# Digital Transformation with Model-Based System Engineering, Design and Early Verification

**Mauro Fusco, Application Engineer, MathWorks**

8th September 2022

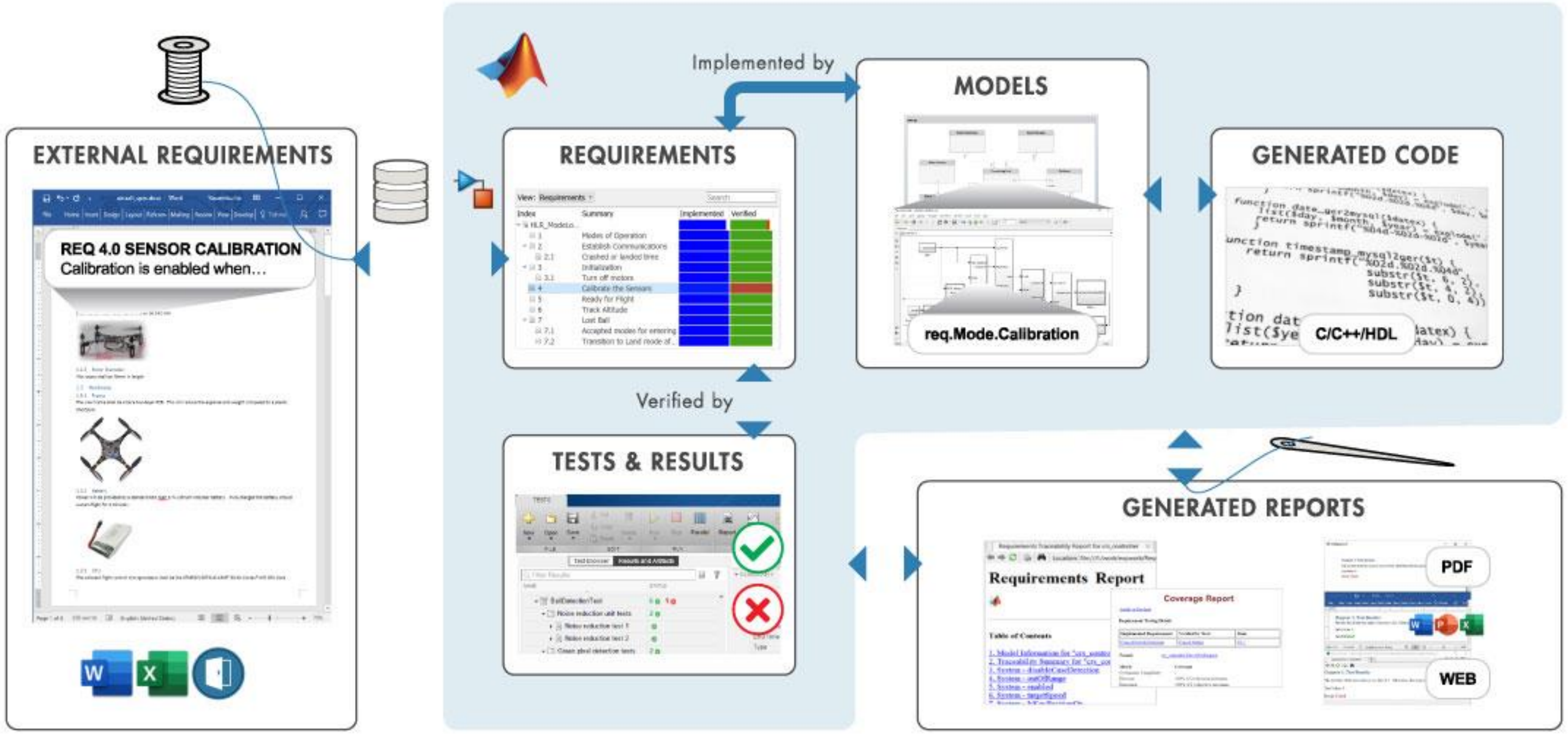
# In summary, digital engineering practices can make standards compliance easier and reduce costs

- **Seek single source of truth using digital thread**

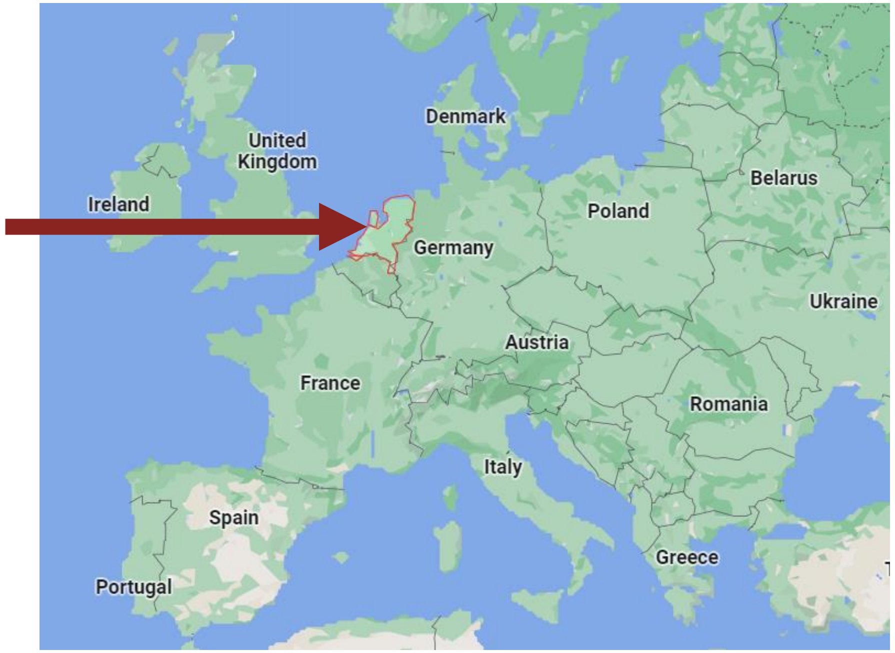
- **Automate tasks that machines are good at**

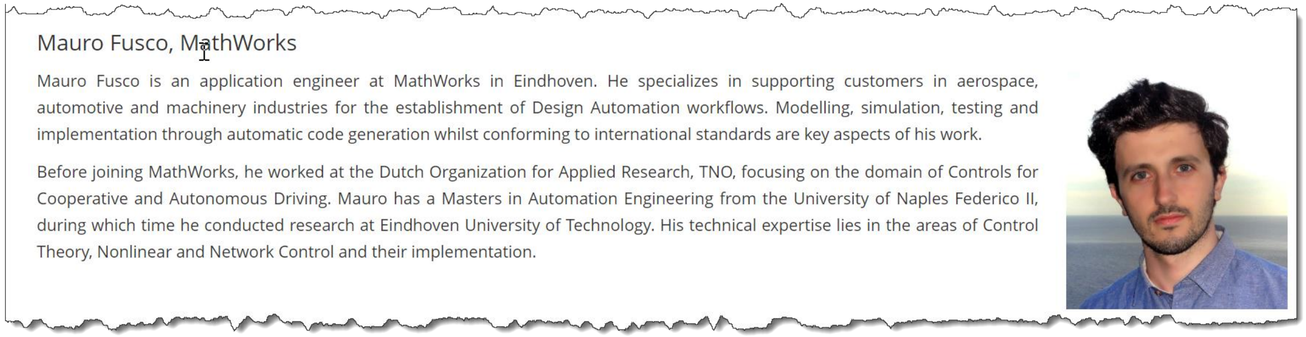
- **Improve efficiency *without* skipping verification steps**

DIGITAL THREAD: Traceability Between Requirements, Architecture, and Design



# Who is talking ??

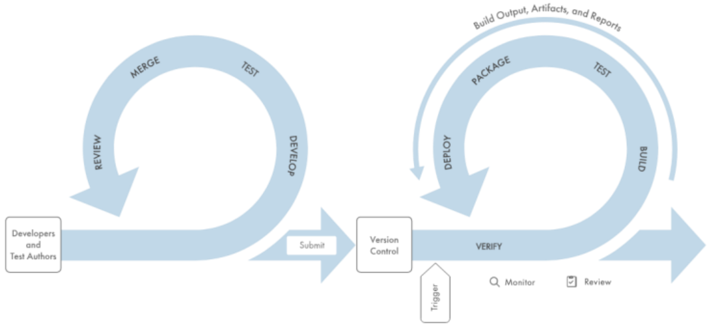




Mauro Fusco, MathWorks

Mauro Fusco is an application engineer at MathWorks in Eindhoven. He specializes in supporting customers in aerospace, automotive and machinery industries for the establishment of Design Automation workflows. Modelling, simulation, testing and implementation through automatic code generation whilst conforming to international standards are key aspects of his work.

Before joining MathWorks, he worked at the Dutch Organization for Applied Research, TNO, focusing on the domain of Controls for Cooperative and Autonomous Driving. Mauro has a Masters in Automation Engineering from the University of Naples Federico II, during which time he conducted research at Eindhoven University of Technology. His technical expertise lies in the areas of Control Theory, Nonlinear and Network Control and their implementation.



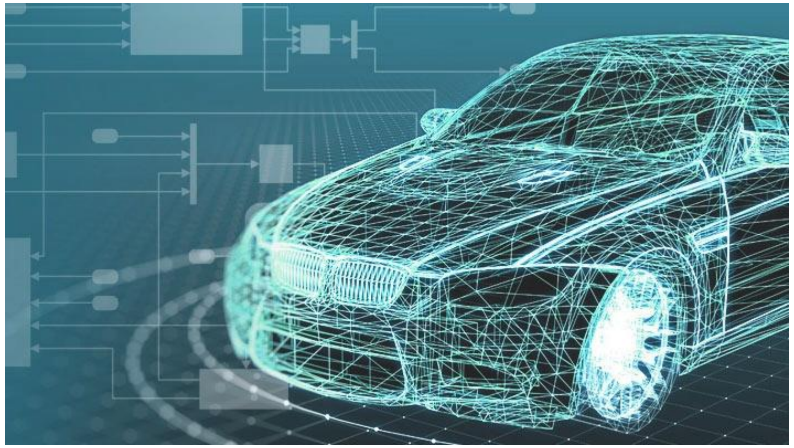
# Agenda

| | Topic |
|---|---|
| 14:15<br>–<br>14:45 | **Digital Transformation with MBSE, MBD and Early Verification** |
| | **Requirements Management and System Architecture Design** |
| | **Requirements-based Testing** |
| | **Q&A** |

# Agenda

| | Topic |
|---|---|
| 14.15<br>–<br>14:45 | Digital Transformation with MBSE, MBD and Early Verification |
| | Requirements Management and System Architecture Design |
| | Requirements-based Testing |
| | Q&A |

# Some industries use established international standards

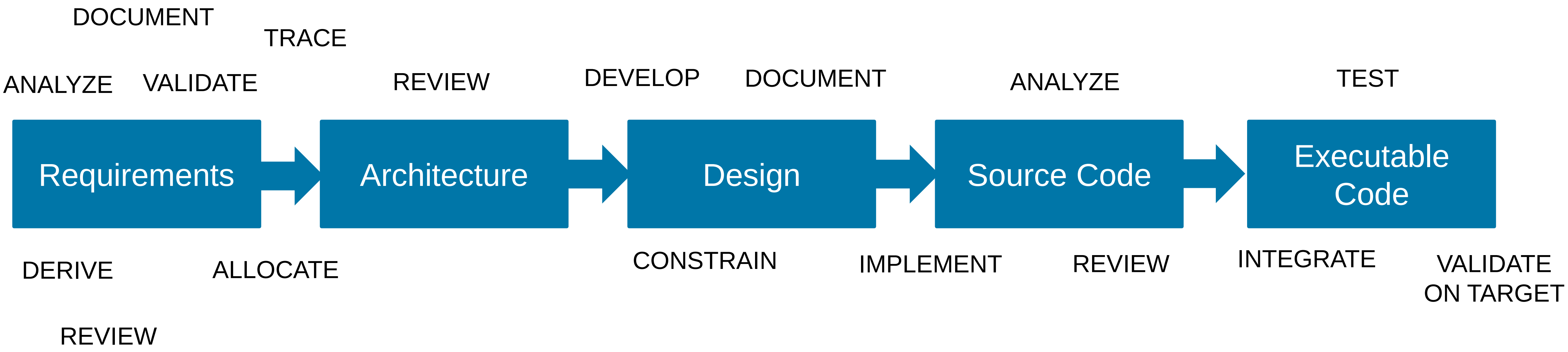


**Automotive:  ISO 26262**

**Medical Devices:  IEC 62304**

**Aeronautics:  DO-178C**

# The standards typically establish "what" should be done, but not necessarily "how" to do it



# The standards typically establish "what" should be done, but not necessarily "how" to do it

Agile

DOCUMENT

TRACE

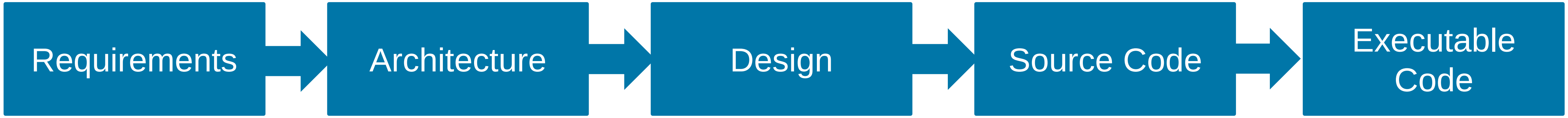
ANALYZE VALIDATE

REVIEW

DEVELOP DOCUMENT

ANALYZE

TEST



Requirements

Architecture

Design

Source Code

Executable Code

DERIVE

ALLOCATE

CONSTRAIN

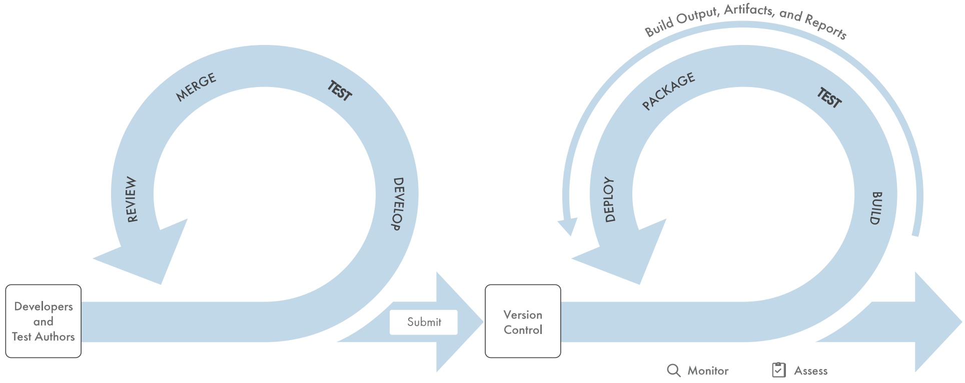
IMPLEMENT

REVIEW

INTEGRATE

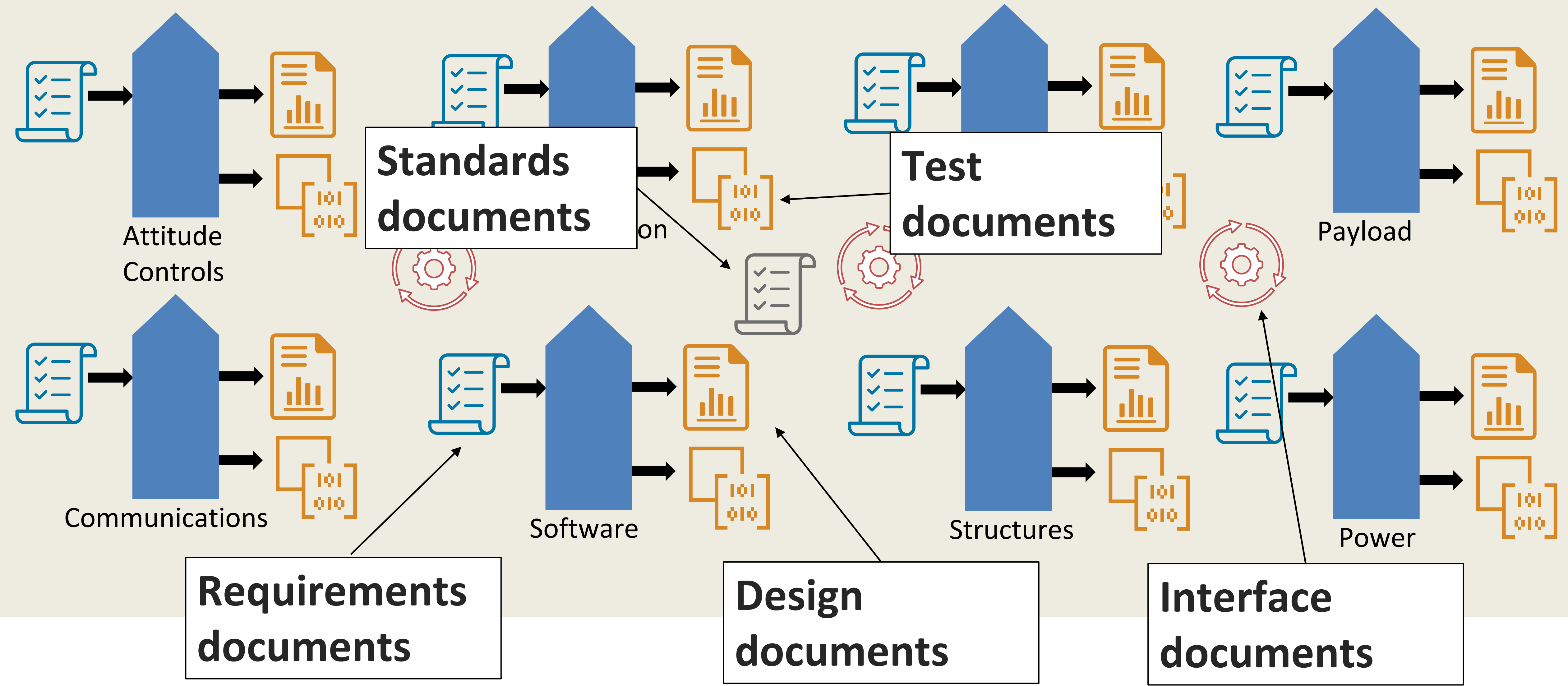
VALIDATE ON TARGET

REVIEW



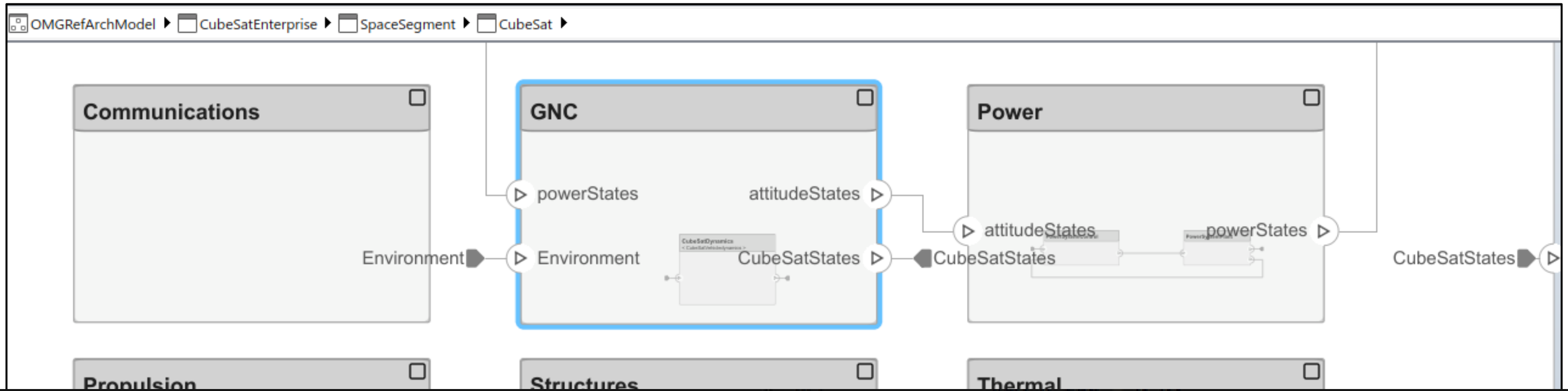
Continuous Integration

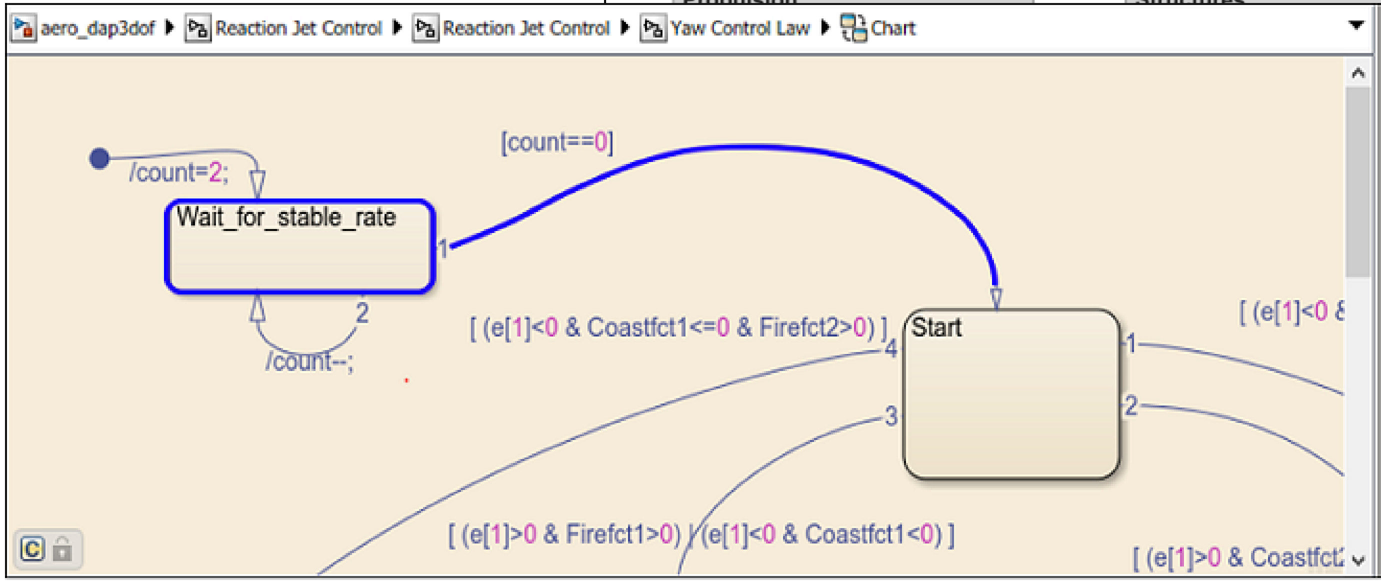
# Over time, our standards-driven process has become document-intensive and challenging to manage

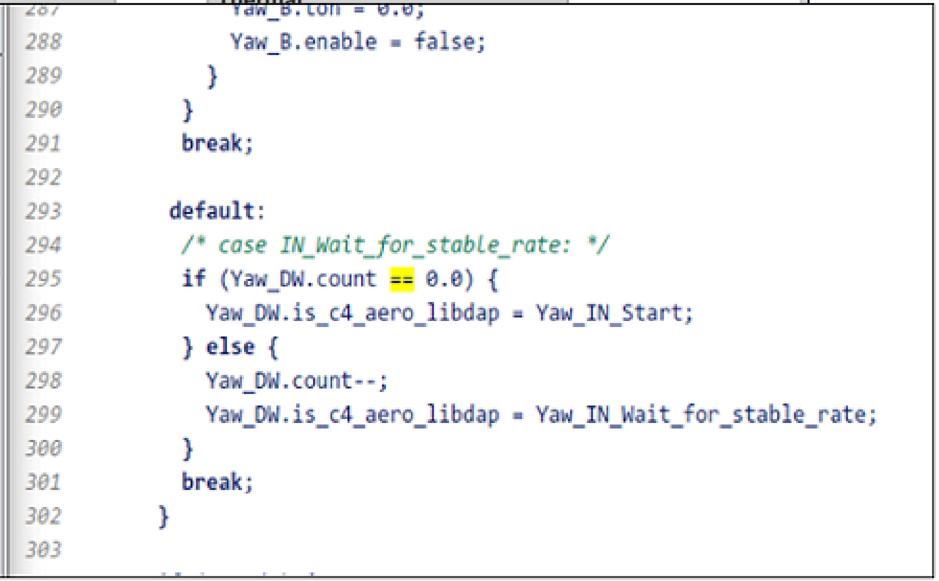


# Digital engineering addresses the disconnects in the traditional development process

## Architecture







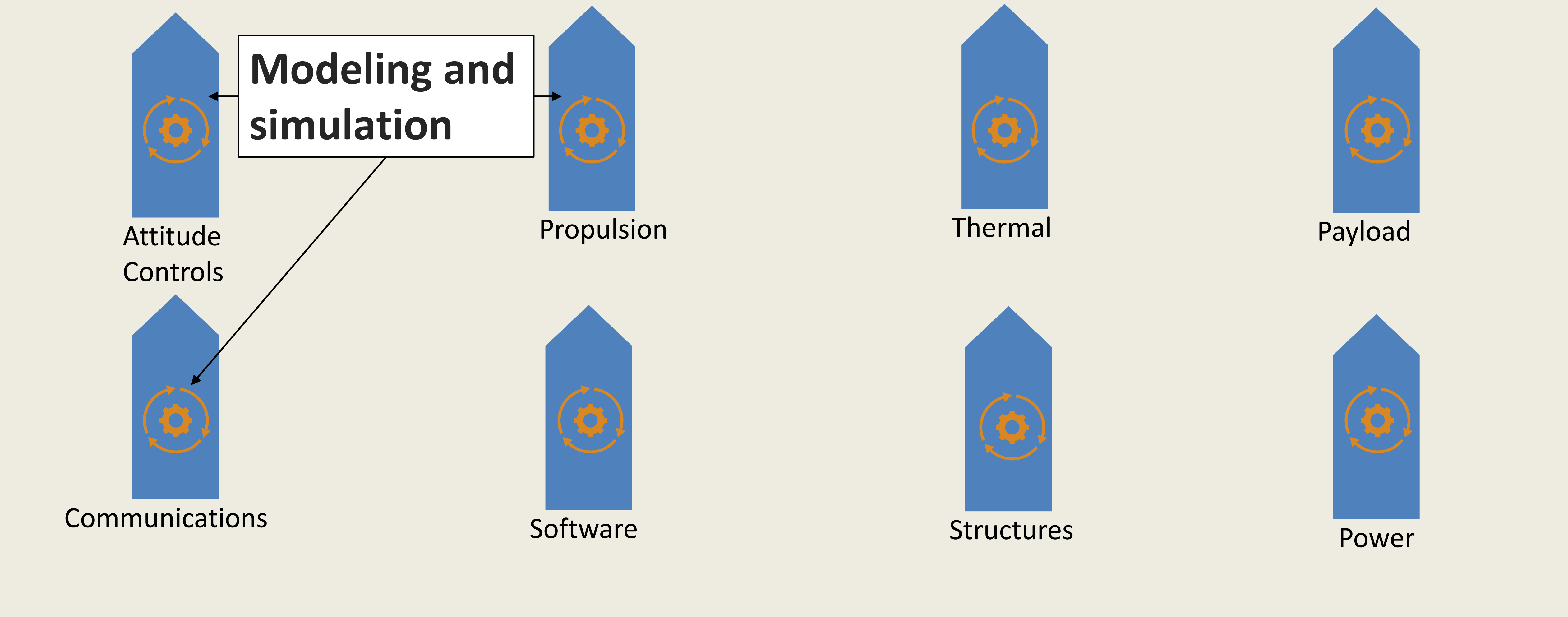
```
          Yaw_B.con = 0.0;
          Yaw_B.enable = false;
        }
      }
      break;

     default:
      /* case IN_Wait_for_stable_rate: */
      if (Yaw_DW.count == 0.0) {
        Yaw_DW.is_c4_aero_libdap = Yaw_IN_Start;
      } else {
        Yaw_DW.count--;
        Yaw_DW.is_c4_aero_libdap = Yaw_IN_Wait_for_stable_rate;
      }
      break;
    }
```

## Design

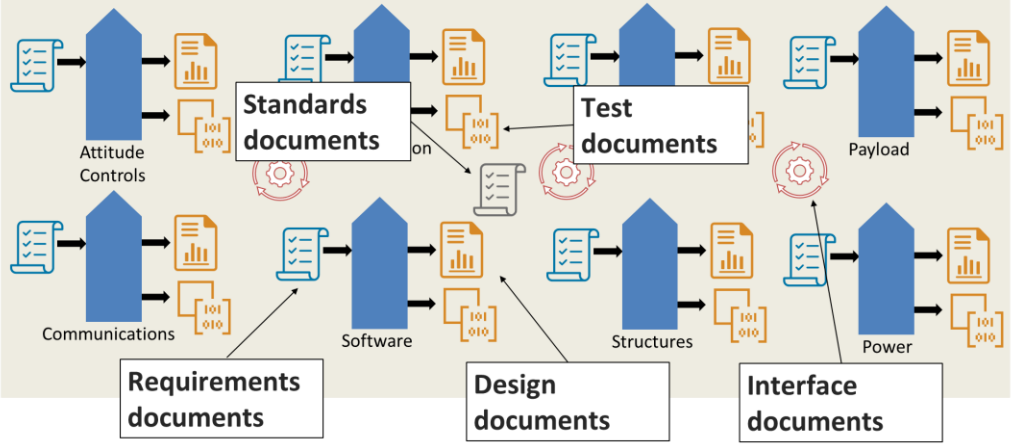
## Code

# The vision for full digital engineering is different from what we have done before

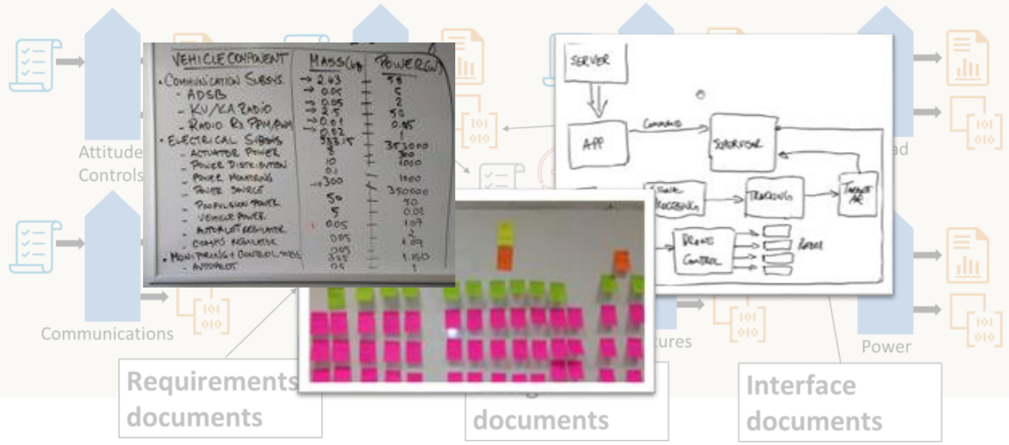


Modeling and simulation

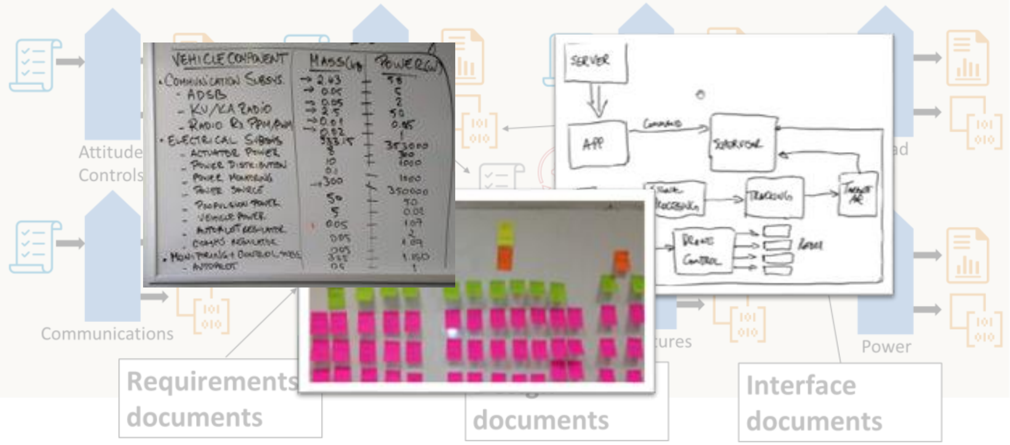
# Establishing a digital thread makes standards compliance simpler and easier

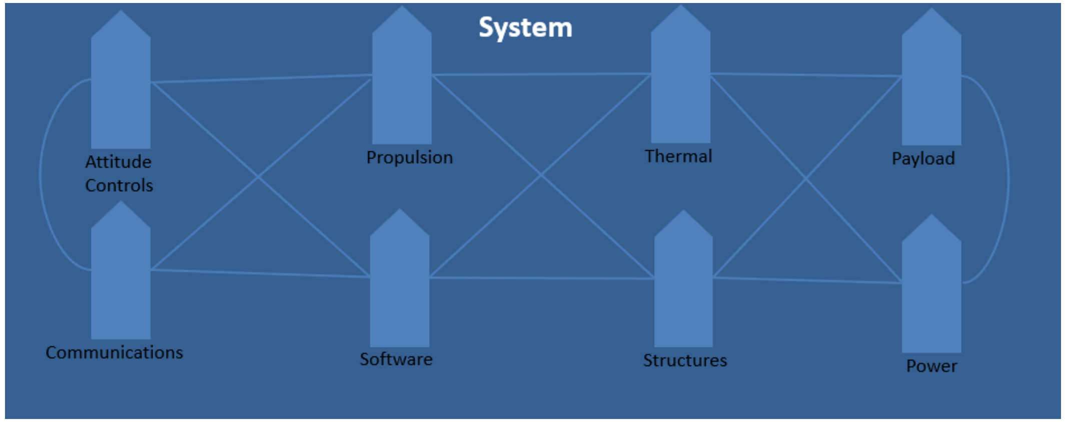


# Establishing a digital thread makes standards compliance simpler and easier

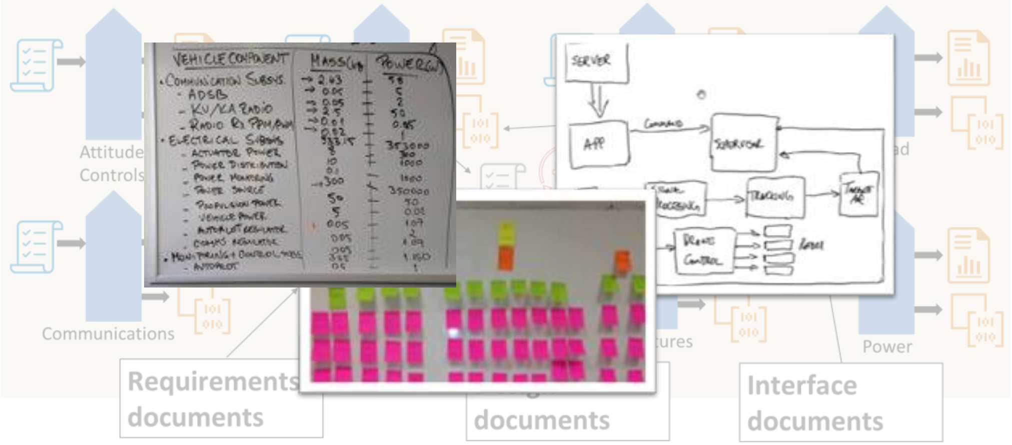


# Establishing a digital thread makes standards compliance simpler and easier

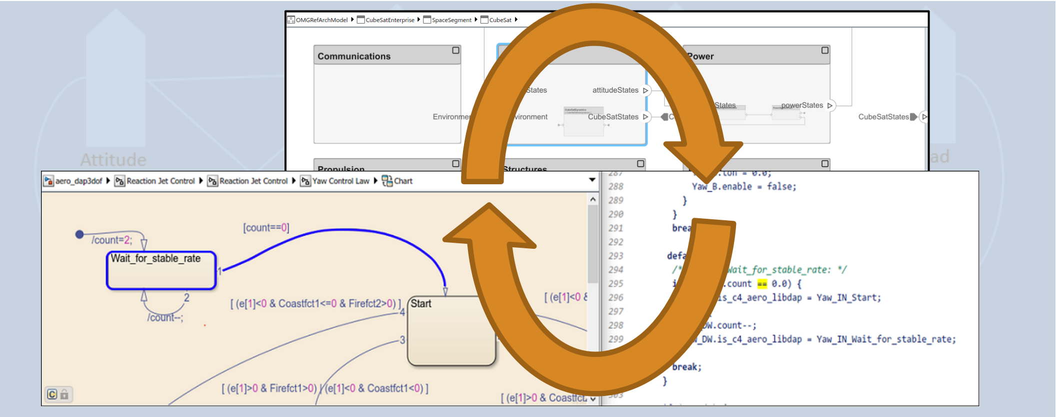




# Establishing a digital thread makes standards compliance simpler and easier

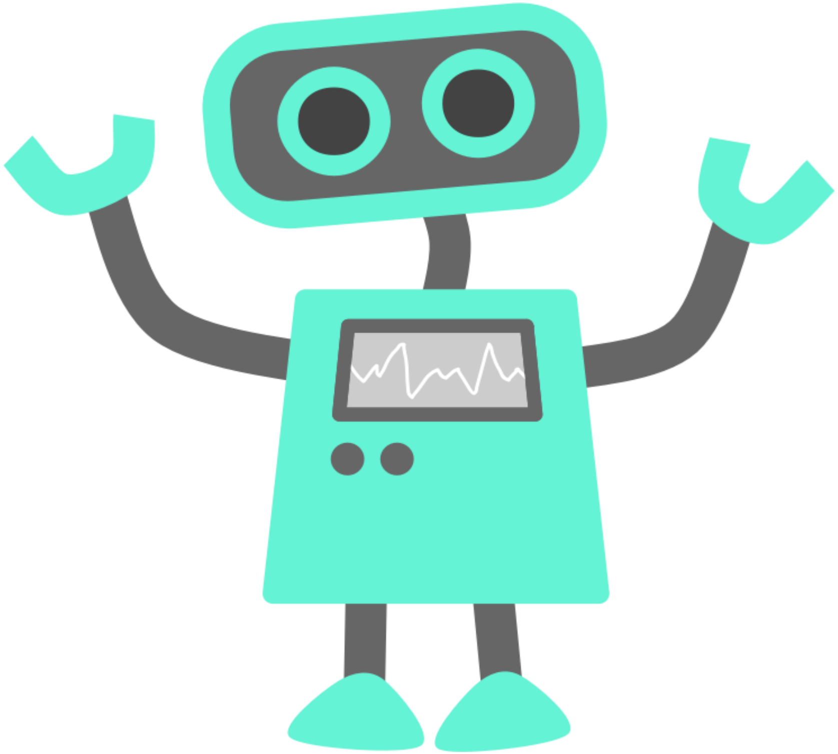


**Disconnected documents are primary artifact**



**Digital model is primary artifact**

# Automation should be approached thoughtfully

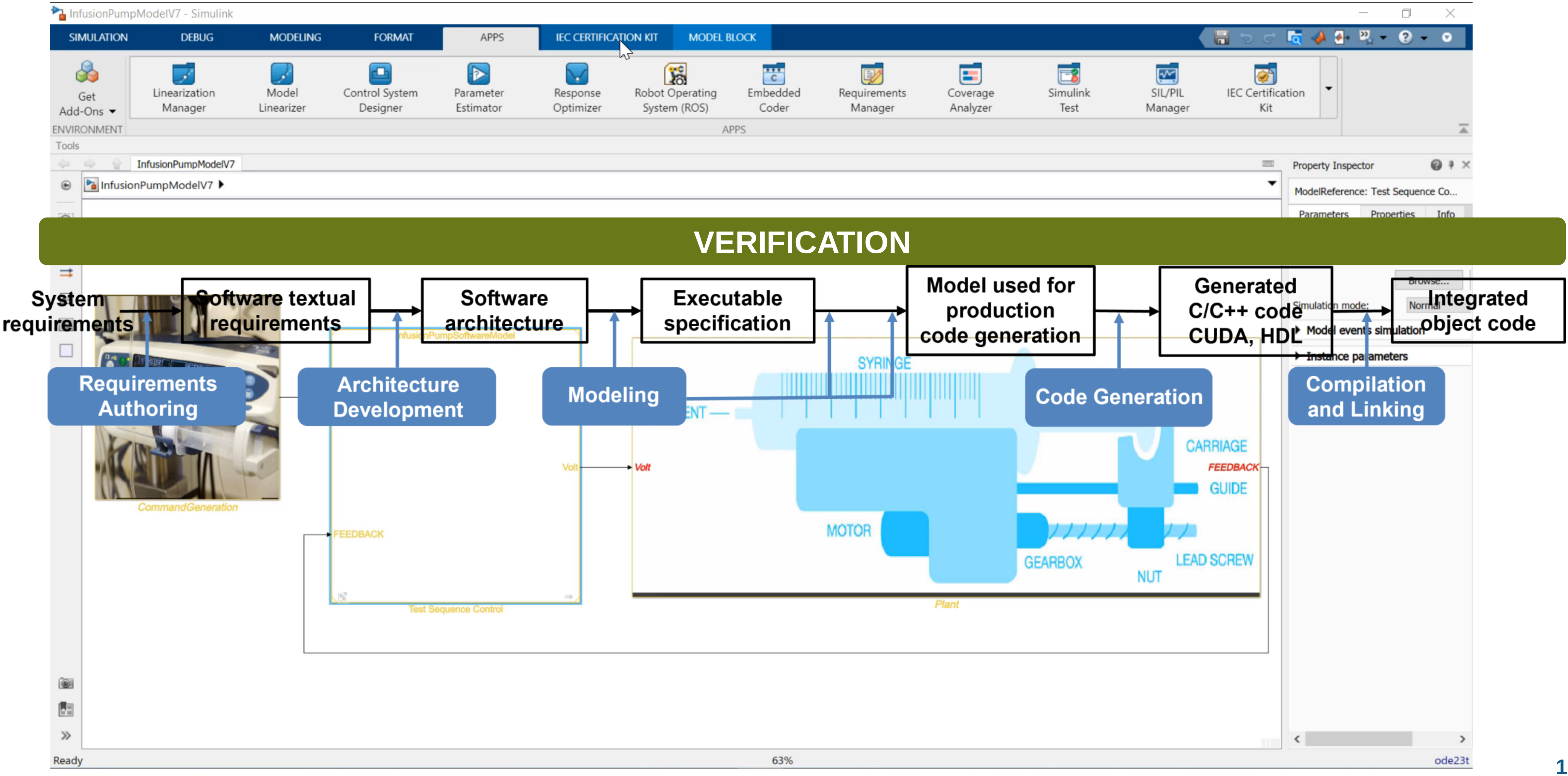


VS

**What are machines better at than humans?**

# How do we create the Digital Thread?

# Model-Based Verification and Validation Workflow



VERIFICATION

System requirements → Software textual requirements → Software architecture → Executable specification → Model used for production code generation → Generated C/C++ code CUDA, HDL → Integrated object code

Requirements Authoring

Architecture Development

Modeling

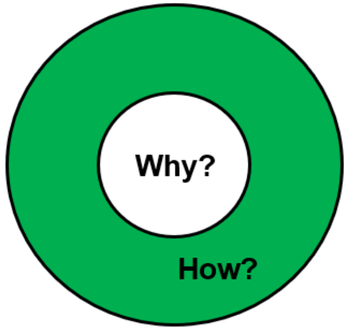
Code Generation

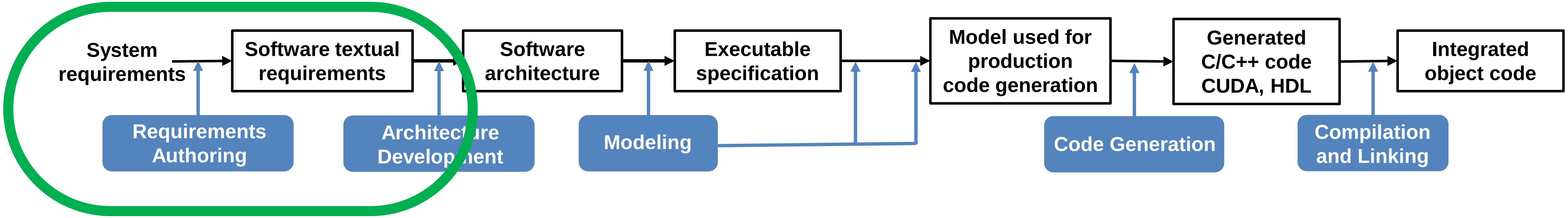
Compilation and Linking

# Agenda

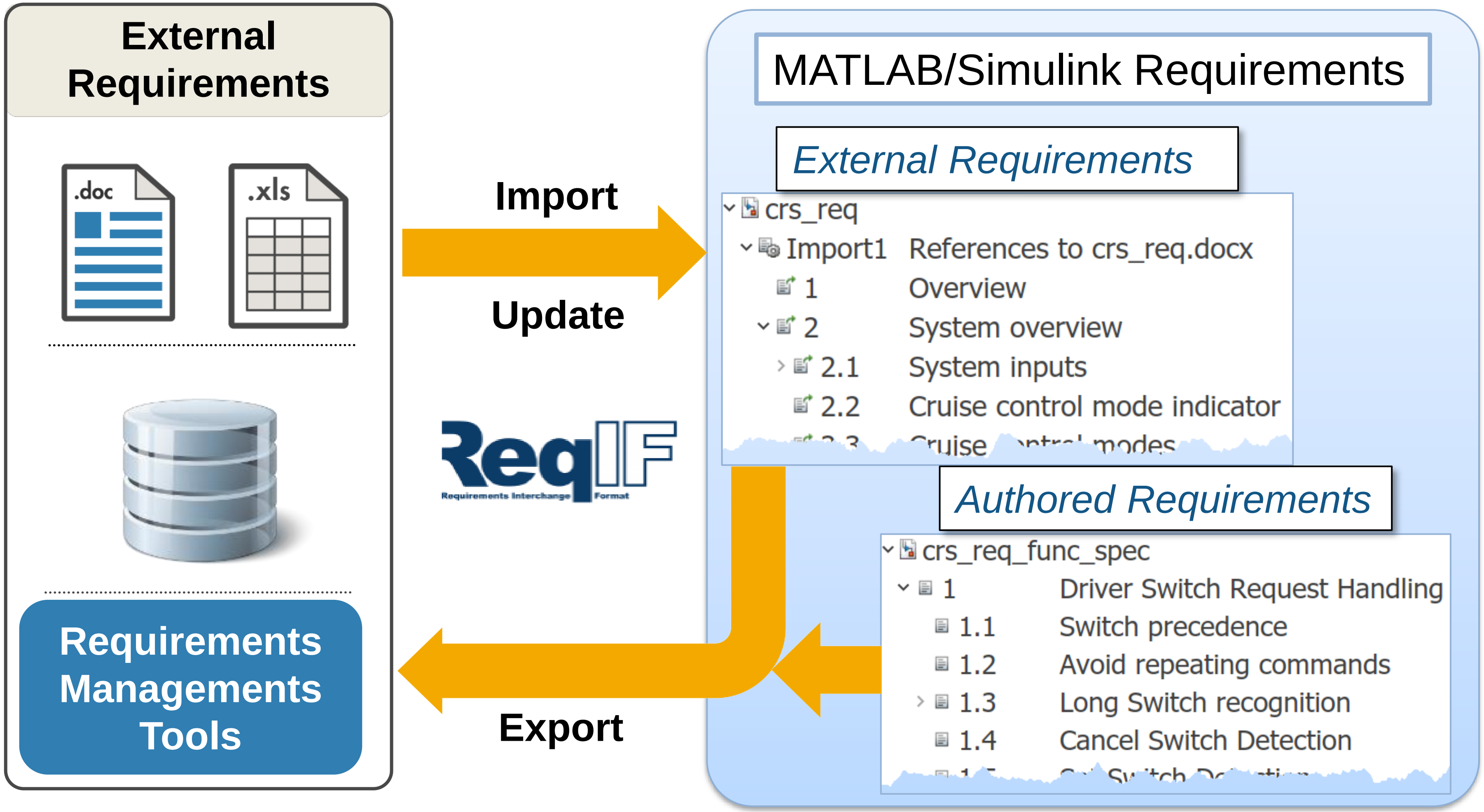
| | Topic |
|---|---|
| 14.15<br>–<br>14:45 | Digital Transformation with MBSE, MBD and Early Verification |
| | Requirements Management and System Architecture Design |
| | Requirements-based Testing |
| | Q&A |

# Complete Model-Based Design with V&V





# Integrate with requirements tools … and author requirements



External Requirements

Requirements Managements Tools

MATLAB/Simulink Requirements

*External Requirements*

| | |
|---|---|
| crs_req | |
| Import1 | References to crs_req.docx |
| 1 | Overview |
| 2 | System overview |
| 2.1 | System inputs |
| 2.2 | Cruise control mode indicator |
| 2.3 | Cruise control modes |

*Authored Requirements*

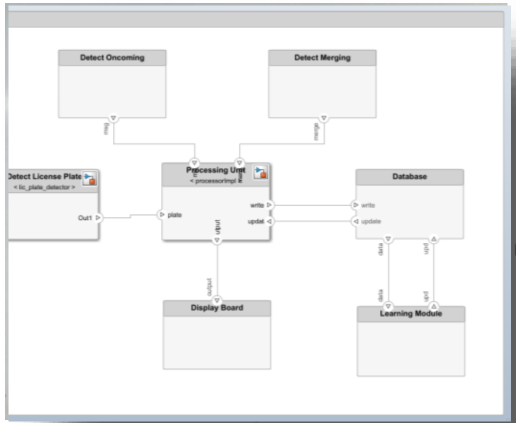
| | |
|---|---|
| crs_req_func_spec | |
| 1 | Driver Switch Request Handling |
| 1.1 | Switch precedence |
| 1.2 | Avoid repeating commands |
| 1.3 | Long Switch recognition |
| 1.4 | Cancel Switch Detection |
| 1.5 | Switch Detection |

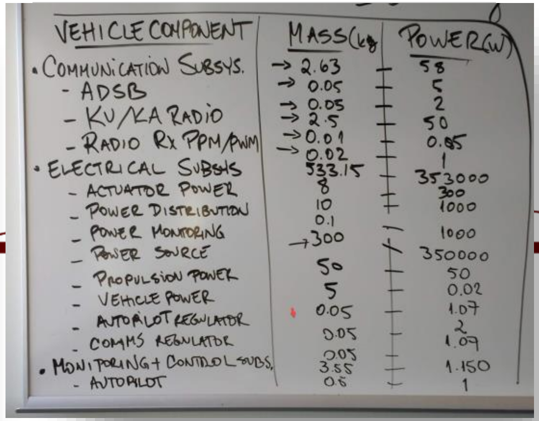
**Import/Export**

- ✓ Polarion®
- ✓ Word / Excel
- ✓ IBM® Rational® DOORS®
- ✓ ReqIF™ Standard

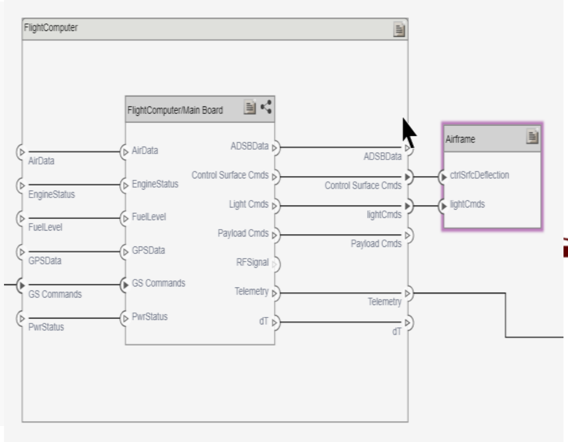
# Architecture Design

✓ Be Intuitive ✓ Facilitate Analysis ✓ Tackle Complexity ✓ Enable Implementation





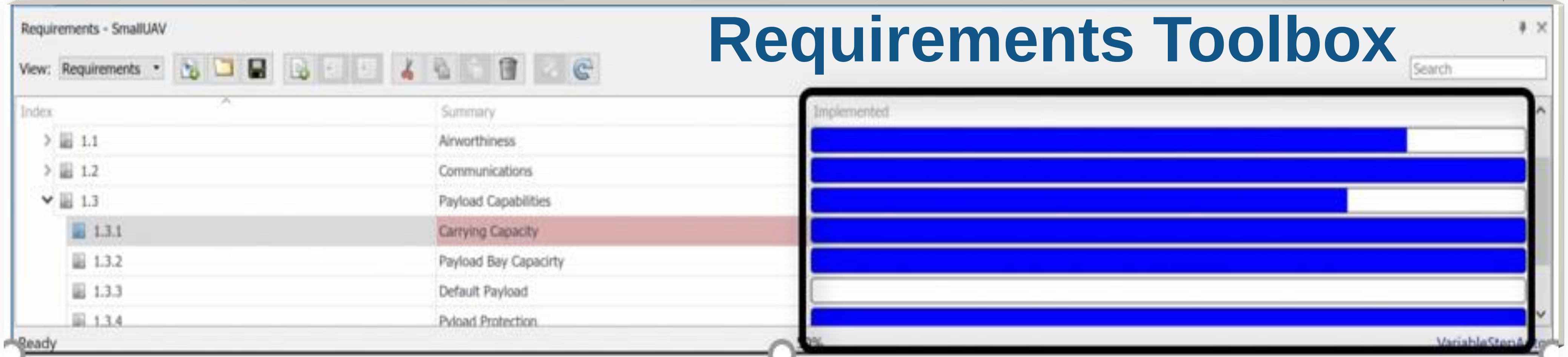
| VEHICLE COMPONENT | MASS(kg) | POWER(W) |
|---|---|---|
| • COMMUNICATION SUBSYS. | → 2.63 | 58 |
| - ADSB | → 0.05 | 5 |
| | → 0.05 | 2 |
| - KU/KA RADIO | → 2.5 | 50 |
| - RADIO RX PPM/PWM | → 0.01 | 0.05 |
| | → 0.02 | 1 |
| • ELECTRICAL SUBSYS | 533.15 | 353000 |
| - ACTUATOR POWER | 8 | 300 |
| - POWER DISTRIBUTION | 10 | 1000 |
| | 0.1 | |
| - POWER MONITORING | → 300 | 1000 |
| - POWER SOURCE | | 350000 |
| - PROPULSION POWER | 50 | 50 |
| - VEHICLE POWER | 5 | 0.02 |
| - AUTOPILOT REGULATOR | 0.05 | 1.07 |
| | | 2 |
| - COMMS REGULATOR | 0.05 | 1.07 |
| | 0.05 | |
| • MONITORING + CONTROL SUBS. | 3.55 | 1.150 |
| - AUTOPILOT | 0.5 | 1 |



update
plate
msg
merge
write
output

Simulink

Requirements Coverage Reporting and Impact Analysis



Requirements Toolbox

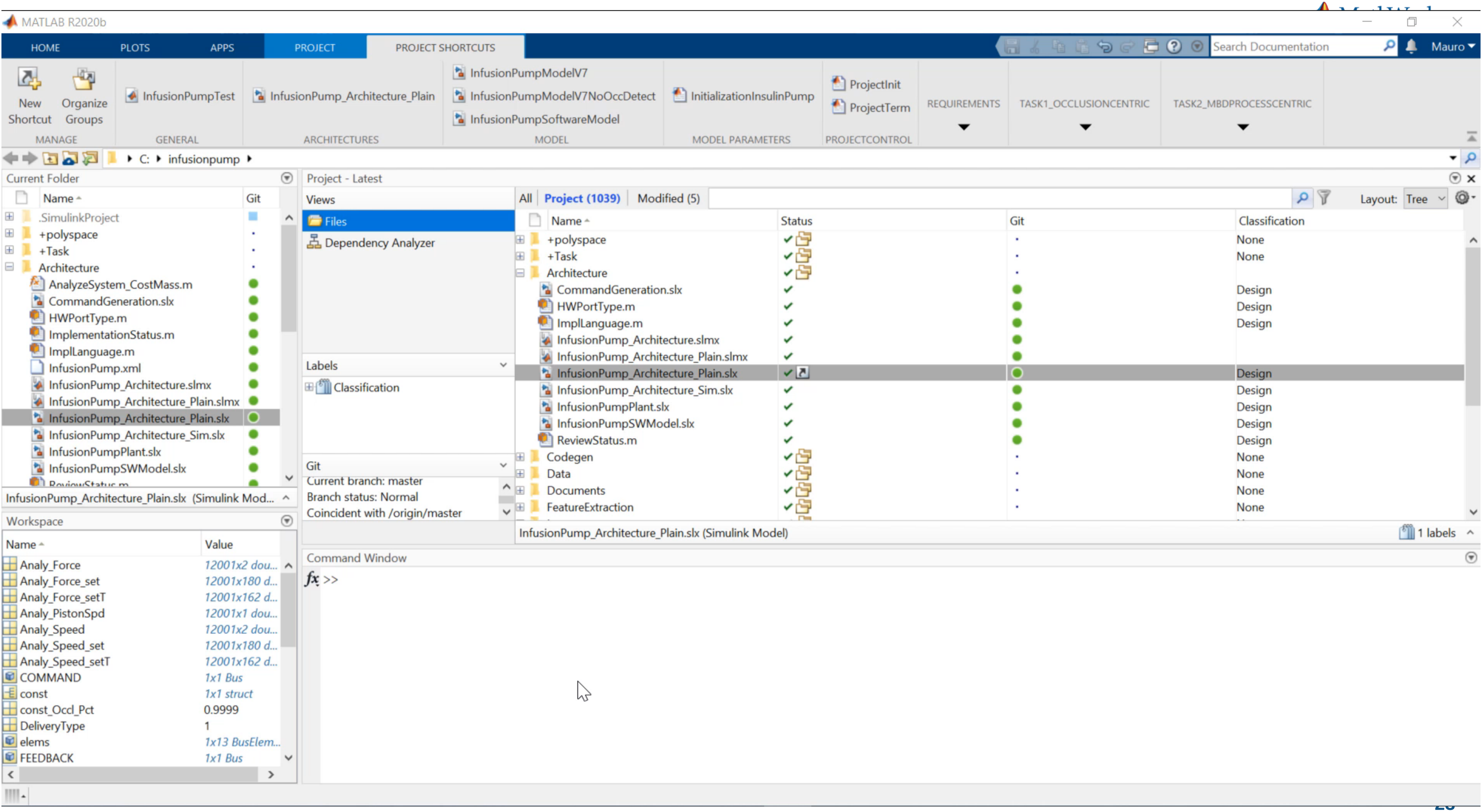
Requirements - SmallUAV

View: Requirements

| Index | Summary | Implemented |
|---|---|---|
| 1.1 | Airworthiness | |
| 1.2 | Communications | |
| 1.3 | Payload Capabilities | |
| 1.3.1 | Carrying Capacity | |
| 1.3.2 | Payload Bay Capacirty | |
| 1.3.3 | Default Payload | |
| 1.3.4 | Pyload Protection | |

Ready

VariableStepA



Current Folder

| Name | Git |
|---|---|
| .SimulinkProject | |
| +polyspace | · |
| +Task | · |
| Architecture | · |
| AnalyzeSystem_CostMass.m | ● |
| CommandGeneration.slx | ● |
| HWPortType.m | ● |
| ImplementationStatus.m | ● |
| ImplLanguage.m | ● |
| InfusionPump.xml | ● |
| InfusionPump_Architecture.slmx | ● |
| InfusionPump_Architecture_Plain.slmx | ● |
| InfusionPump_Architecture_Plain.slx | ● |
| InfusionPump_Architecture_Sim.slx | ● |
| InfusionPumpPlant.slx | ● |
| InfusionPumpSWModel.slx | ● |
| ReviewStatus.m | ● |

InfusionPump_Architecture_Plain.slx (Simulink Mod...

Project - Latest

| Name | Status | Git | Classification |
|---|---|---|---|
| +polyspace | ✓ | · | None |
| +Task | ✓ | · | None |
| Architecture | ✓ | · | |
| CommandGeneration.slx | ✓ | ● | Design |
| HWPortType.m | ✓ | ● | Design |
| ImplLanguage.m | ✓ | ● | Design |
| InfusionPump_Architecture.slmx | ✓ | ● | |
| InfusionPump_Architecture_Plain.slmx | ✓ | ● | |
| InfusionPump_Architecture_Plain.slx | ✓ | ● | Design |
| InfusionPump_Architecture_Sim.slx | ✓ | ● | Design |
| InfusionPumpPlant.slx | ✓ | ● | Design |
| InfusionPumpSWModel.slx | ✓ | ● | Design |
| ReviewStatus.m | ✓ | ● | Design |
| Codegen | ✓ | · | None |
| Data | ✓ | · | None |
| Documents | ✓ | · | None |
| FeatureExtraction | ✓ | · | None |

InfusionPump_Architecture_Plain.slx (Simulink Model)

Git

Current branch: master
Branch status: Normal
Coincident with /origin/master

Workspace

| Name | Value |
|---|---|
| Analy_Force | 12001x2 dou... |
| Analy_Force_set | 12001x180 d... |
| Analy_Force_setT | 12001x162 d... |
| Analy_PistonSpd | 12001x1 dou... |
| Analy_Speed | 12001x2 dou... |
| Analy_Speed_set | 12001x180 d... |
| Analy_Speed_setT | 12001x162 d... |
| COMMAND | 1x1 Bus |
| const | 1x1 struct |
| const_Occl_Pct | 0.9999 |
| DeliveryType | 1 |
| elems | 1x13 BusElem... |
| FEEDBACK | 1x1 Bus |

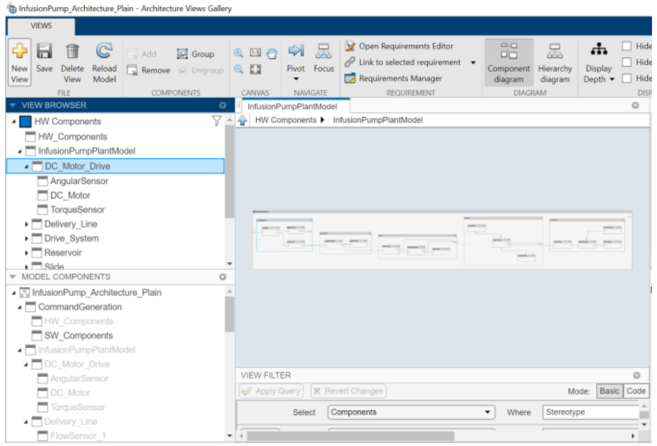
Command Window

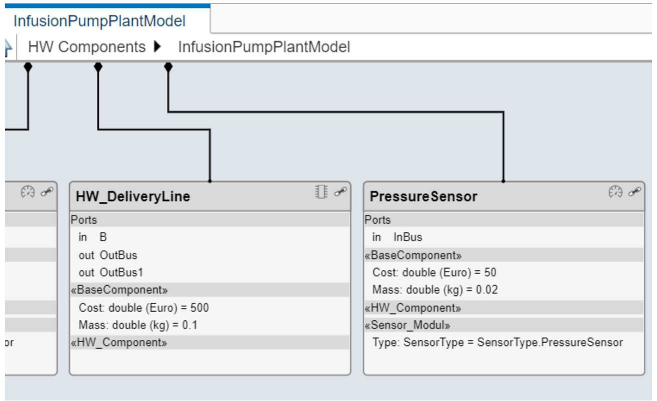
```
fx >>
```

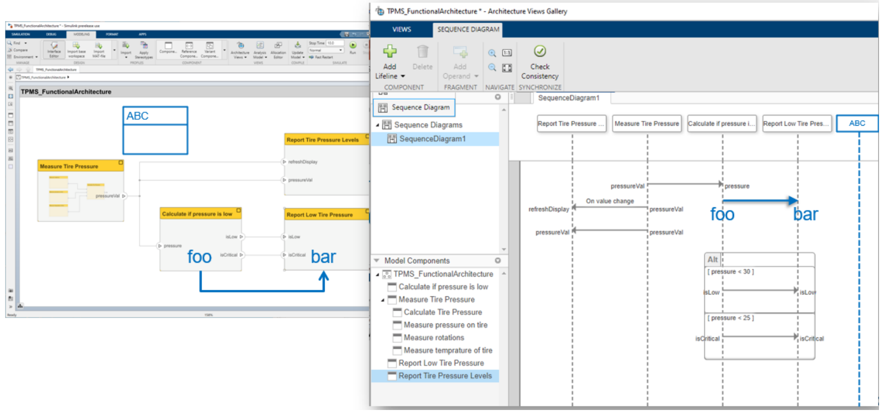
# Architecture Development

Define, analyze and specify architectures and compositions for model-based systems engineering and software design:

- Define profiles to capture properties via stereotyping
- Allocate requirements to architecture
- Define behaviors and keep them synchronized with your architecture
- Perform analysis



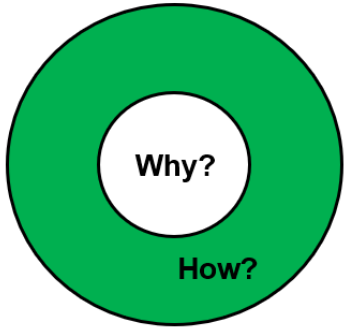


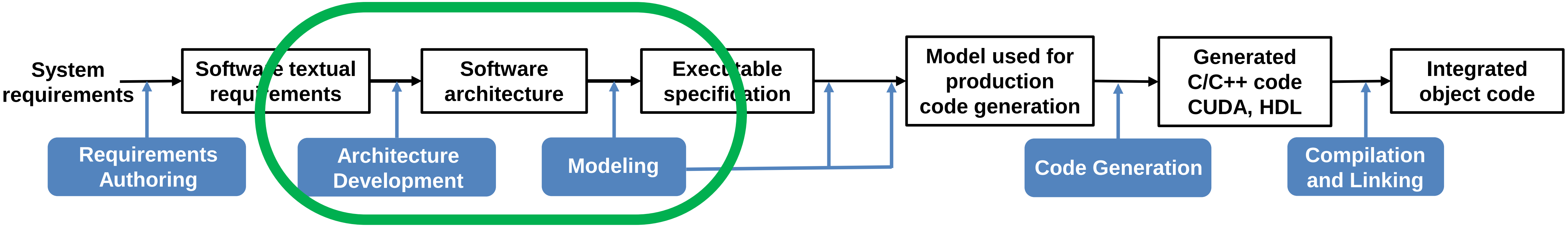


# Agenda

| | Topic |
|---|---|
| 14:15<br>-<br>14:45 | Digital Transformation with MBSE, MBD and Early Verification |
| | Requirements Management and System Architecture Design |
| | Requirements-based Testing |
| | Q&A |

# Complete Model-Based Design with V&V

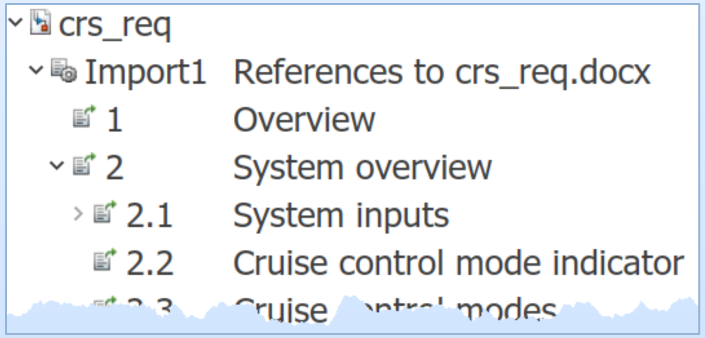




# Requirements and Design Traceability

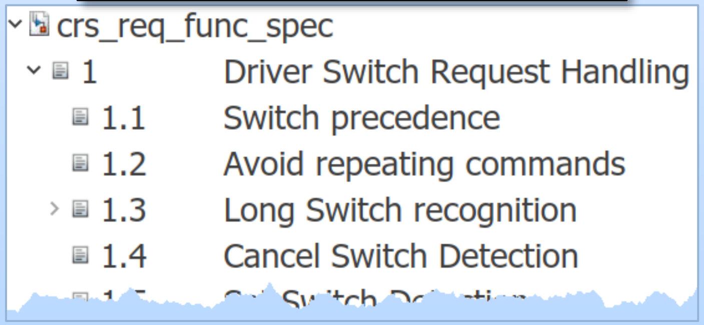
*External Requirements*

**Requirements Managements Tools**



| | | |
|---|---|---|
| crs_req | | |
| Import1 | References to crs_req.docx | |
| 1 | Overview | |
| 2 | System overview | |
| 2.1 | System inputs | |
| 2.2 | Cruise control mode indicator | |
| 2.3 | Cruise control modes | |

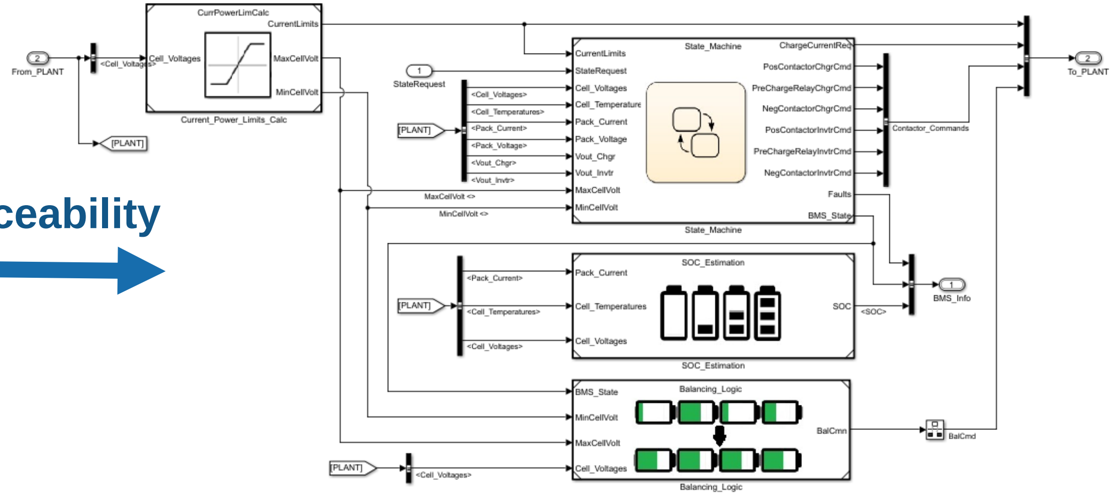
*Authored Requirements*

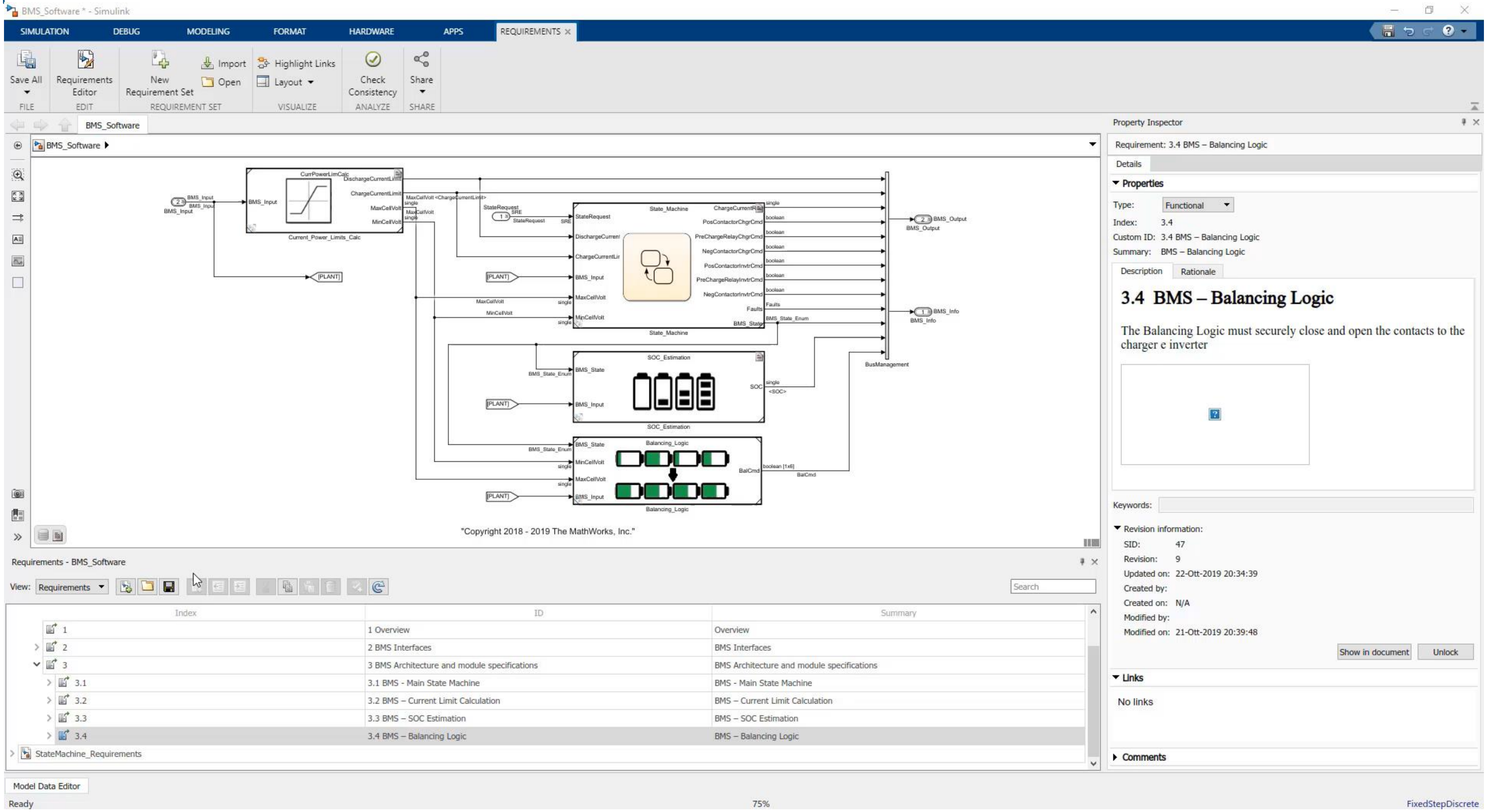


| | |
|---|---|
| crs_req_func_spec | |
| 1 | Driver Switch Request Handling |
| 1.1 | Switch precedence |
| 1.2 | Avoid repeating commands |
| 1.3 | Long Switch recognition |
| 1.4 | Cancel Switch Detection |
| 1.5 | Set Switch Detection |

**Traceability**

*Design in Simulink*





Requirements - BMS_Software

View: Requirements

| Index | ID | Summary |
|---|---|---|
| 1 | 1 Overview | Overview |
| 2 | 2 BMS Interfaces | BMS Interfaces |
| 3 | 3 BMS Architecture and module specifications | BMS Architecture and module specifications |
| 3.1 | 3.1 BMS - Main State Machine | BMS - Main State Machine |
| 3.2 | 3.2 BMS – Current Limit Calculation | BMS – Current Limit Calculation |
| 3.3 | 3.3 BMS – SOC Estimation | BMS – SOC Estimation |
| 3.4 | 3.4 BMS – Balancing Logic | BMS – Balancing Logic |
| StateMachine_Requirements | | |

Property Inspector

Requirement: 3.4 BMS – Balancing Logic

Details

Properties

Type: Functional

Index: 3.4

Custom ID: 3.4 BMS – Balancing Logic

Summary: BMS – Balancing Logic

Description | Rationale

**3.4 BMS – Balancing Logic**

The Balancing Logic must securely close and open the contacts to the charger e inverter

Keywords:

Revision information:

SID: 47

Revision: 9

Updated on: 22-Ott-2019 20:34:39

Created by:

Created on: N/A

Modified by:

Modified on: 21-Ott-2019 20:39:48

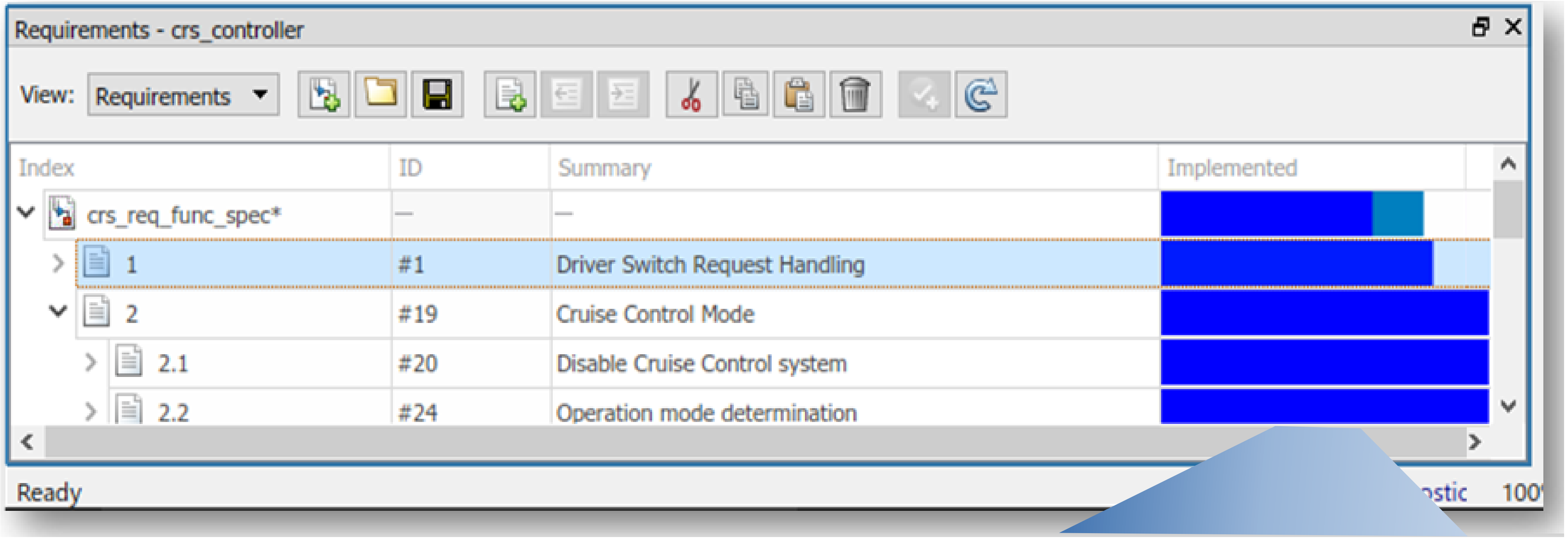
Show in document | Unlock

Links

No links

Comments

# Requirements Implementation Status



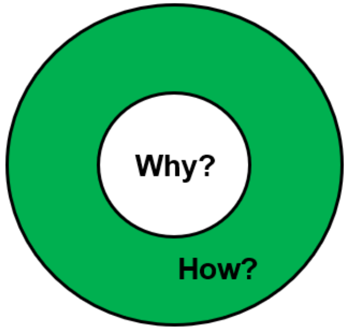
Requirements - crs_controller

| Index | ID | Summary | Implemented |
|---|---|---|---|
| crs_req_func_spec* | — | — | |
| 1 | #1 | Driver Switch Request Handling | |
| 2 | #19 | Cruise Control Mode | |
| 2.1 | #20 | Disable Cruise Control system | |
| 2.2 | #24 | Operation mode determination | |

**Implementation Status**

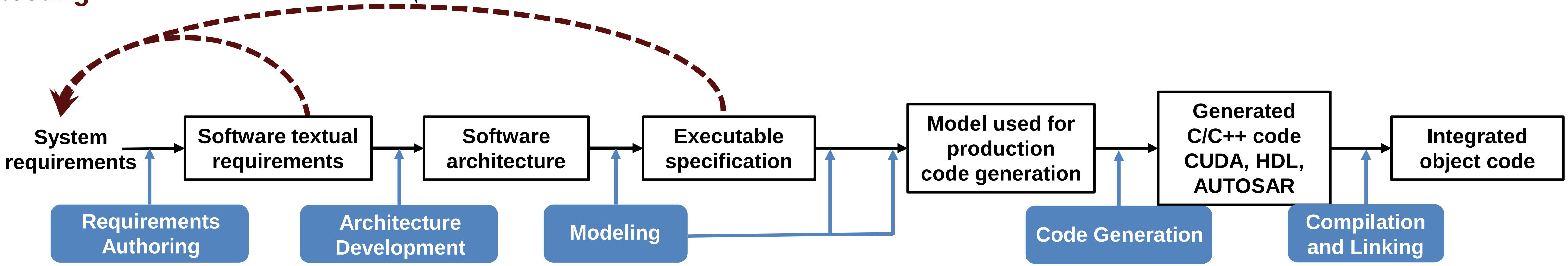
| | |
|---|---|
| ■ | Implemented |
| ▨ | Justified |
| □ | Missing |

# Testing in Simulink



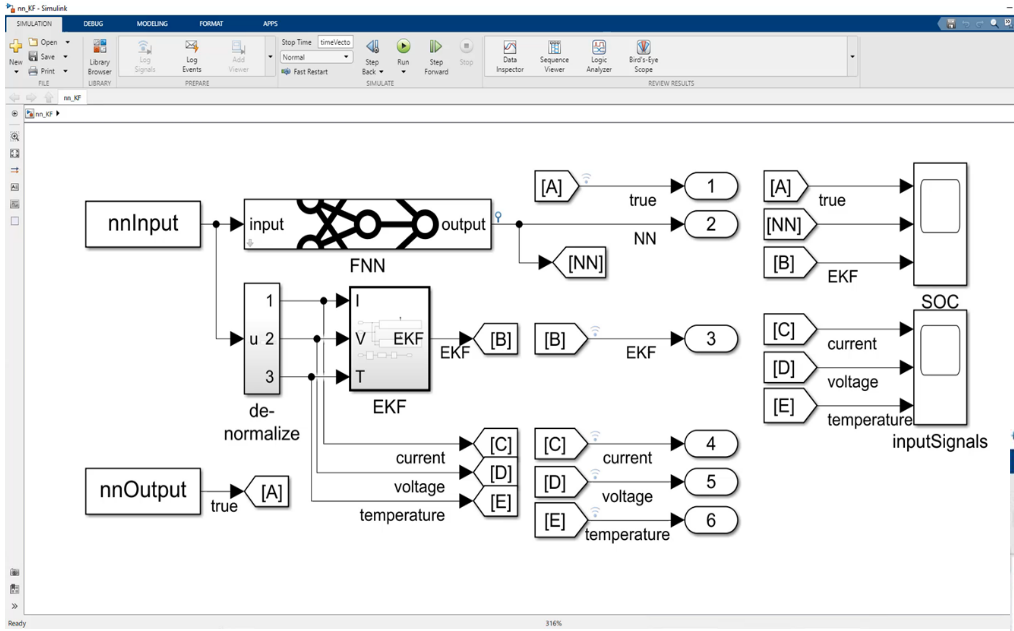
1. Isolate and test components
2. Manage and organize tests
3. Traceability Requirements - Tests
4. Measure model coverage
5. Generate tests for missing coverage

**Component and system testing**

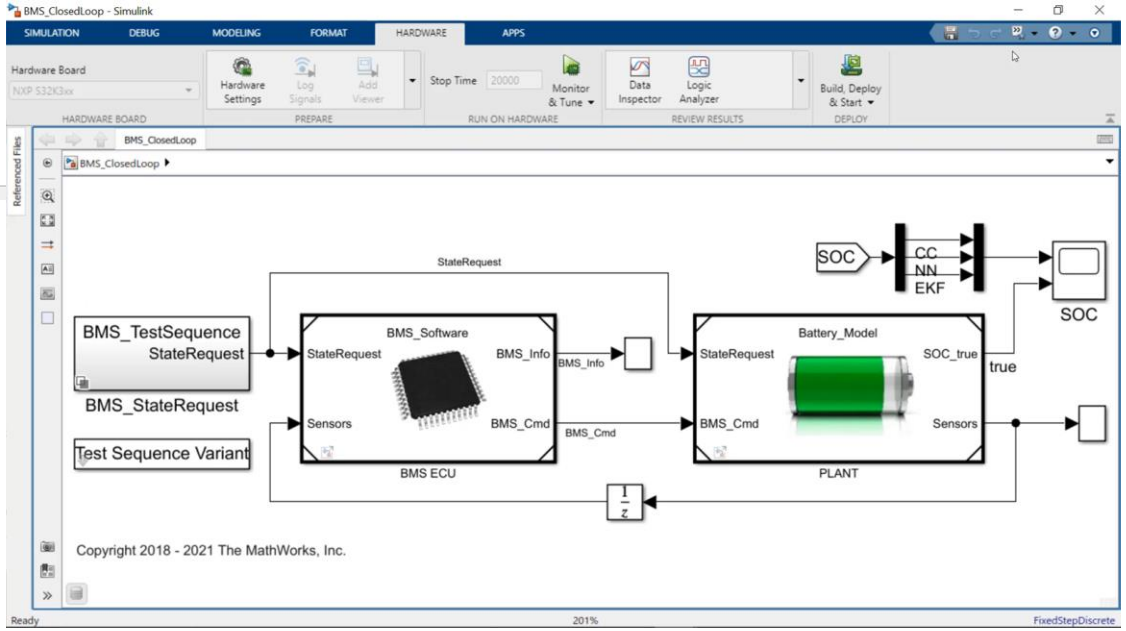


# Test Early at Unit and System Level

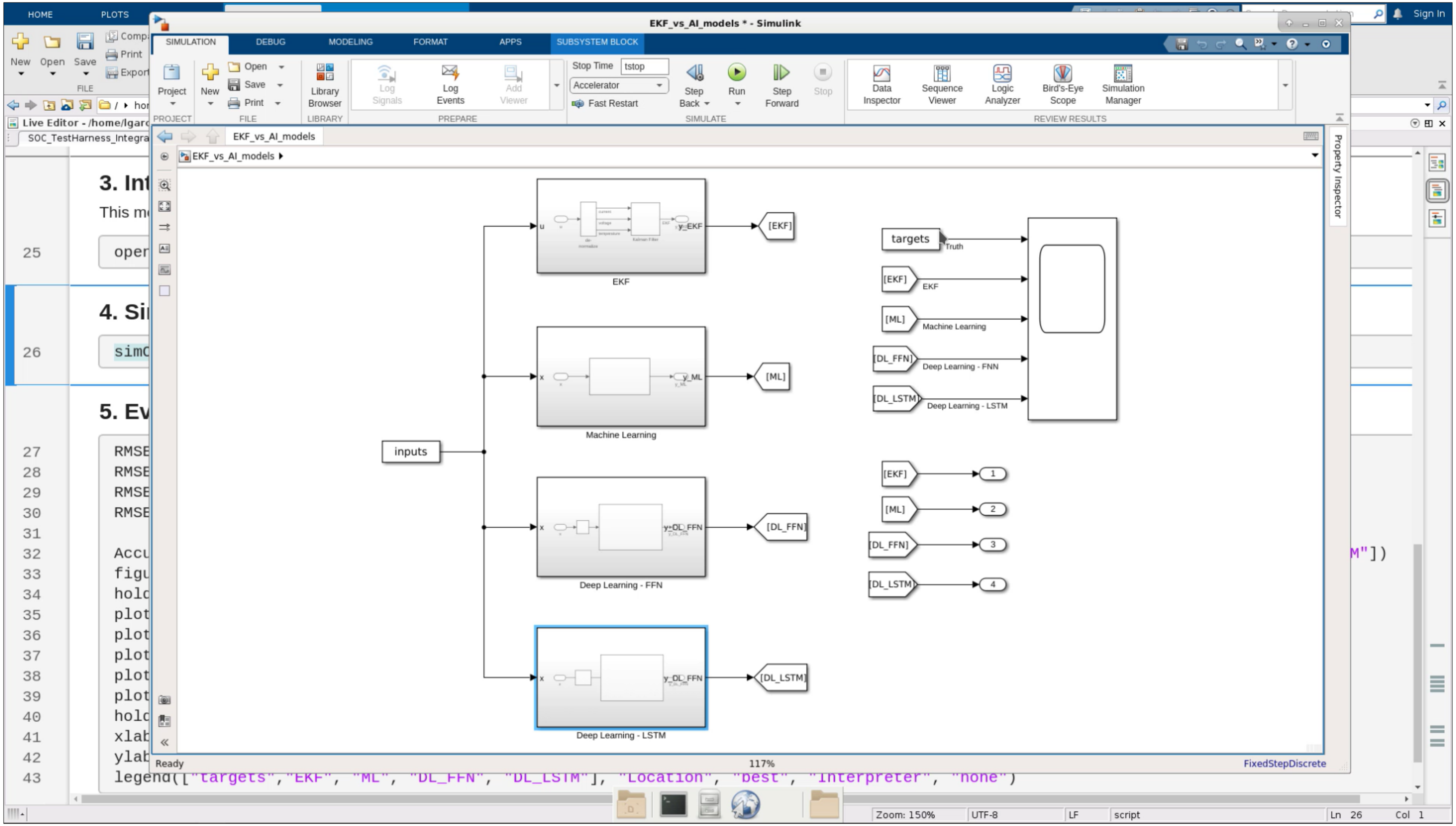
## Unit-level simulation



## System-level simulation



# Integration of trained AI models into Simulink



EKF_vs_AI_models * - Simulink

```
3. Int
This m
open

4. Si
simO

5. Ev
RMSE
RMSE
RMSE
RMSE

Accu
figu
hold
plot
plot
plot
plot
plot
hold
xlab
ylab
legend(["targets","EKF", "ML", "DL_FFN", "DL_LSTM"], "Location", "best", "Interpreter", "none")
```

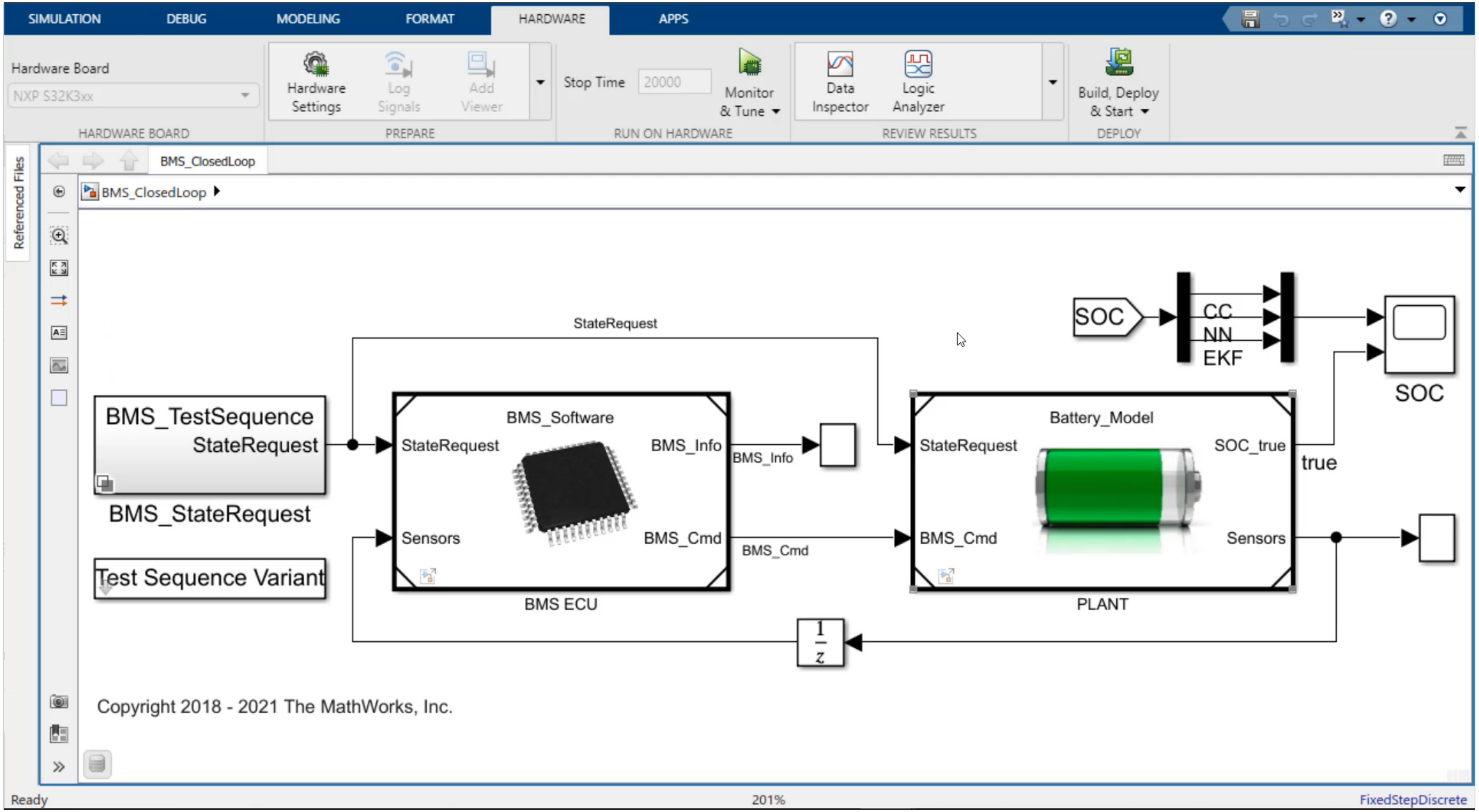
Stop Time: tstop

Accelerator

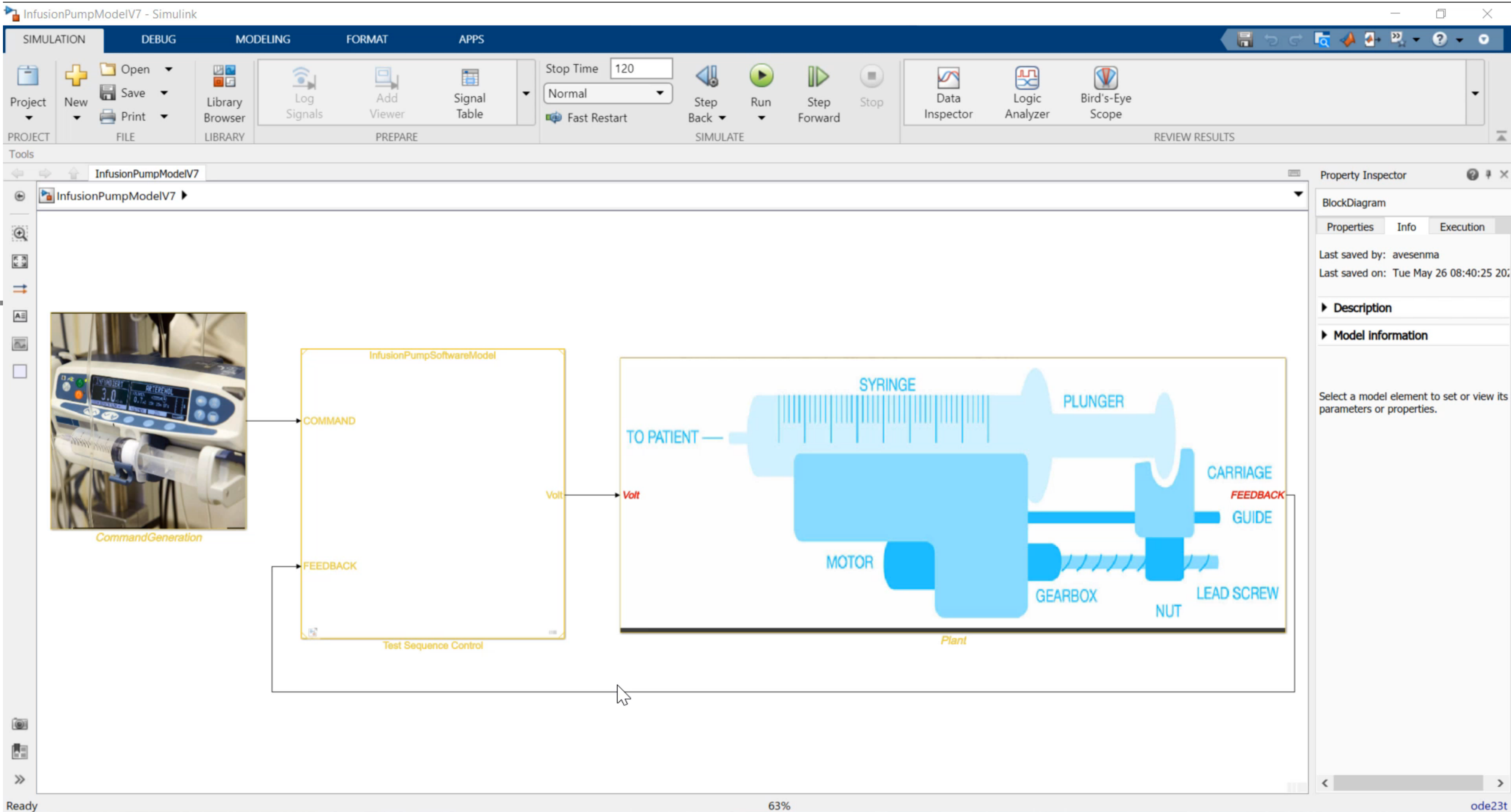
Ready 117% FixedStepDiscrete

Zoom: 150% UTF-8 LF script Ln 26 Col 1

# System-Level Simulation



Copyright 2018 - 2021 The MathWorks, Inc.



Stop Time 120

Normal

Fast Restart

Last saved by: avesenma

Last saved on: Tue May 26 08:40:25 20

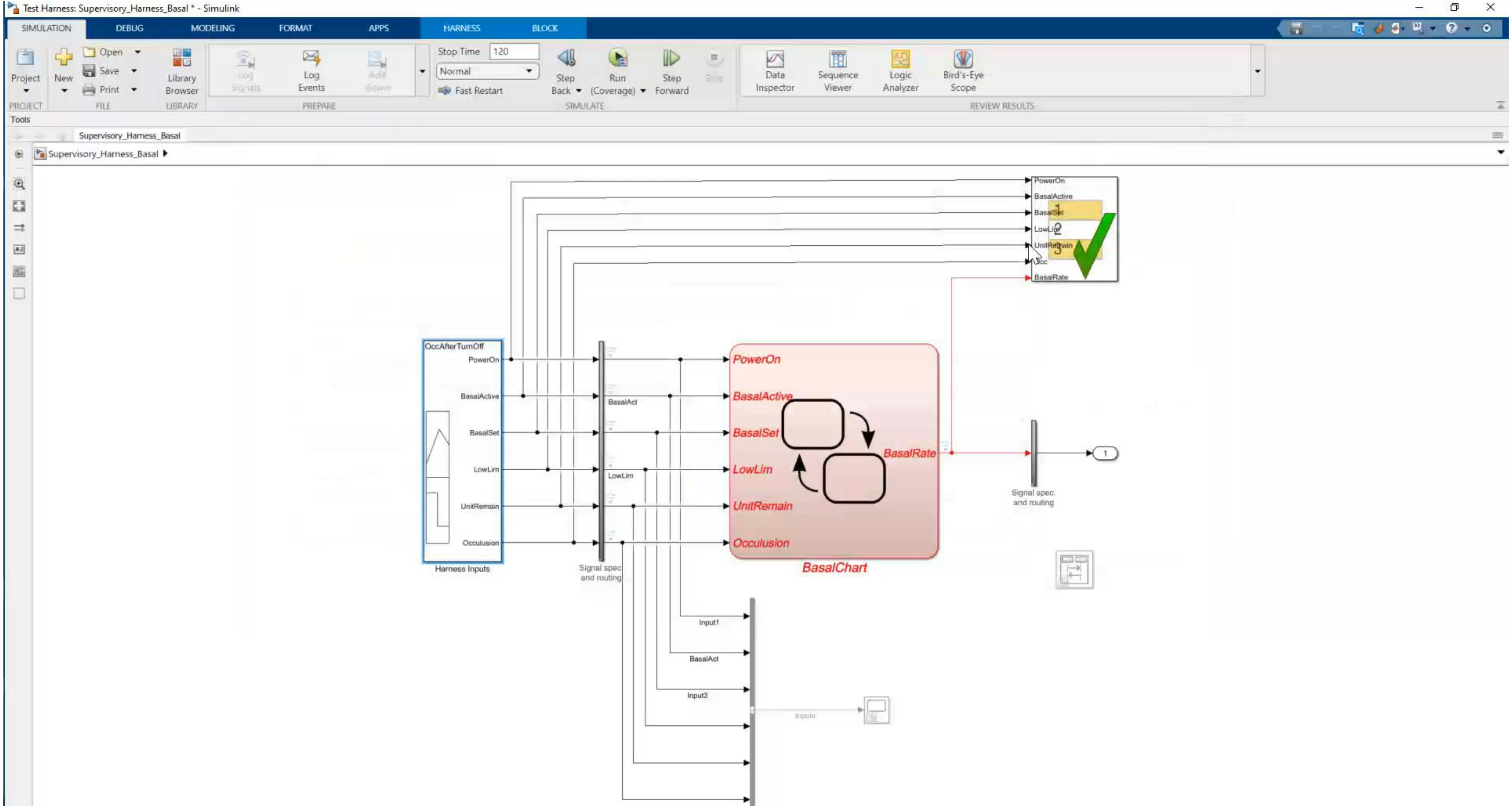
Select a model element to set or view its parameters or properties.

Ready

63%

ode23t

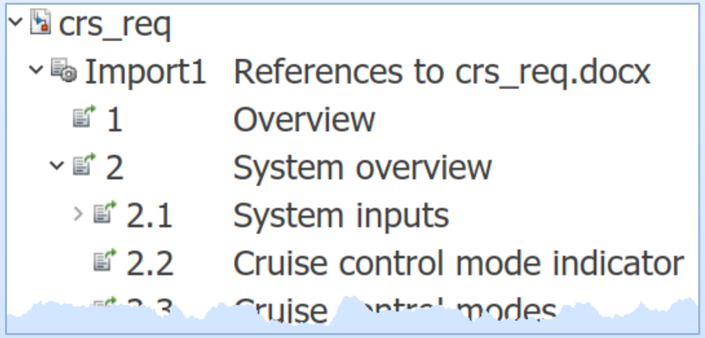
# Manage Tests and Automate



# Traceability Requirements - Tests

*External Requirements*

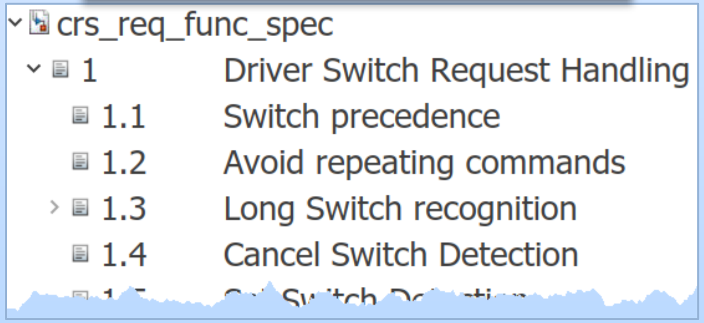
Requirements Managements Tools



crs_req

| | |
|---|---|
| Import1 | References to crs_req.docx |
| 1 | Overview |
| 2 | System overview |
| 2.1 | System inputs |
| 2.2 | Cruise control mode indicator |
| 2.3 | Cruise control modes |

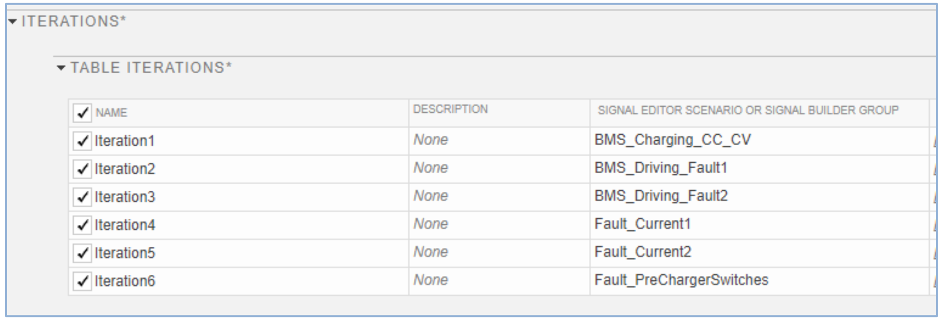
*Authored Requirements*



crs_req_func_spec

| | |
|---|---|
| 1 | Driver Switch Request Handling |
| 1.1 | Switch precedence |
| 1.2 | Avoid repeating commands |
| 1.3 | Long Switch recognition |
| 1.4 | Cancel Switch Detection |
| 1.5 | [illegible] Switch Detection |

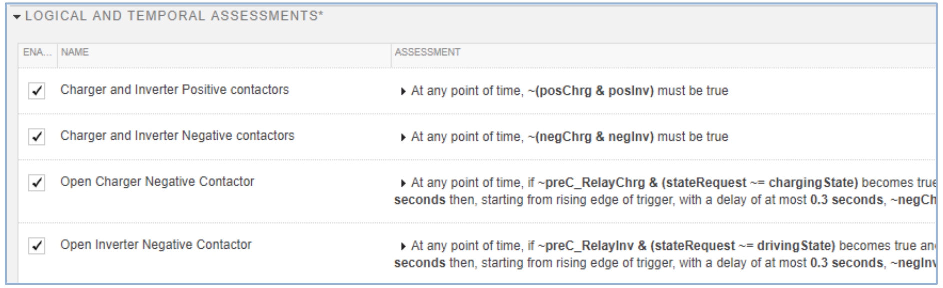
**Traceability**



ITERATIONS*

TABLE ITERATIONS*

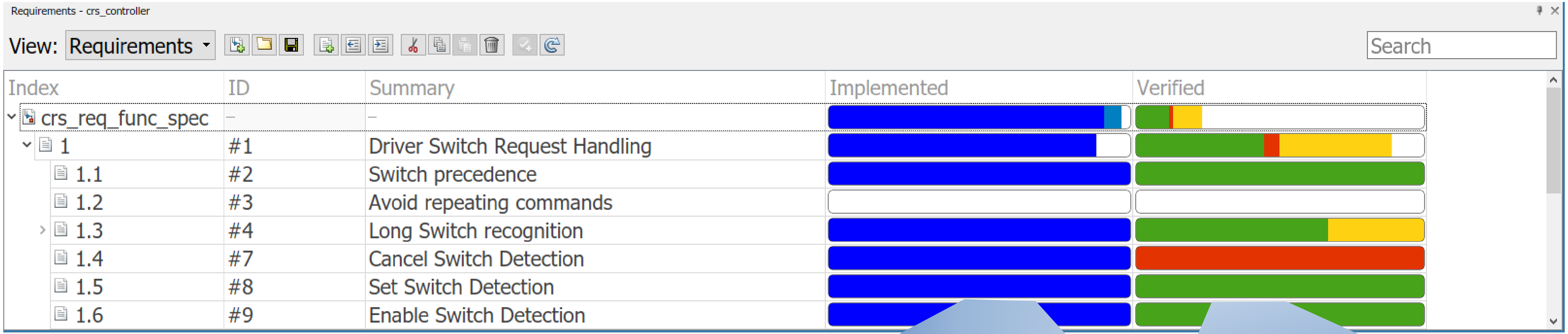
| ☑ NAME | DESCRIPTION | SIGNAL EDITOR SCENARIO OR SIGNAL BUILDER GROUP |
|---|---|---|
| ☑ Iteration1 | *None* | BMS_Charging_CC_CV |
| ☑ Iteration2 | *None* | BMS_Driving_Fault1 |
| ☑ Iteration3 | *None* | BMS_Driving_Fault2 |
| ☑ Iteration4 | *None* | Fault_Current1 |
| ☑ Iteration5 | *None* | Fault_Current2 |
| ☑ Iteration6 | *None* | Fault_PreChargerSwitches |



LOGICAL AND TEMPORAL ASSESSMENTS*

| ENA... | NAME | ASSESSMENT |
|---|---|---|
| ☑ | Charger and Inverter Positive contactors | At any point of time, **~(posChrg & posInv)** must be true |
| ☑ | Charger and Inverter Negative contactors | At any point of time, **~(negChrg & negInv)** must be true |
| ☑ | Open Charger Negative Contactor | At any point of time, if **~preC_RelayChrg & (stateRequest ~= chargingState)** becomes true<br>**seconds** then, starting from rising edge of trigger, with a delay of at most **0.3 seconds**, **~negCh** |
| ☑ | Open Inverter Negative Contactor | At any point of time, if **~preC_RelayInv & (stateRequest ~= drivingState)** becomes true an<br>**seconds** then, starting from rising edge of trigger, with a delay of at most **0.3 seconds**, **~negInv** |

# Track Implementation and Verification



Requirements - crs_controller

View: Requirements

Search

| 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 | | |

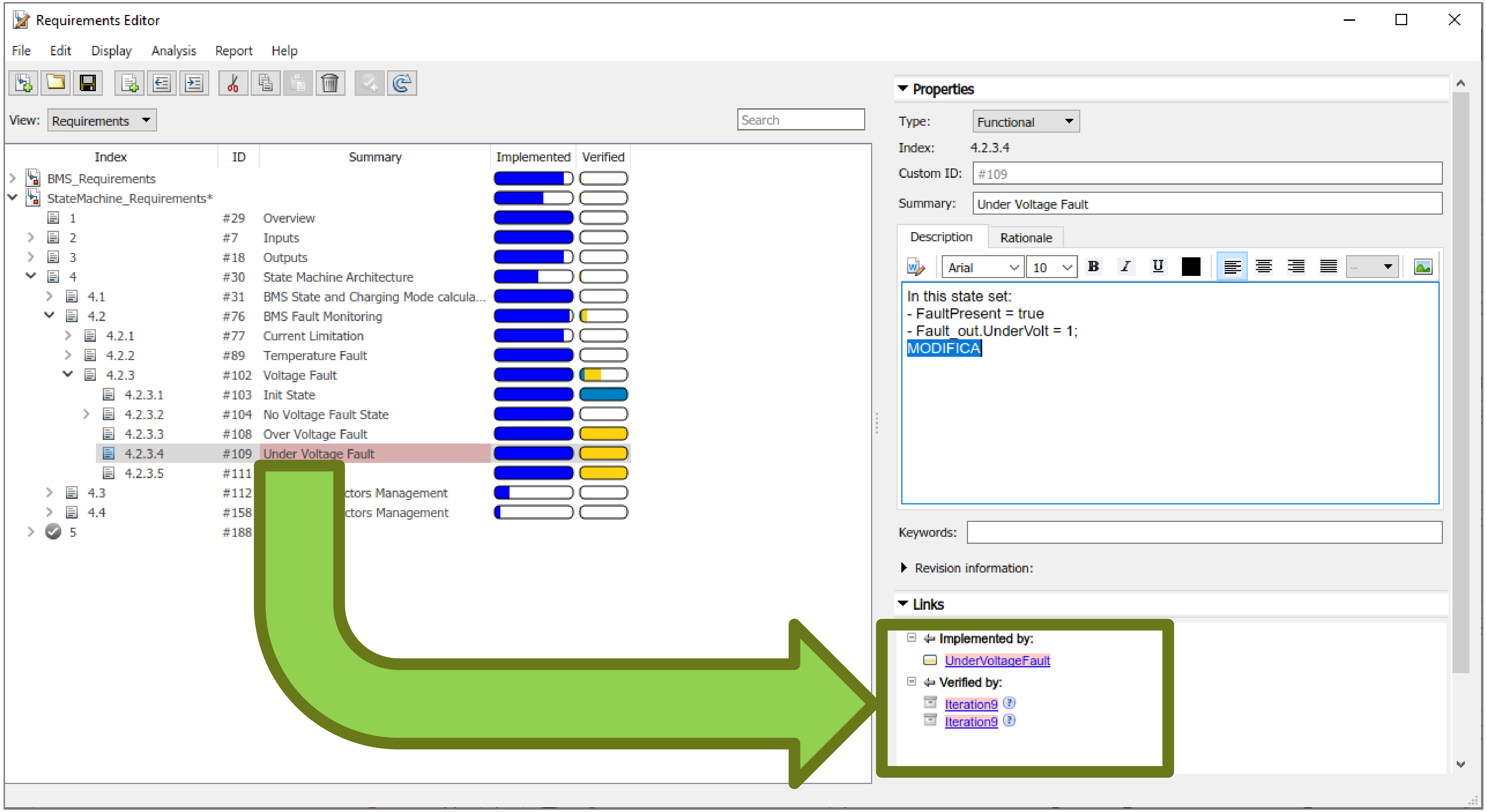
**Implementation Status**

- 🟦 Implemented
- ▨ Justified
- □ Missing

**Verification Status**

- 🟩 Passed
- 🟥 Failed
- 🟨 Unexecuted
- □ Missing

# Respond to Change – Impact Analysis



Requirements Editor

View: Requirements

| Index | ID | Summary | Implemented | Verified |
|---|---|---|---|---|
| BMS_Requirements | | | | |
| StateMachine_Requirements* | | | | |
| 1 | #29 | Overview | | |
| 2 | #7 | Inputs | | |
| 3 | #18 | Outputs | | |
| 4 | #30 | State Machine Architecture | | |
| 4.1 | #31 | BMS State and Charging Mode calcula... | | |
| 4.2 | #76 | BMS Fault Monitoring | | |
| 4.2.1 | #77 | Current Limitation | | |
| 4.2.2 | #89 | Temperature Fault | | |
| 4.2.3 | #102 | Voltage Fault | | |
| 4.2.3.1 | #103 | Init State | | |
| 4.2.3.2 | #104 | No Voltage Fault State | | |
| 4.2.3.3 | #108 | Over Voltage Fault | | |
| 4.2.3.4 | #109 | Under Voltage Fault | | |
| 4.2.3.5 | #111 | | | |
| 4.3 | #112 | ctors Management | | |
| 4.4 | #158 | ctors Management | | |
| 5 | #188 | | | |

Properties

Type: Functional

Index: 4.2.3.4

Custom ID: #109

Summary: Under Voltage Fault

In this state set:
- FaultPresent = true
- Fault_out.UnderVolt = 1;
MODIFICA

Keywords:

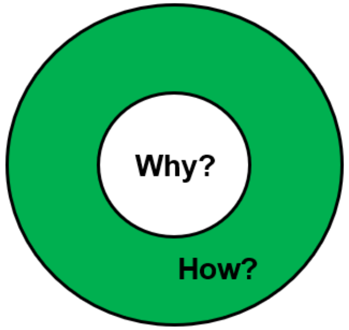
Revision information:

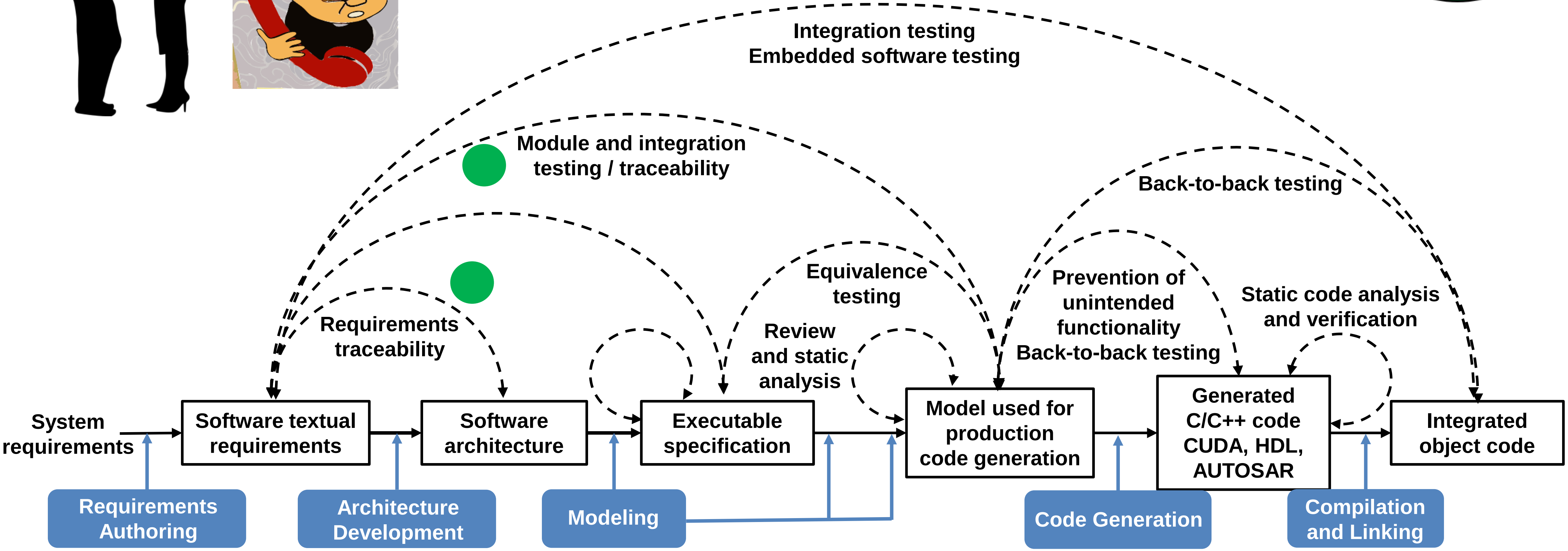
Links

Implemented by:
UnderVoltageFault

Verified by:
Iteration9
Iteration9

# Complete Model-Based Design with V&V



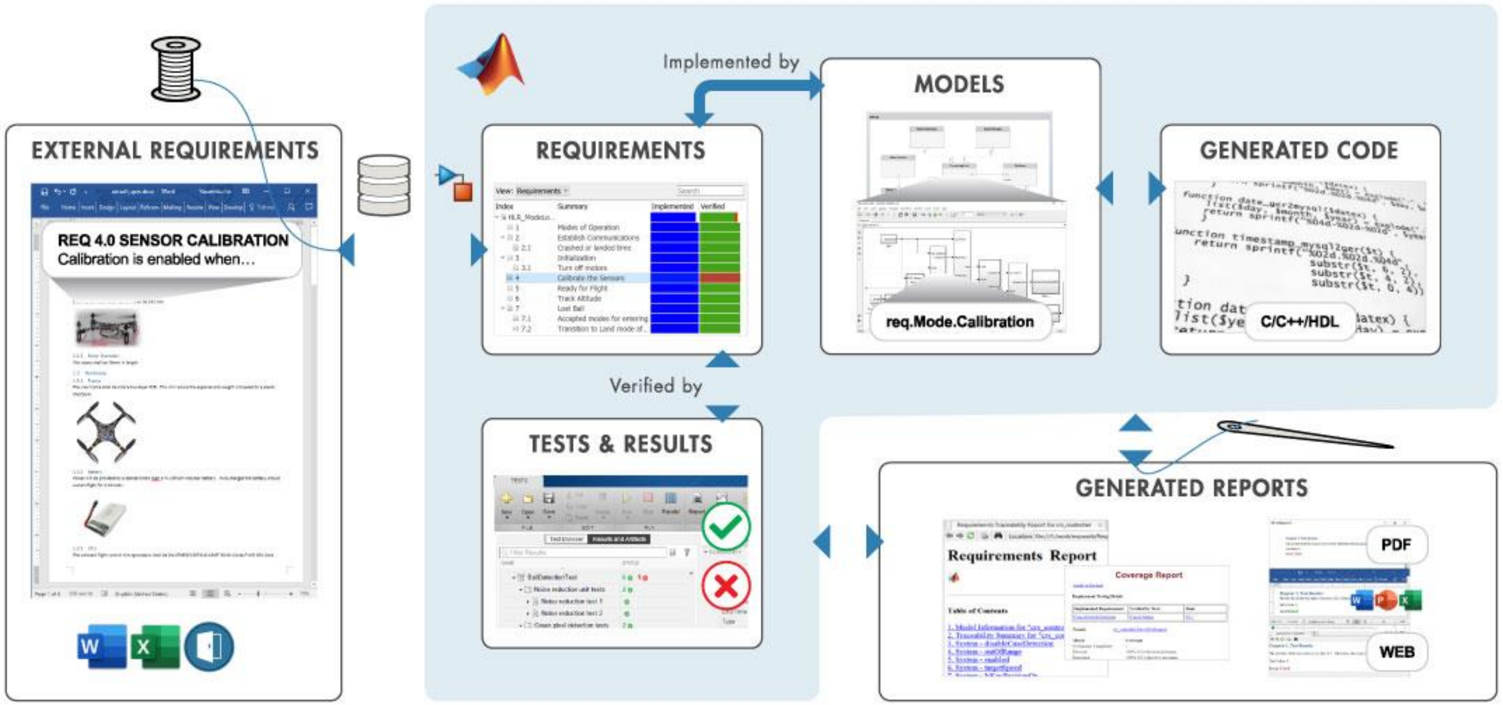


Integration testing
Embedded software testing
Module and integration testing / traceability
Back-to-back testing
Equivalence testing
Prevention of unintended functionality
Back-to-back testing
Static code analysis and verification
Requirements traceability
Review and static analysis
System requirements
Software textual requirements
Software architecture
Executable specification
Model used for production code generation
Generated C/C++ code CUDA, HDL, AUTOSAR
Integrated object code
Requirements Authoring
Architecture Development
Modeling
Code Generation
Compilation and Linking

# In summary, digital engineering practices can make standards compliance easier and reduce costs

- Seek single source of truth using digital thread
- Automate tasks that machines are good at
- Improve efficiency *without* skipping verification steps

DIGITAL THREAD: Traceability Between Requirements, Architecture, and Design



Thank you!

# Agenda

| | Topic |
|---|---|
| 14.15 | **Digital Transformation with MBSE, MBD and Early Verification** |
| | **Requirements Management and System Architecture Design** |
| | **Requirements-based Testing** |
| | **Q&A** |