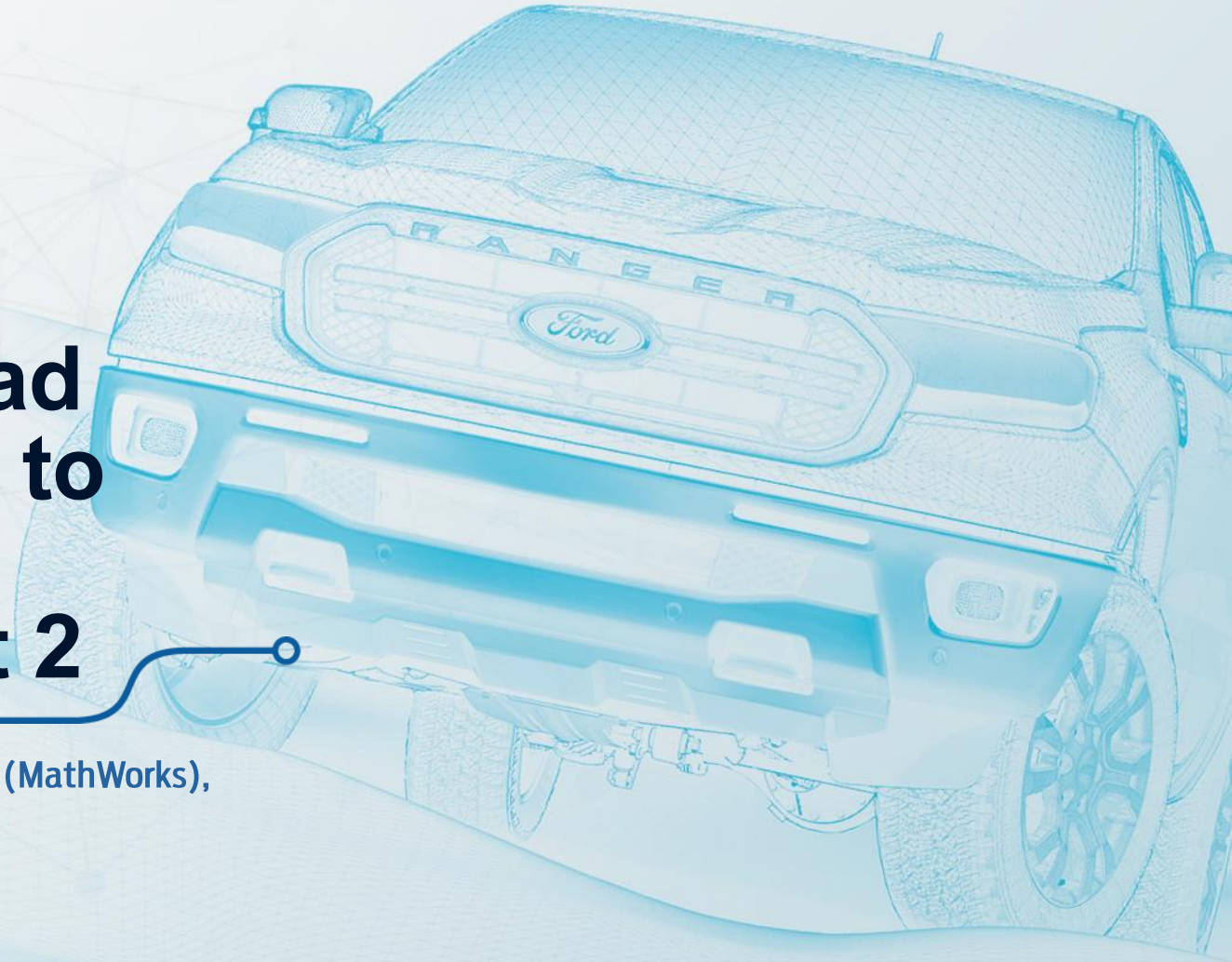
Vehicle Software & Electronic Solutions

Building the Digital Thread between MBSE and MBD to Meet ISO26262 for Embedded Software Part 2



Authors: Joshua McCready (Ford), Hans Gangwar (Ford), Josh Kahn (MathWorks), Eric Browning (Ford)

Part 1 Link

Ford

Problem Statement

Assessing ISO26262 Part 6 compliance for new and existing Ford In House software developed with Model Based Design software has demonstrated the need for additional best practices

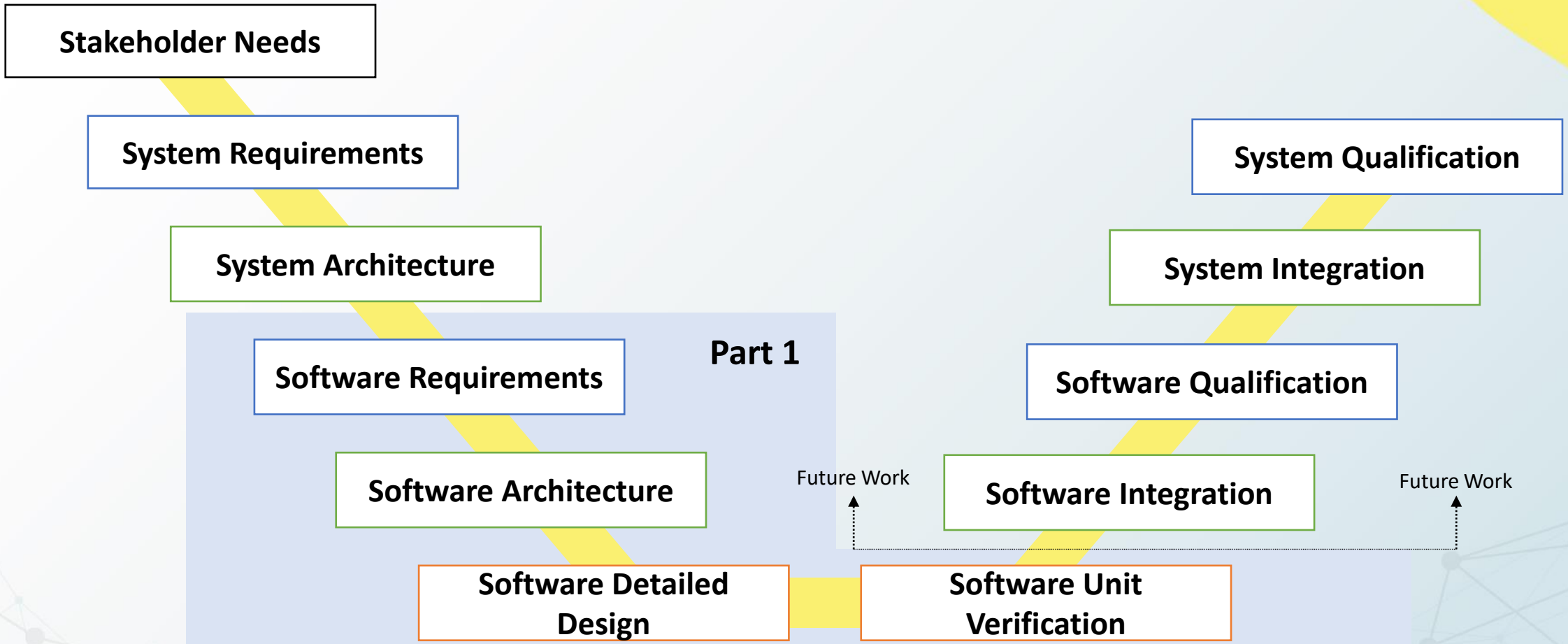
These best practices are needed to achieve connectivity to the System Engineering process and to allow for traceability and thread pulling of SW development artifacts*

Part 1 – SW requirements and design

Part 2 - SW Integration and Qualification traceability & Implementation

*System and software requirements, model and data dictionary, implementation, test cases

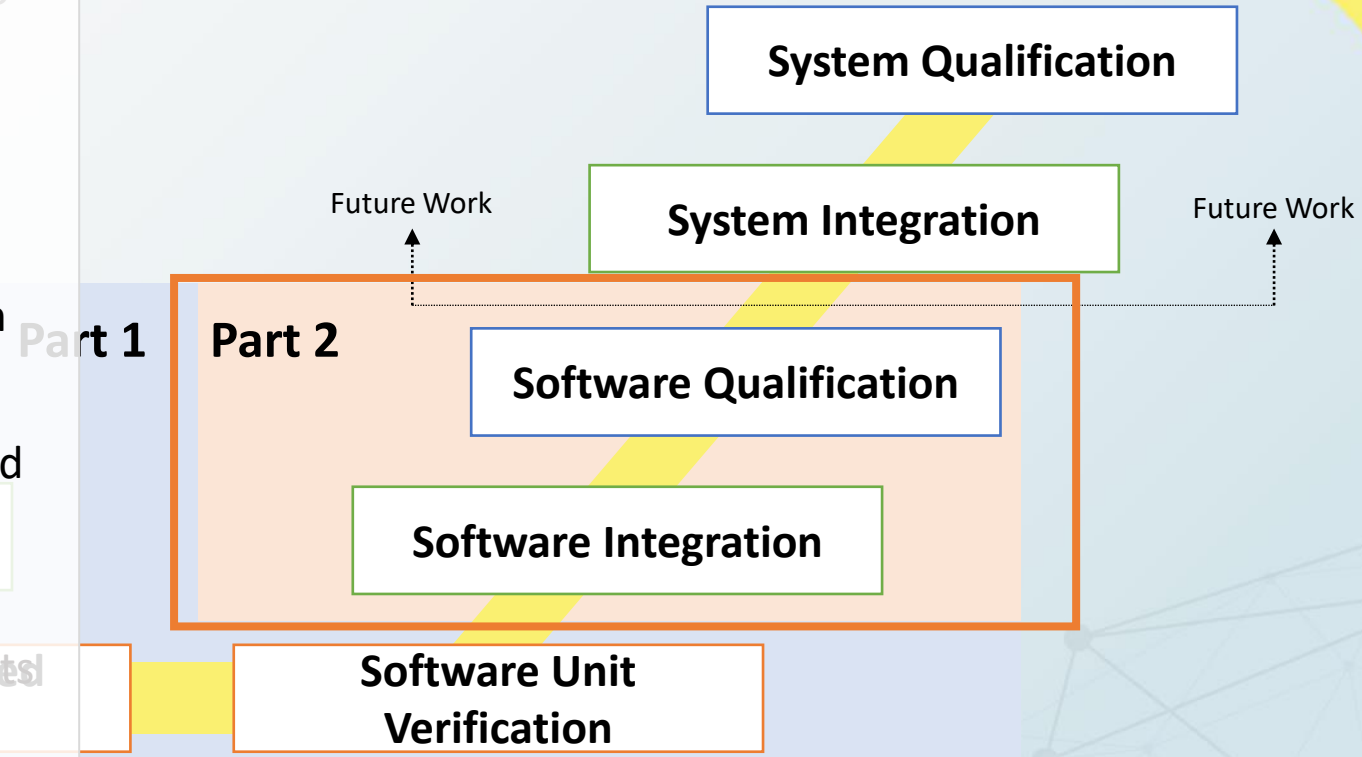
Process Overview – Scope of Part 1



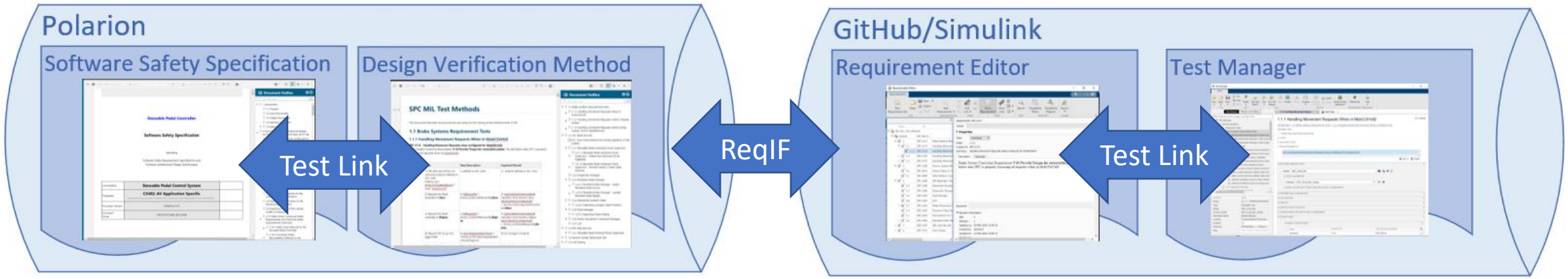
Process Overview – Scope of Part 2

Next Steps

- Create Design Verification Methods
- Link test cases to Design Verification Methods
- Create the Software Integration and Qualification Test Suites
- Identify dependencies of software integration and qualification testing and how to establish traceability across the project artifacts



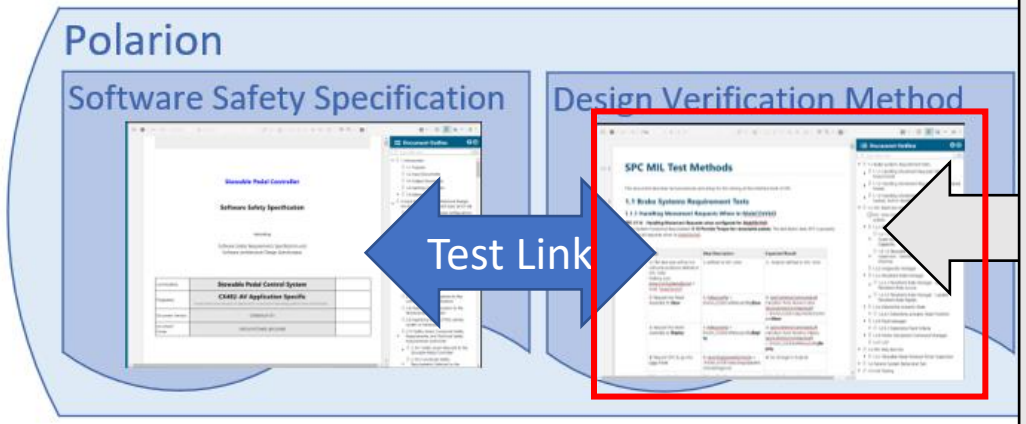
Process Overview – Software Integration – Traceability



Requirement changes impact on test methods ("Suspect Links")

Failed Tests impact on requirements (JIRA)

Process Overview – Software Integration – Create DVMs



Requirement changes im

Failed Tests

Polarion - Design Verification Method

SPC-2189 - When Motor demand Faults while the pedal assembly is moving then the Modify Motor Demand shall set the Motor Command to stop (8 iterations)

Test Steps

Step	Step Description	Expected Result
1) This test case will be run with preconditions defined in SPC-2093.	1) Defined in SPC-2093	1) Outputs defined in SPC-2093
2) Request the Pedal Assembly to: a) **Stow** b) **Deploy**	2) PldAsyRq a) ENUM_CODE1:ePdlAsyActRq**Stow** b) ENUM_CODE3:ePdlAsyActRq**Deploy**	2) The Motor Command Left transition as described below: a) SpcOutMotorCommandLeft transition from None to **Stow** SpcOutMotorCommandLeft -> ENUM_CODE1:eSpcMotorCommand**Stow** b) SpcOutMotorCommandLeft transition from None to **deploy** SpcOutMotorCommandLeft -> ENUM_CODE3:ePdlAsyActRq**Deploy**
3a&b) Input current profile to trigger the faults described in step 4	3a&b) TBD	3) a) SpcOutMotorCommandLeft remains ENUM_CODE1:eSpcMotorCommandStow b)SpcOutMotorCommandLeft remains ENUM_CODE3:ePdlAsyActRq**Deploy**

Work Item Properties

SPC-2189 - When Motor demand Faults while the pedal assembly is moving then the Modify Motor Demand shall set the Motor Command to stop (8 iterations)

Fields

*Severity: **Basic**

Links

Links

Edit Links

relates to

SPC-2214 - Modify Motor Demand Faulted Stop (8 iter

verifies

SPC-2205 - Modify Motor Demand Faulted Stop

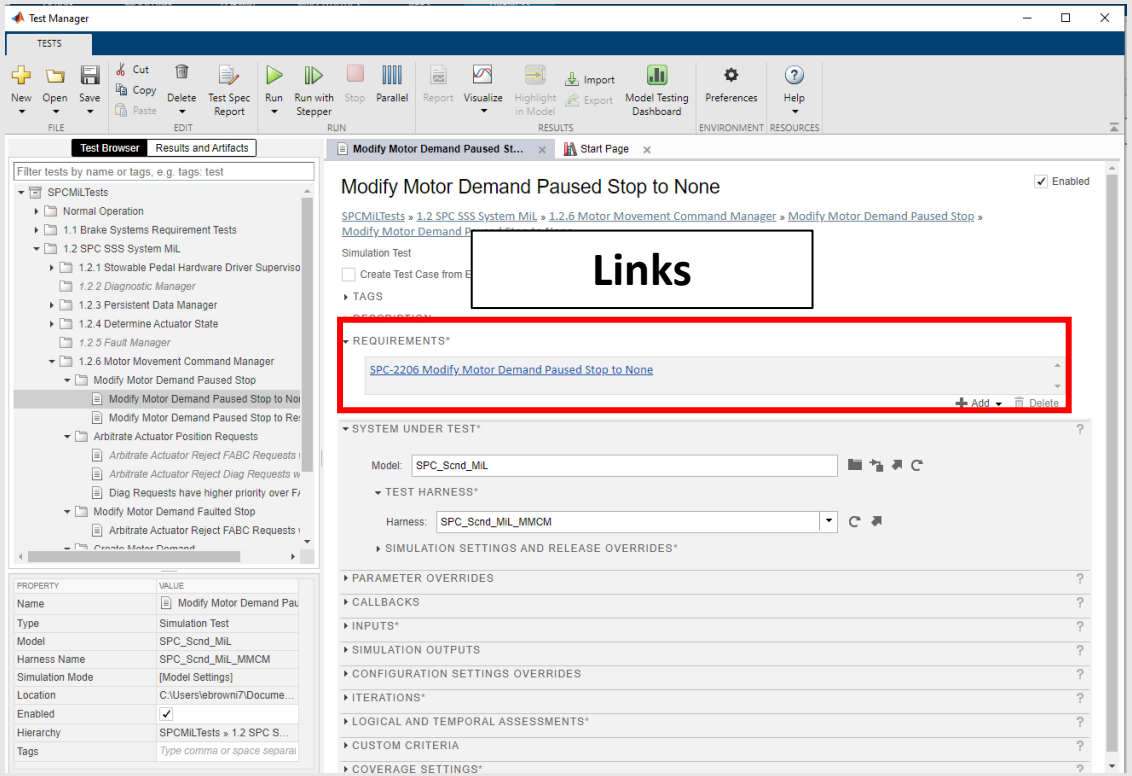
is related to

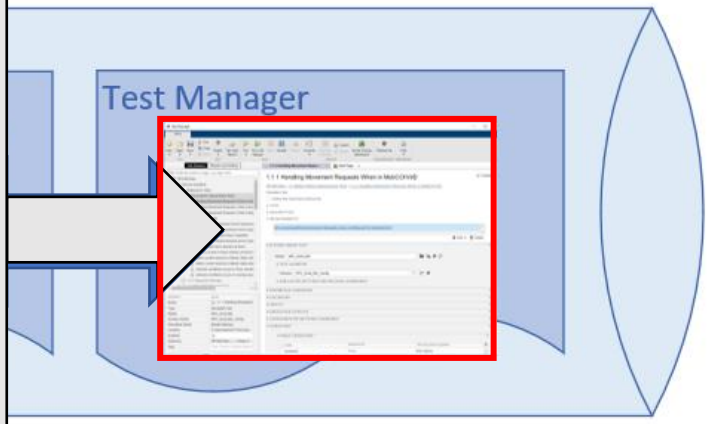
ABCS_Jira-371 - Spc v6 Bug - Remove Fault Status 5 fro

Process Overview – Software Integration – Test Manager

Vehicle Software & Electronic Solutions

Simulink - Test Manager





ks")

Process Overview – Software Integration - Test Manager

Test Manager Test Types

Baseline (Current)

Compares the outputs from a simulation against the expected outputs capture in Excel file or a MAT-file.

Pros
- Quickly identify change/regression in model outputs

Cons
- Not adaptive to model changes (PARAM/Configs/Cal)
- Requirement/model changes require re-running baseline and visually inspecting

Simulation (New)

Assess model timing and event ordering by authoring and including temporal assessments with test cases

Pros
- Hybrid systems with discrete and continuous time behavior can require complex timing-dependent signal logic
- Removes the need to visually inspect/update baselines
- Adaptive to most requirement/model changes
- Iterations within one test

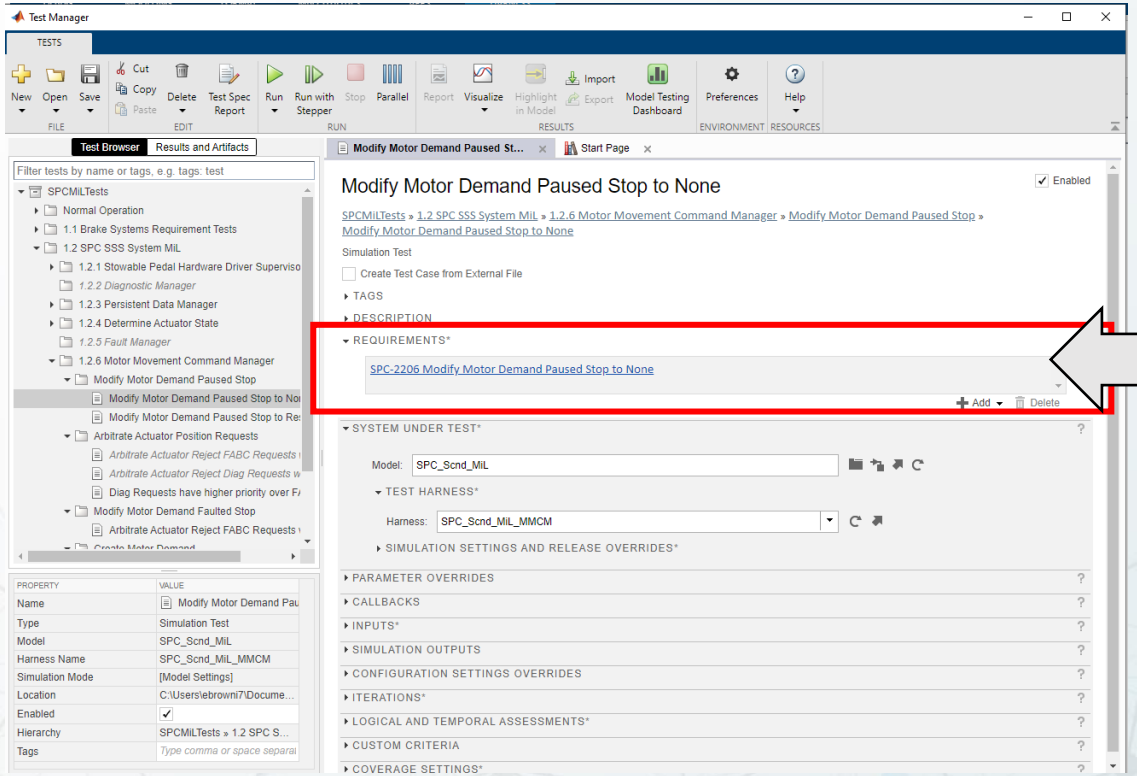
Cons
- Front loads validation
- Based on implementation, potential for missing bug identification (no visual inspection)

Process Overview – Software Integration - Test Manager

Simulation - Requirements

- Link test method Stored in Polarion
- Trace Failed tests to JIRA Bug/Issue

Simulink - Test Manager



Polarion - DVM

SPC-2206 - Modify Motor Demand Paused Stop to None

When Motor demand Pause Stop Triggers for longer than Pause time out, while the pedal assembly is moving then the Modify Motor Demand shall set the Motor Command to stop (8 iterations)

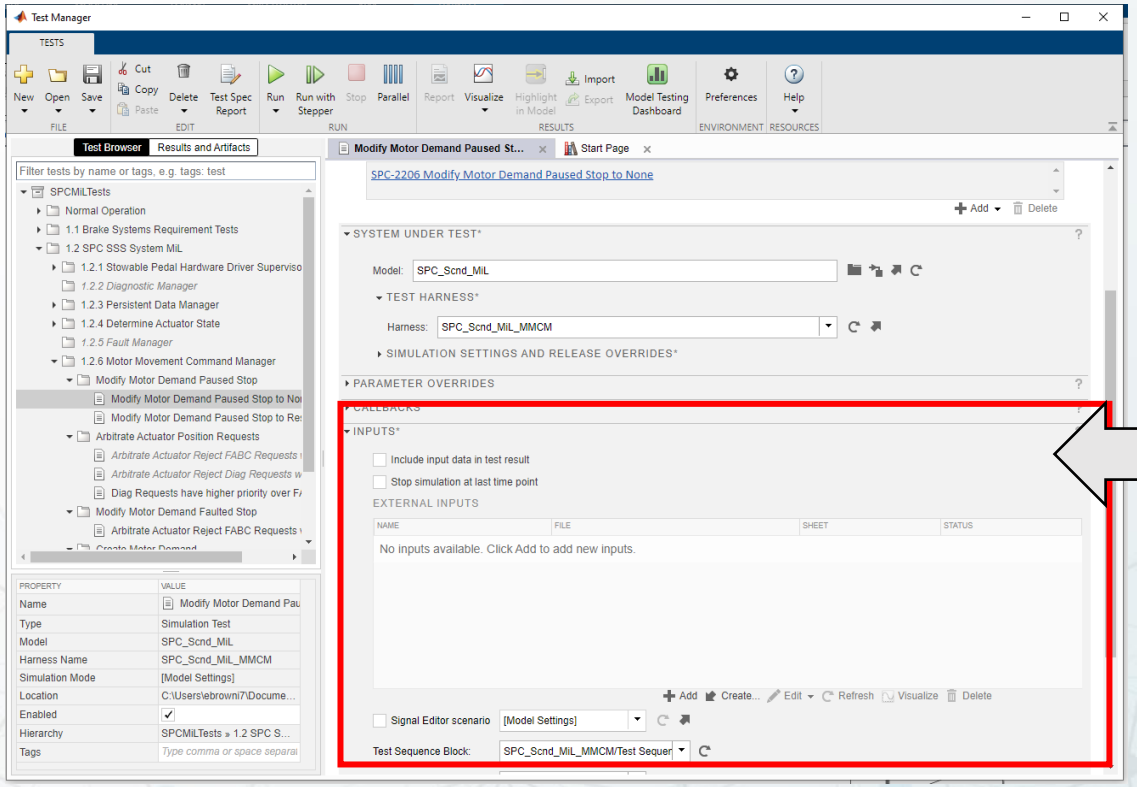
Step	Step Description	Expected Result
1) This test case will be run with preconditions defined in SPC-2093.	1) Defined in SPC-2093	1) Outputs defined in SPC-2093
2) Request the Pedal Assembly to: a) **Stow** b) **Deploy**	2) PldAsyRq a) ENUM_CODE1:ePdlAsyActRq**Stow** b) ENUM_CODE3:ePdlAsyActRq**Deploy**	2) The Motor Command Left transition as described below: a) SpcOutMotorCommandLeft transition from None to **Stow** SpcOutMotorCommandLeft -> ENUM_CODE1:eSpcMotorCommand**Stow** b) SpcOutMotorCommandLeft transition from None to **deploy** SpcOutMotorCommandLeft -> 040:ENUM_CODE2:eSpcMotorCommandDeploy
3a&b) Input current profile	3a&b) TBD	3) a) SpcOutMotorCommandLeft remains ENUM_CODE1:eSpcMotorComma

Process Overview – Software Integration - Test Manager

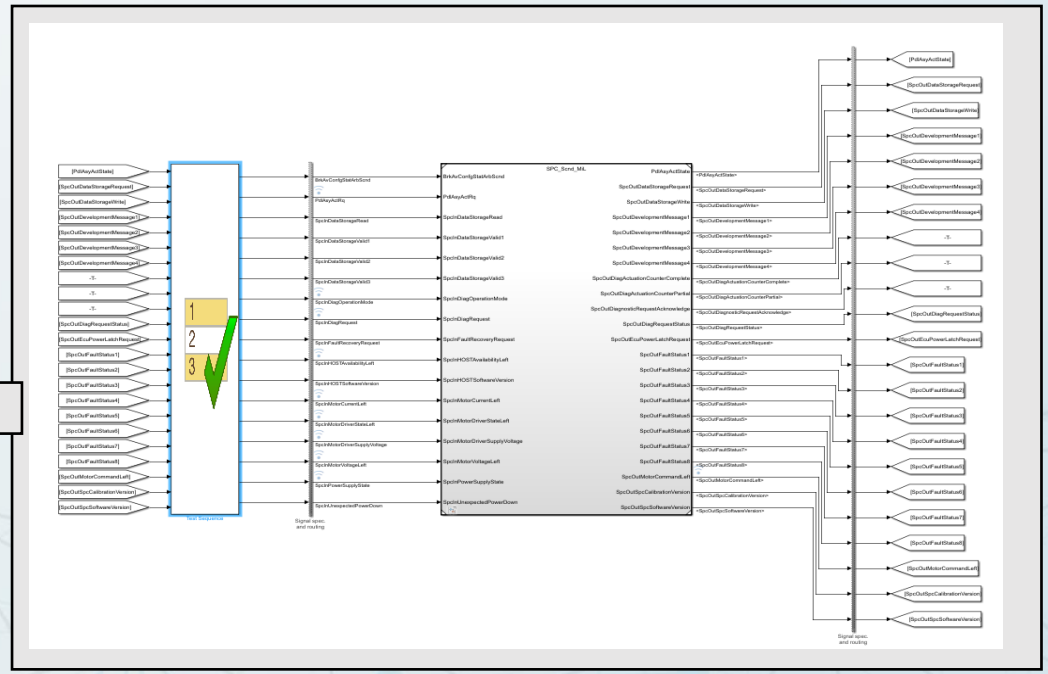
Simulation - Inputs

- Sequence editor allows for I/O closed loop feedback of function under test
- Use of refence model allows for ease of updating to multiple releases

Simulink - Test Manager



Model Test Harness

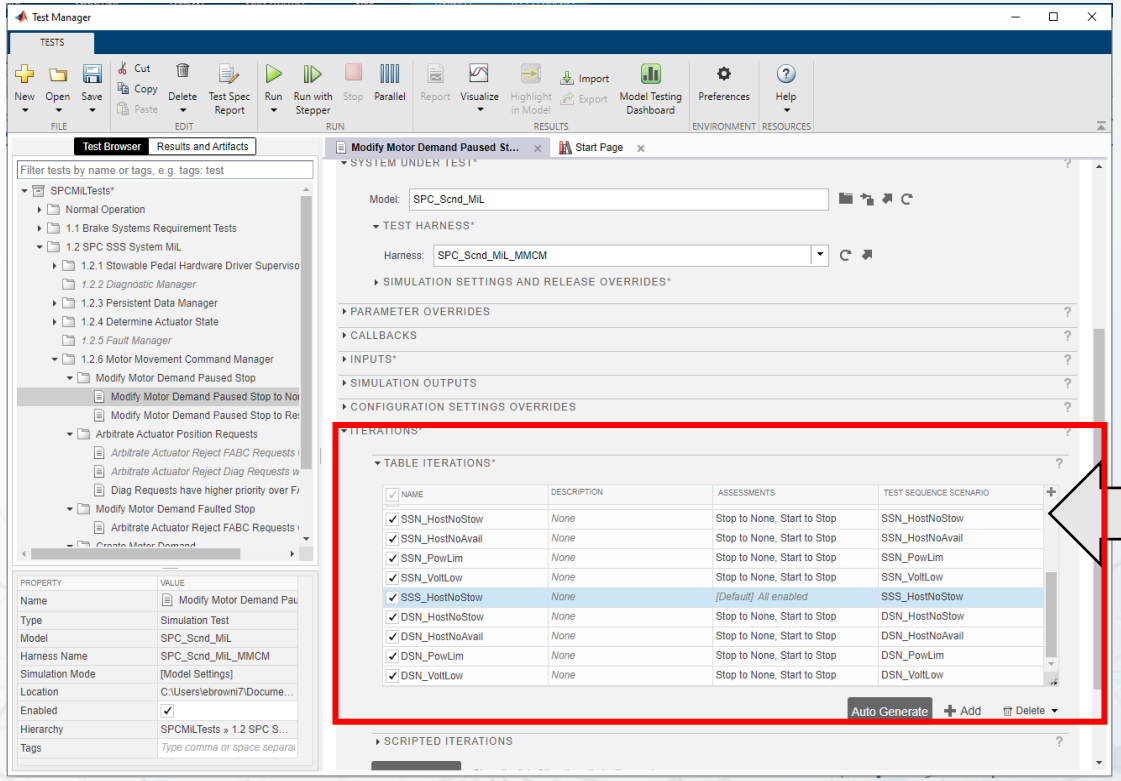


Process Overview – Software Integration - Manager

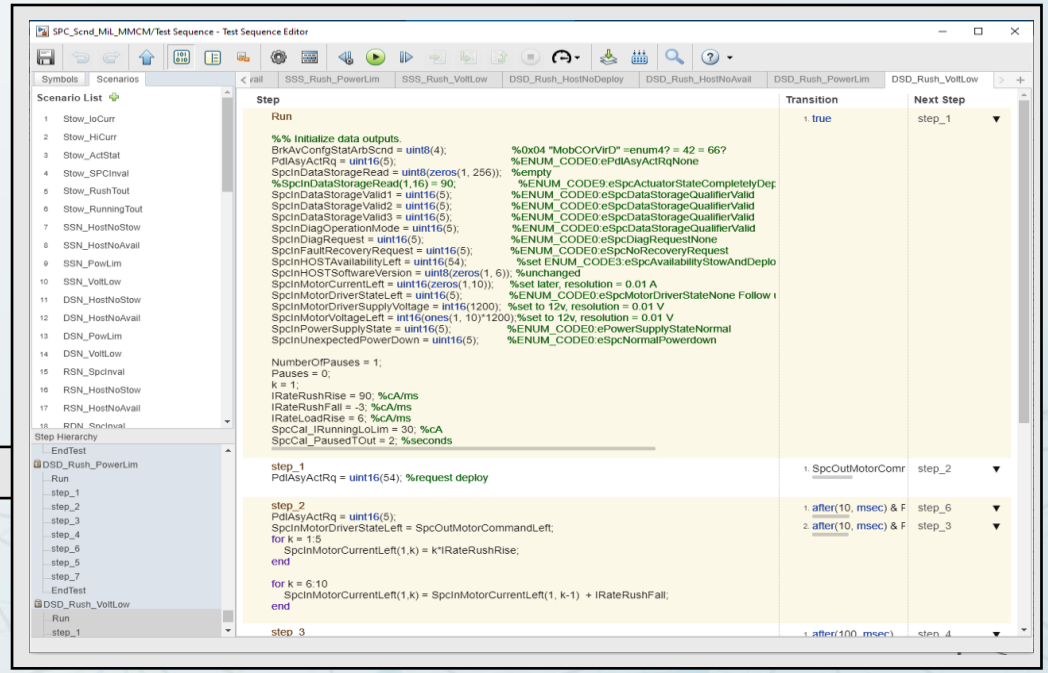
Simulation - Test Sequence/Iterations

- Run multiple iteration changing input parameters in a single test
- Link Model based Trigger & Temporal based Transitions (i.e. Model Cals, I/O)
- Apply specific assessment criteria to specific iterations

Simulink - Test Manager



Test Sequence/Iterations

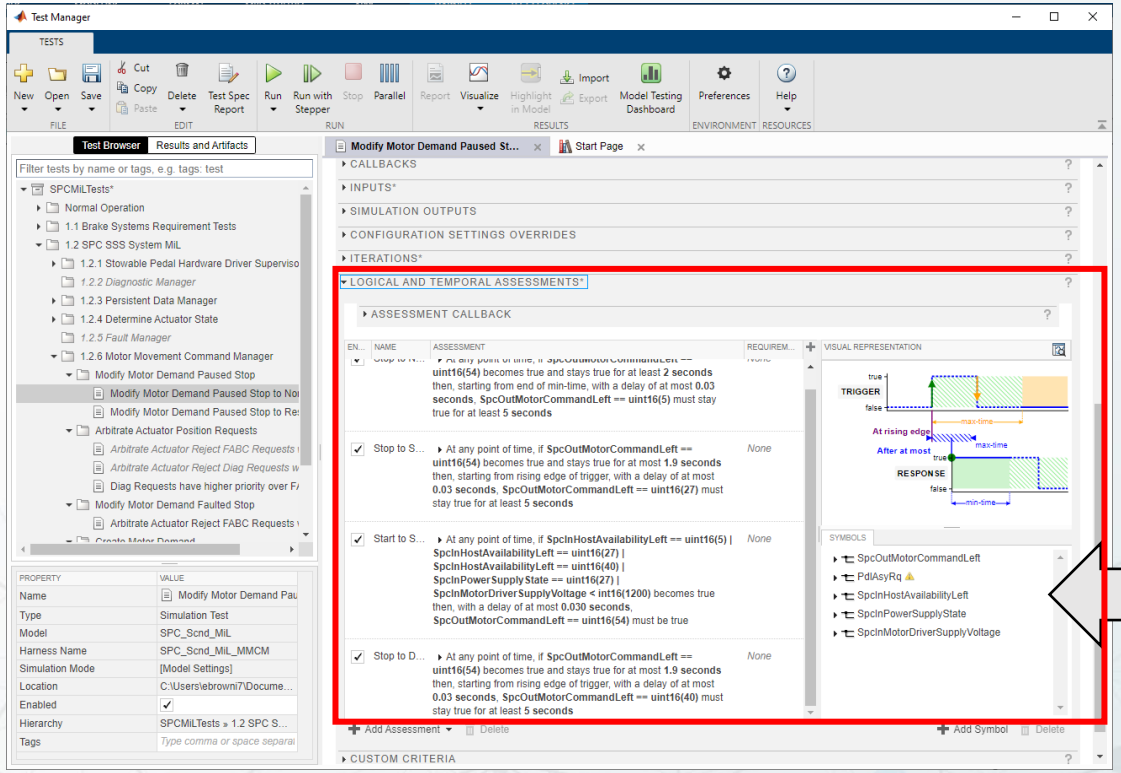


Process Overview – Software Integration – Test Manager

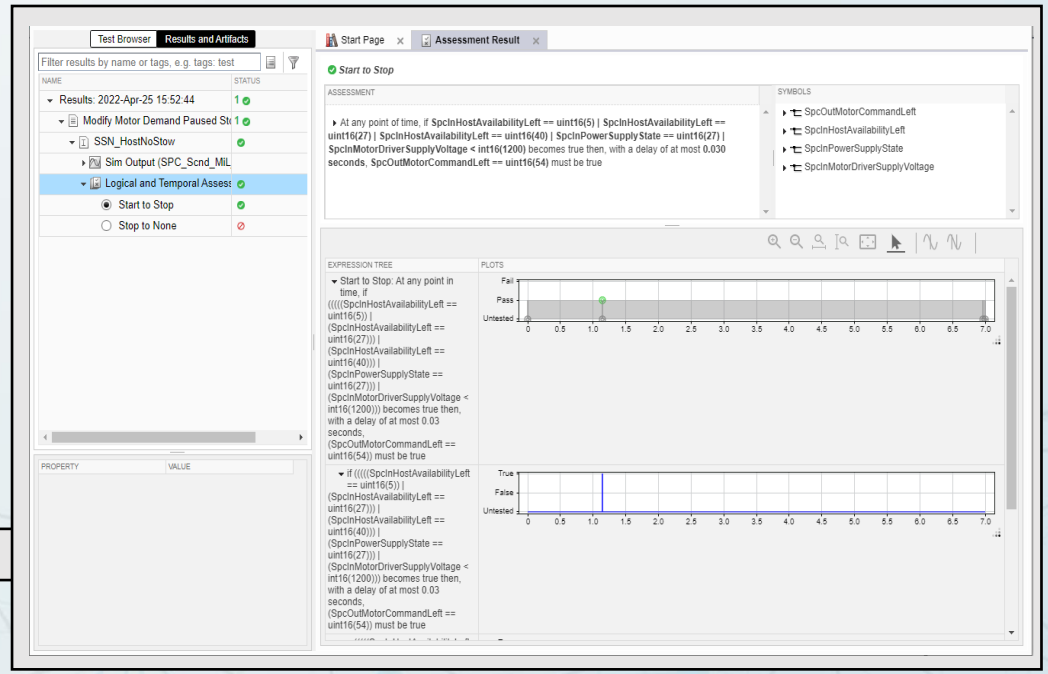
Simulation - Logical and Temporal Assessments

- Create triggered response assessments to be applied to multiple iterations
- Dynamic assessments criteria automates the "Assessment"
- Link Assessments to Model parameters (i.e. Model Cals, I/O)

Simulink - Test Manager



Logical and Temporal Assessments



Conclusion and Lessons Learned

Vehicle Software & Electronic Solutions

Conclusion

- Identified a Traceability method from Software requirement to implemented DVMs/bugs
- Linking used to trace closure of bugs and impact analysis of updated requirements on DVMs
- Simulink Test Manager – Simulation Tests creates an adaptive environment to programmatically implement DVMs

Lessons Learned

- Full top-down decomposition of requirements creates a Higher lvl Software Implementation requirement that can be tested at a function lvl
- Resources/Program delays allowed for time allocation on a process focused, continuous improvement activity

Next Steps

Polarion

- Ongoing work with Siemens for full functionality of *Test Steps*
- Linking of HIL/Vehicle lvl DVM and Test results
- Resolve Polarion -> JIRA link (currently uni-directional, not bi-directional)
- Compatibility/transition to JIRA Cloud

Simulink

- Investigate SIL implementation of *Logical and Temporal Assessments*
- Potentially re-run Simulink Test manager test suite on HIL using vector tools
- Collaborate with Ford VC Core/IT to automatically run tests when new model is pushed to GitHub

Thank you for joining us today.

Please direct any follow-up questions to:

Eric Browning
Ebrowni7@ford.com

Joshua McCready
jmccrea8@ford.com