MATLAB EXPO

INDIA

"Shaping Future Software Factories: Leveraging Model-Based Design for Scalability from Desktop to Cloud"

Rajat Arora,
Automotive Products

Gaurav Dubey,
Aerospace Products

MathWorks®

Panelists

Nukul Sehgal

Code Generation, Virtualization & DevOps

Gaurav Ahuja

Safety Standards, V&V & Code Generation

Vamshi Kumbham

MBSE, Systems & Software Simulation

The rush for ~~Gold~~ **Software**

From the news…

*“**Software strategy** is one of the **key building blocks** of Stellantis' overall strategy to build the most sustainable mobility for our customers.”*
Carlos Tavares – Stellantis CEO

*“The vehicle is no longer the central point of the automotive value chain, as **software, electronics and on-board intelligence increasingly determine both the value and use** of the vehicle for new mobility needs and services.”*
Luca de Meo – Renault Group CEO

***Build products to evolve.** As a progression from the historical development approach of “build to last,” **Aerospace & Defense developers** are now looking to build products to evolve. From satellites constructed at a fraction of the cost with **software that can be updated over the air** with commercially available technology, to on-the-spot defense solutions to conflict and warfare, leaders must evolve models to keep pace*.
Excerpt from Bain & Co Press Release by Jim Harris, Partner

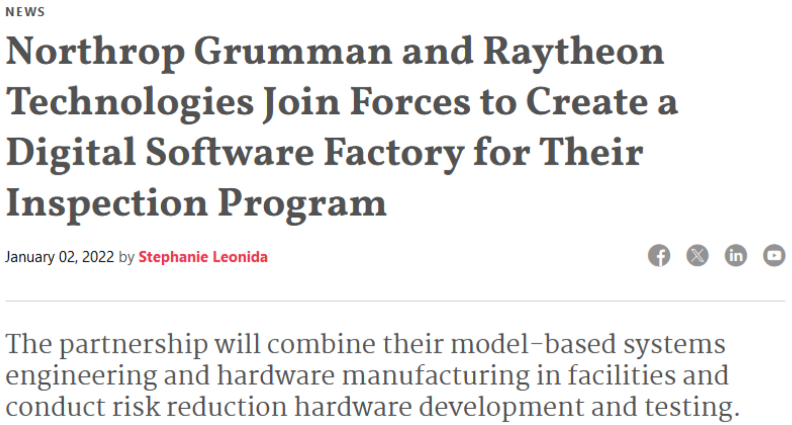
https://www.stellantis.com/en/investors/events/sw-day-2021

https://www.renaultgroup.com/en/news-on-air/news/the-software-republique-a-new-ecosystem-for-innovation-in-intelligent-and-sustainable-mobility/

https://www.volkswagenag.com/en/strategy/software.html

https://www.bain.com/about/media-center/press-releases/2023/aerospace-and-defense-executives-to-increase-engineering-and-rd-investment-over-the-next-three-years-to-digitize-value-chains-meet-sustainability-targets/

Software Factory



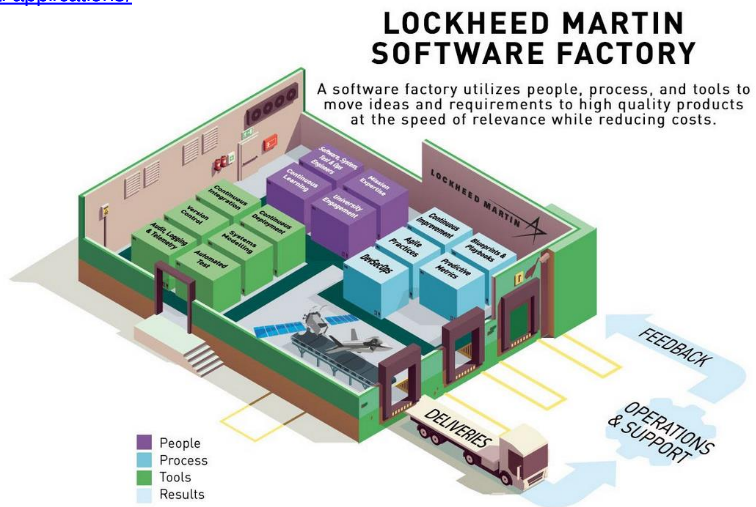
NEWS

Northrop Grumman and Raytheon Technologies Join Forces to Create a Digital Software Factory for Their Inspection Program

January 02, 2022 by Stephanie Leonida

The partnership will combine their model-based systems engineering and hardware manufacturing in facilities and conduct risk reduction hardware development and testing.

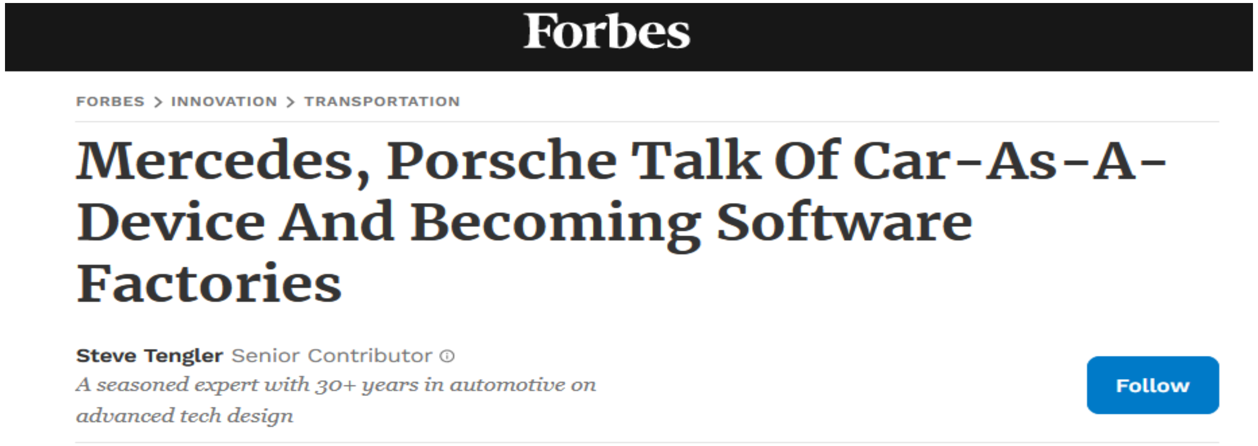
https://control.com/news/northrop-grumman-and-raytheon-technologies-join-forces-to-design-inspection-systems-for-industrial-applications/



LOCKHEED MARTIN SOFTWARE FACTORY

A software factory utilizes people, process, and tools to move ideas and requirements to high quality products at the speed of relevance while reducing costs.

https://www.lockheedmartin.com/en-us/capabilities/digital-transformation/software-factory.html



Forbes

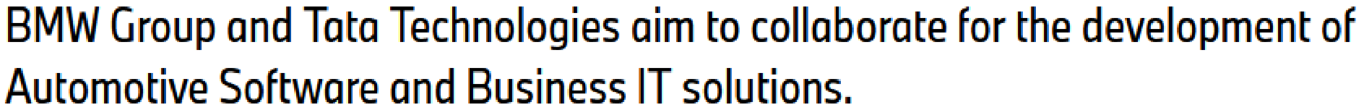
FORBES > INNOVATION > TRANSPORTATION

Mercedes, Porsche Talk Of Car-As-A-Device And Becoming Software Factories

Steve Tengler Senior Contributor

A seasoned expert with 30+ years in automotive on advanced tech design

https://www.forbes.com/sites/stevetengler/2023/10/10/mercedes-porsche-talk-of-car-as-a-device-and-becoming-software-factories/



BMW Group and Tata Technologies aim to collaborate for the development of Automotive Software and Business IT solutions.

https://www.press.bmwgroup.com/global/article/detail/T0439143EN/bmw-group-and-tata-technologies-aim-to-collaborate-for-the-development-of-automotive-software-and-business-it-solutions?language=en/

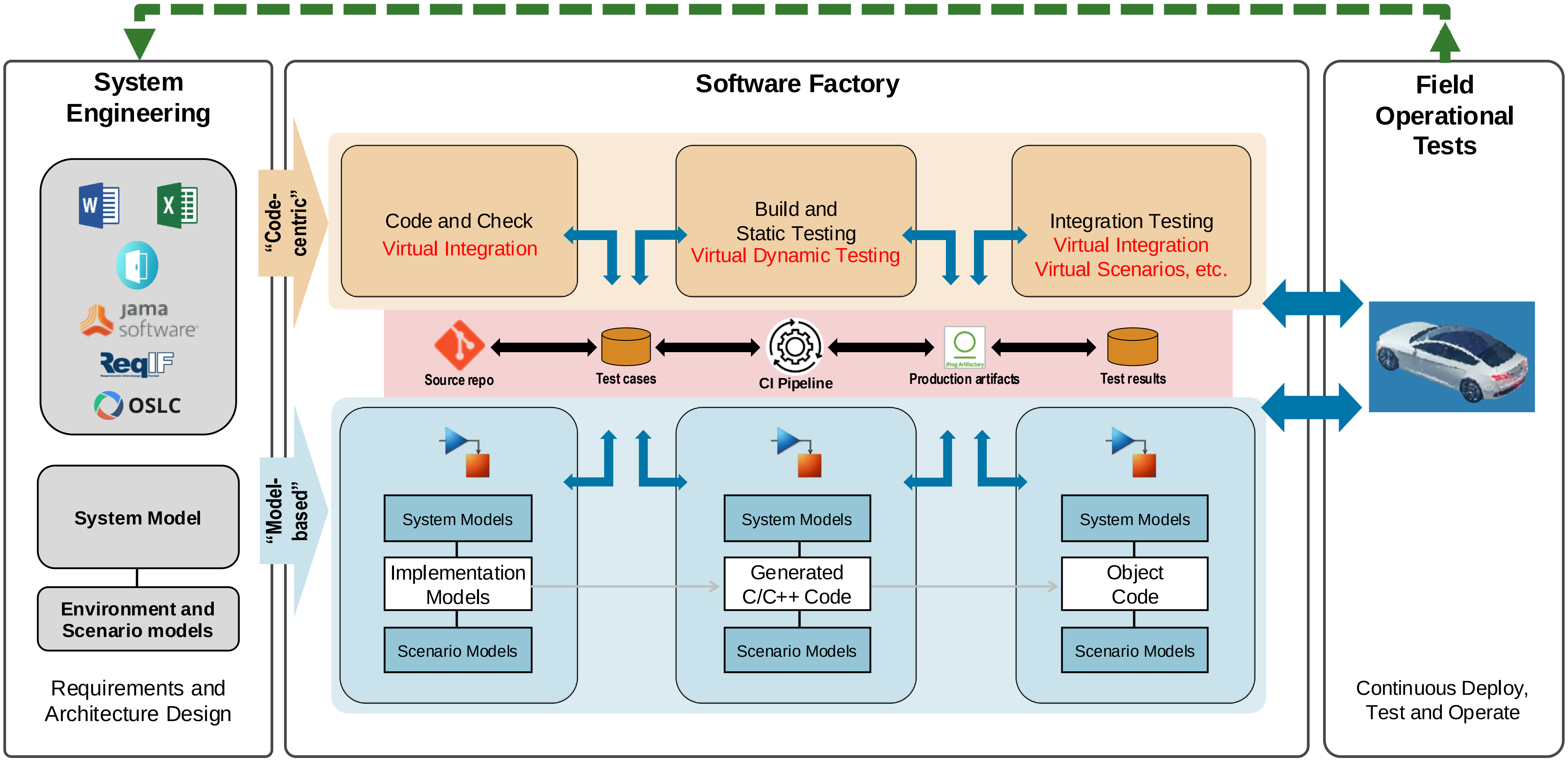
Technology Driven Amalgamation of ***Process, People, Methods and Standards*** to ship ***safe and secure products*** with ***high level of agility*** that ***enhance customer comfort*** and ***experience*** and ***unlock new revenue*** streams

The path forward requires four strategic clusters of action:

1. **Process**
   - Align ***software development and system engineering*** approaches to ***handle complexity***
2. **People**
   - Collaborative, building synergies with new teams to ***enhance productivity***
   - Domain skills, re- and up-skilling the existing work force
3. **Methods**
   - Agile, DevOps ***to react to changes***
   - Parallelize and virtualize development ***to reduce dependency on physical prototypes***
   - “Software factory” mindset of development-process automation ***for speed and consistency***
4. **Standards**
   - Legislative regulations, functional safety, cyber-security, AUTOSAR compliance, etc. ***to ensure safety, security and reliability***

Software Factory- A Shift From Desktop to Cloud – An Industry View

Aligning and Automating MBD and Code-Centric Approaches



System Engineering

jama software

ReqIF

OSLC

System Model

Environment and Scenario models

Requirements and Architecture Design

Software Factory

"Code-centric"

Code and Check
Virtual Integration

Build and Static Testing
Virtual Dynamic Testing

Integration Testing
Virtual Integration
Virtual Scenarios, etc.

Source repo

Test cases

CI Pipeline

Production artifacts

Test results

"Model-based"

System Models

Implementation Models

Scenario Models

System Models

Generated C/C++ Code

Scenario Models

System Models

Object Code

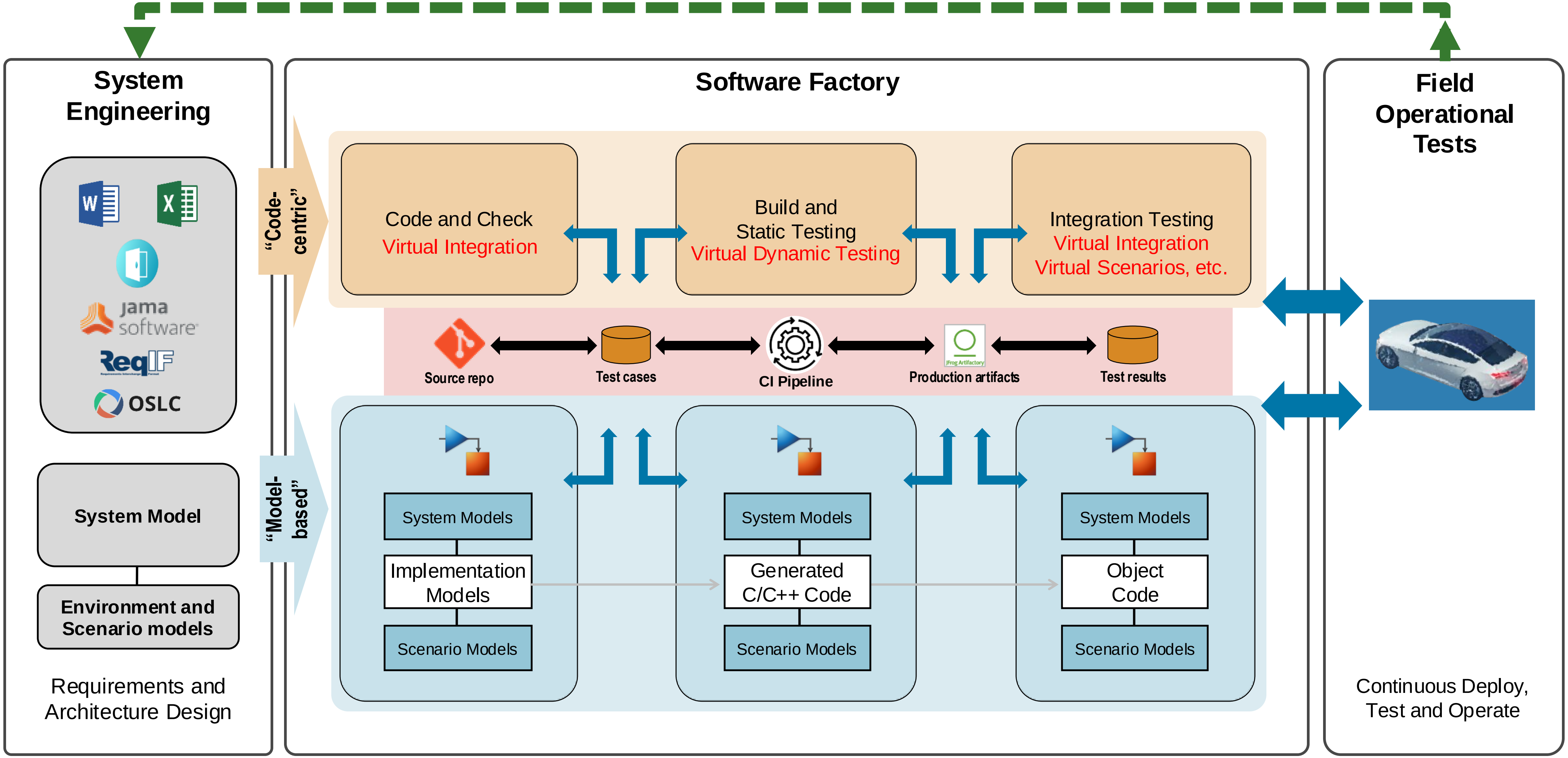
Scenario Models

Field Operational Tests

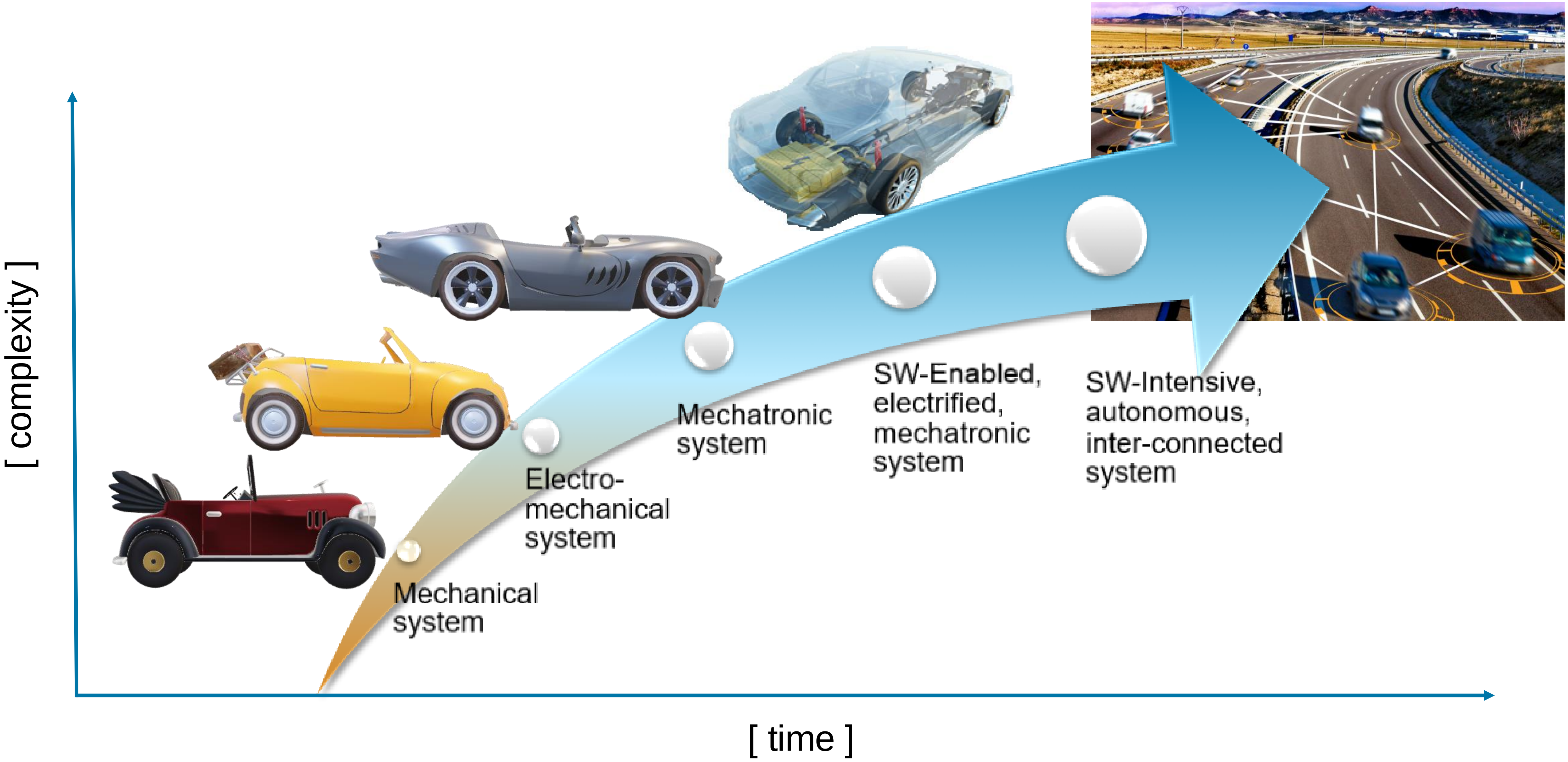
Continuous Deploy, Test and Operate

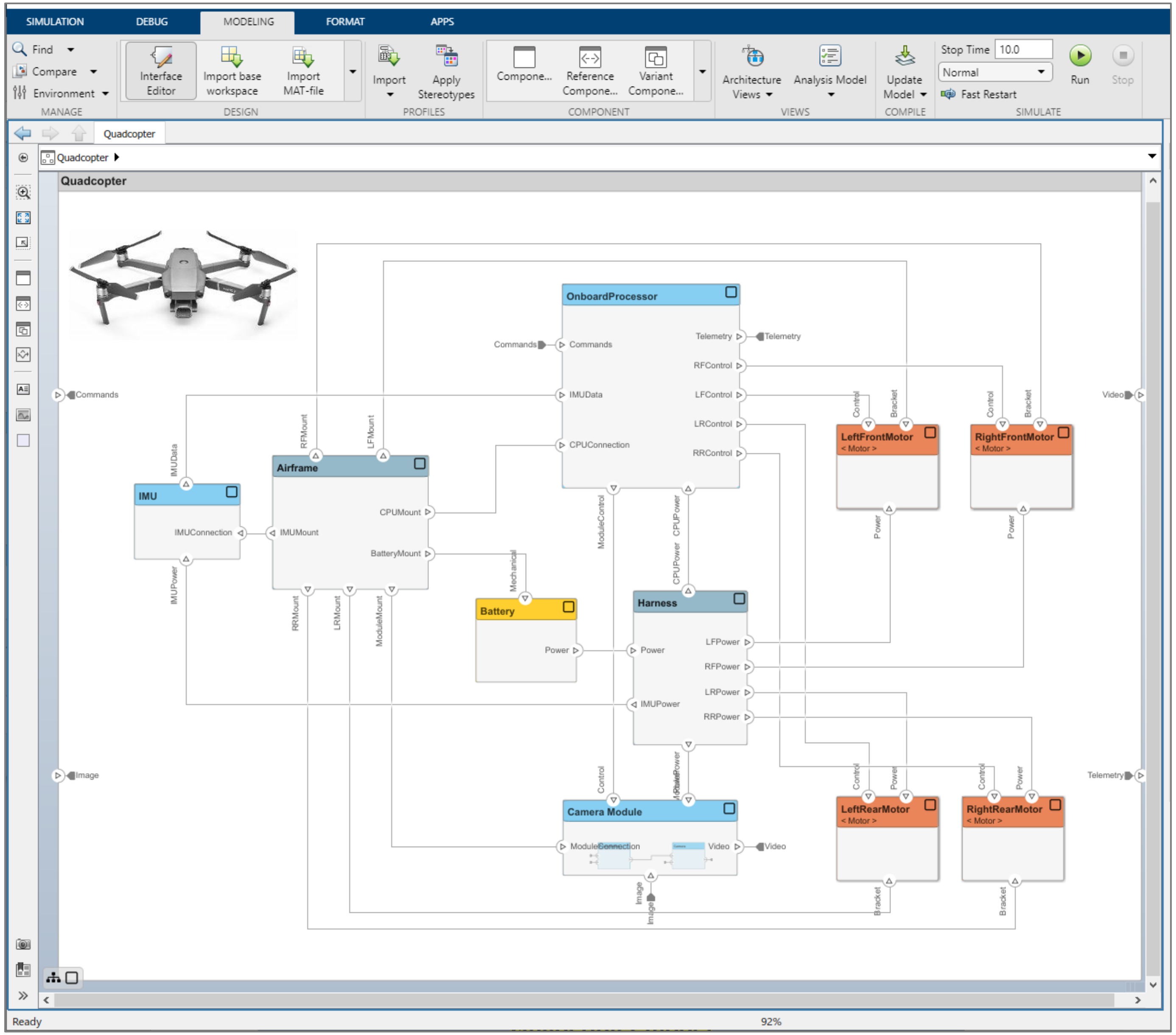
Software Factory – Handling the complexities❓

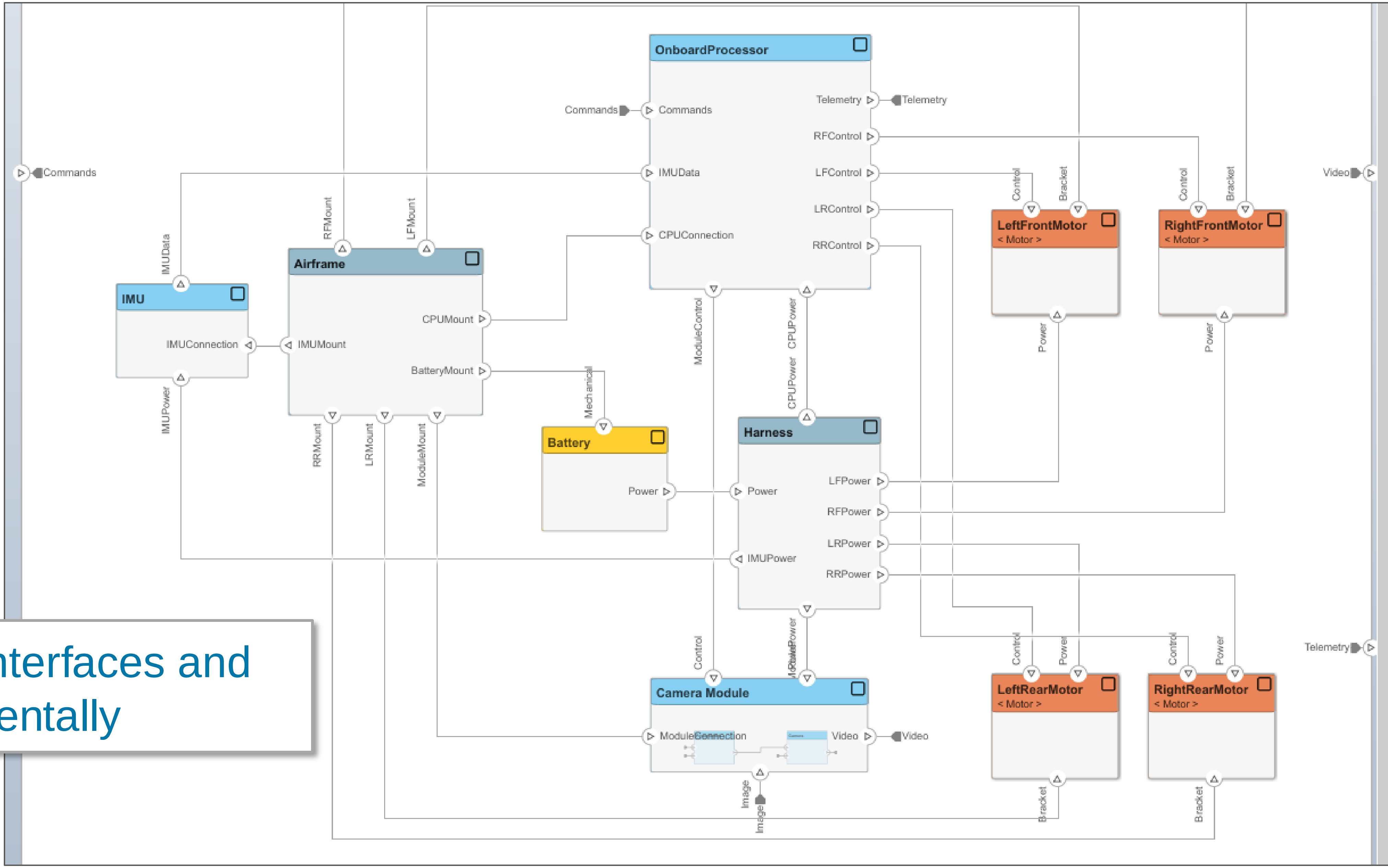
Aligning and Automating MBD and Code-Centric Approaches



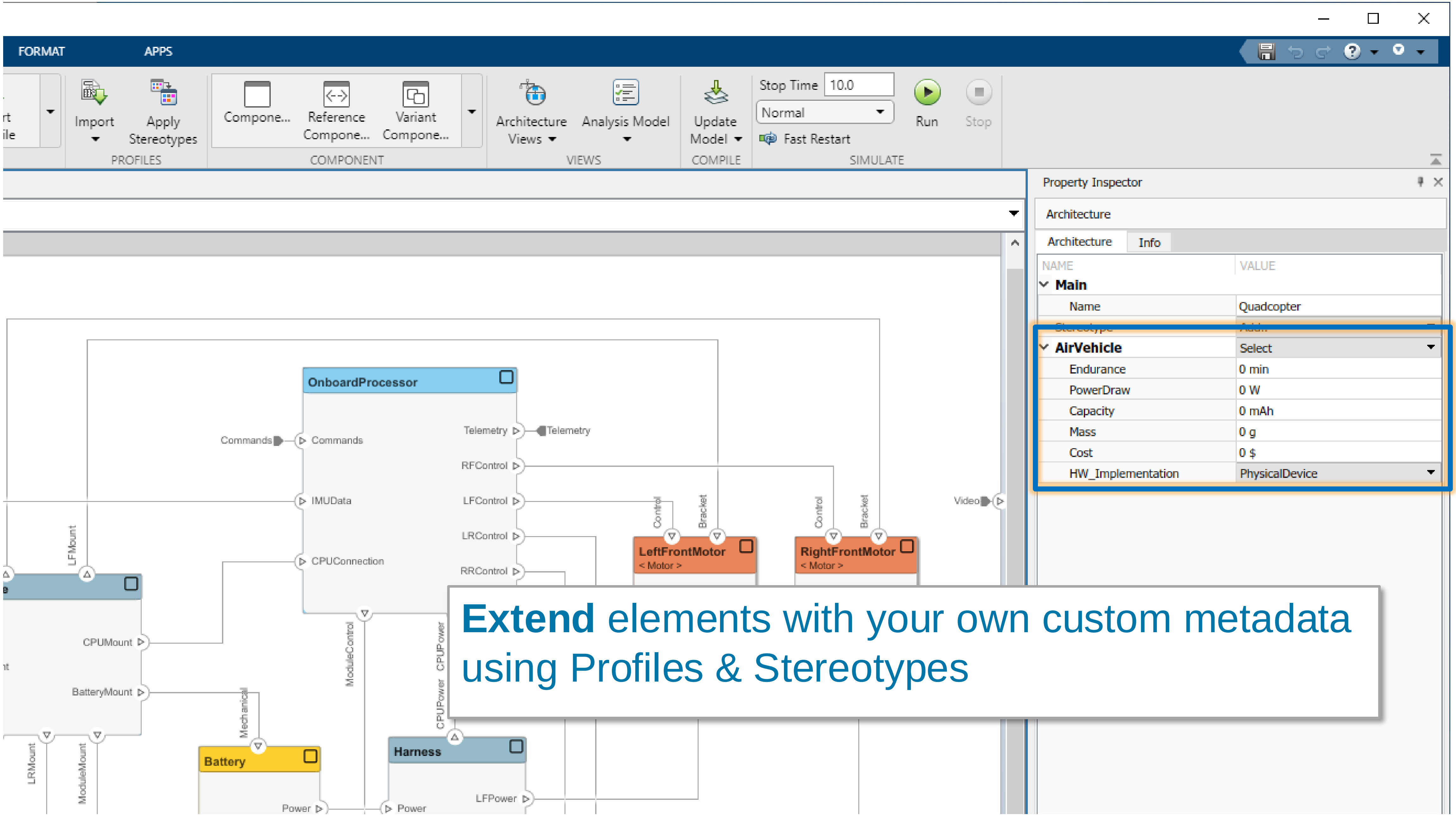
System Complexity


[complexity]
Mechanical system
Electro-mechanical system
Mechatronic system
SW-Enabled, electrified, mechatronic system
SW-Intensive, autonomous, inter-connected system
[time]


SIMULATION
DEBUG
MODELING
FORMAT
APPS
Find
Compare
Environment
MANAGE
Interface Editor
Import base workspace
Import MAT-file
DESIGN
Import
Apply Stereotypes
PROFILES
Compone...
Reference Compone...
Variant Compone...
COMPONENT
Architecture Views
Analysis Model
VIEWS
Update Model
COMPILE
Stop Time 10.0
Normal
Fast Restart
Run
Stop
SIMULATE
Quadcopter
Quadcopter
Quadcopter
OnboardProcessor
Commands
Telemetry
RFControl
LFControl
LRControl
RRControl
IMUData
CPUConnection
ModuleControl
CPUPower
Airframe
CPUMount
IMUMount
BatteryMount
RFMount
LFMount
RRMount
LRMount
ModuleMount
IMU
IMUConnection
IMUData
IMUPower
Battery
Power
Mechanical
Harness
Power
LFPower
RFPower
LRPower
RRPower
IMUPower
CPUPower
ModulePower
LeftFrontMotor
< Motor >
RightFrontMotor
< Motor >
LeftRearMotor
< Motor >
RightRearMotor
< Motor >
Control
Bracket
Power
Camera Module
ModuleConnection
Video
Control
Image
Commands
Image
Video
Telemetry
Ready
92%



Sketch system interfaces and elaborate incrementally



Stop Time 10.0

Normal

Property Inspector

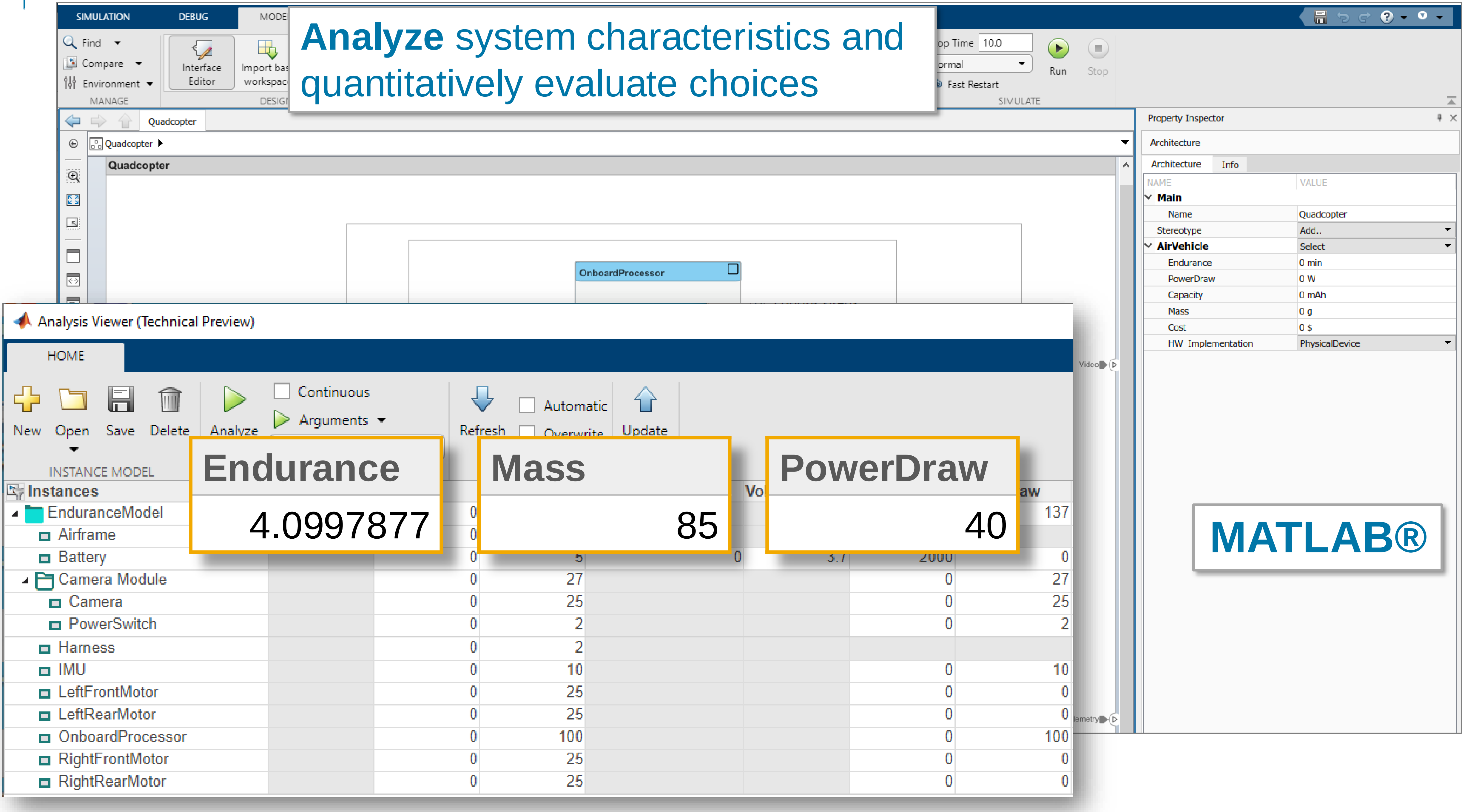
Architecture

Architecture | Info

NAME	VALUE
Main	
Name	Quadcopter
Stereotype	Add...
AirVehicle	Select
Endurance	0 min
PowerDraw	0 W
Capacity	0 mAh
Mass	0 g
Cost	0 $
HW_Implementation	PhysicalDevice

Extend elements with your own custom metadata using Profiles & Stereotypes

Analyze system characteristics and quantitatively evaluate choices



Property Inspector

Architecture

Architecture | Info

NAME	VALUE
Main	
Name	Quadcopter
Stereotype	Add..
AirVehicle	Select
Endurance	0 min
PowerDraw	0 W
Capacity	0 mAh
Mass	0 g
Cost	0 $
HW_Implementation	PhysicalDevice

Endurance
4.0997877

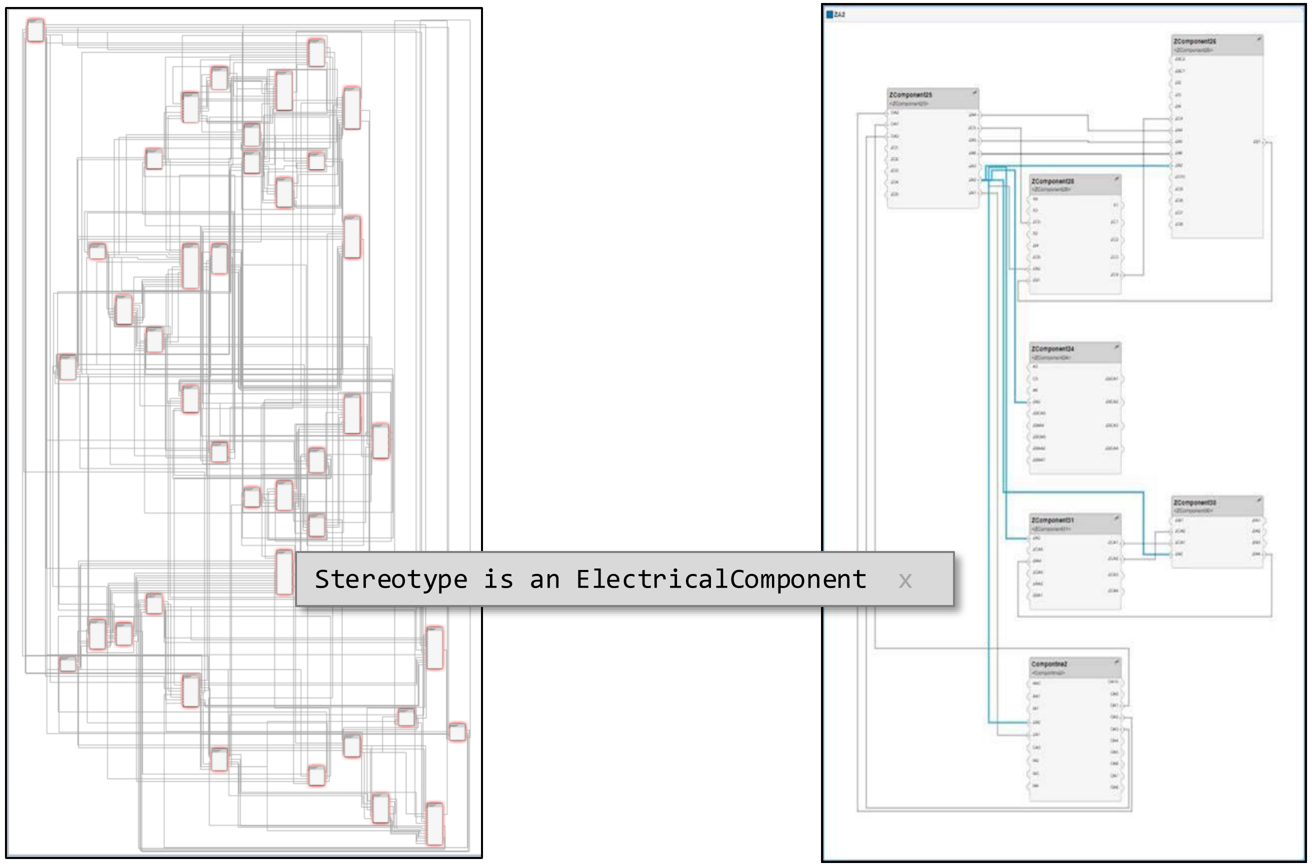
Mass
85

PowerDraw
40

Instances					Vo		aw
EnduranceModel		0					137
Airframe		0					
Battery		0	5	0	3.7	2000	0
Camera Module		0	27			0	27
Camera		0	25			0	25
PowerSwitch		0	2			0	2
Harness		0	2				
IMU		0	10			0	10
LeftFrontMotor		0	25			0	0
LeftRearMotor		0	25			0	0
OnboardProcessor		0	100			0	100
RightFrontMotor		0	25			0	0
RightRearMotor		0	25			0	0

MATLAB®

Simplify the complex with Filters and autogenerated Views

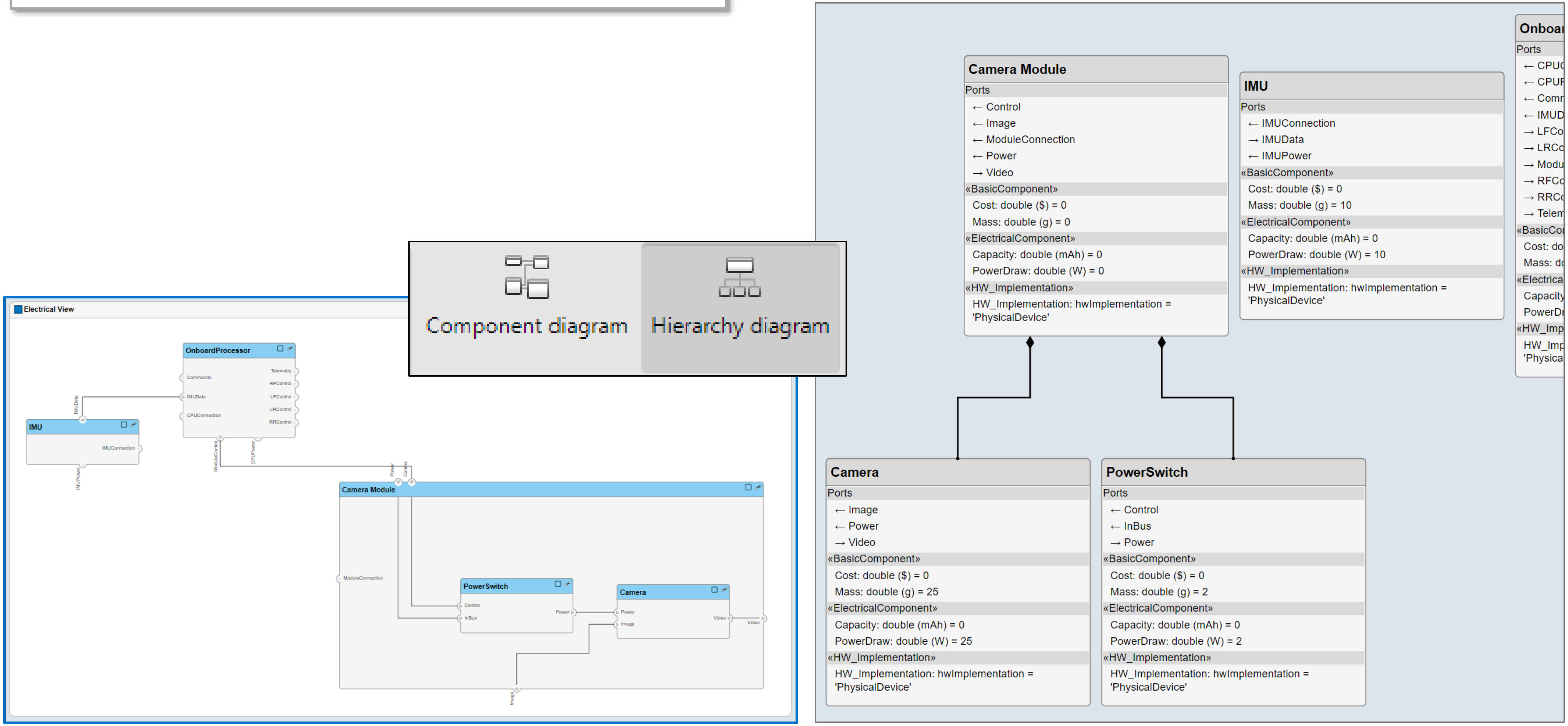


Stereotype is an ElectricalComponent x

Full system model

Filtered view

Simplify the complex with Filters and autogenerated Views



Camera Module
Ports
← Control
← Image
← ModuleConnection
← Power
→ Video
«BasicComponent»
Cost: double ($) = 0
Mass: double (g) = 0
«ElectricalComponent»
Capacity: double (mAh) = 0
PowerDraw: double (W) = 0
«HW_Implementation»
HW_Implementation: hwImplementation = 'PhysicalDevice'

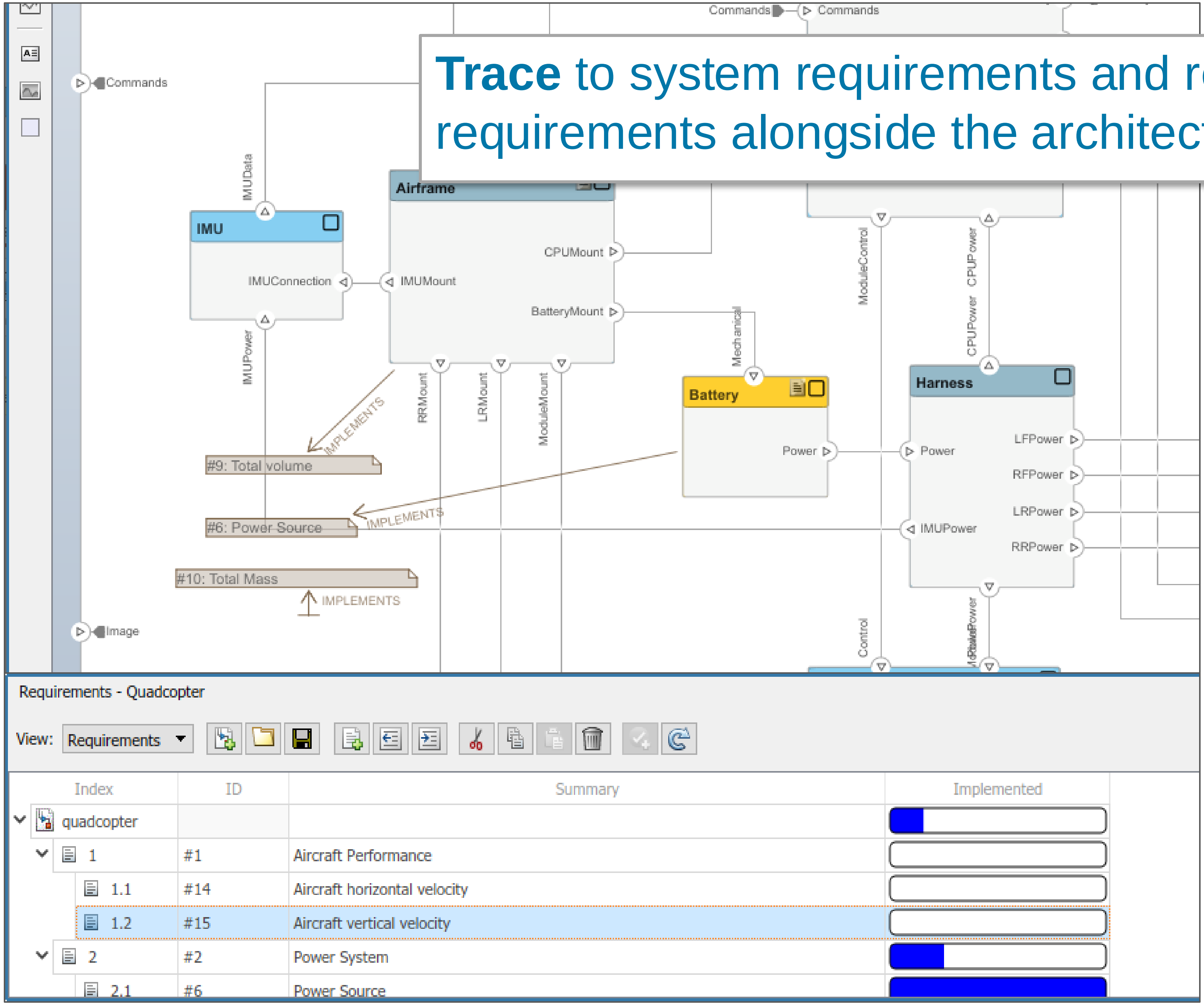
IMU
Ports
← IMUConnection
→ IMUData
← IMUPower
«BasicComponent»
Cost: double ($) = 0
Mass: double (g) = 10
«ElectricalComponent»
Capacity: double (mAh) = 0
PowerDraw: double (W) = 10
«HW_Implementation»
HW_Implementation: hwImplementation = 'PhysicalDevice'

Onboar
Ports
← CPUC
← CPUF
← Comr
← IMUD
→ LFCo
→ LRCo
→ Modu
→ RFCo
→ RRCo
→ Telem
«BasicCo
Cost: do
Mass: do
«Electrica
Capacity
PowerDr
«HW_Imp
HW_Imp
'Physica

Camera
Ports
← Image
← Power
→ Video
«BasicComponent»
Cost: double ($) = 0
Mass: double (g) = 25
«ElectricalComponent»
Capacity: double (mAh) = 0
PowerDraw: double (W) = 25
«HW_Implementation»
HW_Implementation: hwImplementation = 'PhysicalDevice'

PowerSwitch
Ports
← Control
← InBus
→ Power
«BasicComponent»
Cost: double ($) = 0
Mass: double (g) = 2
«ElectricalComponent»
Capacity: double (mAh) = 0
PowerDraw: double (W) = 2
«HW_Implementation»
HW_Implementation: hwImplementation = 'PhysicalDevice'

Trace to system requirements and refine requirements alongside the architecture

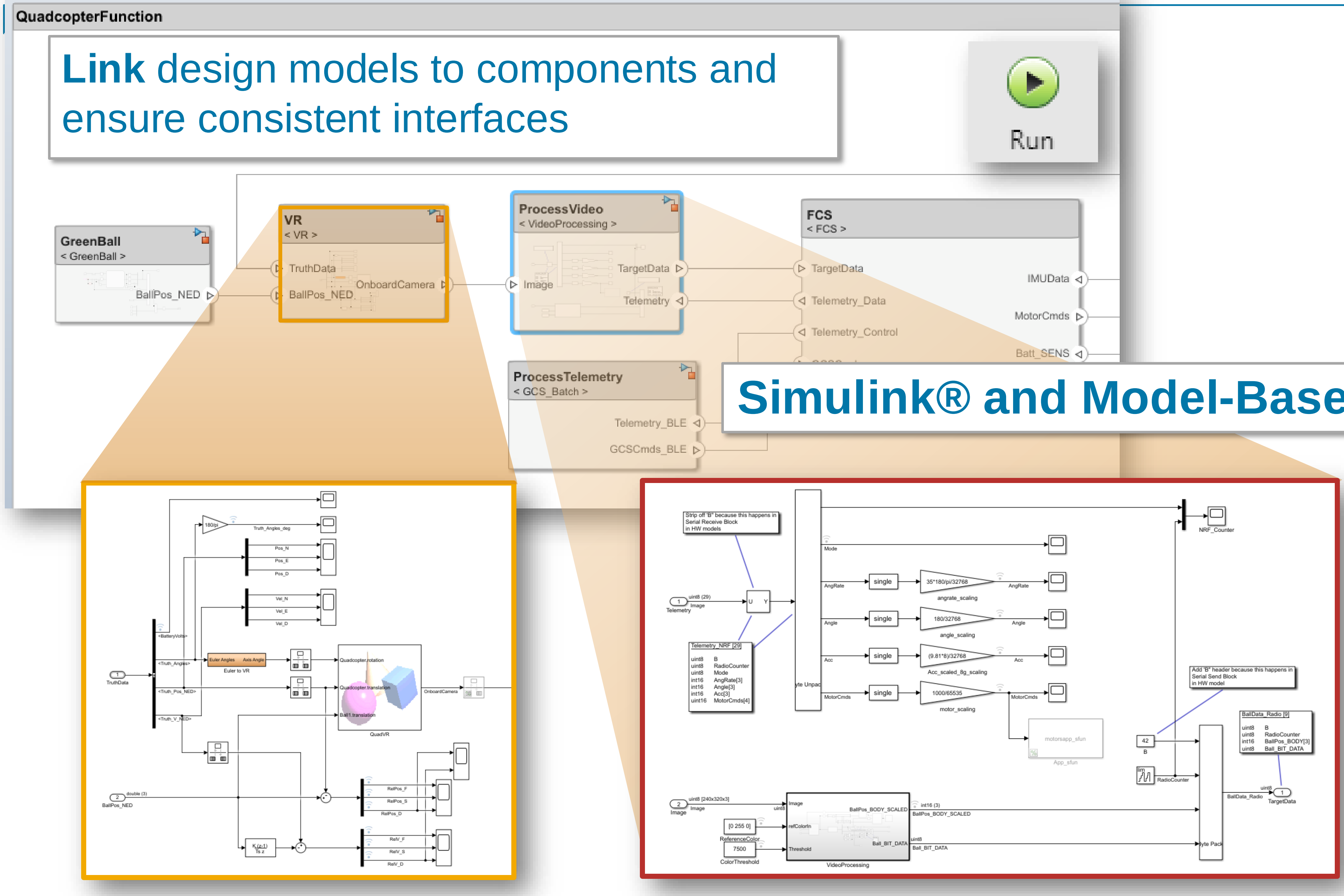


Requirements - Quadcopter

View: Requirements

Index	ID	Summary	Implemented
quadcopter			
1	#1	Aircraft Performance	
1.1	#14	Aircraft horizontal velocity	
1.2	#15	Aircraft vertical velocity	
2	#2	Power System	
2.1	#6	Power Source	

Link design models to components and ensure consistent interfaces



Simulink® and Model-Based Design

Software Factory – Handling the complexities ✓

Safety and reliability ❓

> "The more certain we are about our knowledge, the more we should question it.", **Aristotle.**

- High integrity applications development follows standards and guidelines
- Demonstrate compliance…

DO 178C
Functional Safety Avionics

ISO 26262
Functional Safety Automotive

EN 50128
Functional Safety Railway

IEC 62304
Functional Safety Medical

DO 254
Functional Safety Avionics

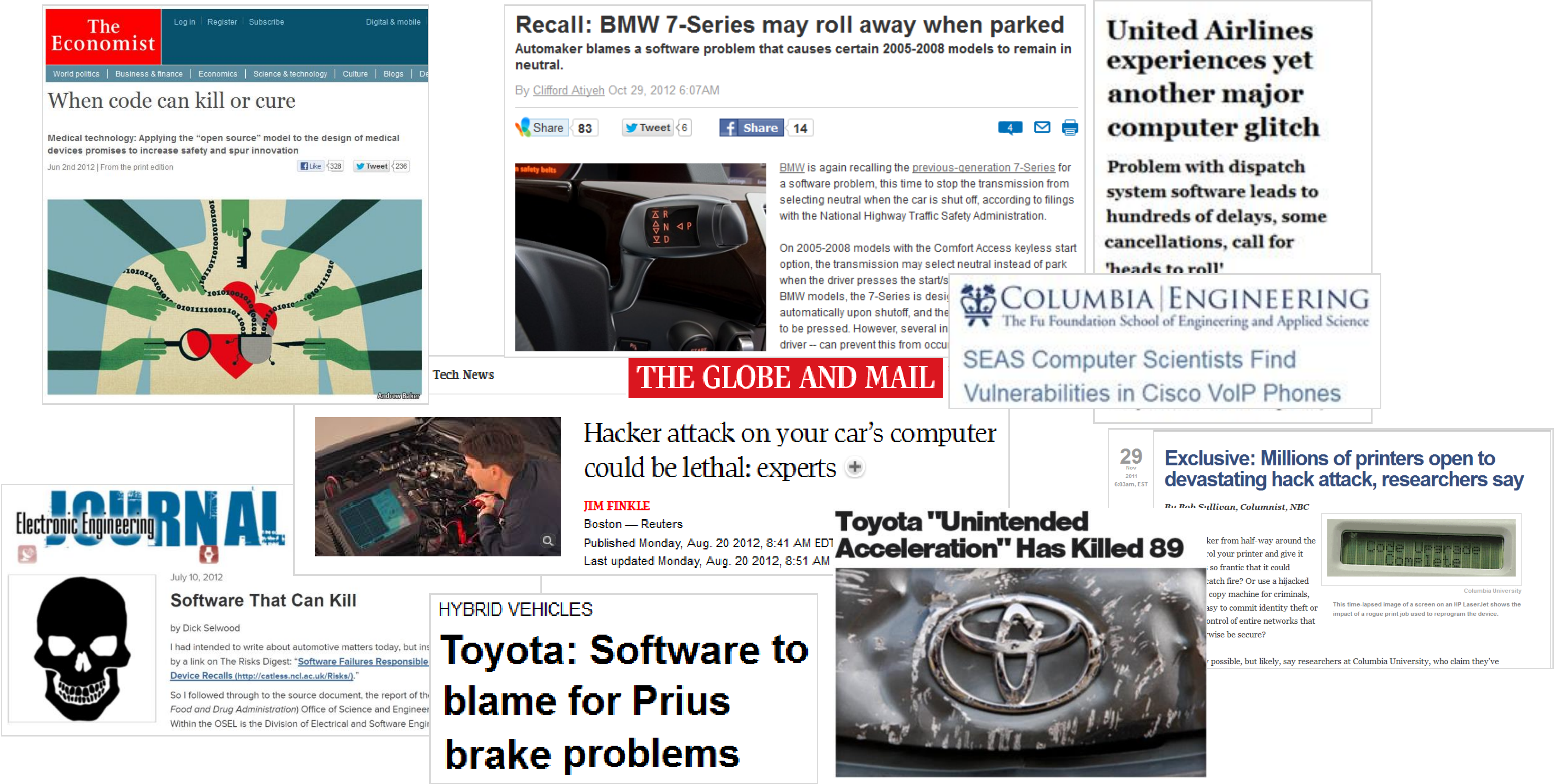
IEC 61508
Functional Safety Industrial Automation

IEC 62061
Functional Safety Machinery

ISO 25119
Functional Safety Agricultural Machines

“Even when you think you’ve tested everything that you can possibly imagine, you’re wrong.” [3]

- **Glenn E. Reeves**, Mars Pathfinder Software Team Leader



The Economist

Log in | Register | Subscribe | Digital & mobile

World politics | Business & finance | Economics | Science & technology | Culture | Blogs | De

When code can kill or cure

Medical technology: Applying the “open source” model to the design of medical devices promises to increase safety and spur innovation

Jun 2nd 2012 | From the print edition

Like 328 Tweet 236

Andrew Baker

Recall: BMW 7-Series may roll away when parked

Automaker blames a software problem that causes certain 2005-2008 models to remain in neutral.

By Clifford Atiyeh Oct 29, 2012 6:07AM

Share 83 Tweet 6 Share 14

BMW is again recalling the previous-generation 7-Series for a software problem, this time to stop the transmission from selecting neutral when the car is shut off, according to filings with the National Highway Traffic Safety Administration.

On 2005-2008 models with the Comfort Access keyless start option, the transmission may select neutral instead of park when the driver presses the start/s
BMW models, the 7-Series is desi
automatically upon shutoff, and the
to be pressed. However, several in
driver -- can prevent this from occu

United Airlines experiences yet another major computer glitch

Problem with dispatch system software leads to hundreds of delays, some cancellations, call for 'heads to roll'

COLUMBIA | ENGINEERING
The Fu Foundation School of Engineering and Applied Science

SEAS Computer Scientists Find Vulnerabilities in Cisco VoIP Phones

Tech News

THE GLOBE AND MAIL

Hacker attack on your car’s computer could be lethal: experts

JIM FINKLE
Boston — Reuters
Published Monday, Aug. 20 2012, 8:41 AM EDT
Last updated Monday, Aug. 20 2012, 8:51 AM

29 Nov 2011 6:03am, EST

Exclusive: Millions of printers open to devastating hack attack, researchers say

By Bob Sullivan, Columnist, NBC

ker from half-way around the
ol your printer and give it
so frantic that it could
atch fire? Or use a hijacked
copy machine for criminals,
sy to commit identity theft or
ontrol of entire networks that
wise be secure?

possible, but likely, say researchers at Columbia University, who claim they've

Code Upgrade Complete

Columbia University

This time-lapsed image of a screen on an HP LaserJet shows the impact of a rogue print job used to reprogram the device.

Electronic Engineering JOURNAL

July 10, 2012

Software That Can Kill

by Dick Selwood

I had intended to write about automotive matters today, but ins
by a link on The Risks Digest: “Software Failures Responsible
Device Recalls (http://catless.ncl.ac.uk/Risks/).”

So I followed through to the source document, the report of the
Food and Drug Administration) Office of Science and Engineer
Within the OSEL is the Division of Electrical and Software Engi

HYBRID VEHICLES

Toyota: Software to blame for Prius brake problems

Toyota "Unintended Acceleration" Has Killed 89

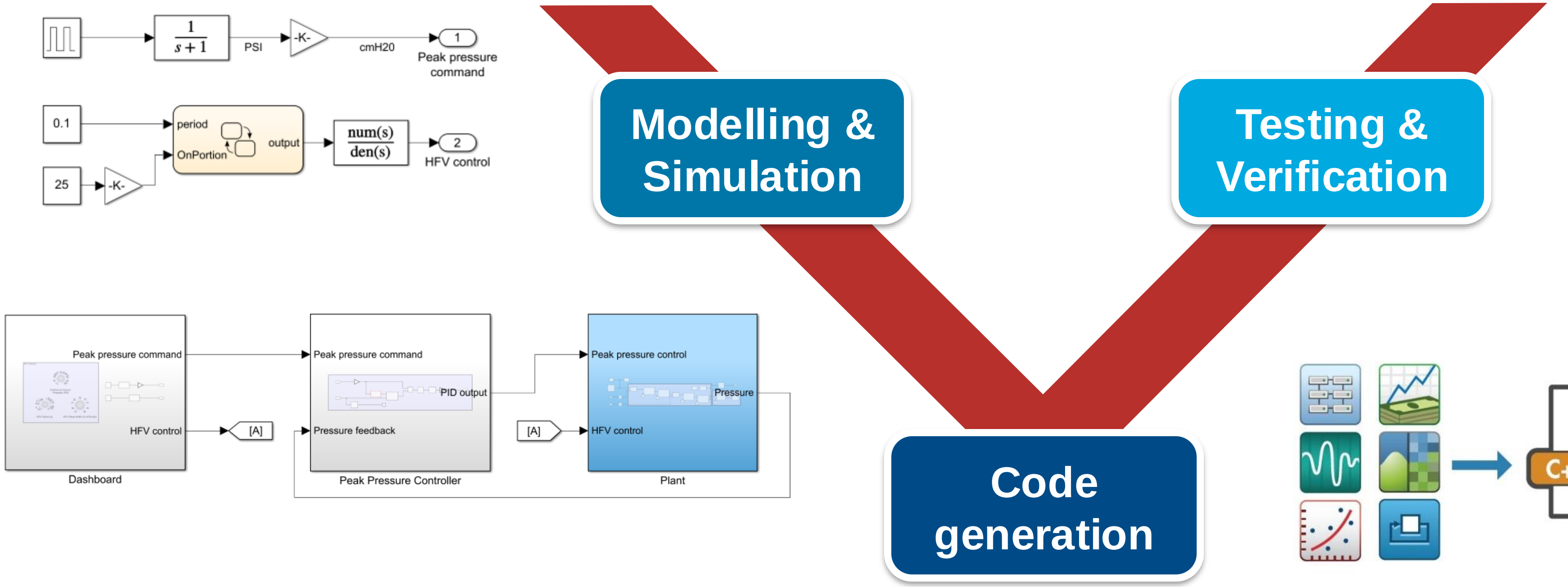
Shift Left

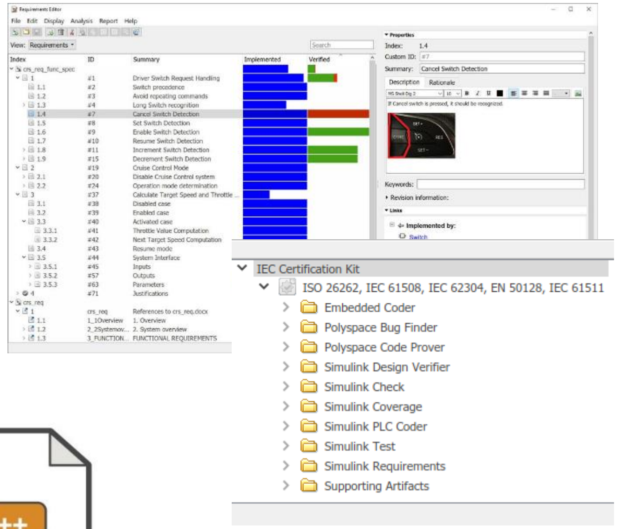
SIMULINK®

Model and Simulate Your System

Automatically Generate Code

Test Early and Often





Index	ID	Summary	Implemented	Verified
crs_req_func_spec				
1	#1	Driver Switch Request Handling		
1.1	#2	Switch precedence		
1.2	#3	Avoid repeating commands		
1.3	#4	Long Switch recognition		
1.4	#7	Cancel Switch Detection		
1.5	#8	Set Switch Detection		
1.6	#9	Enable Switch Detection		
1.7	#10	Resume Switch Detection		
1.8	#11	Increment Switch Detection		
1.9	#15	Decrement Switch Detection		
2	#19	Cruise Control Mode		
2.1	#20	Disable Cruise Control system		
2.2	#24	Operation mode determination		
3	#37	Calculate Target Speed and Throttle ...		
3.1	#38	Disabled case		
3.2	#39	Enabled case		
3.3	#40	Activated case		
3.3.1	#41	Throttle Value Computation		
3.3.2	#42	Next Target Speed Computation		
3.4	#43	Resume mode		
3.5	#44	System Interface		
3.5.1	#45	Inputs		
3.5.2	#57	Outputs		
3.5.3	#63	Parameters		
4	#71	Justifications		
crs_req				
1	crs_req	References to crs_req.docx		
1.1	1_1Overview	1. Overview		
1.2	2_2Systemov...	2. System overview		
1.3	3_FUNCTION...	FUNCTIONAL REQUIREMENTS		

Properties

Index: 1.4

Custom ID: #7

Summary: Cancel Switch Detection

Description | Rationale

If Cancel switch is pressed, it should be recognized.

Keywords:

Revision information:

Links

Implemented by:

Switch

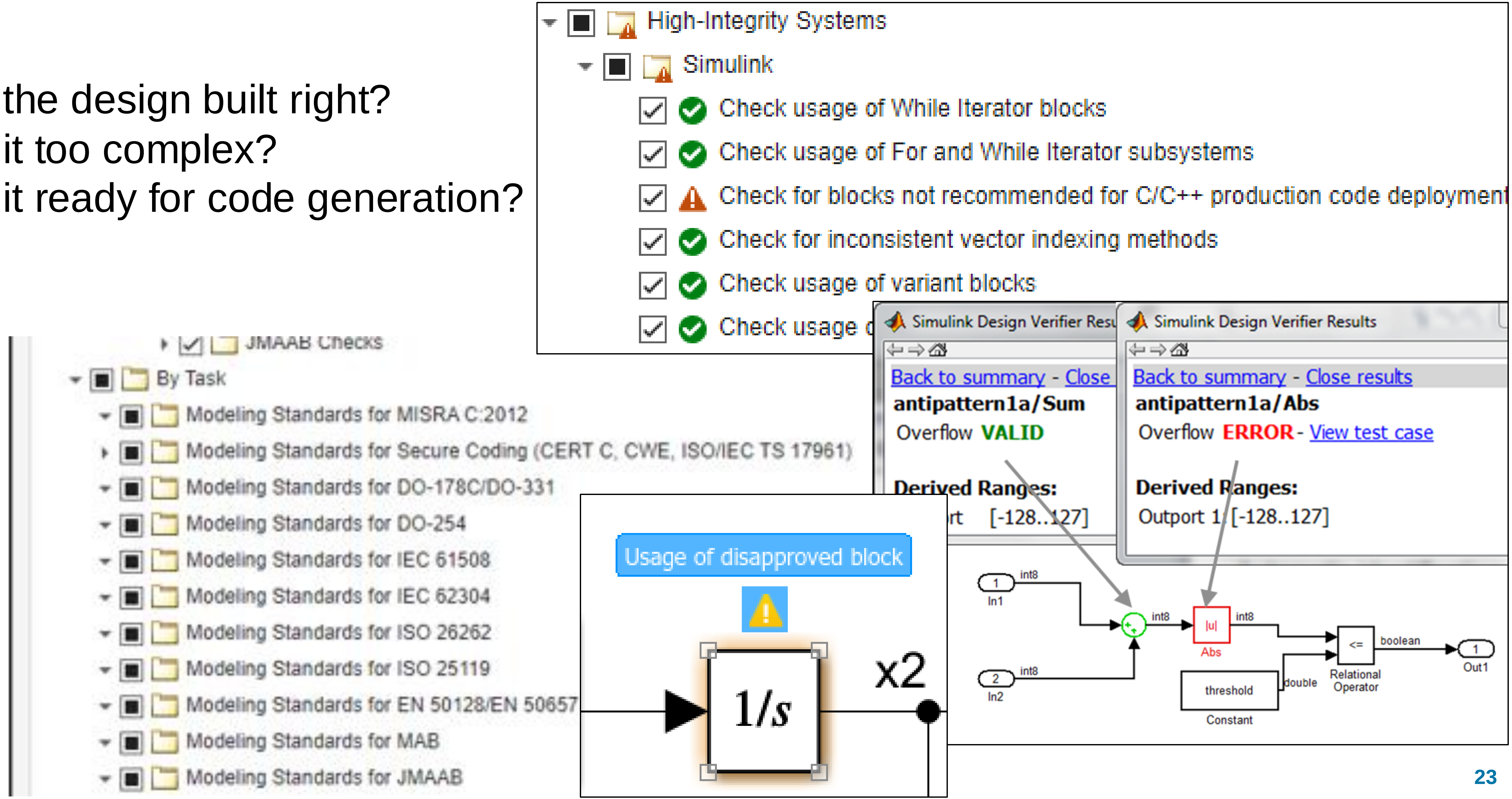
IEC Certification Kit

- ISO 26262, IEC 61508, IEC 62304, EN 50128, IEC 61511
  - Embedded Coder
  - Polyspace Bug Finder
  - Polyspace Code Prover
  - Simulink Design Verifier
  - Simulink Check
  - Simulink Coverage
  - Simulink PLC Coder
  - Simulink Test
  - Simulink Requirements
  - Supporting Artifacts

Meet Standards

Compliance to Standards and Guidelines

Is the design built right?
Is it too complex?
Is it ready for code generation?



- ■ High-Integrity Systems
  - ■ Simulink
    - ☑ ✓ Check usage of While Iterator blocks
    - ☑ ✓ Check usage of For and While Iterator subsystems
    - ☑ ⚠ Check for blocks not recommended for C/C++ production code deploymen
    - ☑ ✓ Check for inconsistent vector indexing methods
    - ☑ ✓ Check usage of variant blocks
    - ☑ ✓ Check usage c

- ☑ JMAAB Checks
- ■ By Task
  - ■ Modeling Standards for MISRA C:2012
  - ■ Modeling Standards for Secure Coding (CERT C, CWE, ISO/IEC TS 17961)
  - ■ Modeling Standards for DO-178C/DO-331
  - ■ Modeling Standards for DO-254
  - ■ Modeling Standards for IEC 61508
  - ■ Modeling Standards for IEC 62304
  - ■ Modeling Standards for ISO 26262
  - ■ Modeling Standards for ISO 25119
  - ■ Modeling Standards for EN 50128/EN 50657
  - ■ Modeling Standards for MAB
  - ■ Modeling Standards for JMAAB

Usage of disapproved block

Simulink Design Verifier Resu

Back to summary - Close

antipattern1a/Sum

Overflow **VALID**

Derived Ranges:

rt [-128..127]

Simulink Design Verifier Results

Back to summary - Close results

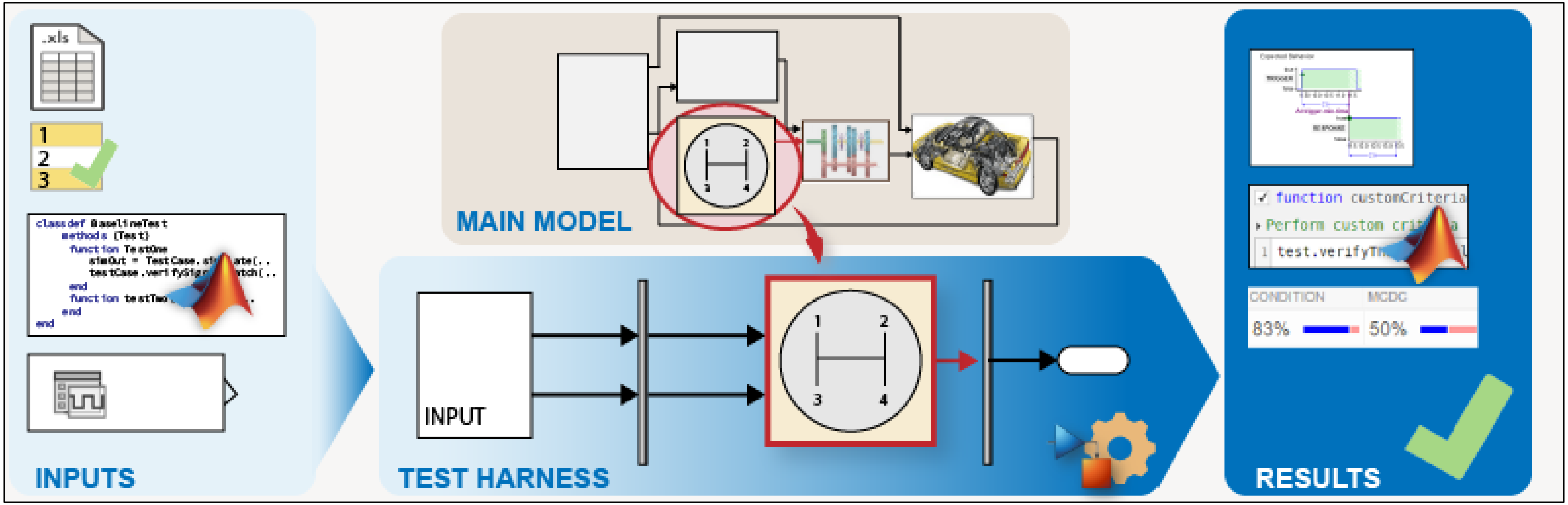
antipattern1a/Abs

Overflow **ERROR** - View test case

Derived Ranges:

Outport 1: [-128..127]

Systematic Functional **Testing**

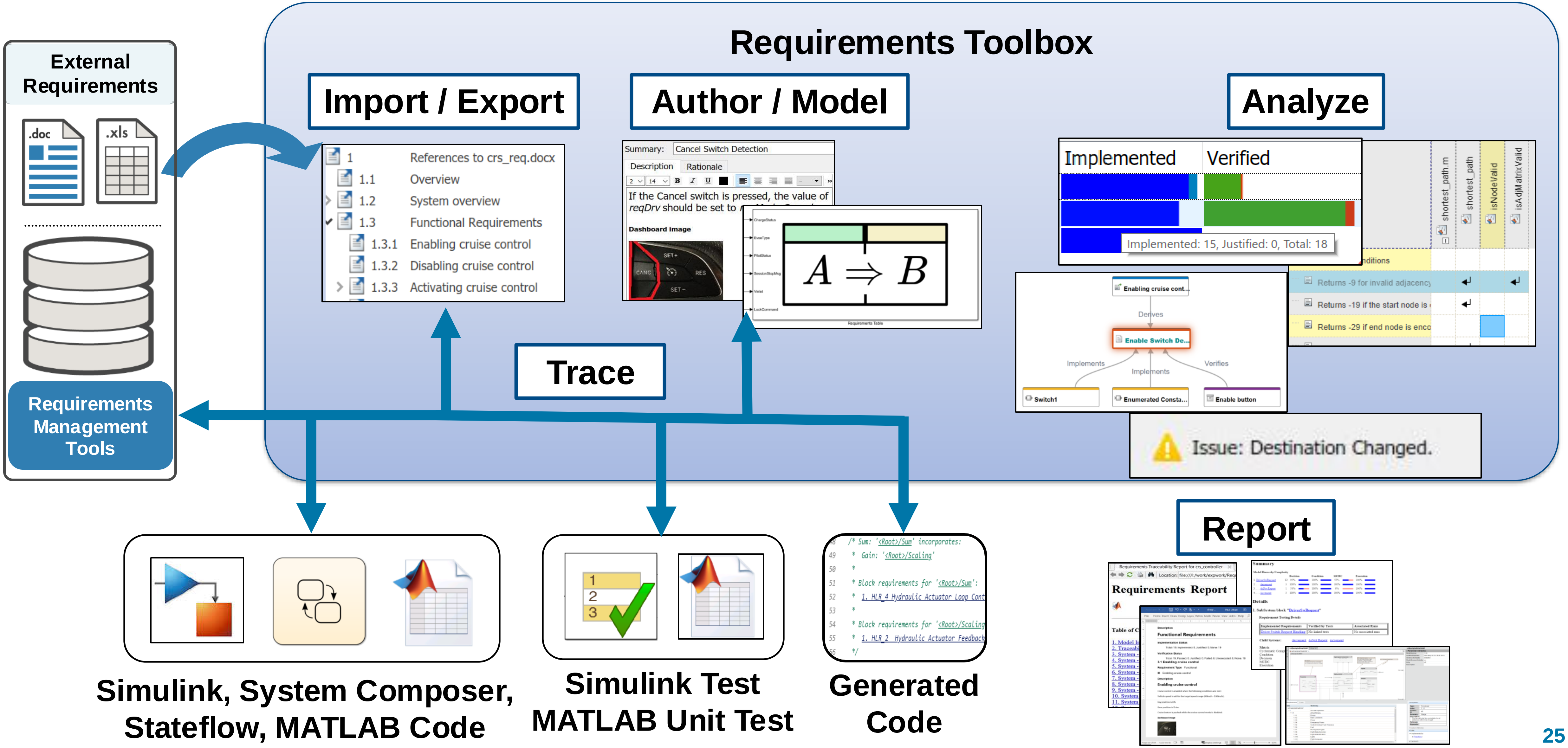


```
classdef BaselineTest
    methods (Test)
      function TestOne
          simOut = TestCase.si   ate(..
          testCase.verifySign   atch(..
      end
      function testTwo        .
    end
end
```

```
☑ function customCriteria
▸ Perform custom crit  a
1 test.verifyTh
```

Does the design meet requirements?
Is it functioning correctly?
Is it completely tested?

Author, link, and validate requirements for designs and tests



1	References to crs_req.docx
1.1	Overview
1.2	System overview
1.3	Functional Requirements
1.3.1	Enabling cruise control
1.3.2	Disabling cruise control
1.3.3	Activating cruise control

Summary: Cancel Switch Detection

Description | Rationale

If the Cancel switch is pressed, the value of *reqDrv* should be set to

Dashboard image

Implemented: 15, Justified: 0, Total: 18

nditions

Returns -9 for invalid adjacency

Returns -19 if the start node is

Returns -29 if end node is enco

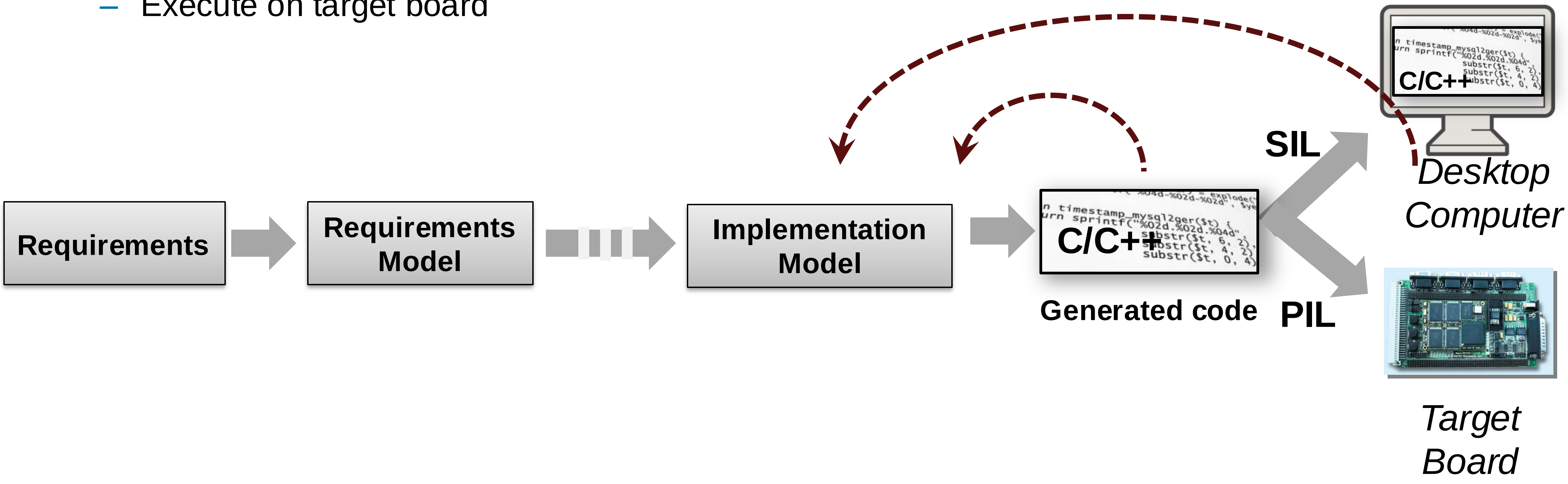
Issue: Destination Changed.

```
/* Sum: '<Root>/Sum' incorporates:
 *  Gain: '<Root>/Scaling'
 *
 * Block requirements for '<Root>/Sum':
 *  1. HLR_4 Hydraulic Actuator Loop Cont
 *
 * Block requirements for '<Root>/Scaling
 *  1. HLR_2  Hydraulic Actuator Feedback
 */
```

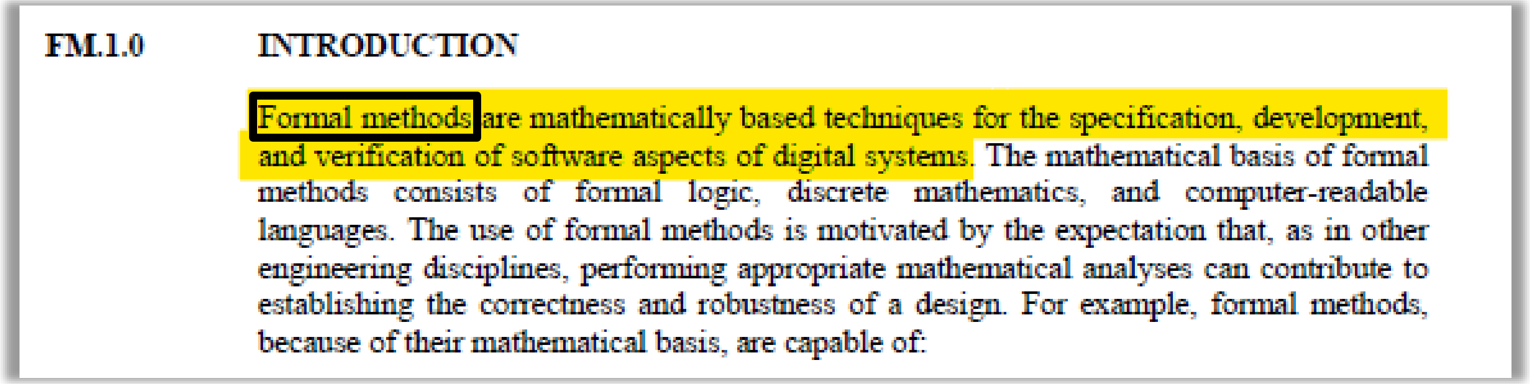
Equivalence Testing

- Software in the Loop (SIL)
  - Show functional equivalence, model to code
  - Execute on desktop / laptop computer
- Processor in the Loop (PIL)
  - Numerical equivalence, model to target code
  - Execute on target board

- Re-use tests developed for model to test code
- Collect code coverage

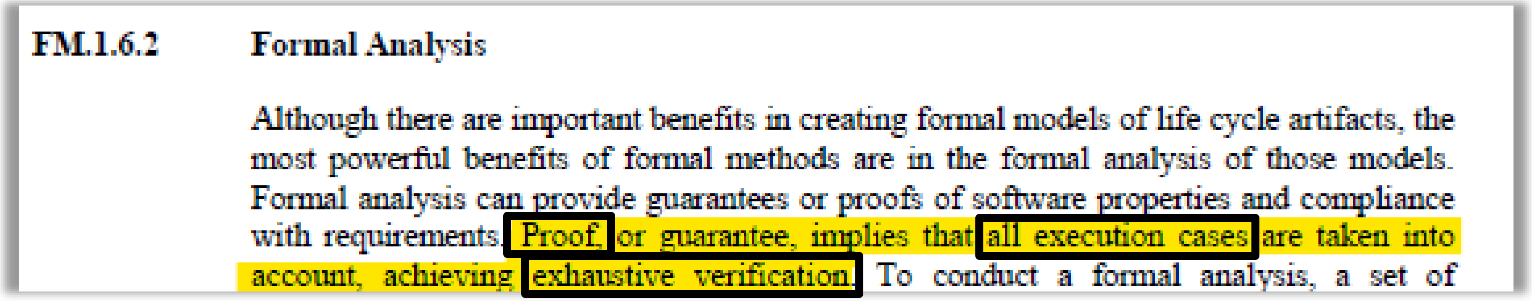


Formal Methods for Functional Safety



FM.1.0 INTRODUCTION

Formal methods are mathematically based techniques for the specification, development, and verification of software aspects of digital systems. The mathematical basis of formal methods consists of formal logic, discrete mathematics, and computer-readable languages. The use of formal methods is motivated by the expectation that, as in other engineering disciplines, performing appropriate mathematical analyses can contribute to establishing the correctness and robustness of a design. For example, formal methods, because of their mathematical basis, are capable of:



FM.1.6.2 Formal Analysis

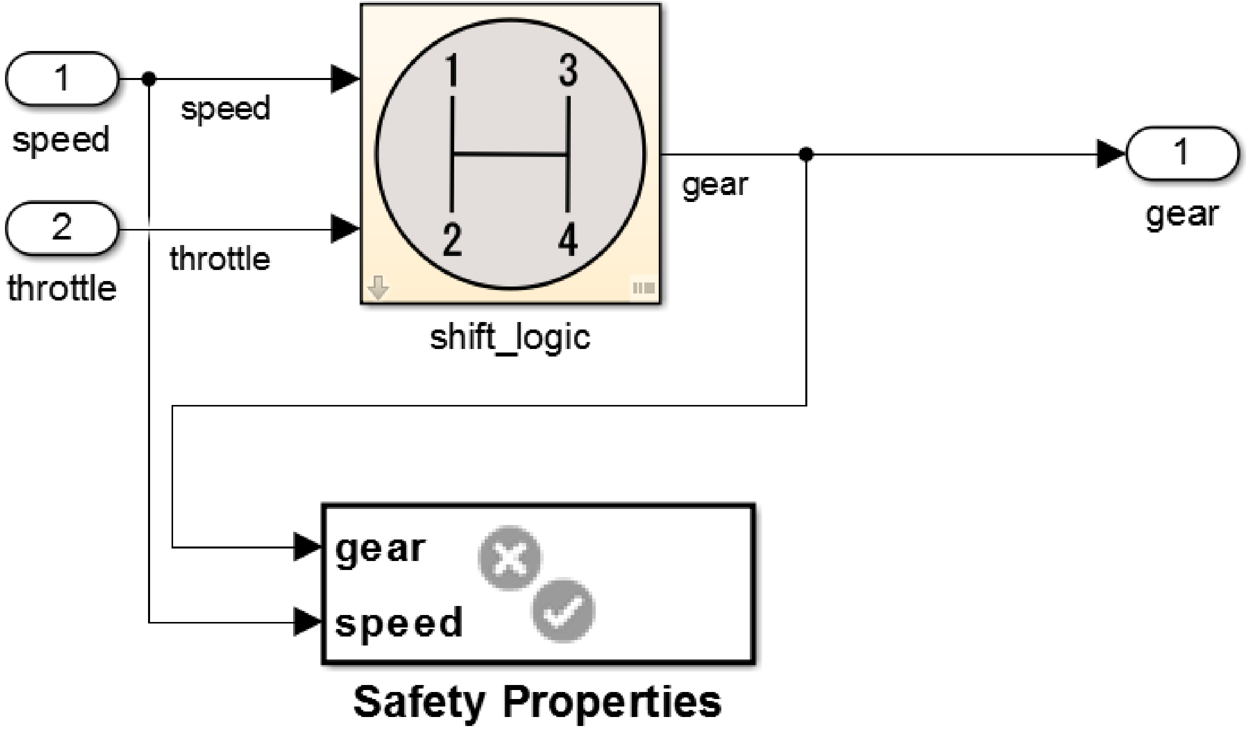
Although there are important benefits in creating formal models of life cycle artifacts, the most powerful benefits of formal methods are in the formal analysis of those models. Formal analysis can provide guarantees or proofs of software properties and compliance with requirements. Proof, or guarantee, implies that all execution cases are taken into account, achieving exhaustive verification. To conduct a formal analysis, a set of

DO-333 Formal Methods Supplement

Sound analysis means that the method never asserts a property to be true when it may not be true” : False Negative

Source: DO-333 Supplement on Formal Methods

"No amount of experimentation can ever prove me right; a single experiment can prove me wrong.“, **Albert Einstein**



- Prove design properties using formal requirement models
- Model functional and safety requirements
- Generates counter example for analysis and debugging

Prove That Design Meets Requirements

“Missed” Runtime Errors with Catastrophic Results

Ariane 5

“The world’s most expensive firework”

GNC system malfunction.

$500M (uninsured) payload

+ $7B in development costs

$7.5B loss

Overflow error

USS Yorktown

Dead in the water

Propulsion system repeatedly shut down.

Divide-by-zero error

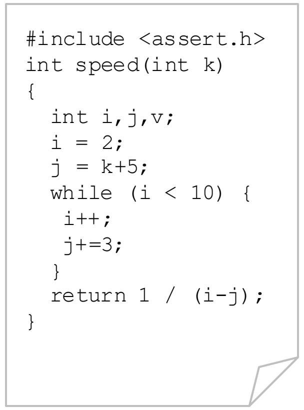
Therac 25

Fatal overdose

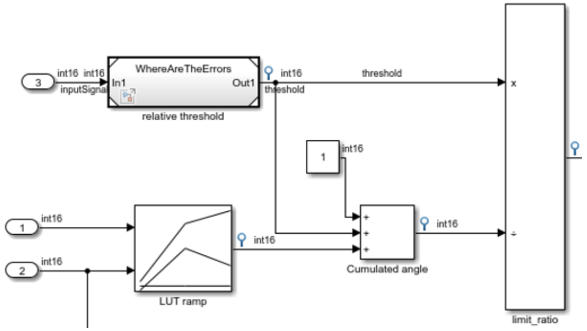
Patients severely overdosed.

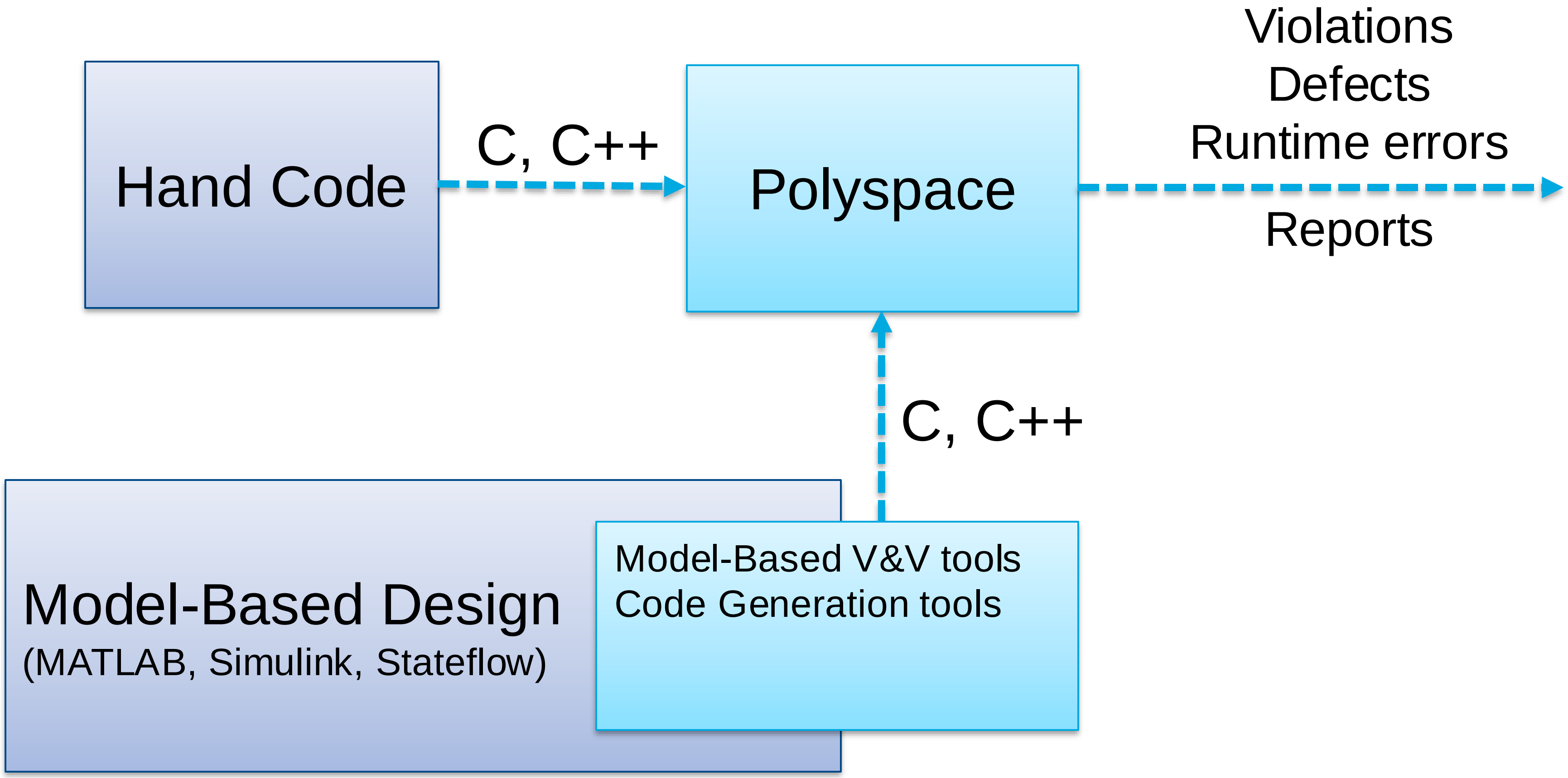
6 Killed.

Race Condition
Overflow Error



```
#include <assert.h>
int speed(int k)
{
  int i,j,v;
  i = 2;
  j = k+5;
  while (i < 10) {
   i++;
   j+=3;
  }
  return 1 / (i-j);
}
```





Hand Code

C, C++

Polyspace

Violations
Defects
Runtime errors

Reports

C, C++

Model-Based Design
(MATLAB, Simulink, Stateflow)

Model-Based V&V tools
Code Generation tools

Polyspace Tools

Bug Finder

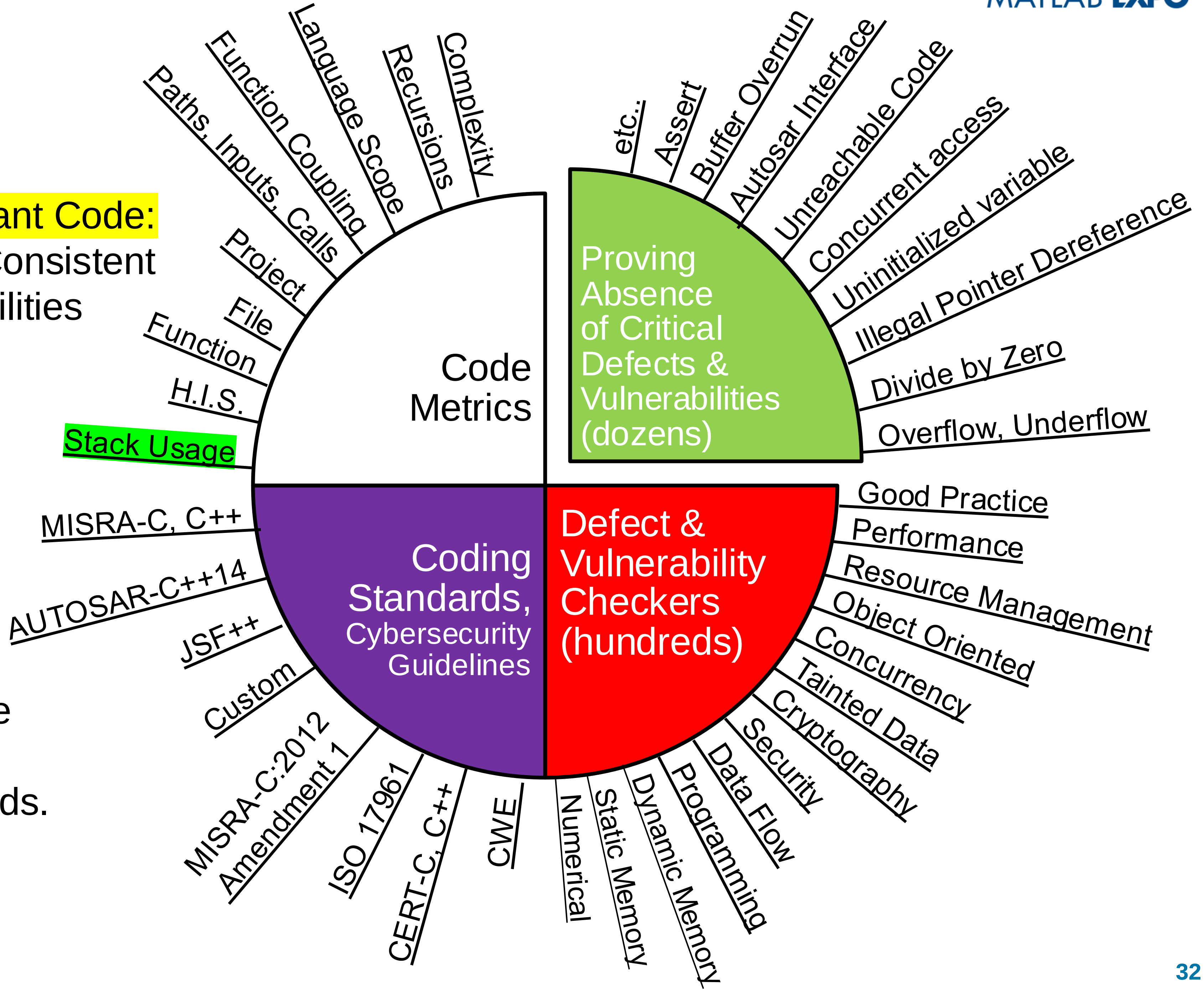
→High Quality, Secure, Compliant Code:

- Measurable, Maintainable, Consistent
- Very few defects or vulnerabilities
- Credits for functional safety, cybersecurity standards.

Code Prover

→Fully Trusted Components:

- Robust, Safe, Secure
- Proven free of critical runtime defects and vulnerabilities
- Additional credits for standards.



Code Metrics

Paths, Inputs, Calls
Function Coupling
Language Scope
Recursions
Complexity
Project
File
Function
H.I.S.
Stack Usage

Proving Absence of Critical Defects & Vulnerabilities (dozens)

etc..
Assert
Buffer Overrun
Autosar Interface
Unreachable Code
Concurrent access
Uninitialized variable
Illegal Pointer Dereference
Divide by Zero
Overflow, Underflow

Coding Standards, Cybersecurity Guidelines

MISRA-C, C++
AUTOSAR-C++14
JSF++
Custom
MISRA-C:2012 Amendment 1
ISO 17961
CERT-C, C++
CWE

Defect & Vulnerability Checkers (hundreds)

Good Practice
Performance
Resource Management
Object Oriented
Concurrency
Tainted Data
Cryptography
Security
Data Flow
Programming
Dynamic Memory
Static Memory
Numerical

Volvo Cars Software Factory Increases Pace and Quality of Development with Polyspace

Challenge

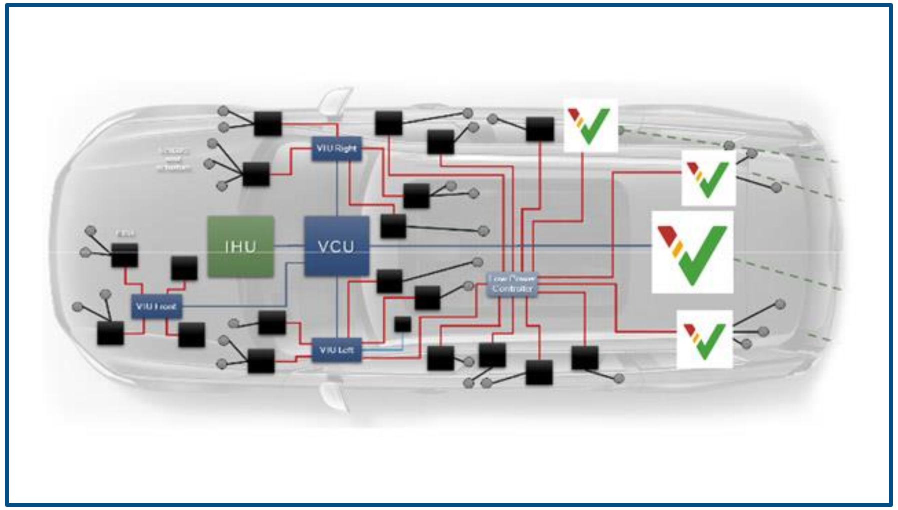
Develop reliable, standards-compliant software for the next generation of cars

Solution

Run static code analysis with Polyspace throughout the software development lifecycle

Results

- Critical run-time errors detected before field testing
- Improved productivity with better code reuse
- ASPICE, ISO 26262, and ISO/SAE 21434 certification requirements met

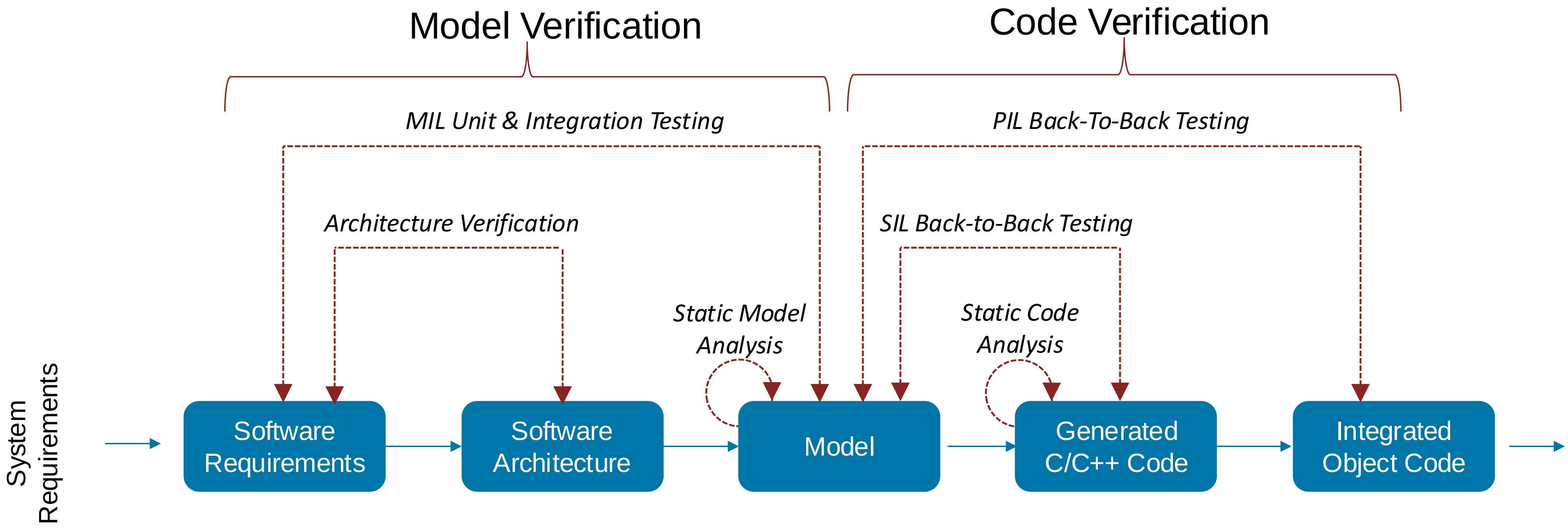


Volvo Cars uses Polyspace for static code checking throughout the development cycle.

> *“With Polyspace, we can ensure software security and quality by identifying and fixing critical run-time errors before every code merge.”*
>
> *- Johannes Foufas, Volvo Cars*

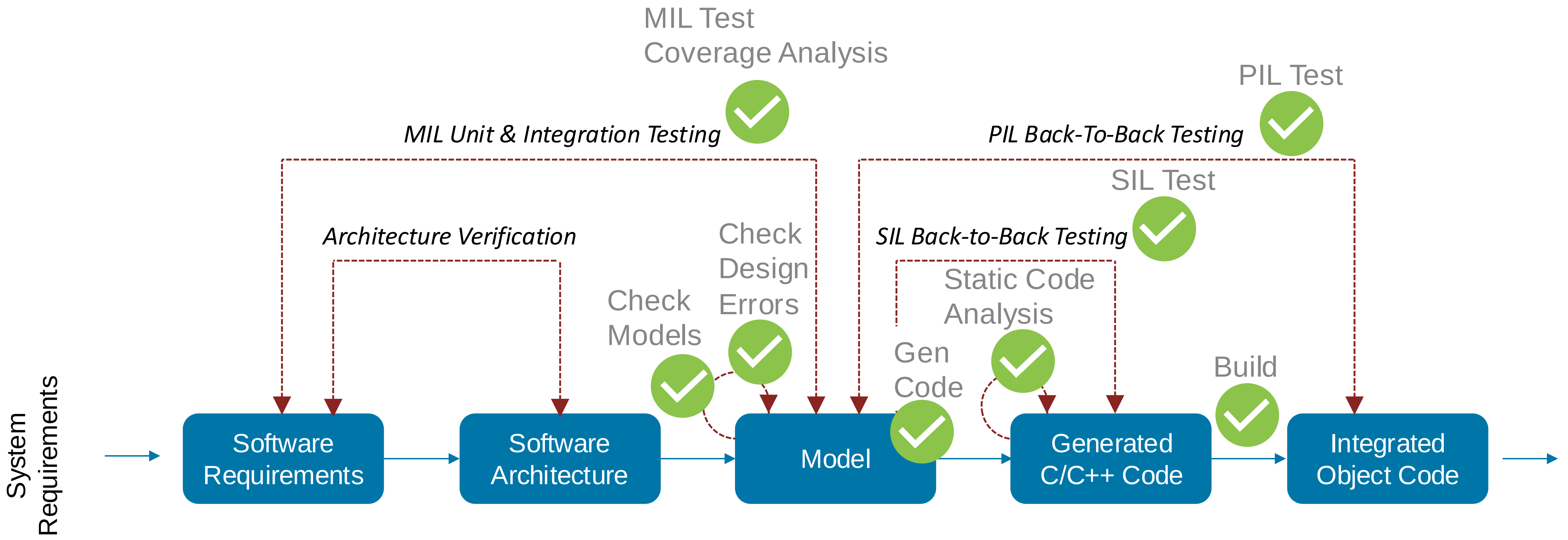
Link to user story

What have we seen !

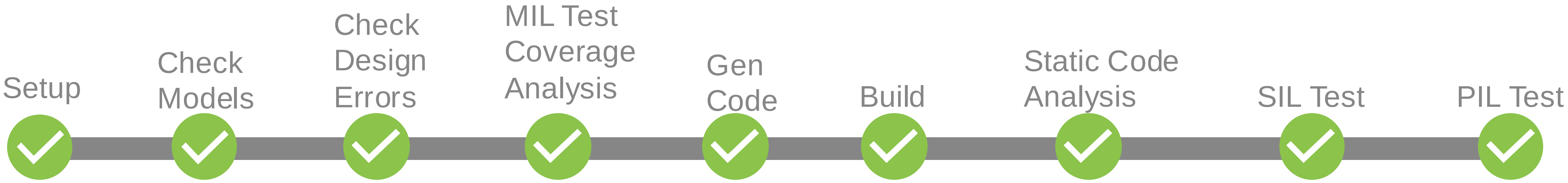


Software Factory – Handling the complexities ✓
Safety and reliability ✓
Speed, Agility and Scalability ❓

Model-Based Design Reference Workflow



Model-Based Design Reference Workflow



Setup

Check Models

Check Design Errors

MIL Test Coverage Analysis

Gen Code

Build

Static Code Analysis

SIL Test

PIL Test

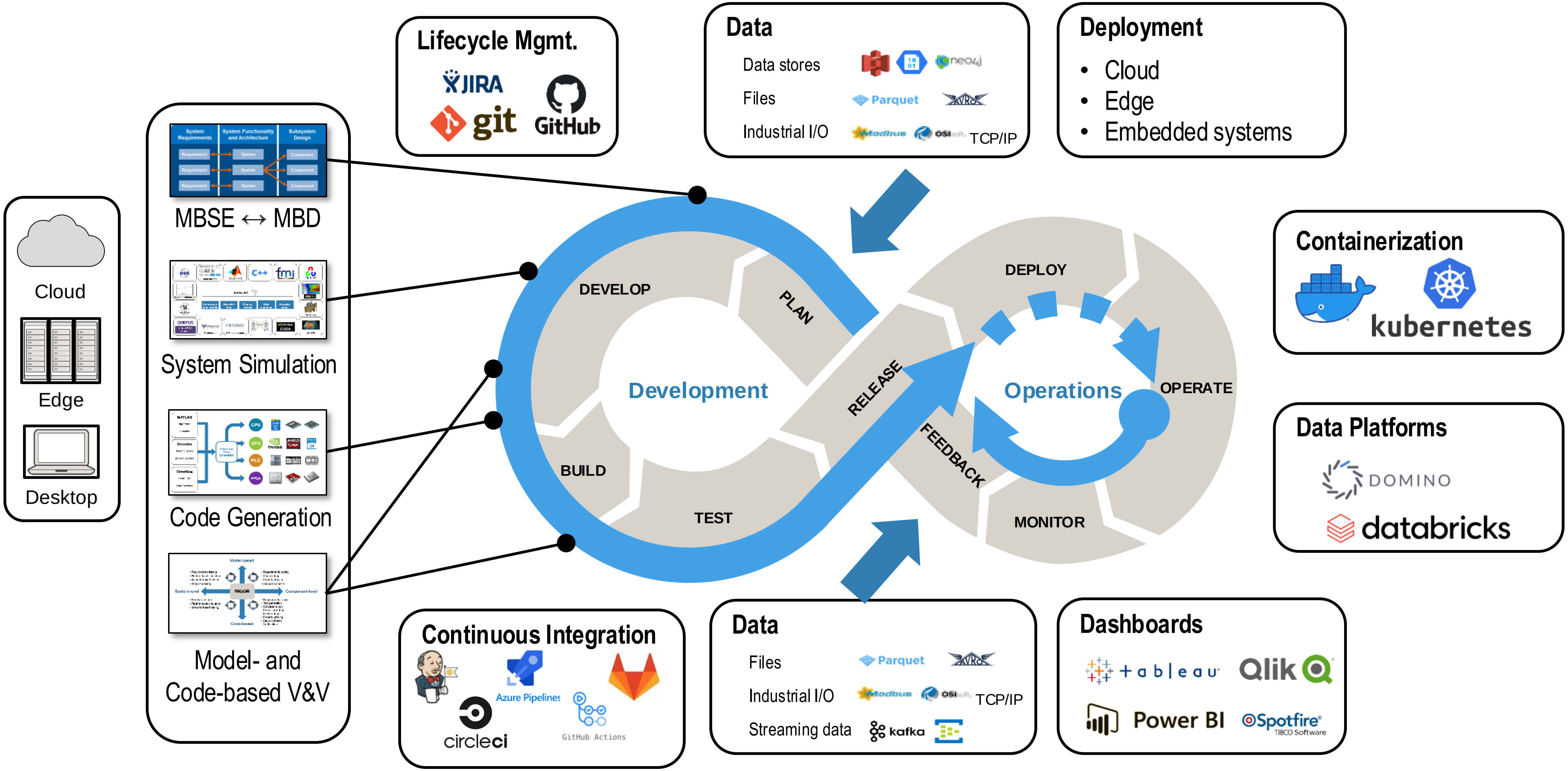
- Define Process and Automate
  - Identify Tasks
  - Define Sequence
  - Define Outputs
  - Script the Tools

build.m

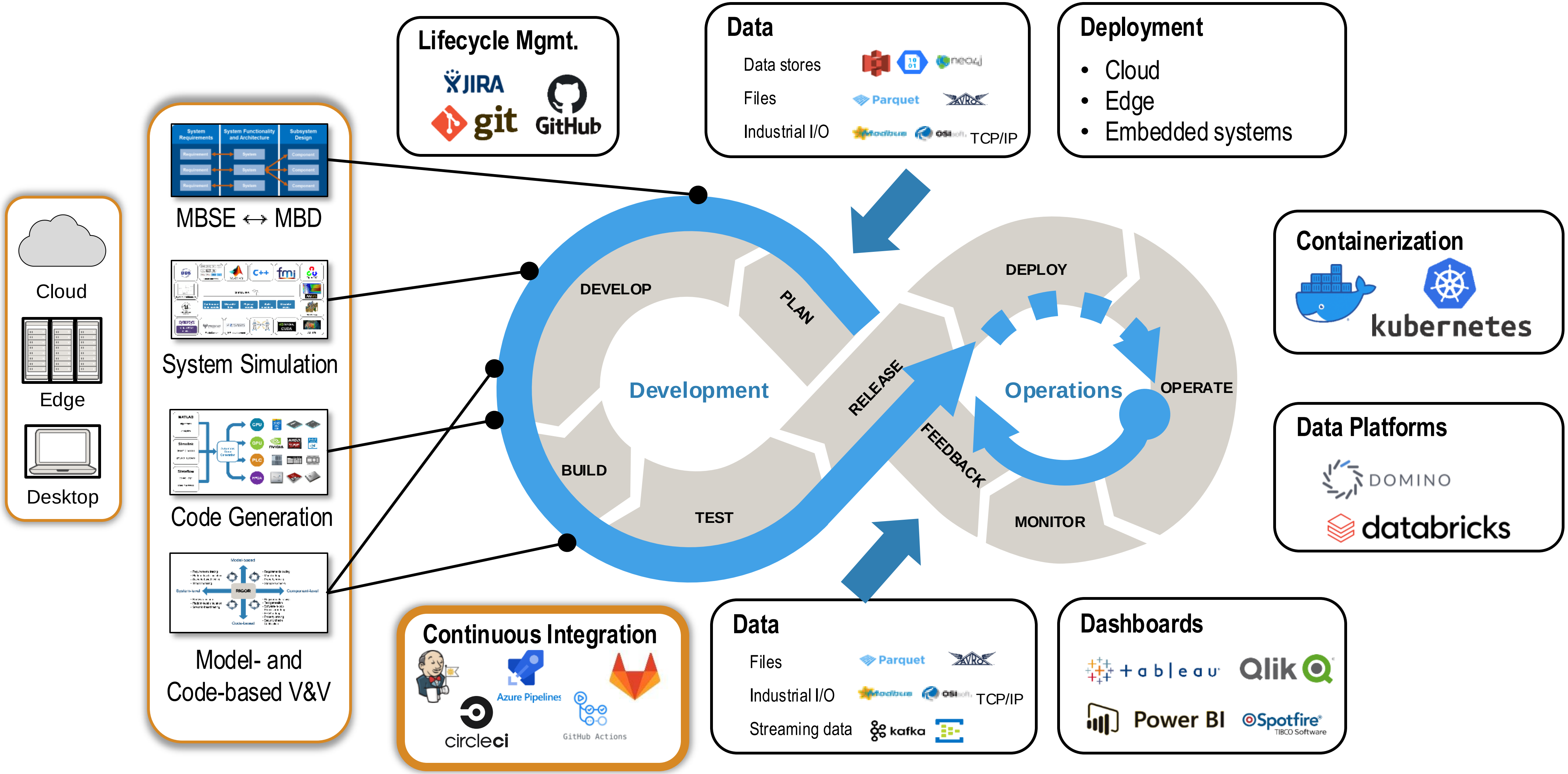
genCode.m

⋮

DevOps building blocks for Embedded Production SW



Continuous Integration for embedded production SW



Lifecycle Mgmt.

JIRA

git

GitHub

Data

Data stores

Files Parquet AVRO

Industrial I/O Modbus OSIsoft TCP/IP

Deployment

- Cloud
- Edge
- Embedded systems

Containerization

kubernetes

Data Platforms

DOMINO

databricks

Cloud

Edge

Desktop

MBSE ↔ MBD

System Simulation

Code Generation

Model- and Code-based V&V

Continuous Integration

Azure Pipelines

circleci

GitHub Actions

Data

Files Parquet AVRO

Industrial I/O Modbus OSIsoft TCP/IP

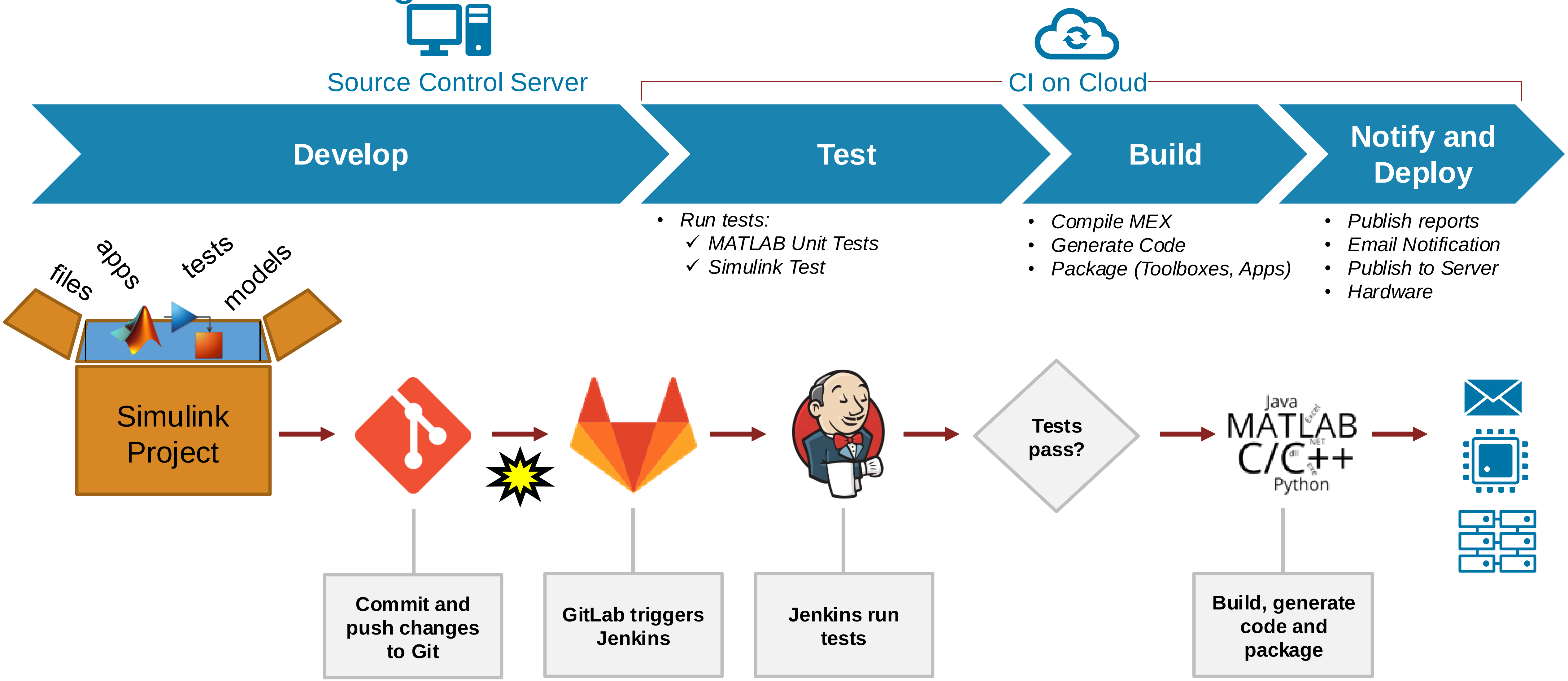
Streaming data kafka

Dashboards

tableau Qlik

Power BI Spotfire TIBCO Software

Continuous Integration Workflow with MATLAB and Simulink



Source Control Server

CI on Cloud

Develop

Test

Build

Notify and Deploy

- *Run tests:*
  - ✓ *MATLAB Unit Tests*
  - ✓ *Simulink Test*

- *Compile MEX*
- *Generate Code*
- *Package (Toolboxes, Apps)*

- *Publish reports*
- *Email Notification*
- *Publish to Server*
- *Hardware*

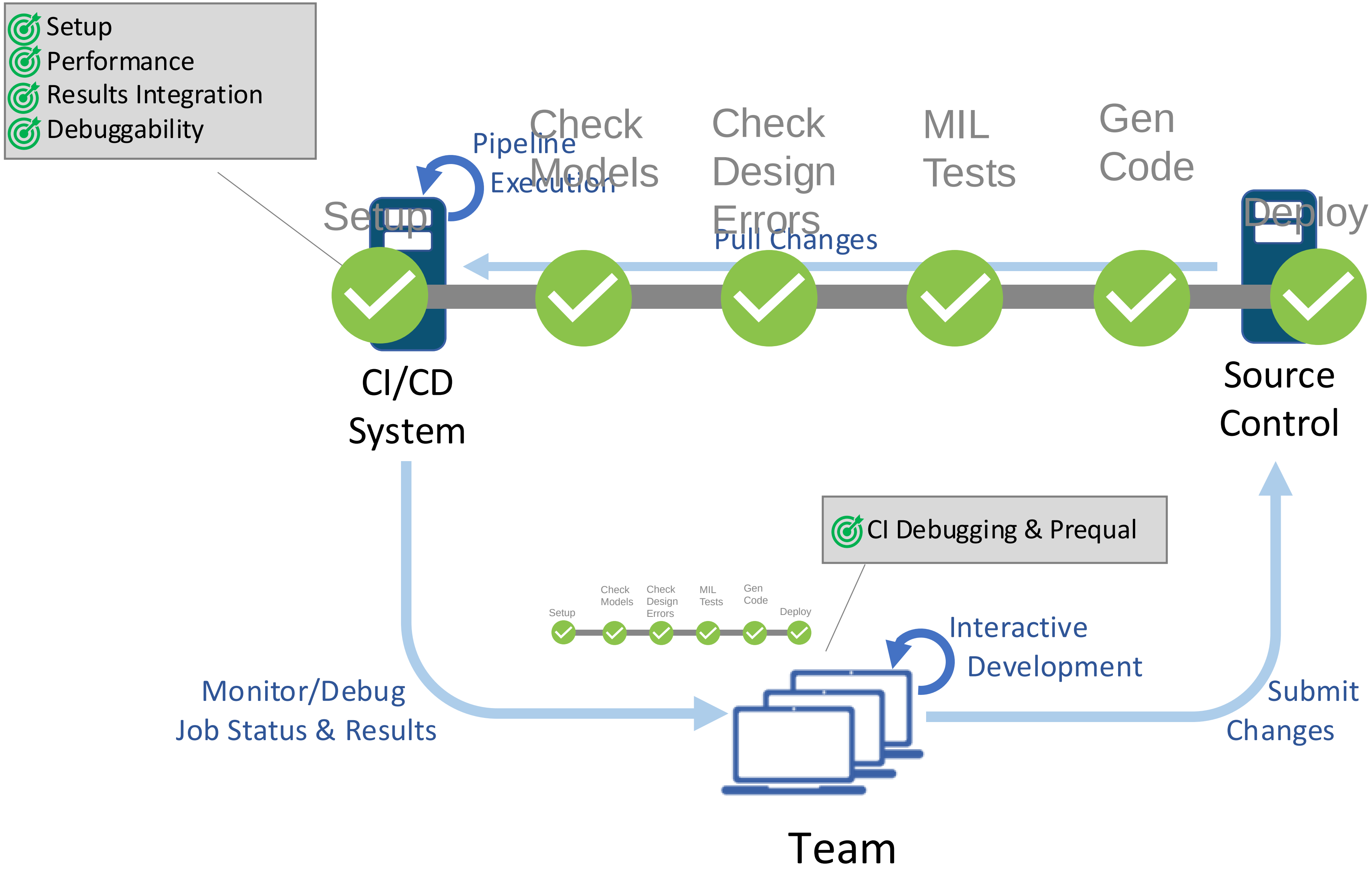
Commit and push changes to Git

GitLab triggers Jenkins

Jenkins run tests

Build, generate code and package

Accelerating Adoption and Optimizing CI/CD for MBD



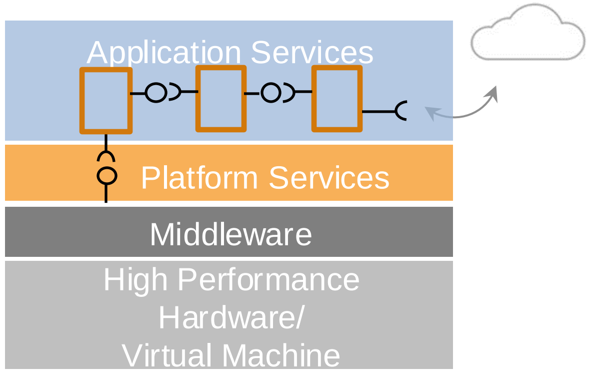
- ✓ Setup
- ✓ Performance
- ✓ Results Integration
- ✓ Debuggability

- ✓ CI Debugging & Prequal

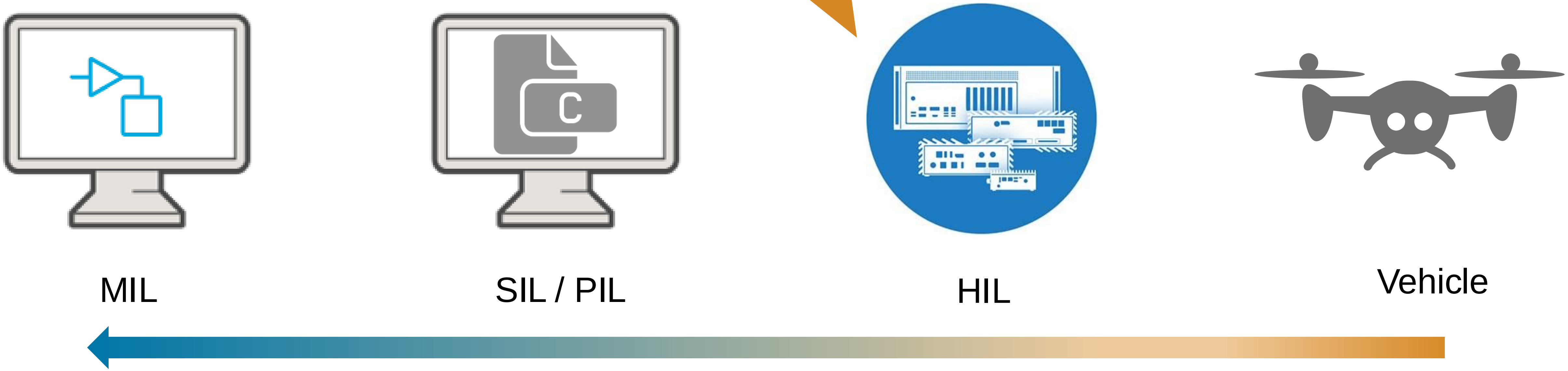
Development in Action

Virtual HW deployment and testing

Can we test and refine more virtually?
Reduce the need Controller or
Peripheral hardware?

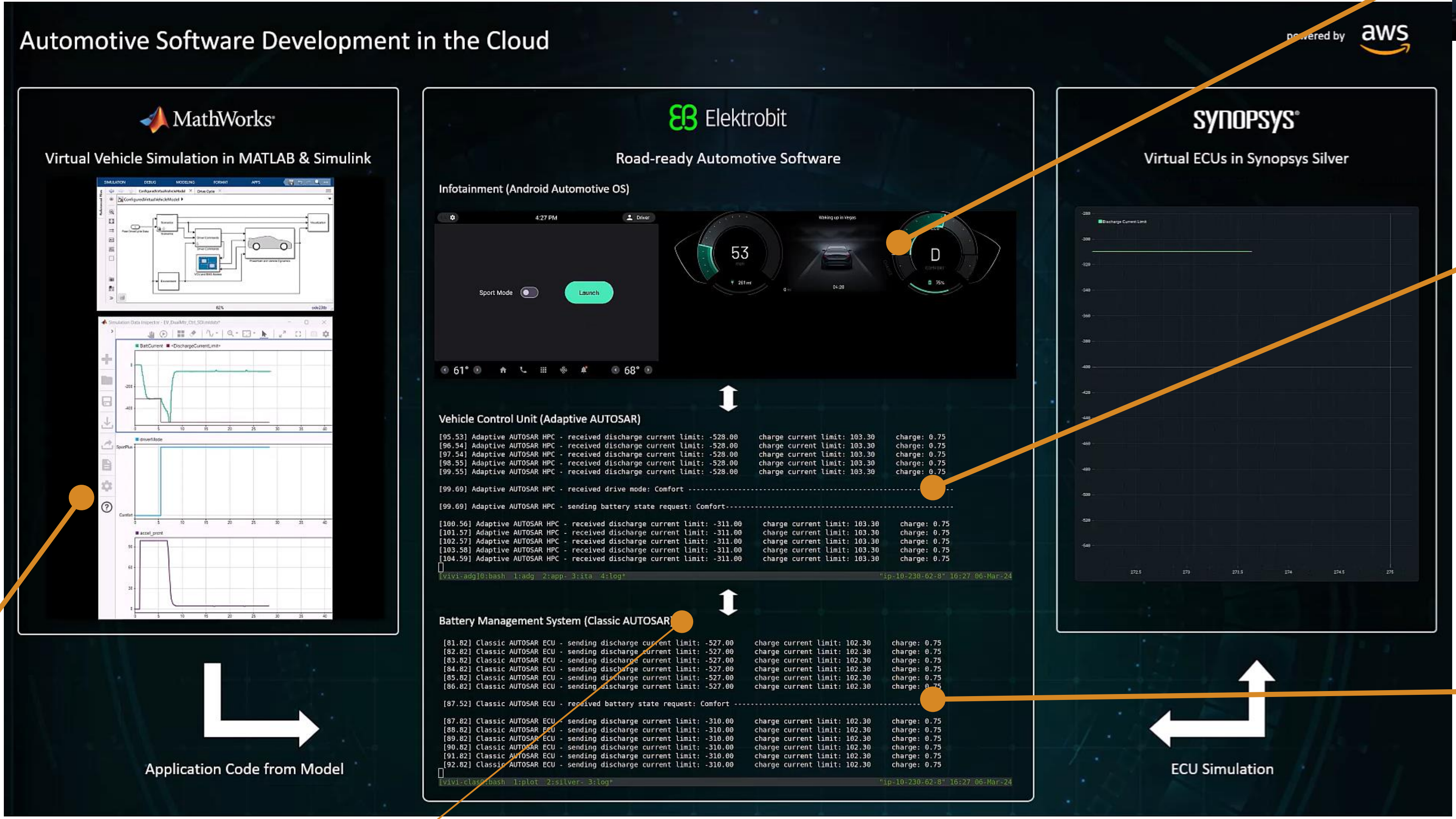


Higher HW abstraction:
Service-oriented architectures



From Analysis Models → Production Software Testing

Test level 3 virtual ECUs on the cloud

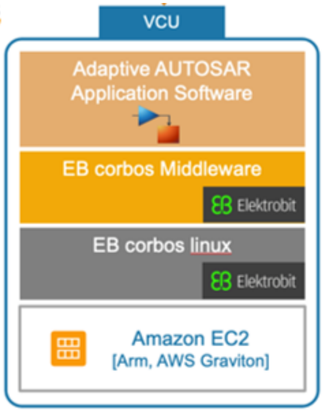

Automotive Software Development in the Cloud
powered by aws
MathWorks
Virtual Vehicle Simulation in MATLAB & Simulink
Elektrobit
Road-ready Automotive Software
Infotainment (Android Automotive OS)
4:27 PM
Sport Mode
Launch
53
D
61°
68°
SYNOPSYS
Virtual ECUs in Synopsys Silver
Application Code from Model
ECU Simulation

Vehicle Control Unit (Adaptive AUTOSAR)

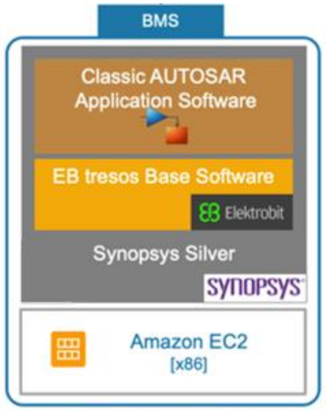
```
[95.53] Adaptive AUTOSAR HPC - received discharge current limit: -528.00     charge current limit: 103.30     charge: 0.75
[96.54] Adaptive AUTOSAR HPC - received discharge current limit: -528.00     charge current limit: 103.30     charge: 0.75
[97.54] Adaptive AUTOSAR HPC - received discharge current limit: -528.00     charge current limit: 103.30     charge: 0.75
[98.55] Adaptive AUTOSAR HPC - received discharge current limit: -528.00     charge current limit: 103.30     charge: 0.75
[99.55] Adaptive AUTOSAR HPC - received discharge current limit: -528.00     charge current limit: 103.30     charge: 0.75

[99.69] Adaptive AUTOSAR HPC - received drive mode: Comfort -------------------------------------------------------------

[99.69] Adaptive AUTOSAR HPC - sending battery state request: Comfort-------------------------------------------------------

[100.56] Adaptive AUTOSAR HPC - received discharge current limit: -311.00     charge current limit: 103.30     charge: 0.75
[101.57] Adaptive AUTOSAR HPC - received discharge current limit: -311.00     charge current limit: 103.30     charge: 0.75
[102.57] Adaptive AUTOSAR HPC - received discharge current limit: -311.00     charge current limit: 103.30     charge: 0.75
[103.58] Adaptive AUTOSAR HPC - received discharge current limit: -311.00     charge current limit: 103.30     charge: 0.75
[104.59] Adaptive AUTOSAR HPC - received discharge current limit: -311.00     charge current limit: 103.30     charge: 0.75

[vivi-adg]0:bash  1:adg  2:app- 3:ita  4:log*                                    "ip-10-230-62-8" 16:27 06-Mar-24
```

Battery Management System (Classic AUTOSAR)

```
 [81.82] Classic AUTOSAR ECU - sending discharge current limit: -527.00     charge current limit: 102.30     charge: 0.75
 [82.82] Classic AUTOSAR ECU - sending discharge current limit: -527.00     charge current limit: 102.30     charge: 0.75
 [83.82] Classic AUTOSAR ECU - sending discharge current limit: -527.00     charge current limit: 102.30     charge: 0.75
 [84.82] Classic AUTOSAR ECU - sending discharge current limit: -527.00     charge current limit: 102.30     charge: 0.75
 [85.82] Classic AUTOSAR ECU - sending discharge current limit: -527.00     charge current limit: 102.30     charge: 0.75
 [86.82] Classic AUTOSAR ECU - sending discharge current limit: -527.00     charge current limit: 102.30     charge: 0.75

 [87.52] Classic AUTOSAR ECU - received battery state request: Comfort ------------------------------------------------

 [87.82] Classic AUTOSAR ECU - sending discharge current limit: -310.00     charge current limit: 102.30     charge: 0.75
 [88.82] Classic AUTOSAR ECU - sending discharge current limit: -310.00     charge current limit: 102.30     charge: 0.75
 [89.82] Classic AUTOSAR ECU - sending discharge current limit: -310.00     charge current limit: 102.30     charge: 0.75
 [90.82] Classic AUTOSAR ECU - sending discharge current limit: -310.00     charge current limit: 102.30     charge: 0.75
 [91.82] Classic AUTOSAR ECU - sending discharge current limit: -310.00     charge current limit: 102.30     charge: 0.75
 [92.82] Classic AUTOSAR ECU - sending discharge current limit: -310.00     charge current limit: 102.30     charge: 0.75

[vivi-clas0:bash  1:plot  2:silver- 3:log*                                       "ip-10-230-62-8" 16:27 06-Mar-24
```



Virtual ECU running on a POSIX containerized environment on EC2

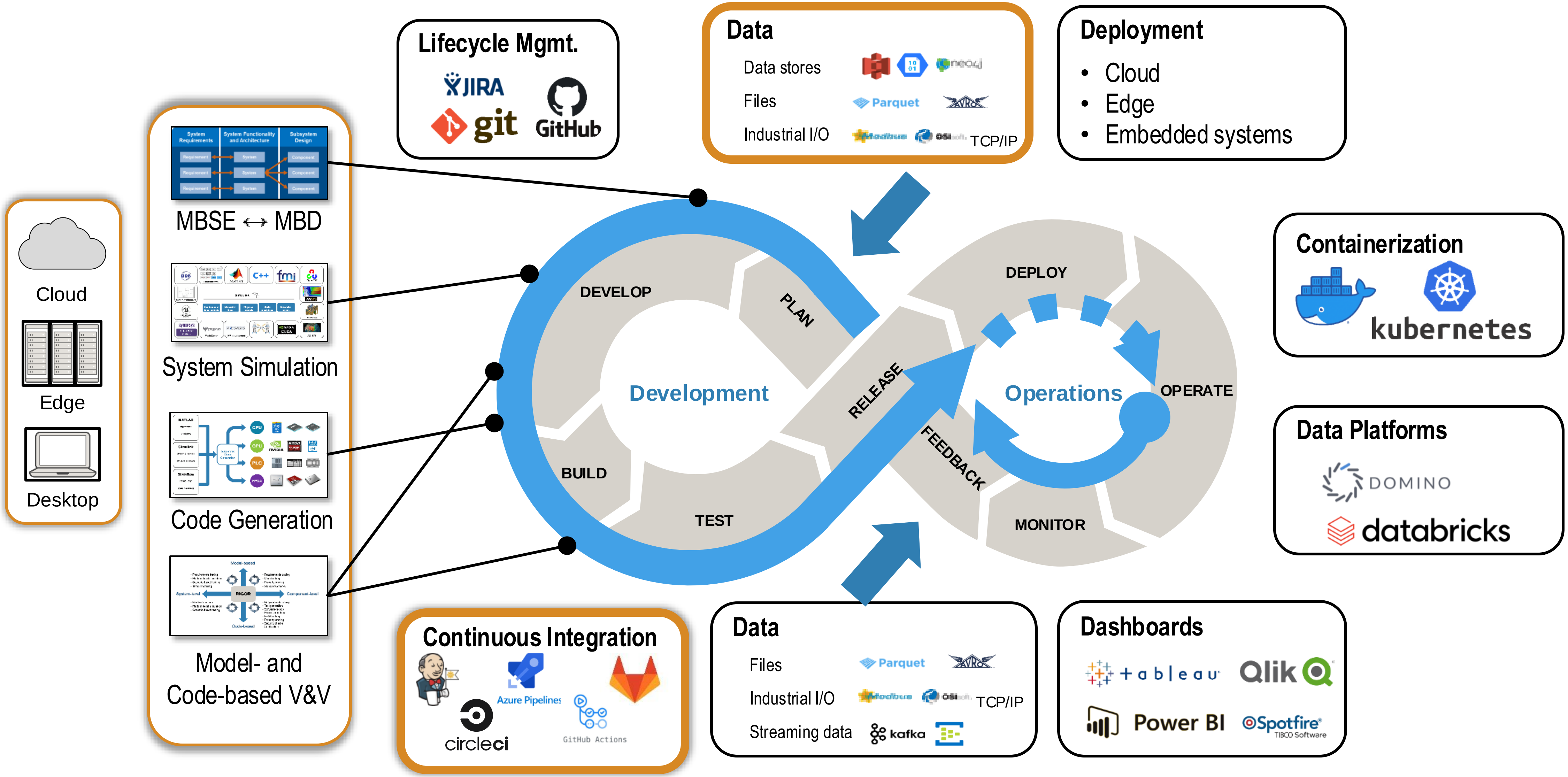


Virtual ECU running on a POSIX containerized environment on EC2

Test Vectors & Vehicle Behavior exported from Simulink (Injected into vECUs via SOME/IP)

Inter ECU Communication (via SOME/IP)

DevOps building blocks for Embedded Production SW



Lifecycle Mgmt.

Data

Data stores

Files

Industrial I/O

TCP/IP

Deployment

- Cloud
- Edge
- Embedded systems

Cloud

Edge

Desktop

MBSE ↔ MBD

System Simulation

Code Generation

Model- and Code-based V&V

Containerization

Data Platforms

Continuous Integration

Data

Files

Industrial I/O

TCP/IP

Streaming data

Dashboards

Scaling up with `parsim` on the Cloud

Different cloud computing resources for different jobs

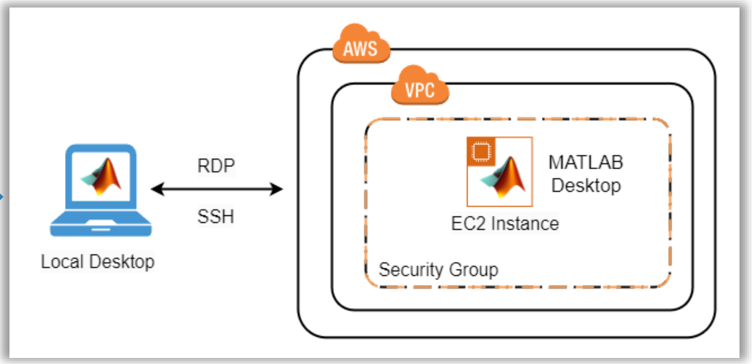
MathWorks Reference Architectures

Launch Stack

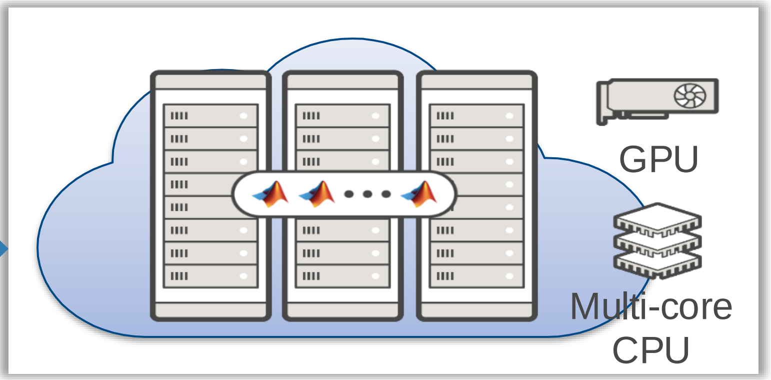
aws

```
simOut = parsim(in)
```

Parallel Computing Toolbox
MATLAB



MATLAB on AWS EC2



MATLAB Parallel Server

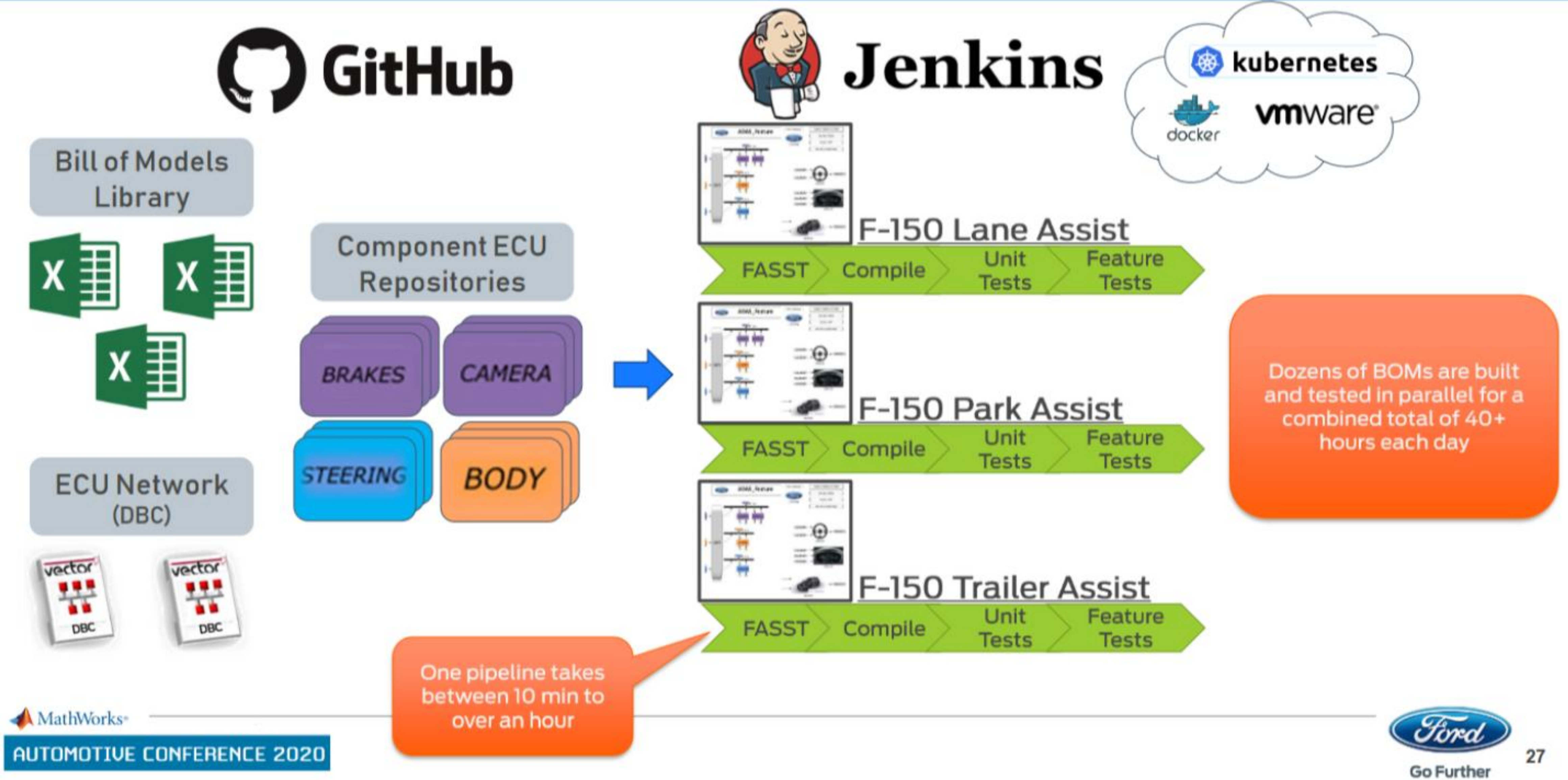
Running 1352 Simulations
~ **18 hours** in series
~ **5.2 hours** on Quadcore Laptop
~ **59 mins** on an m5.12xlarge EC2 instance, 24 core

Worker Machine = m5.12xlarge (24 cores)

Running 1352 Simulations
~ **22.7 mins** on 5 Worker machines, 120 cores
~**17 mins** on 10 Worker machines, 240 cores

Learn more: MATLAB on AWS, MathWorks Reference Architecture, MathWorks CloudCenter

CONTINOUS INTEGRATION: JENKINS TO AUTOMATE VEHICLE BUILDS



Dozens of BOMs are built and tested in parallel for a combined total of 40+ hours each day

One pipeline takes between 10 min to over an hour

MathWorks

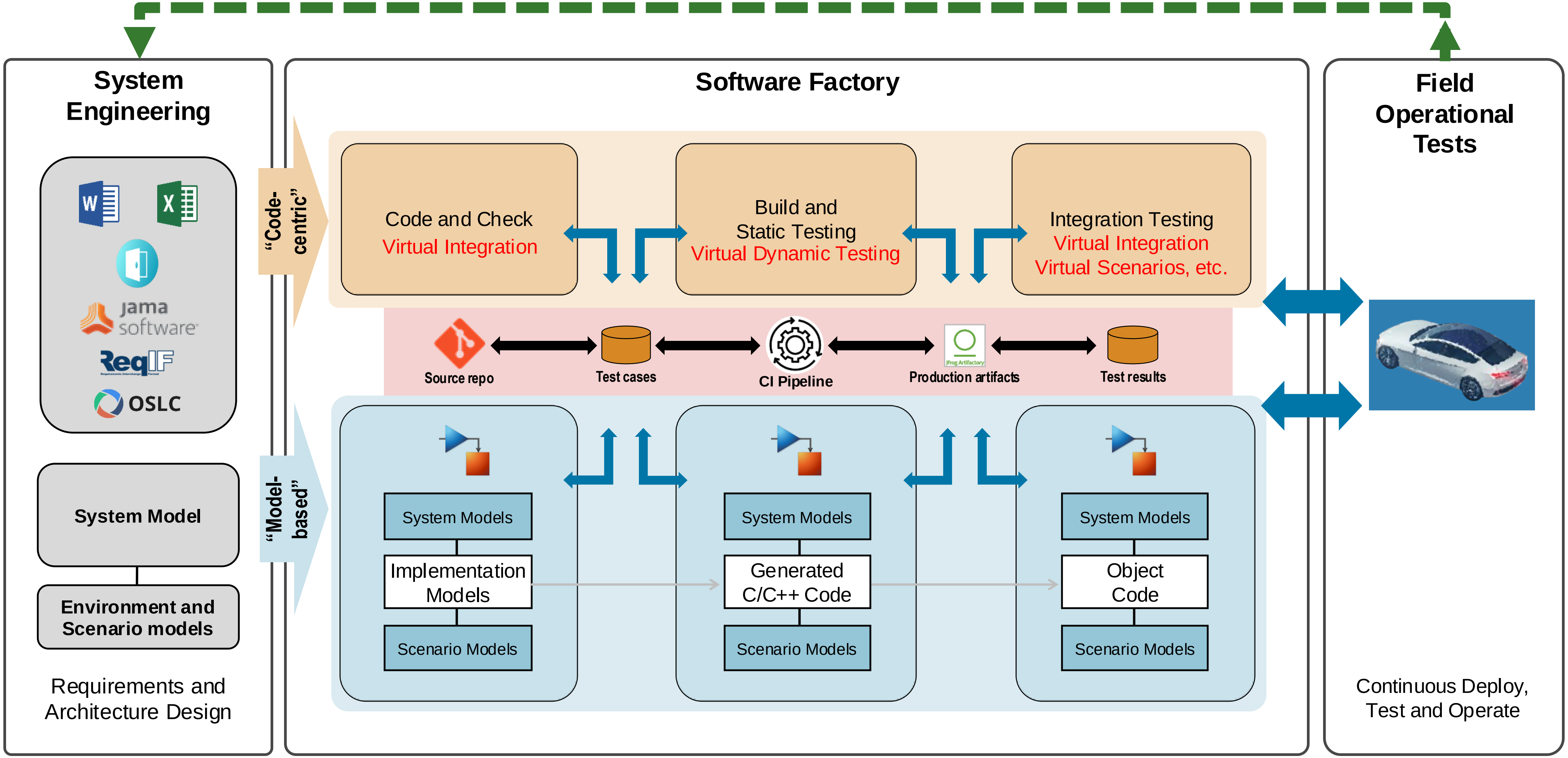
AUTOMOTIVE CONFERENCE 2020

Go Further

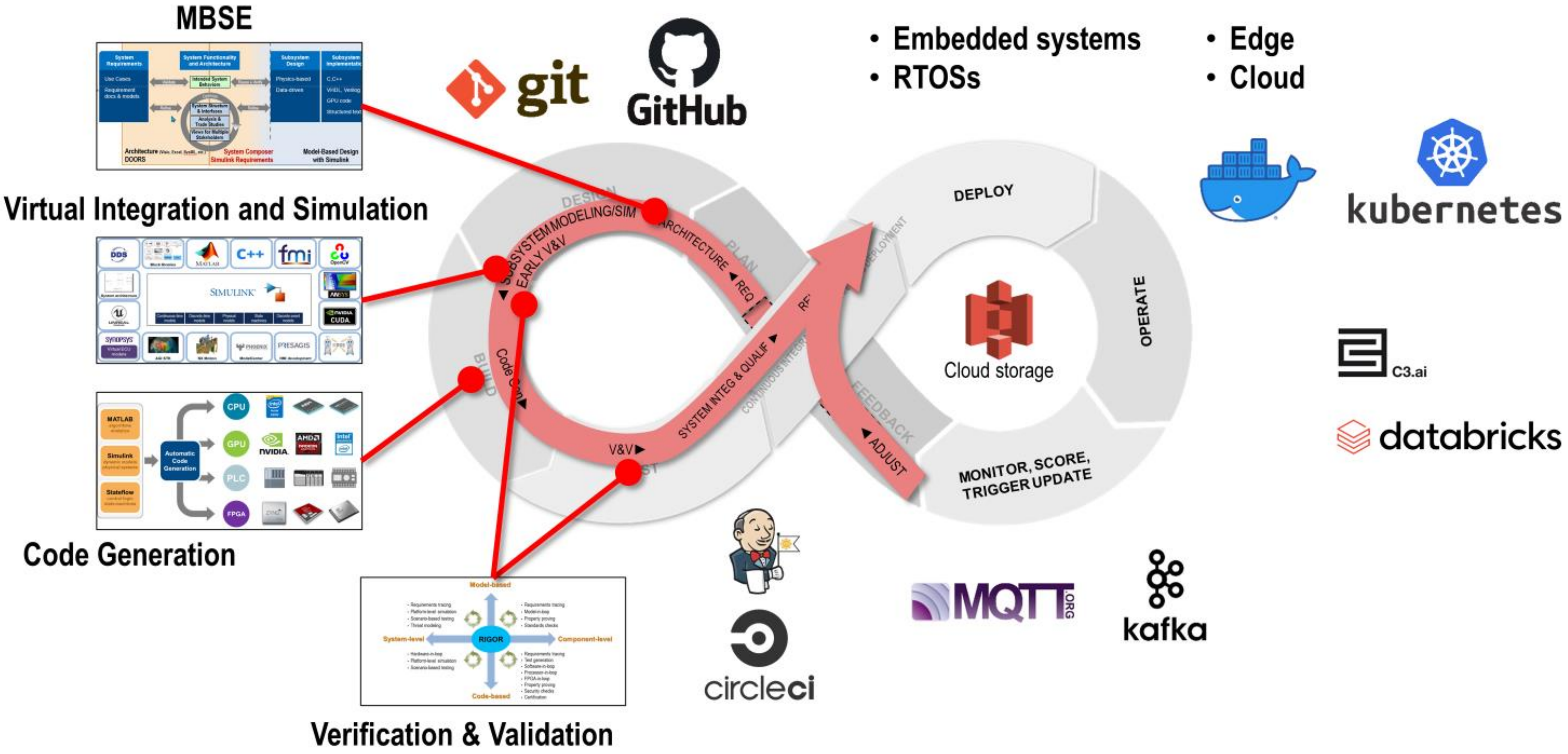
27

Software Factory – Handling the Complexities ✓
Safety and Reliability ✓
Speed, Agility and Scalability ✓

Aligning and Automating MBD and Code-Centric Approaches



Software Factory From a DevOps View



MBSE

Virtual Integration and Simulation

Code Generation

Verification & Validation

- Embedded systems
- RTOSs

- Edge
- Cloud

MathWorks
@MathWorks

Share the EXPO experience
#MATLABEXPO

Q&A

MATLAB EXPO

INDIA

Thank You!!

MathWorks