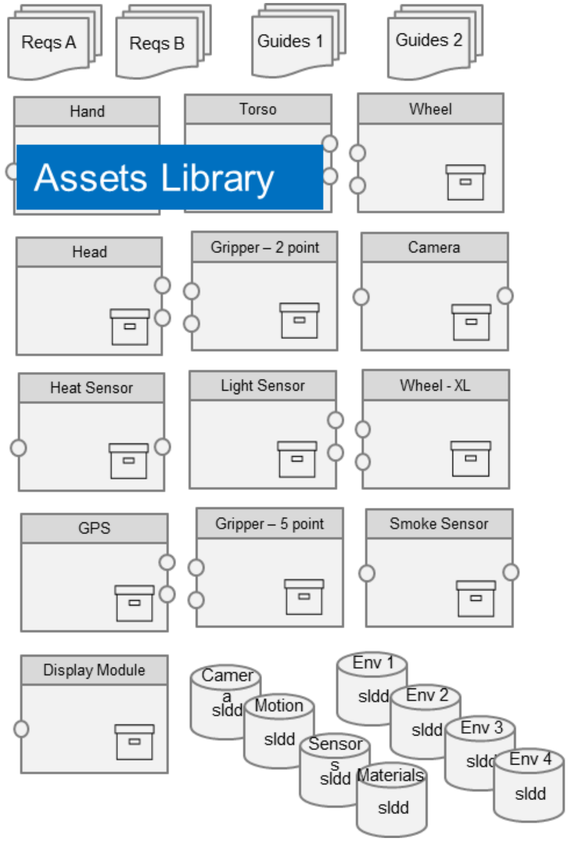


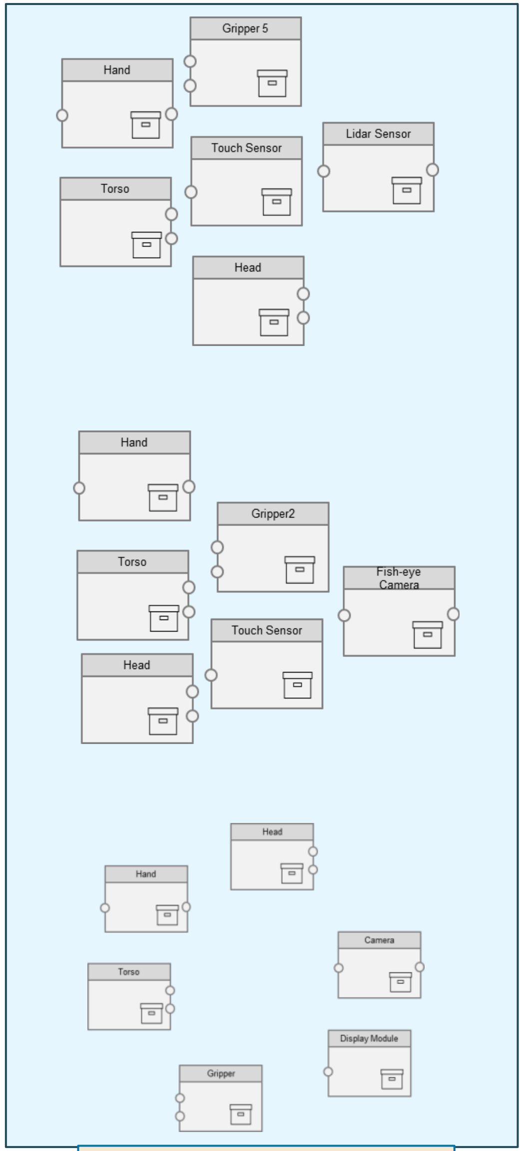
Selection of Parts



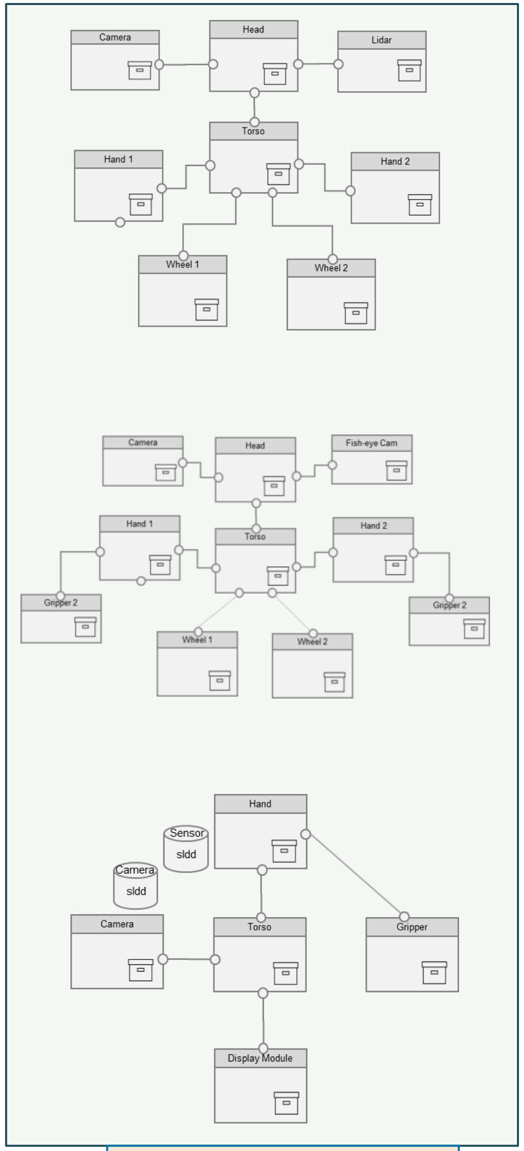
Model Assembly

Automated model assembly is being emphasized in many workflows



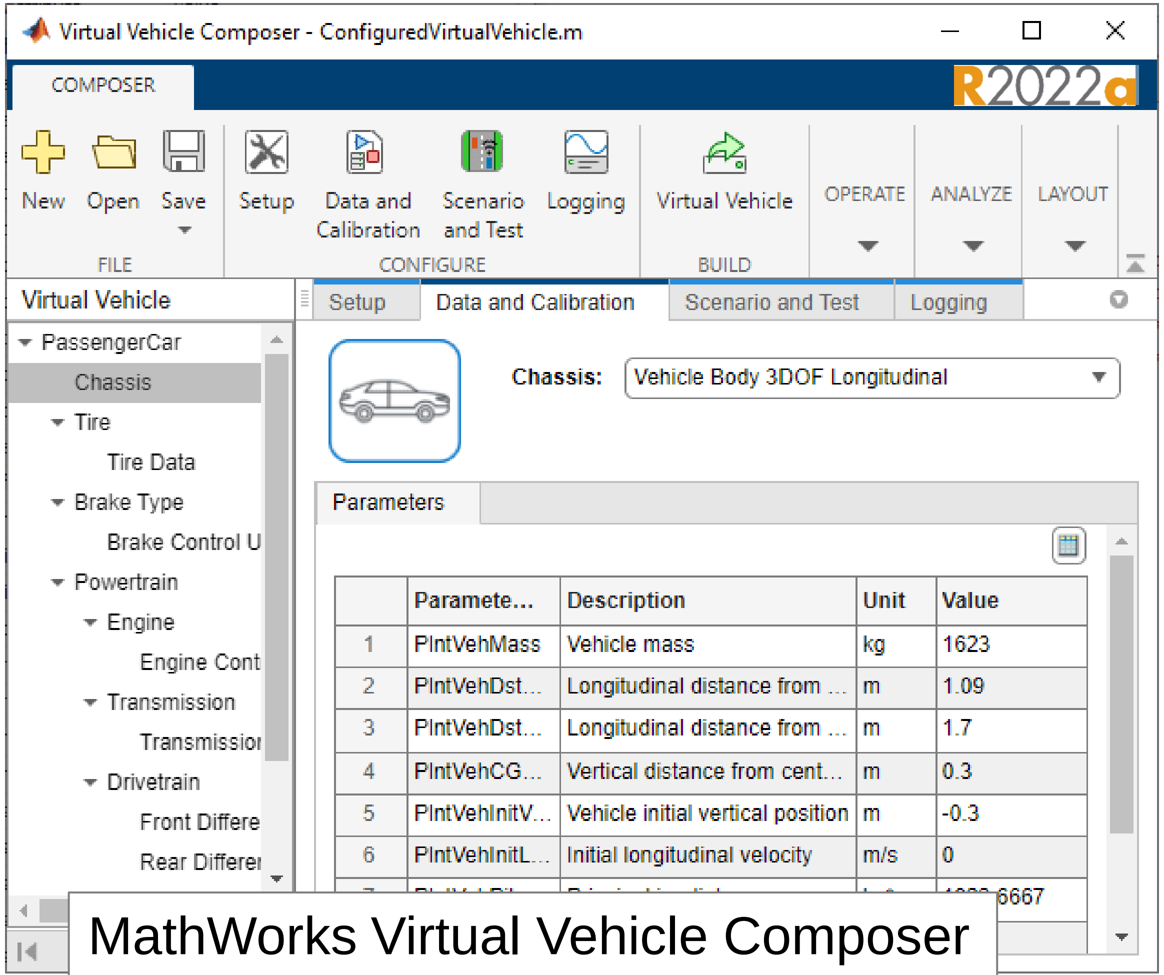


Selection of Parts



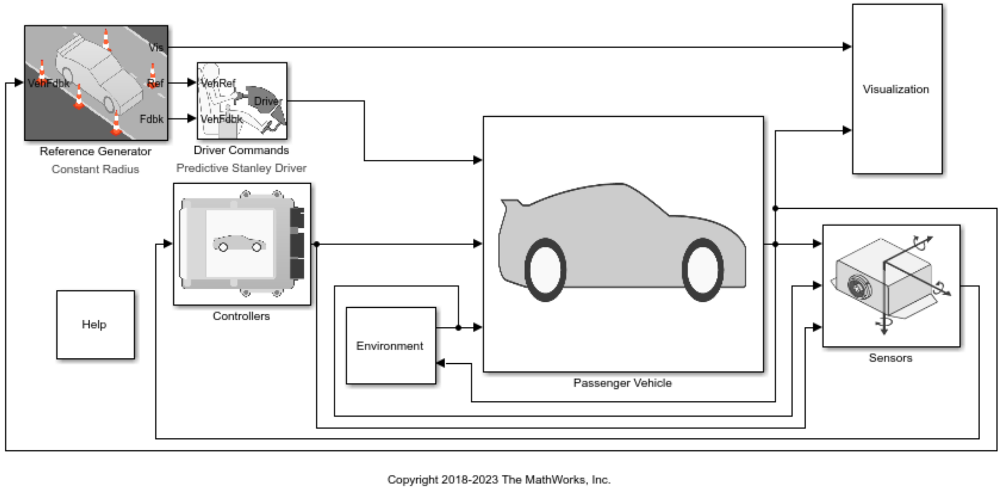
Model Assembly

We are creating a "Feature-Driven" Approach to picking the Parts to Assemble into Models



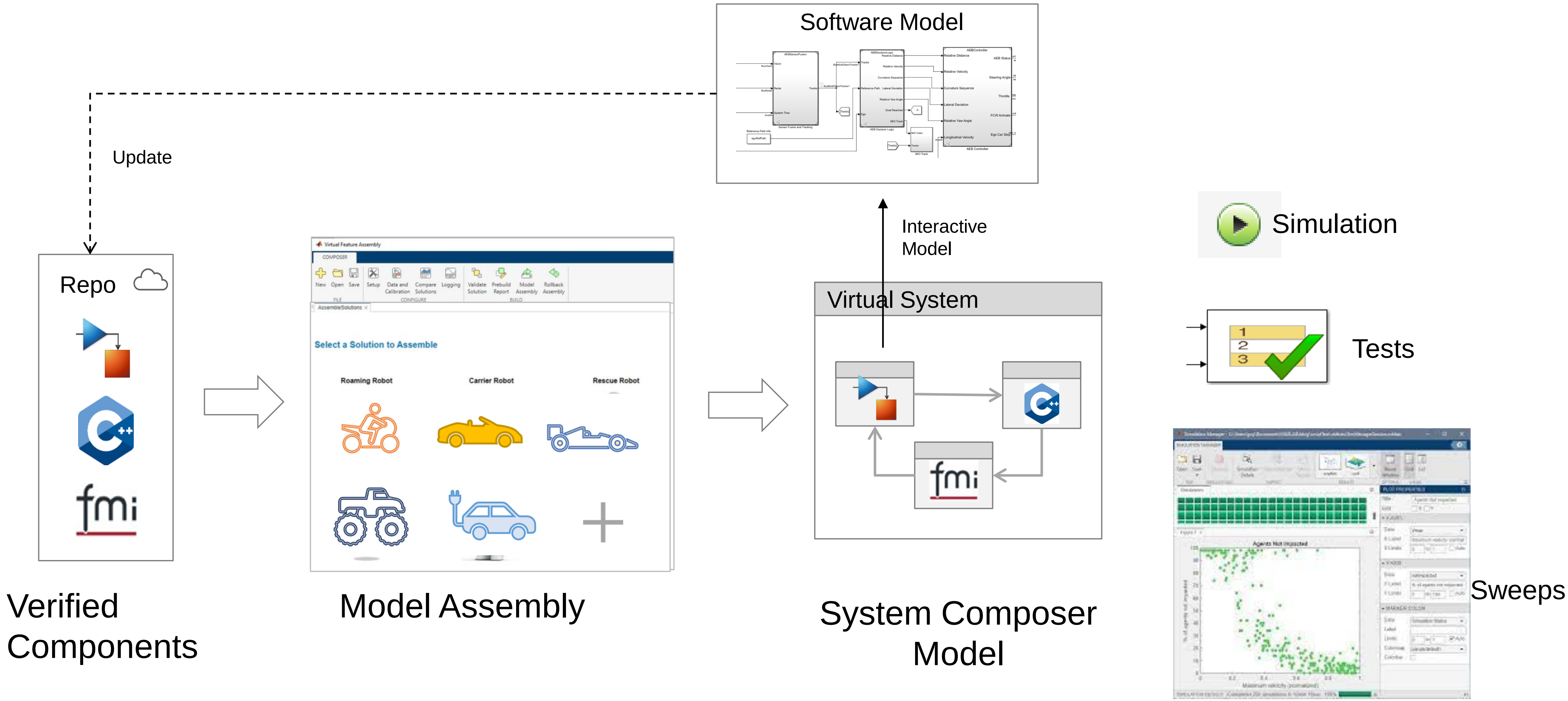
MathWorks Virtual Vehicle Composer

Select Vehicle features & characteristics



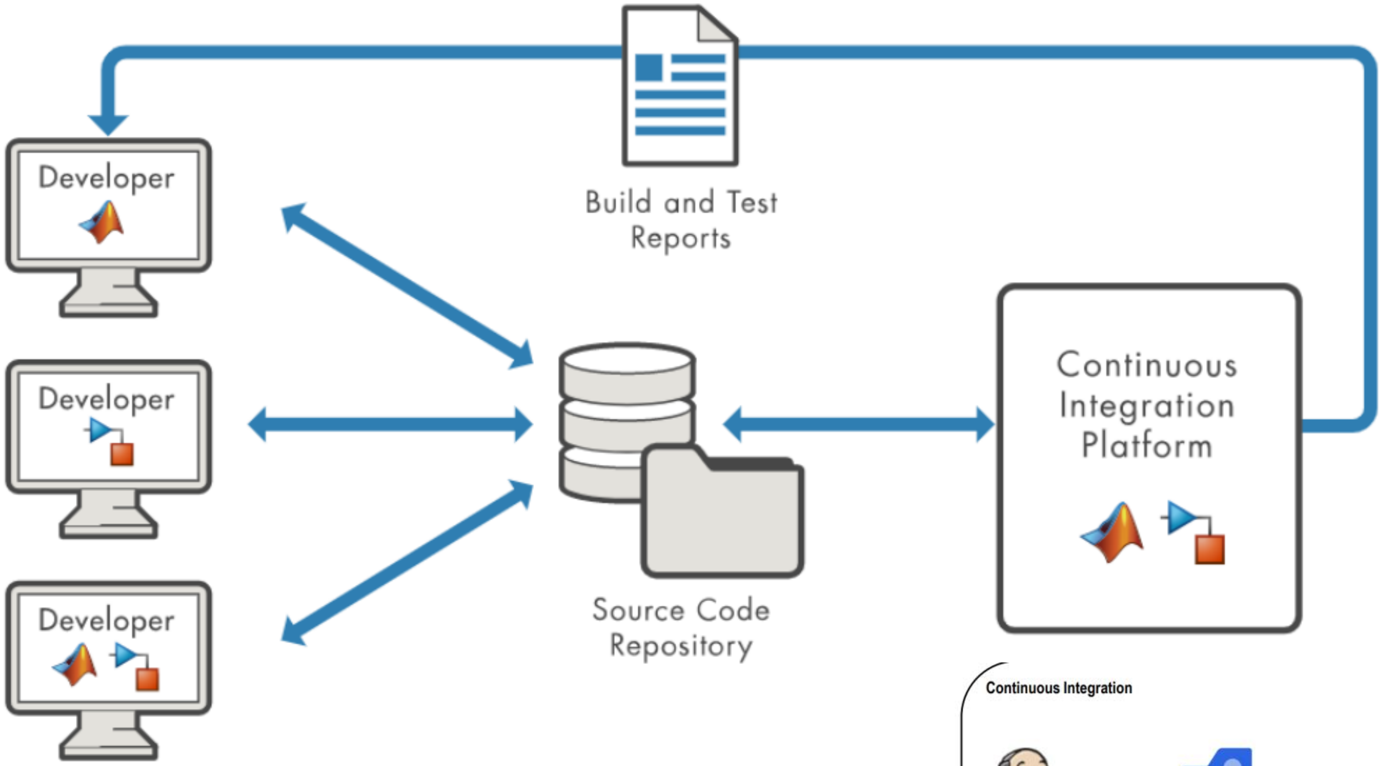
Assembled Vehicle Model

Towards a fully distributed “model and code” integration framework

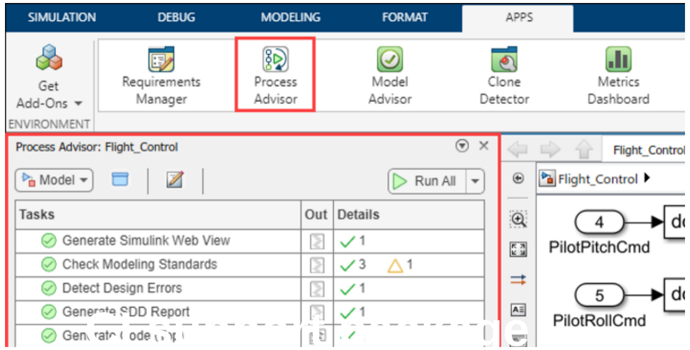


Evolving the modern ecosystem

Seamless onramp to CI/CD Workflows

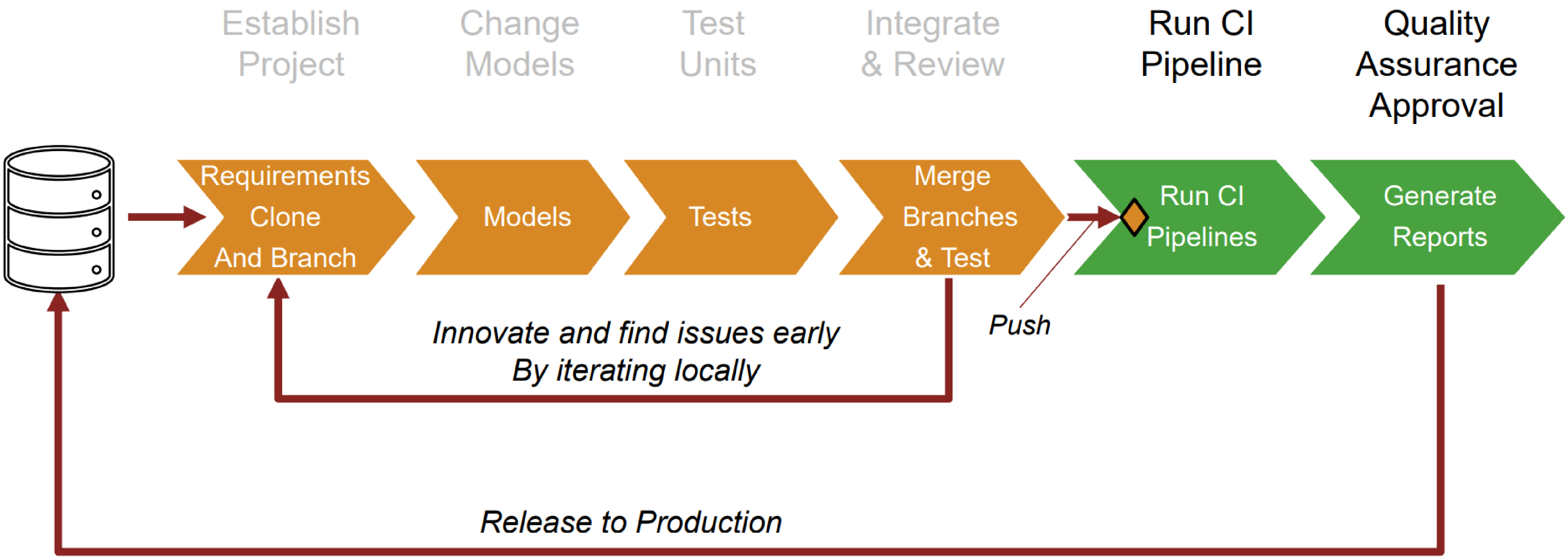


CI Support Package



R2022a

Mapping Model-based Design on to a CI Workflow



Masterclass

Accelerating Development for Software-Defined Vehicles Using CI/CD

Software-Defined Vehicle Track

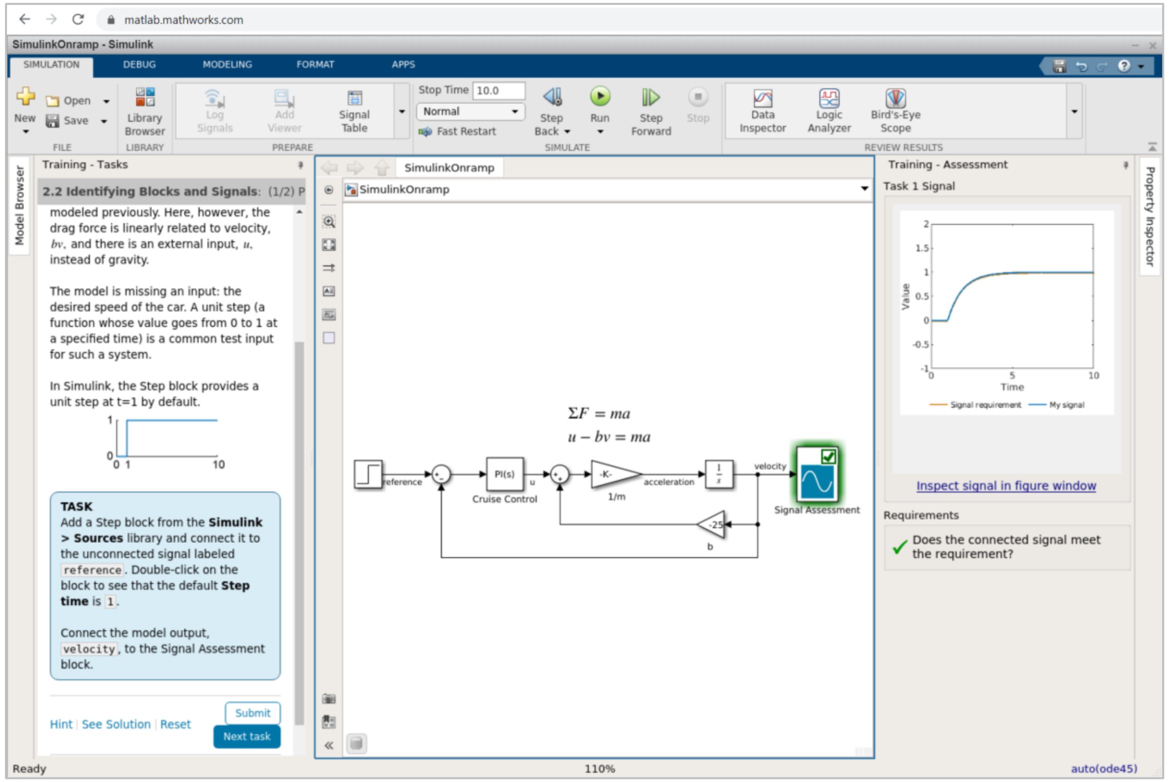
Nukul Sehgal *Vamshi Kumbham*

Evolving a Cloud Ecosystem

Access

Scaling

Collaboration



Simulink Online

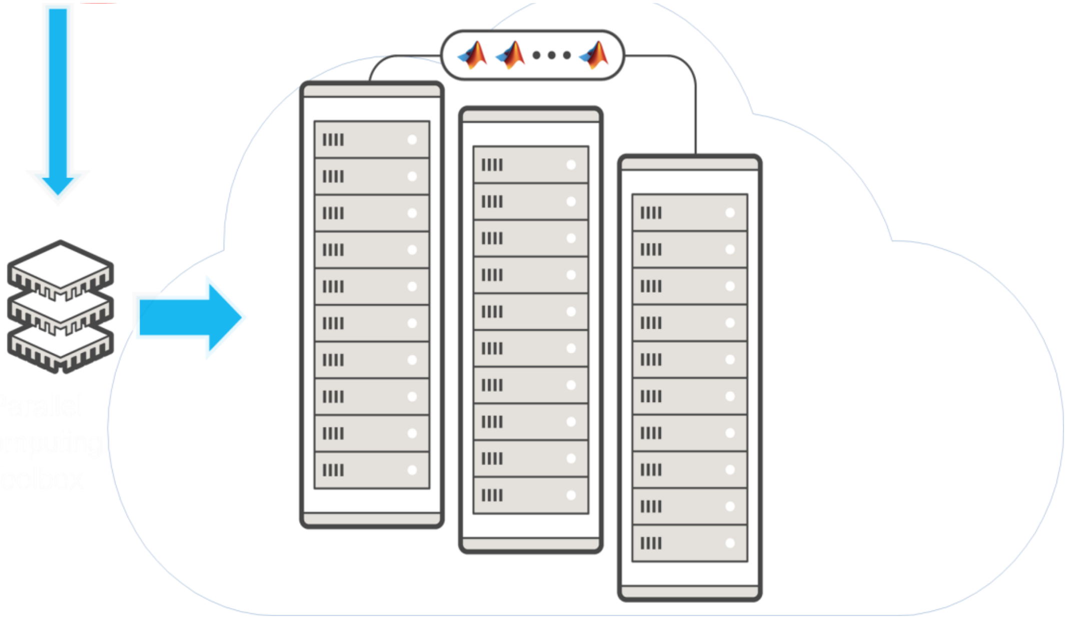
Evolving a Cloud Ecosystem

Access

Scaling

Collaboration

```
for i = 1:10000
    in(i) = Simulink.SimulationInput(my_model)
    in(i) = setVariable(my_var, i);
end
out = parsim(in);
```



MATLAB Parallel Server

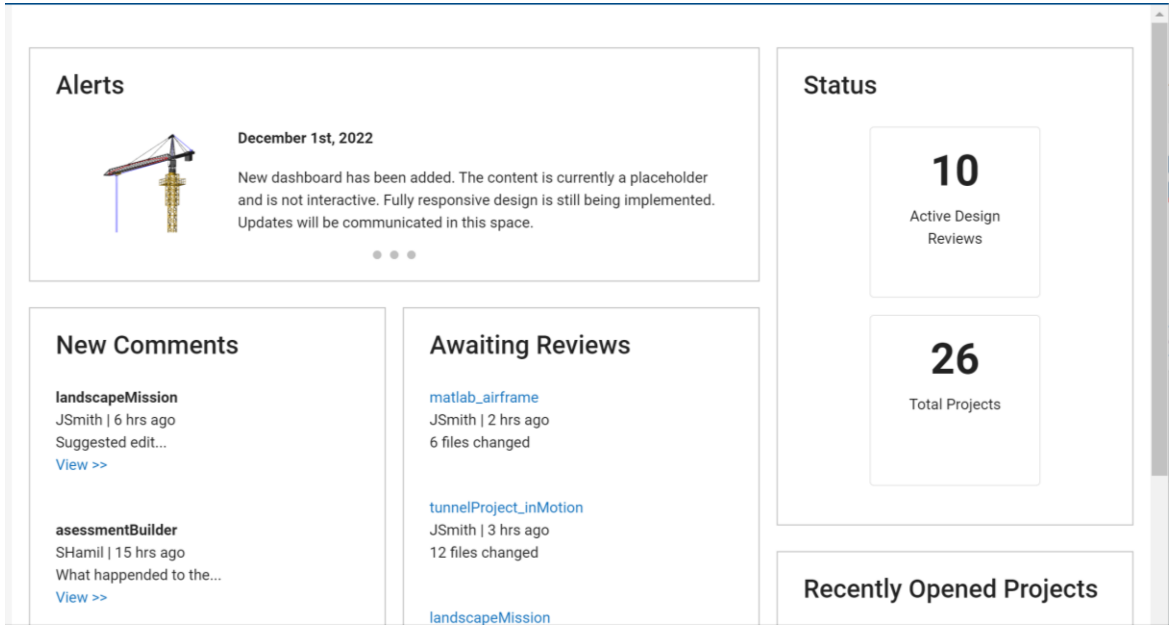
Massive simulation jobs

Evolving a Cloud Ecosystem

Access

Scaling

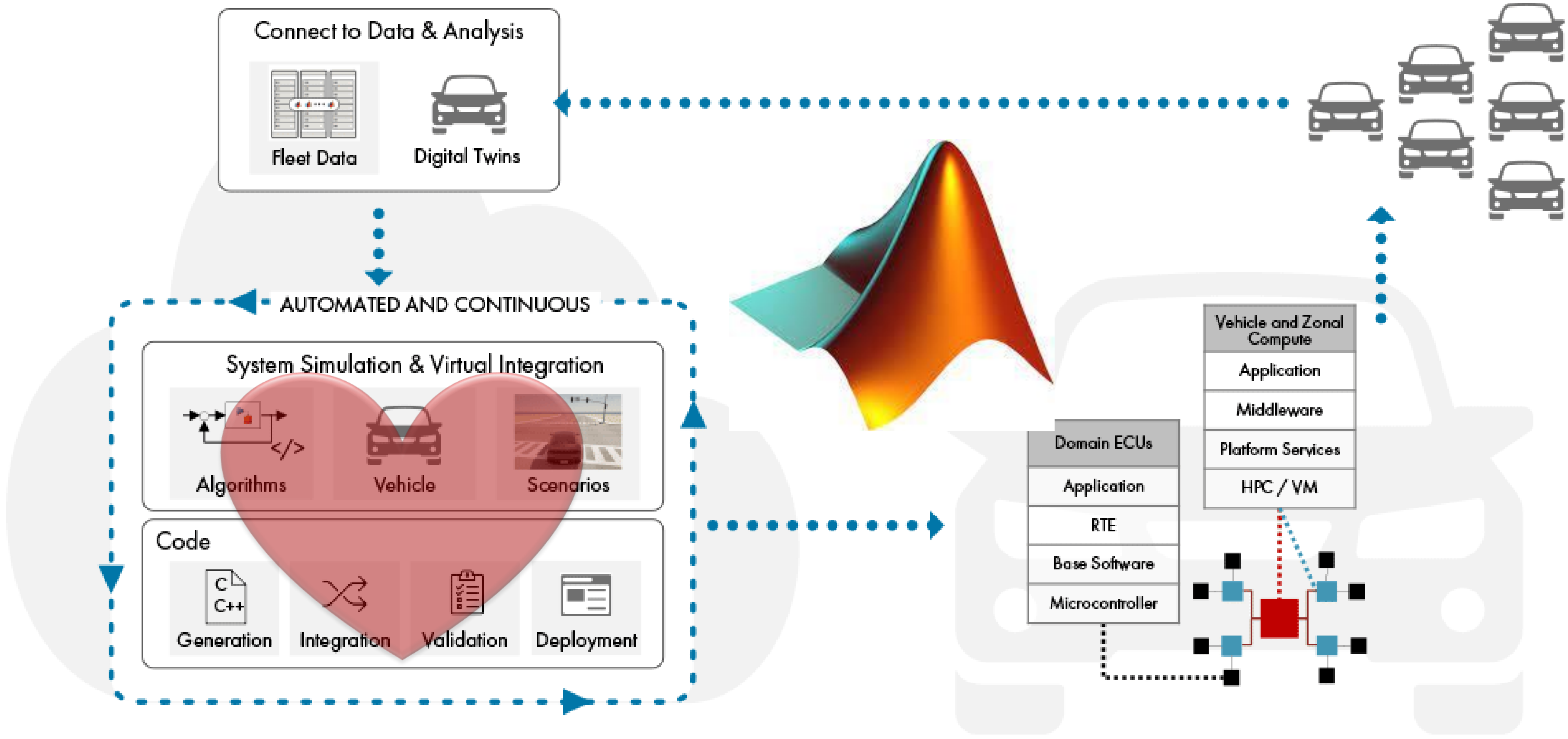
Collaboration



Project dashboard

Design review

Instant search


Connect to Data & Analysis
Fleet Data
Digital Twins
AUTOMATED AND CONTINUOUS
System Simulation & Virtual Integration
Algorithms
Vehicle
Scenarios
Code
C
C++
Generation
Integration
Validation
Deployment
Vehicle and Zonal Compute
Application
Middleware
Platform Services
HPC / VM
Domain ECUs
Application
RTE
Base Software
Microcontroller

MathWorks Deep Solutions for SDV Domains are key enablers

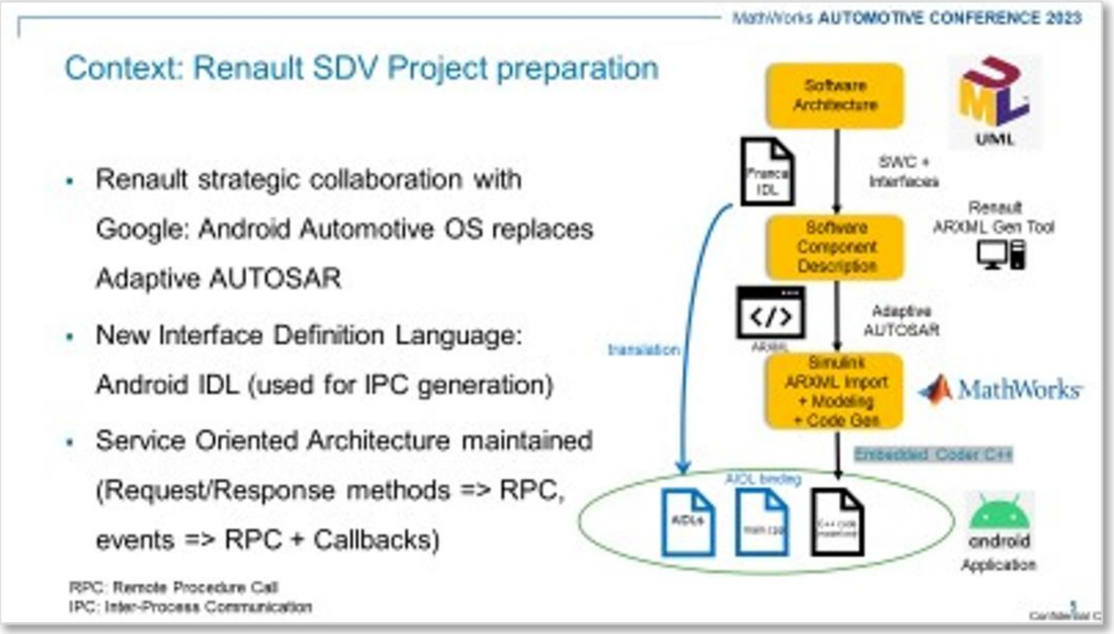
Extensive support in MathWorks Toolboxes

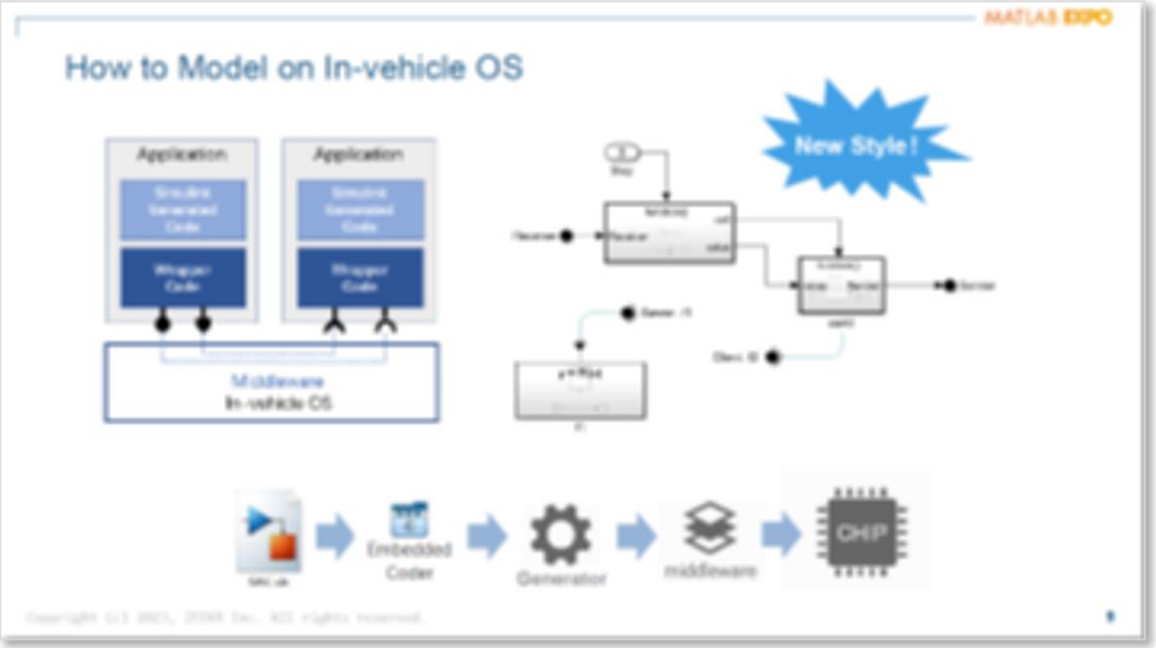
SDV Platform

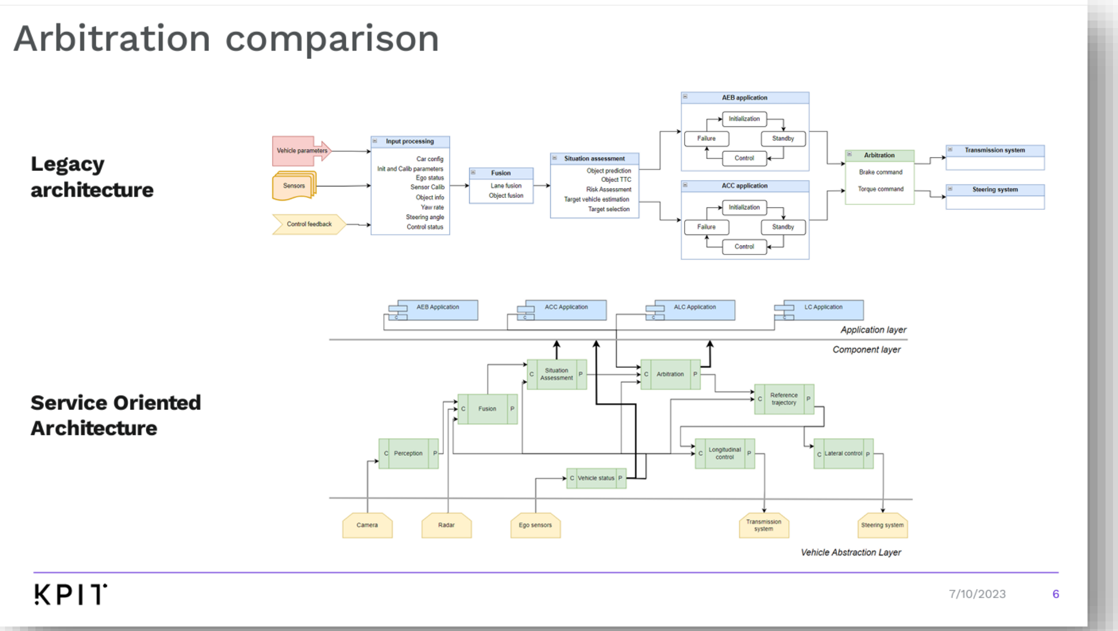
How are we taking this journey?

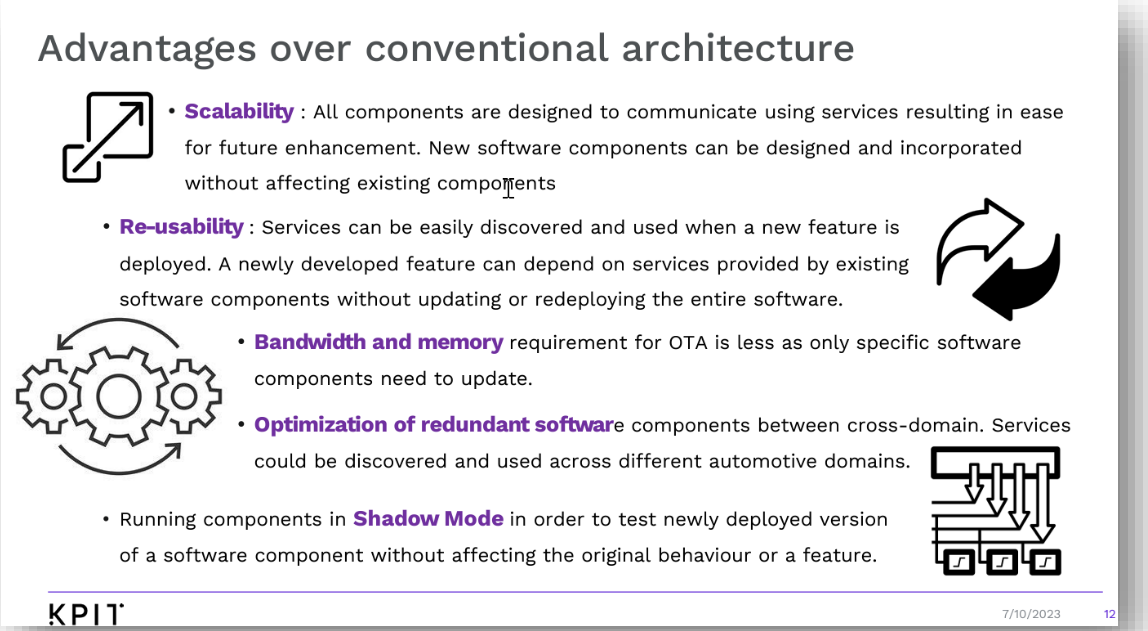

SDV: Integrating Simulink C++ Code in Android Automotive Environment
Rémy Brugnon, Renault Group


ZEEKR
Use Model-Based Design to Develop SOA Application Running on In-vehicle OS
Yining Luo, ZEEKR TECHNOLOGY LIMITED

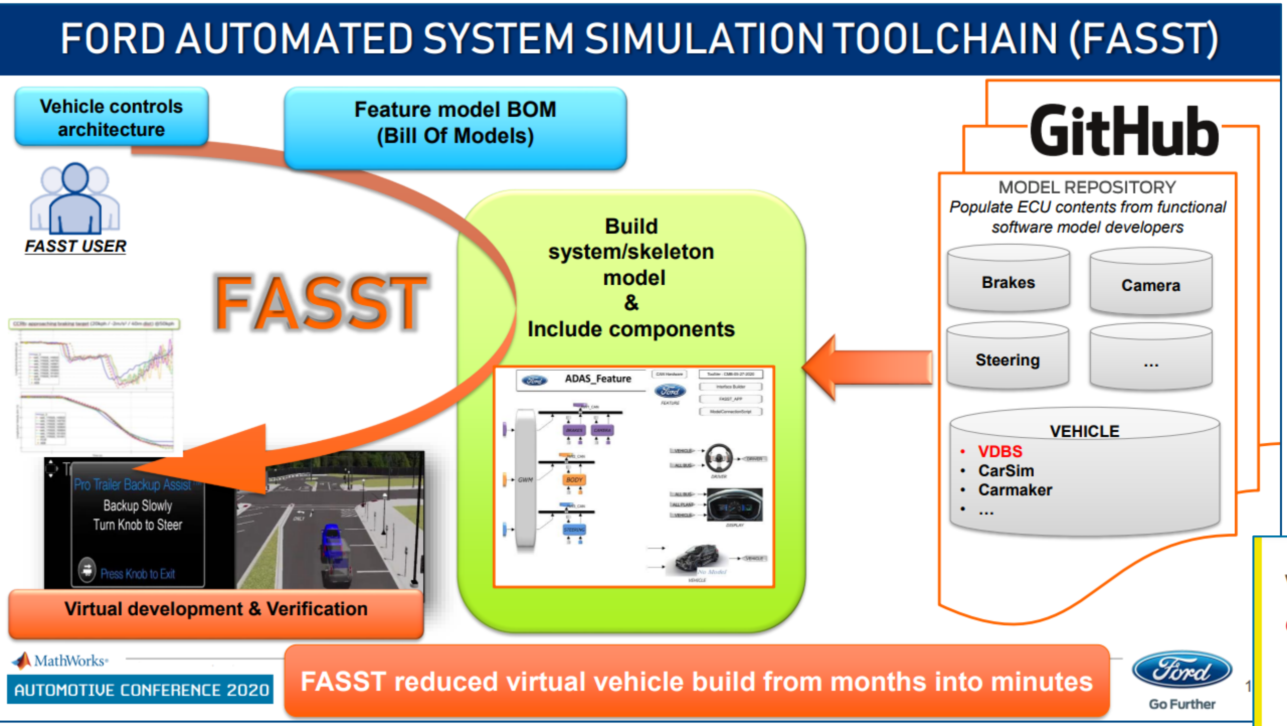

MathWorks AUTOMOTIVE CONFERENCE 2023
Context: Renault SDV Project preparation
Renault strategic collaboration with Google: Android Automotive OS replaces Adaptive AUTOSAR
New Interface Definition Language: Android IDL (used for IPC generation)
Service Oriented Architecture maintained (Request/Response methods => RPC, events => RPC + Callbacks)
Software Architecture
UML
SWC + Interfaces
Renault ARXML Gen Tool
Software Component Description
Adaptive AUTOSAR
ARXML
translation
Simulink ARXML import + Modeling + Code Gen
MathWorks
Embedded Coder C++
AIDL binding
AIDLs
android
Application
RPC: Remote Procedure Call
IPC: Inter-Process Communication

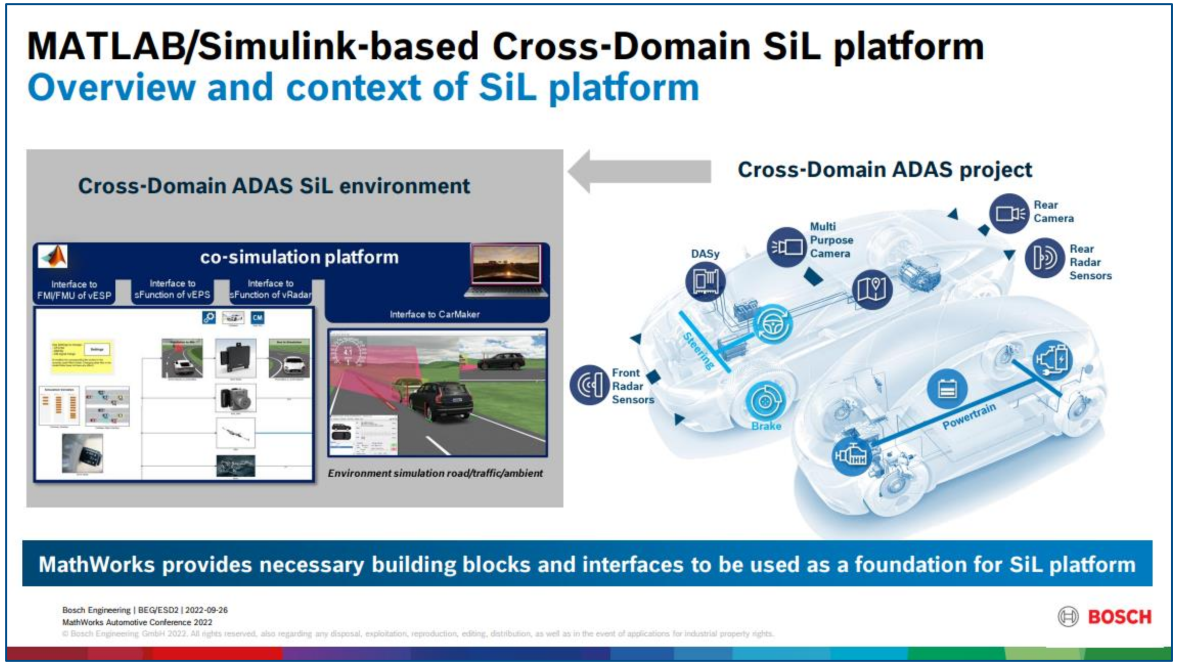

MATLAB EXPO
How to Model on In-vehicle OS
New Style!
Application
Application
Middleware
In-vehicle OS
Embedded Coder
Generator
middleware
CHIP

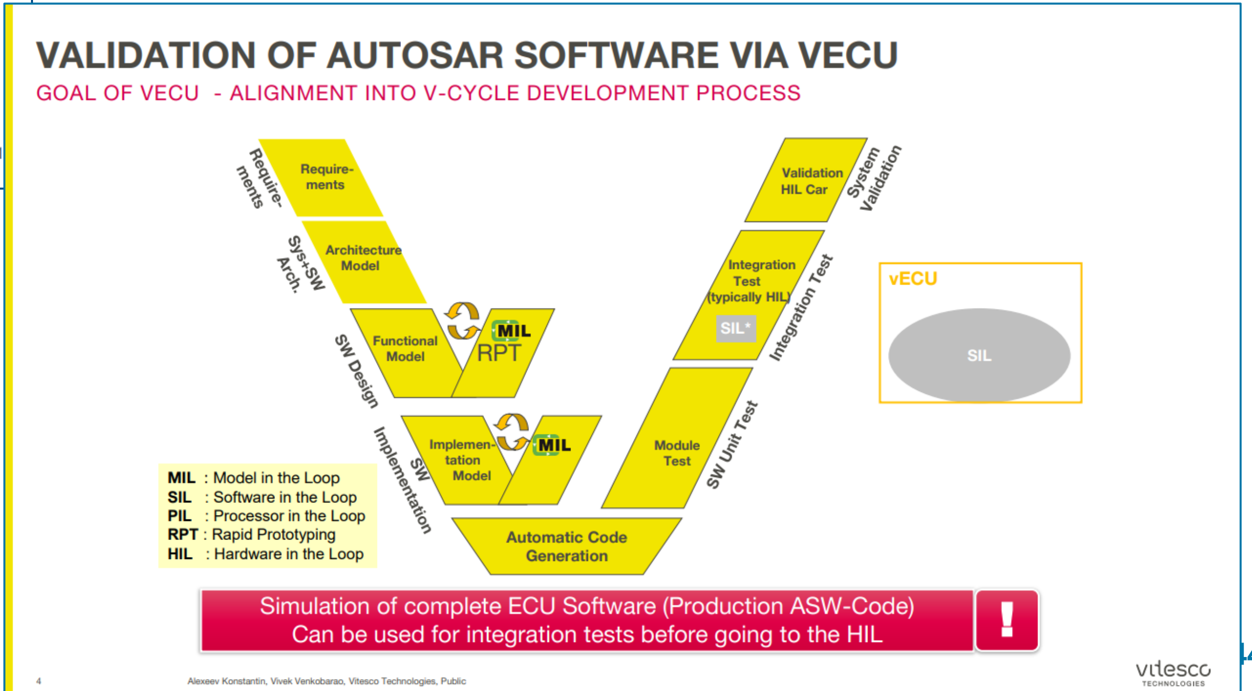


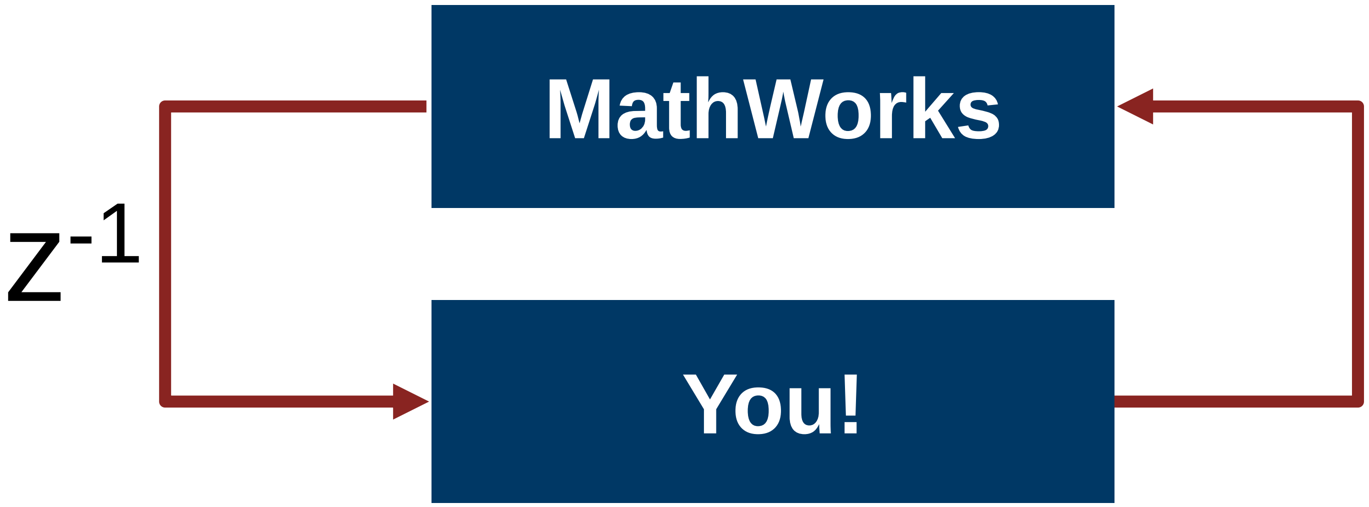


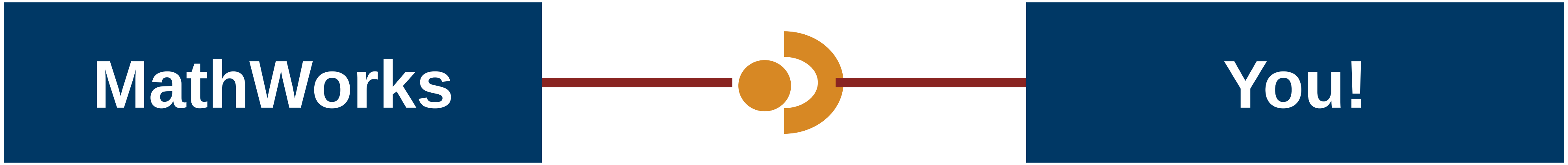
[Link to the talk]


FORD AUTOMATED SYSTEM SIMULATION TOOLCHAIN (FASST)
Vehicle controls architecture
Feature model BOM (Bill Of Models)
FASST USER
FASST
Build system/skeleton model & Include components
ADAS_Feature
GitHub
MODEL REPOSITORY
Populate ECU contents from functional software model developers
Brakes
Camera
Steering
...
VEHICLE
VDBS
CarSim
Carmaker
...
Pro Trailer Backup Assist
Backup Slowly
Turn Knob to Steer
Press Knob to Exit
Virtual development & Verification
MathWorks
AUTOMOTIVE CONFERENCE 2020
FASST reduced virtual vehicle build from months into minutes
Ford
Go Further


MATLAB/Simulink-based Cross-Domain SiL platform
Overview and context of SiL platform
Cross-Domain ADAS SiL environment
Cross-Domain ADAS project
co-simulation platform
Interface to FMI/FMU of vESP
Interface to sFunction of vEPS
Interface to sFunction of vRadar
Interface to CarMaker
Environment simulation road/traffic/ambient
DASy
Multi Purpose Camera
Rear Camera
Rear Radar Sensors
Front Radar Sensors
Steering
Brake
Powertrain
MathWorks provides necessary building blocks and interfaces to be used as a foundation for SiL platform
Bosch Engineering | BEG/ESD2 | 2022-09-26
MathWorks Automotive Conference 2022
BOSCH


VALIDATION OF AUTOSAR SOFTWARE VIA VECU
GOAL OF VECU - ALIGNMENT INTO V-CYCLE DEVELOPMENT PROCESS
Require-ments
Requirements
Sys+SW Arch.
Architecture Model
SW Design
Functional Model
MIL
RPT
SW Implementation
Implementation Model
MIL
Automatic Code Generation
SW Unit Test
Module Test
Integration Test
Integration Test (typically HIL)
SIL*
System Validation
Validation HIL Car
vECU
SIL
MIL : Model in the Loop
SIL : Software in the Loop
PIL : Processor in the Loop
RPT : Rapid Prototyping
HIL : Hardware in the Loop
Simulation of complete ECU Software (Production ASW-Code)
Can be used for integration tests before going to the HIL
!
Alexeev Konstantin, Vivek Venkobarao, Vitesco Technologies, Public
vitesco TECHNOLOGIES


MathWorks
z^{-1}
You!


MathWorks
You!

Questions