MATLAB EXPO 2019

Simplifying Requirements Based Verification with Model-Based Design

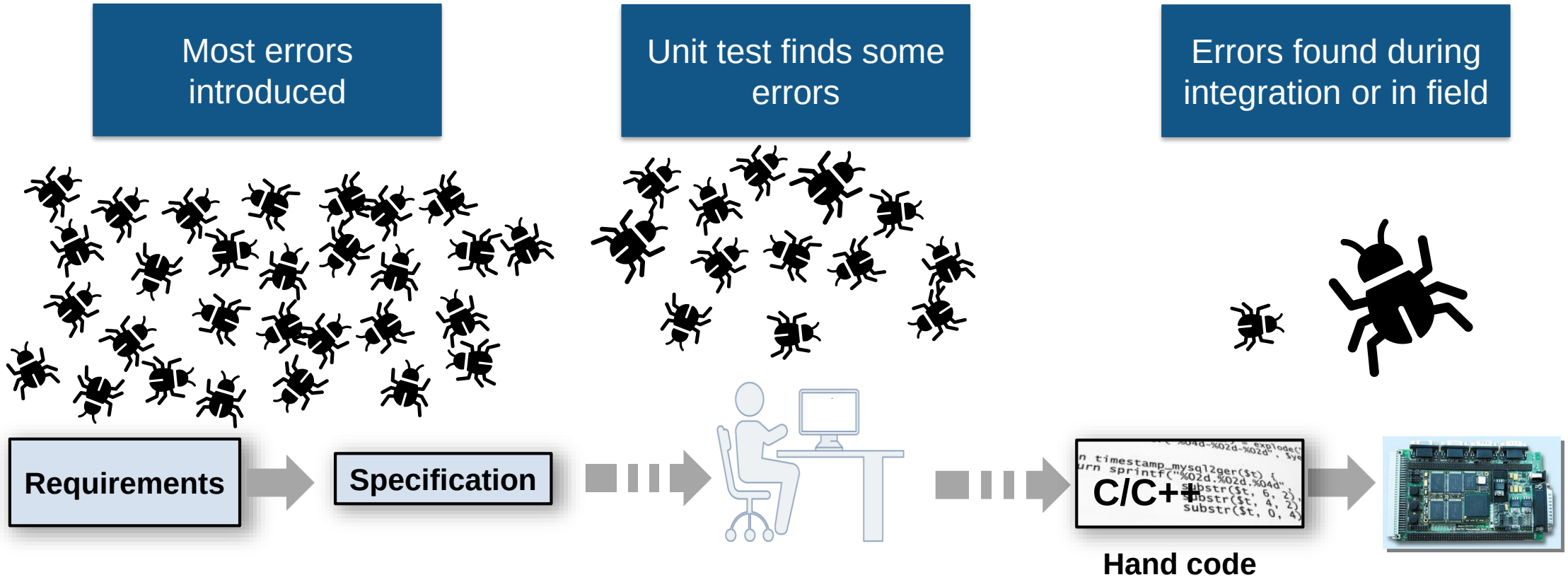
Dr. Tjorben Groß, Senior Application Engineer

Key takeaways

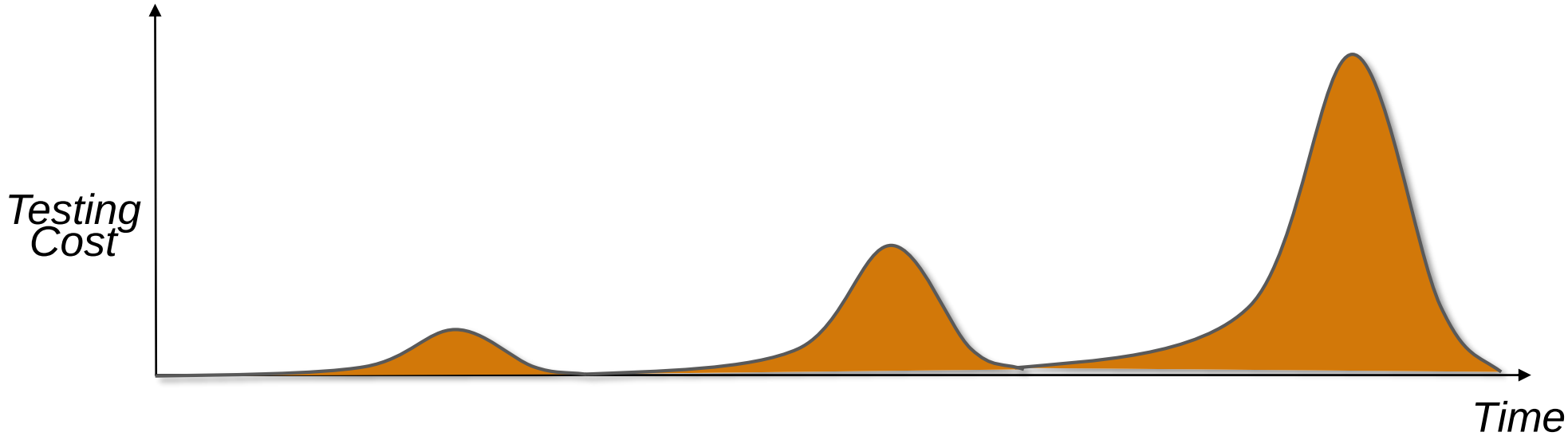
- Verify and validate requirements earlier
- Identify inconsistencies in requirements by using unambiguous assessments
- Traceability from requirements to design and test

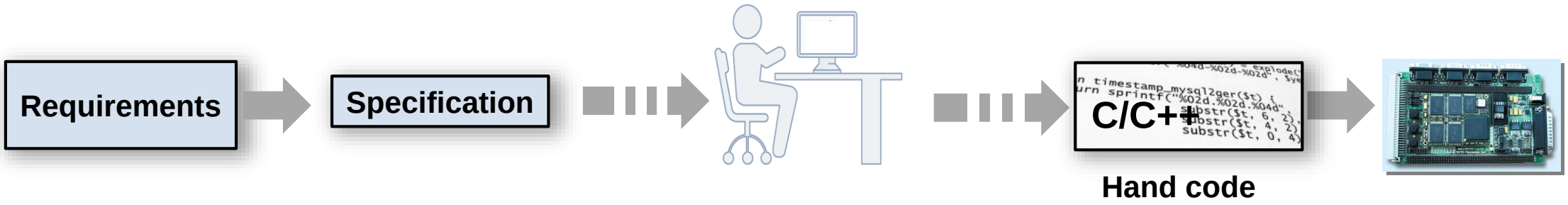
> *"By enabling us to analyze requirements quickly, reuse designs from previous products, and eliminate manual coding errors, Model-Based Design has reduced development times and enabled us to shorten schedules to meet the needs of our customers."*
> *- MyoungSuk Ko, LS Automotive*

Challenge: Errors introduced early but found late



Cost of finding errors increases over time





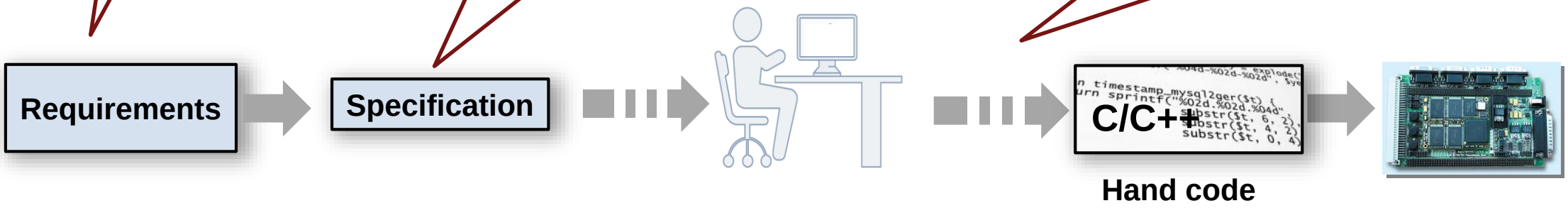
Challenges with requirements-based verification

Are **all** requirements implemented?

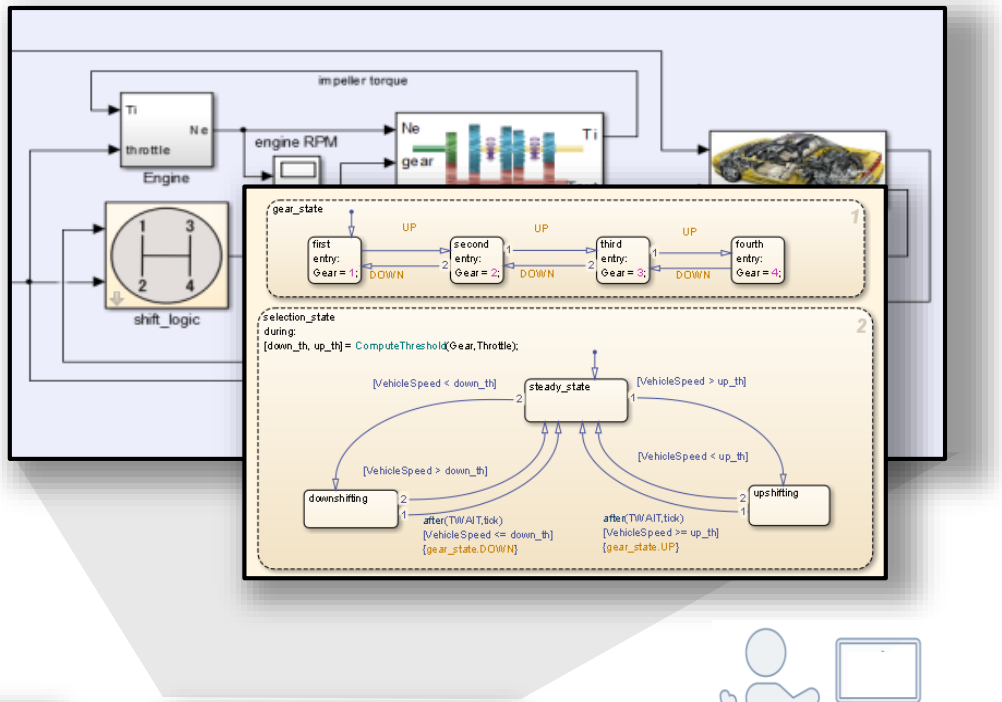
Is requirement interpreted correctly?

Is the implementation functioning correctly?

How to avoid modifying the design for test?

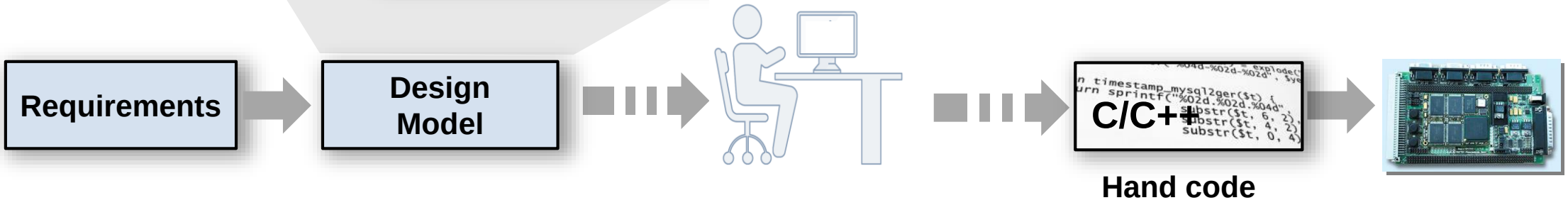


Simulink models for specification

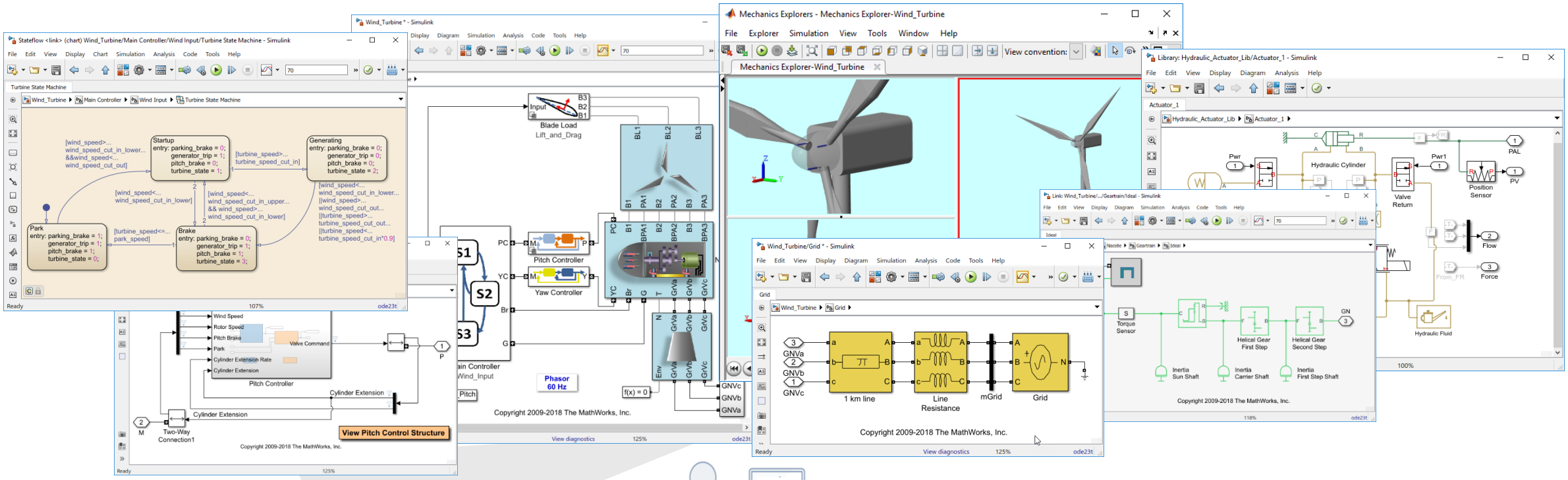


Model-Based Design enables:

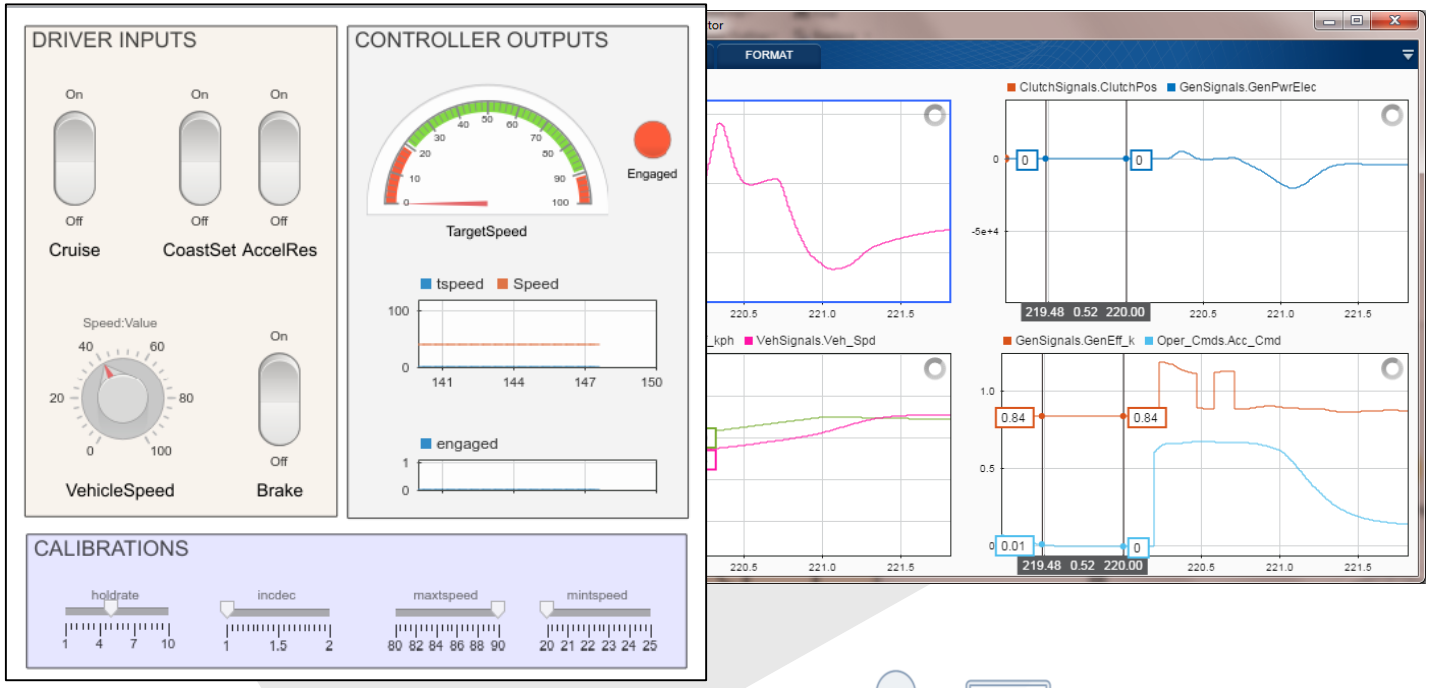
- *Early testing to increase confidence in your design*
- *Delivery of higher quality software throughout the workflow*



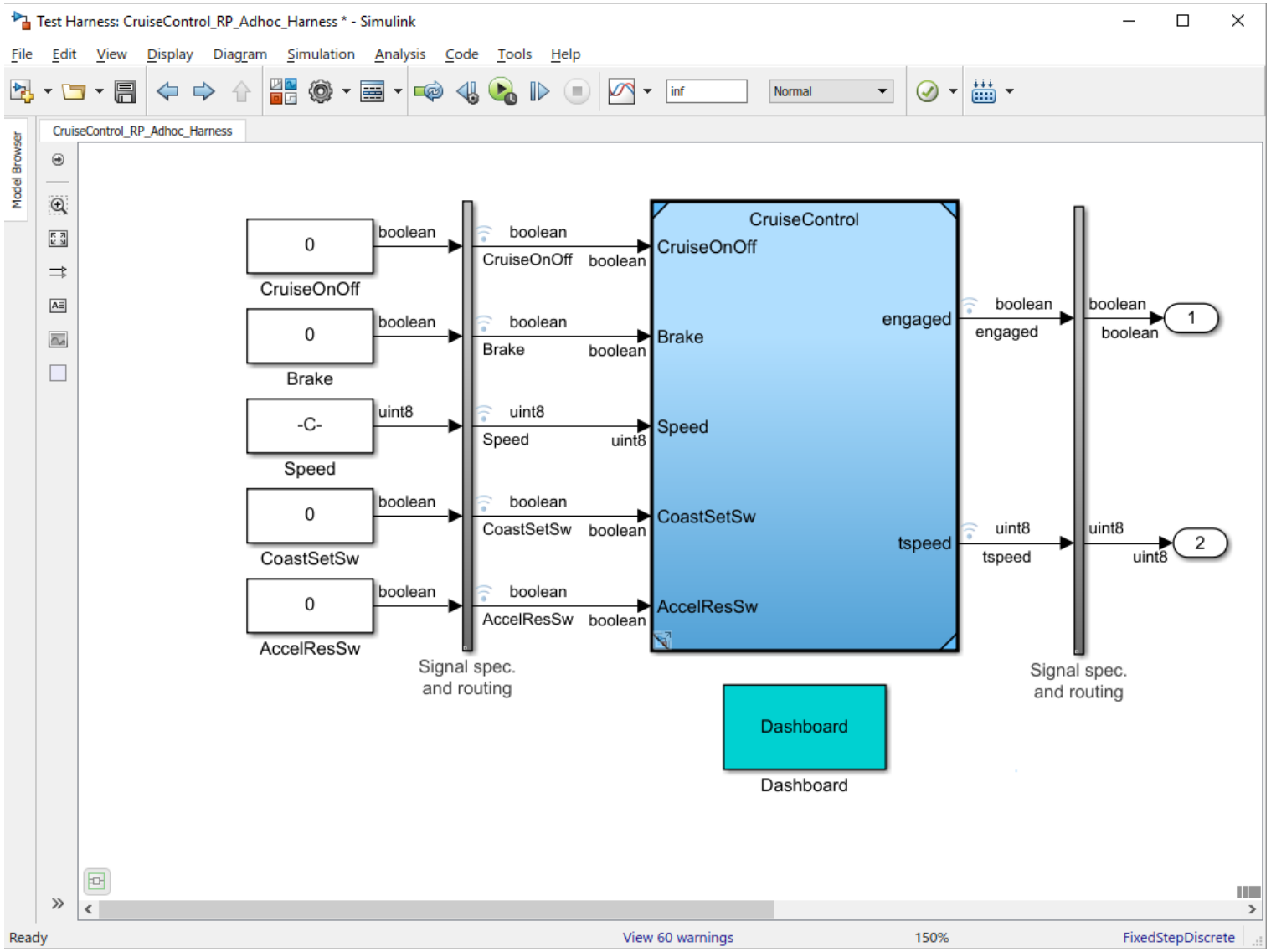
Multiple languages to describe complex systems



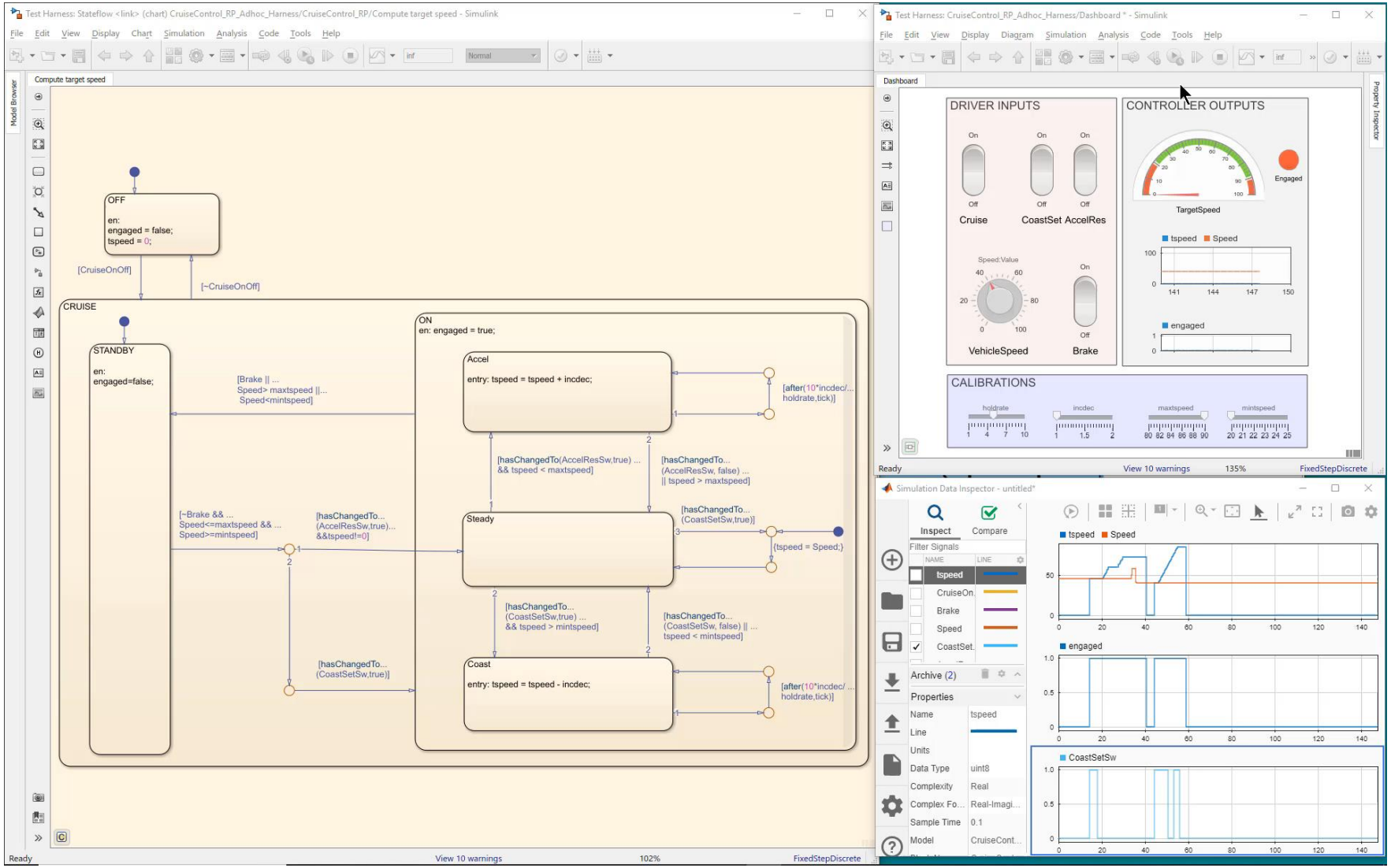
Ad-Hoc Testing: Explore behavior and design alternatives



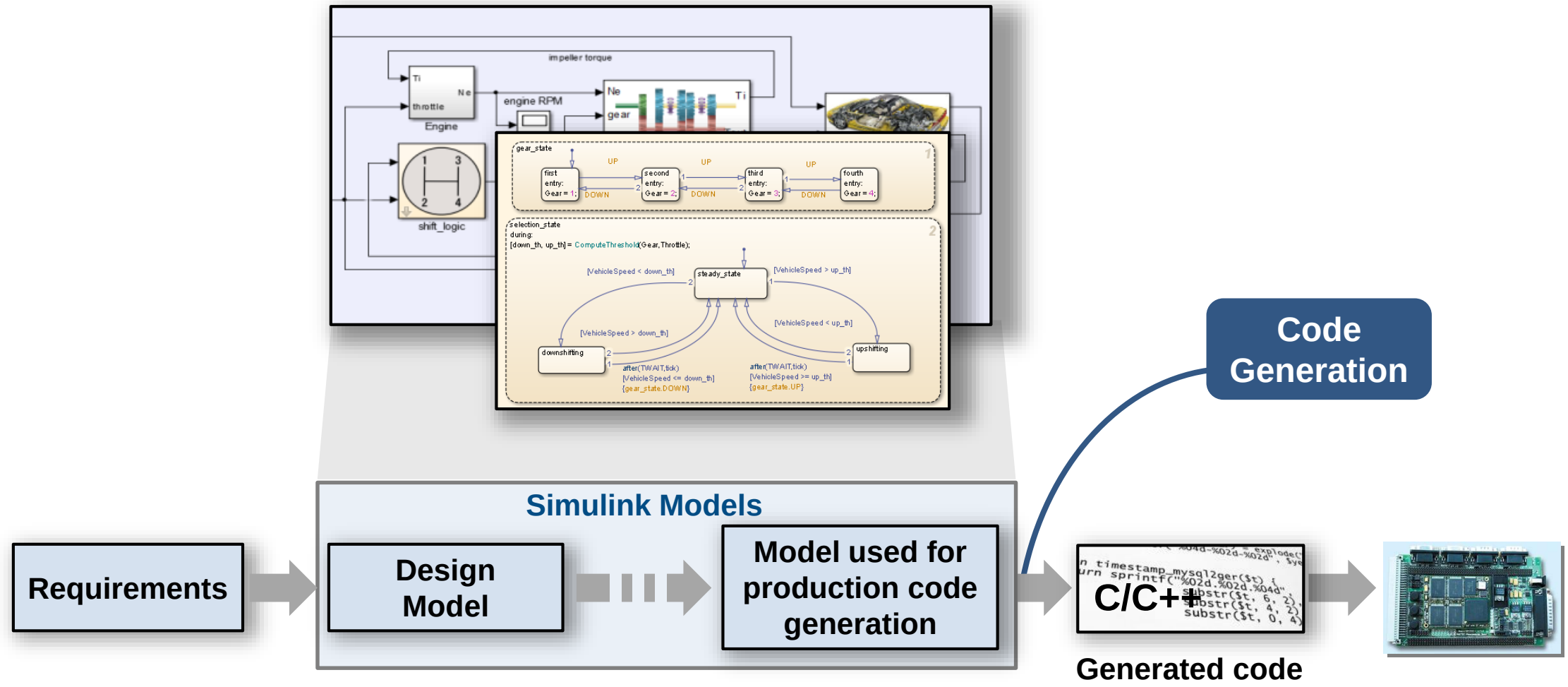
Validate behavior earlier with simulation



Validate Behavior Earlier with Simulation



Complete Model Based Design



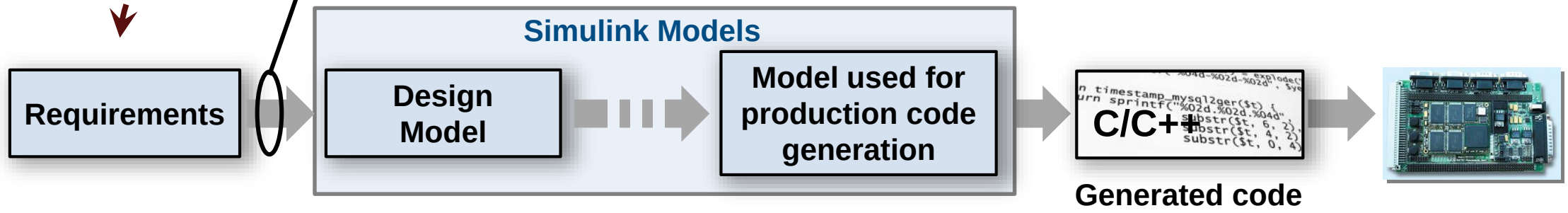
Systematically verify requirements

Are all requirements implemented?

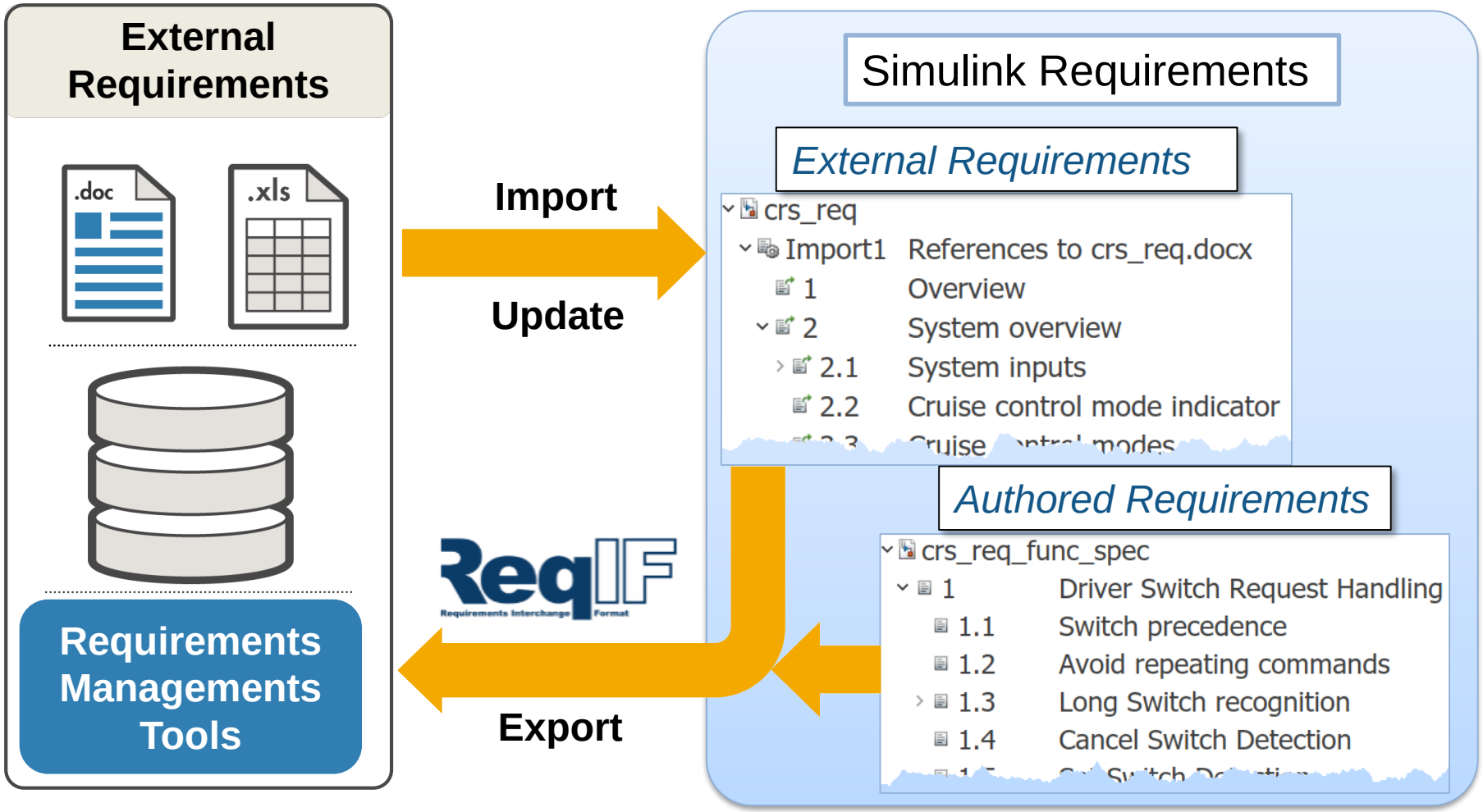
Is the implementation functioning correctly?

Are designs and requirements consistent?

Requirements Based Testing

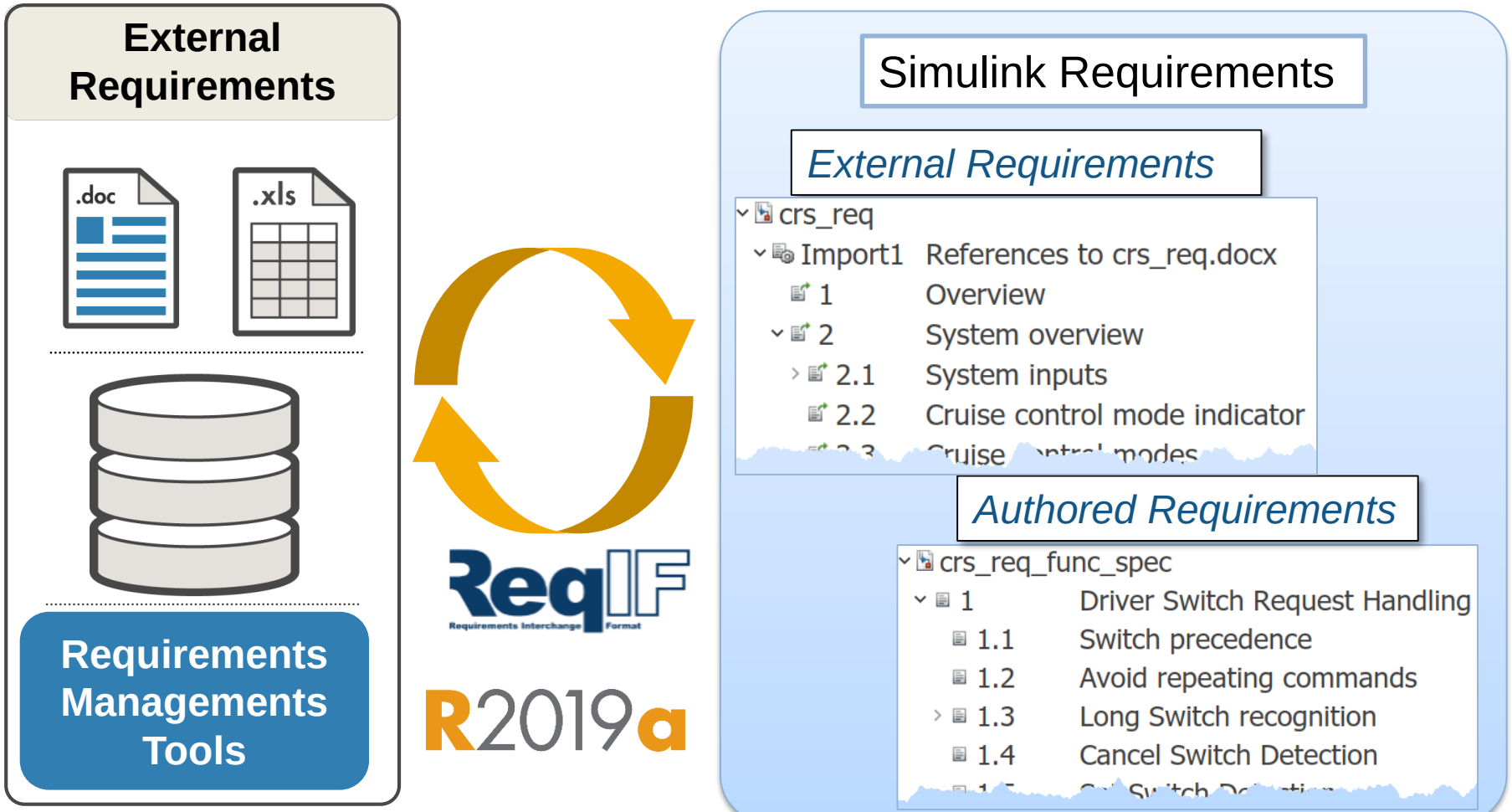


Integrate with requirements tools and author requirements



- Import from:
  - Word / Excel
  - IBM® Rational® DOORS®
  - ReqIF™ standard
- Update synchronizes changes from source
- Edit and add further details to import R2019a
- Author requirements
- Export ReqIF R2019a
  - Enables roundtrip with external tools

Roundtrip workflow with external tools thru ReqIF



- Import from:
  - Word / Excel
  - IBM® Rational® DOORS®
  - ReqIF™ standard
- Update synchronizes changes from source
- Edit and add further details to import R2019a
- Author requirements
- Export ReqIF R2019a
  - Enables roundtrip with external tools

Requirements Verification with Simulink

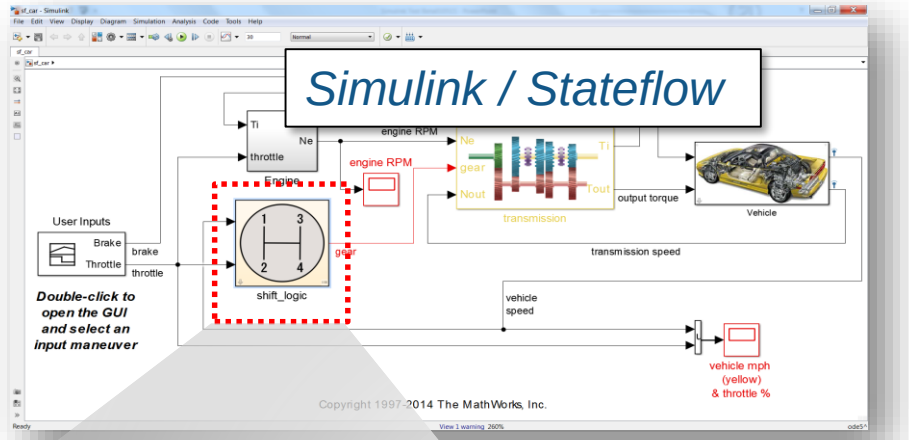
Requirements

TransmissionReq

1	Transmission Operating Modes
1.1	Reverse cannot be entered from drive
1.2	Engine only starts in Park

Implemented By

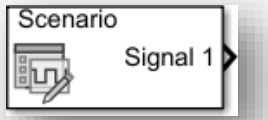
Verified By



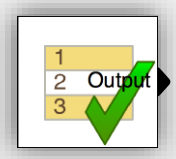
Test Case

Inputs

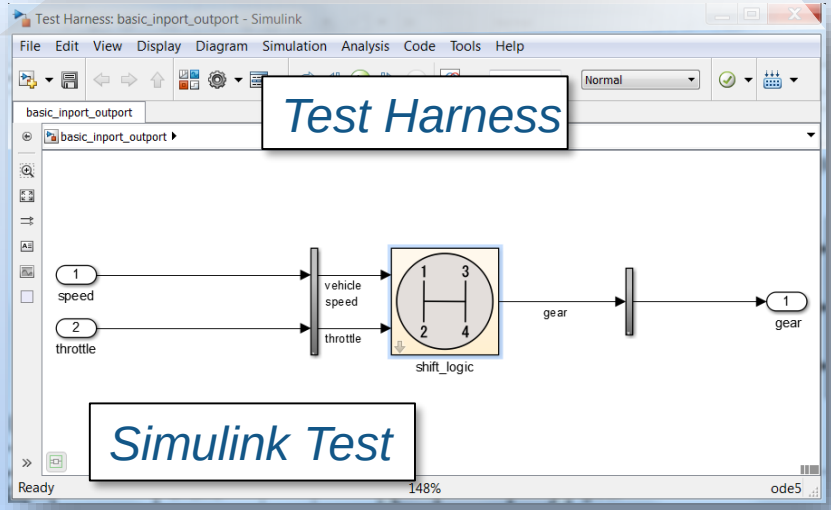
MAT / Excel file (input)



Signal Editor

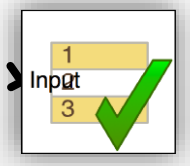


Test Sequence

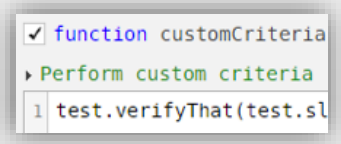


Assessments

MAT / Excel File (baseline)



Test Assessments



MATLAB Unit Test

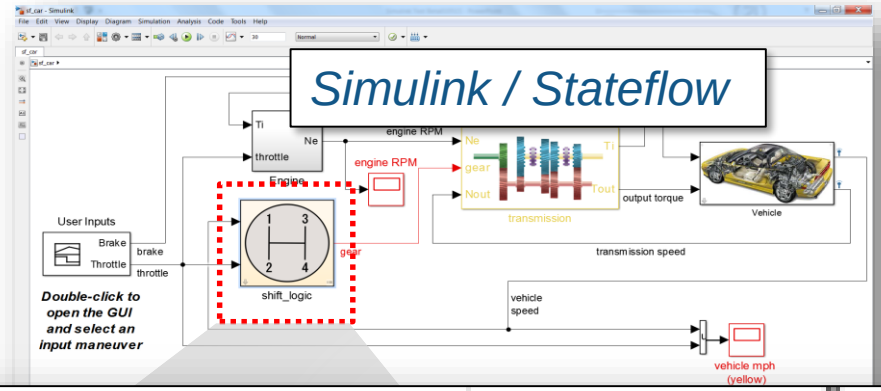
Requirements Verification with Simulink

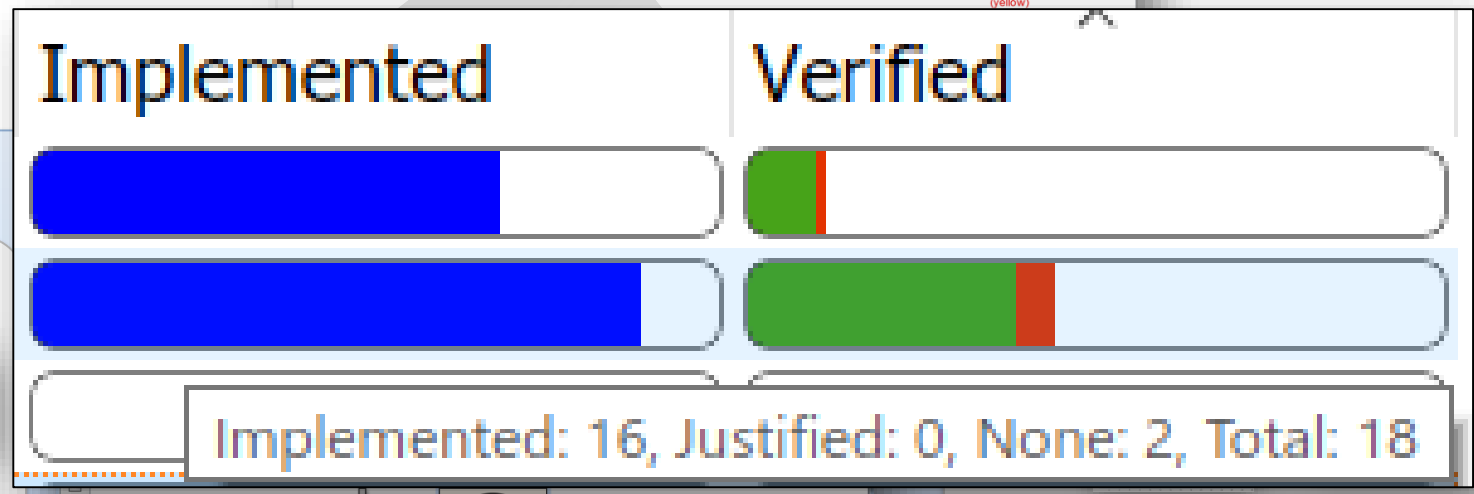
Requirements

- crs_req_func_spec
  - 1 Driver Switch Request Handling
    - 1.1 Switch precedence
    - 1.2 Avoid repeating commands

Implemented By

Verified By



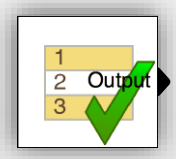


Test Case

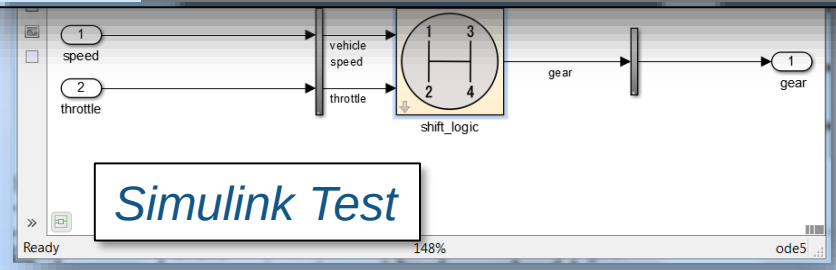
Inputs

MAT / Excel file (input)

Signal Editor

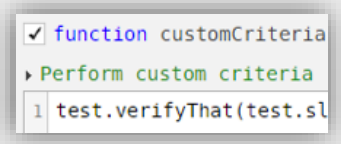


Test Sequence



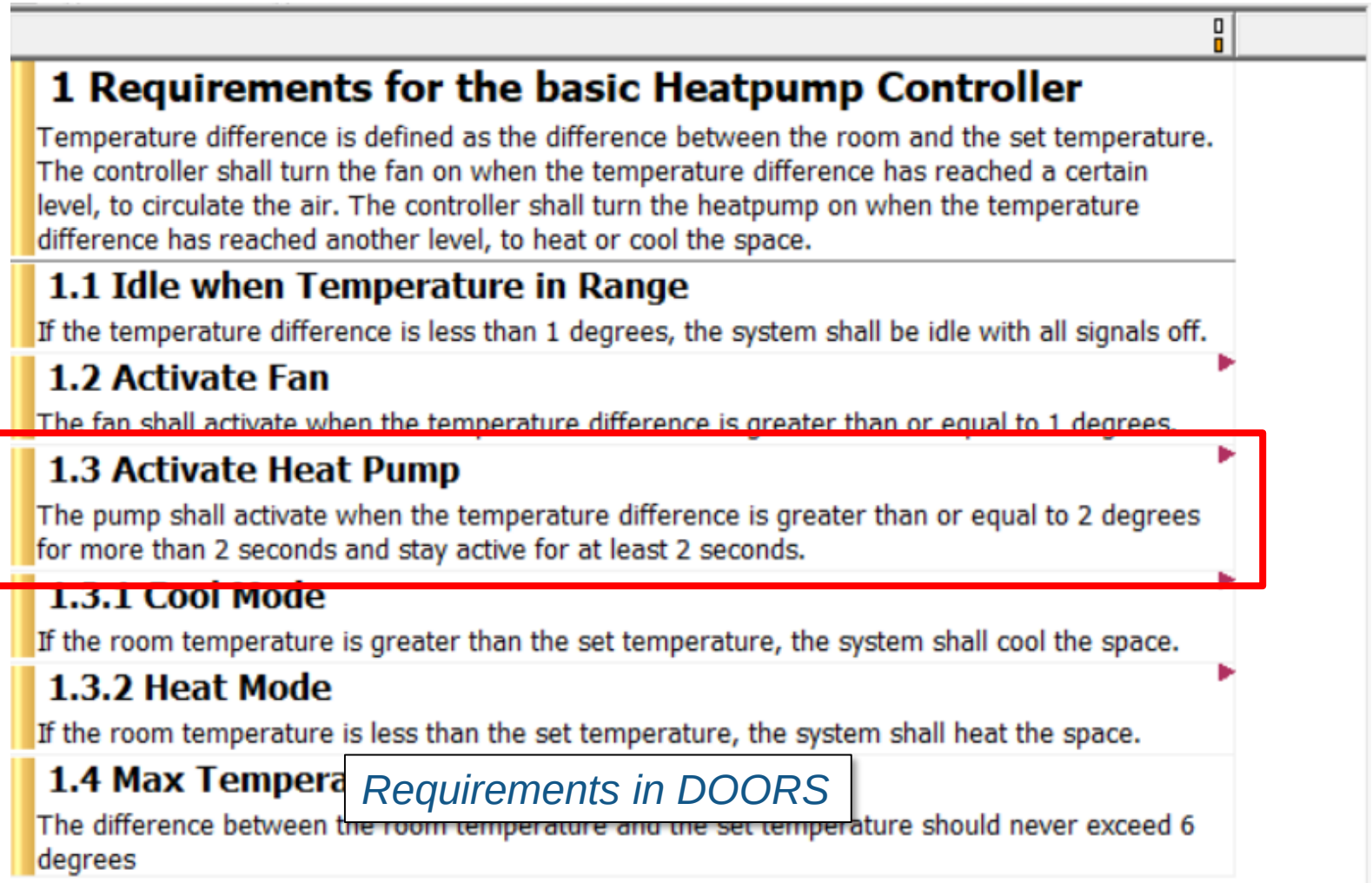
MAT / Excel File (baseline)

Test Assessments

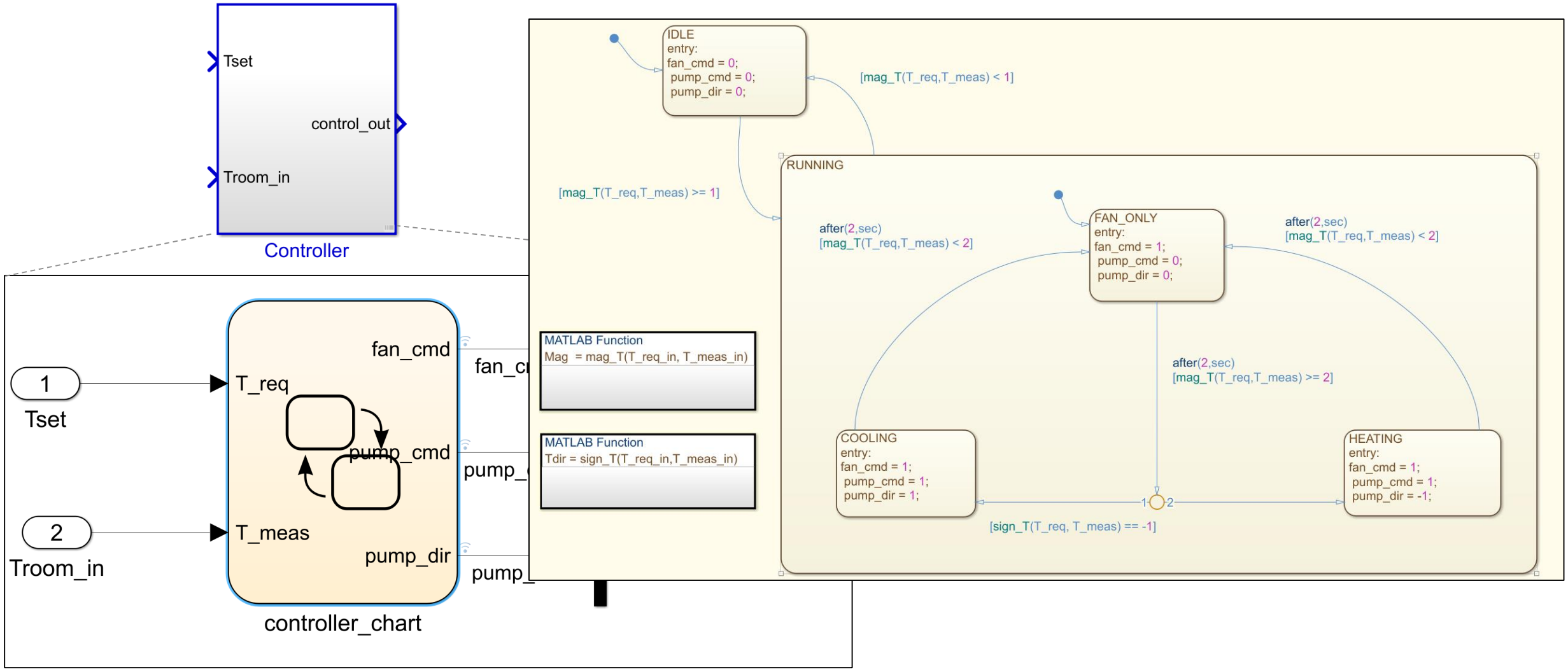


MATLAB Unit Test

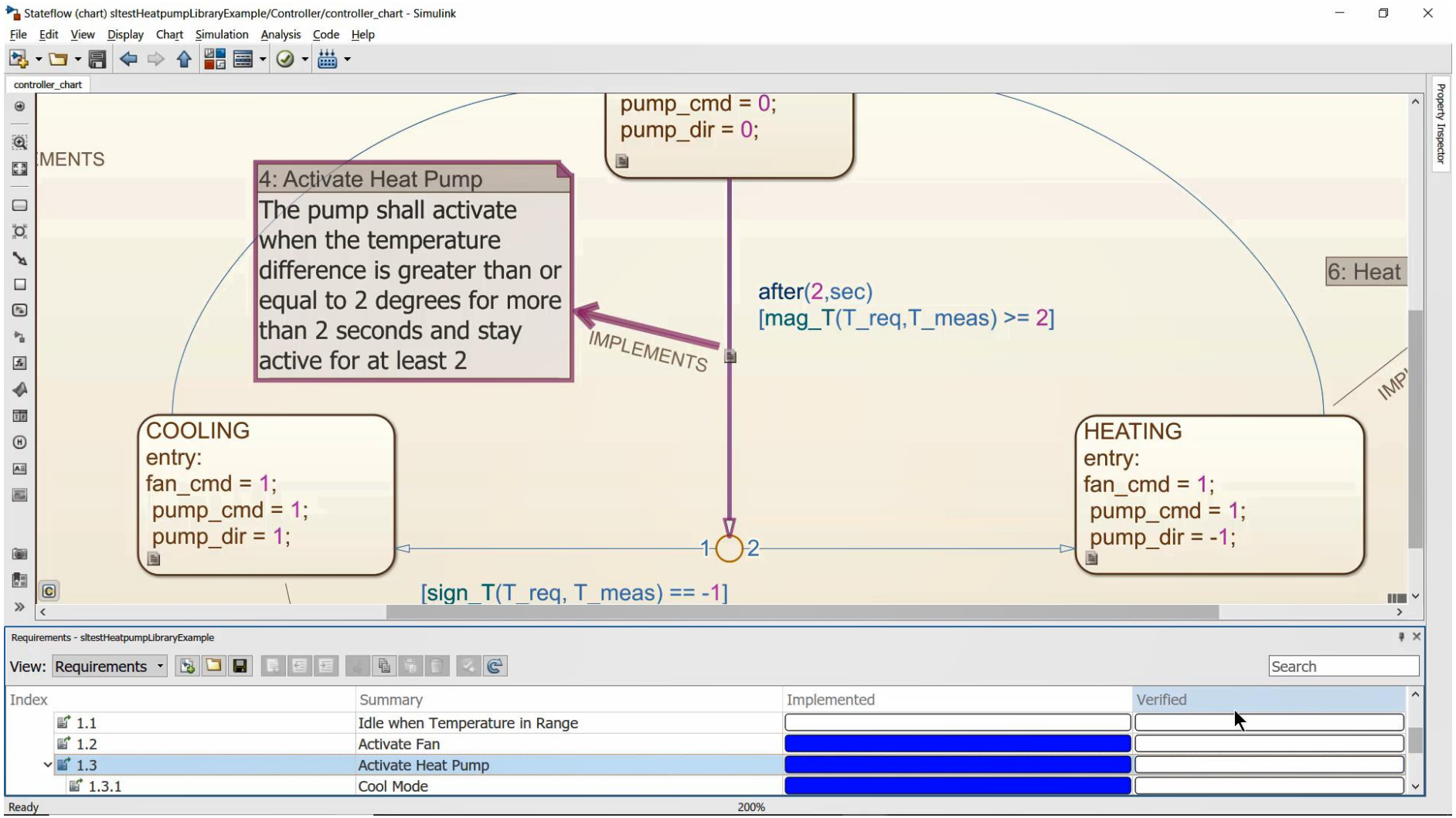
Example: Verifying Heat Pump Controller Requirements



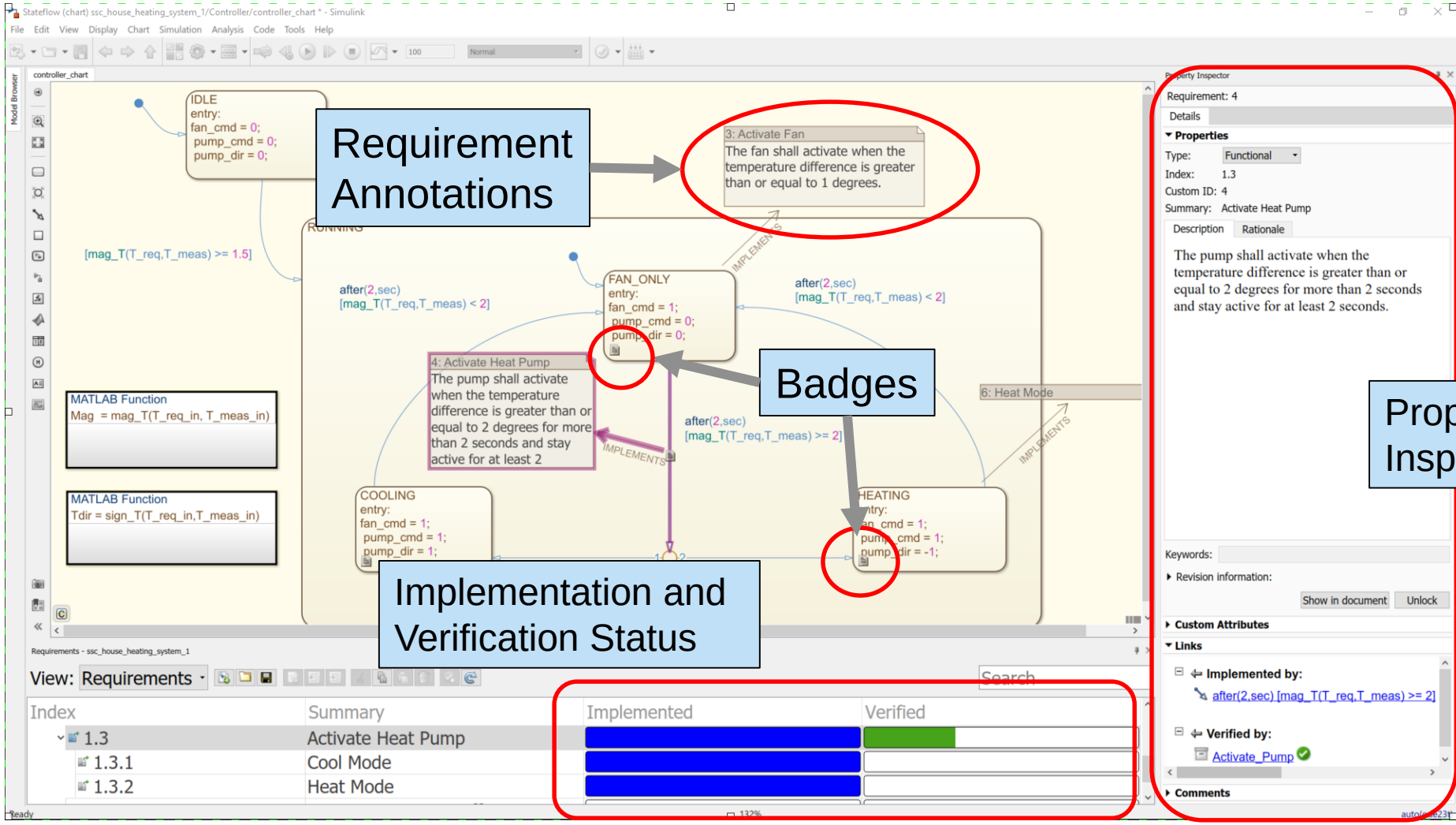
Example: Heat Pump Controller Implementation



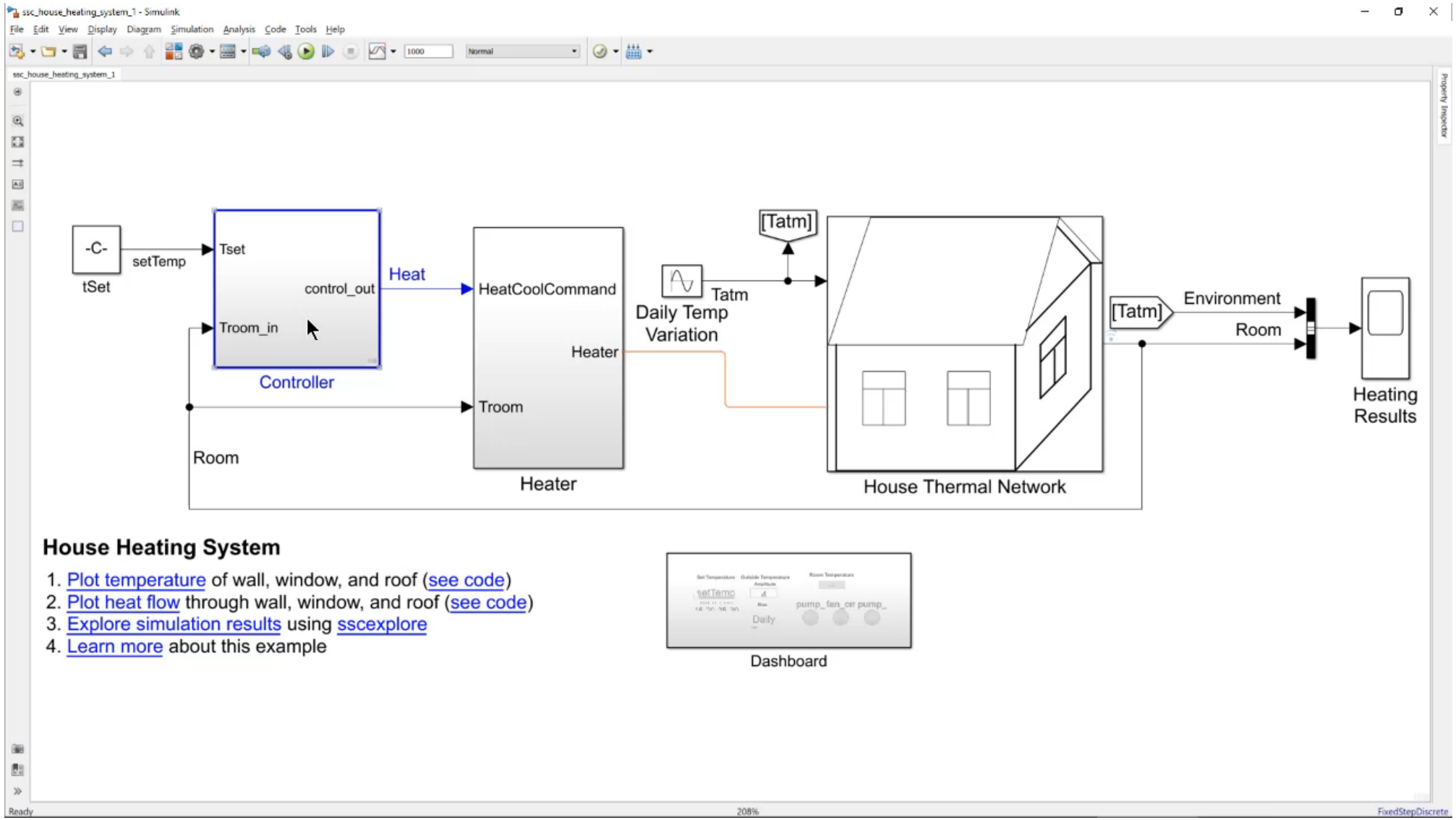
Link requirements to implementation in model



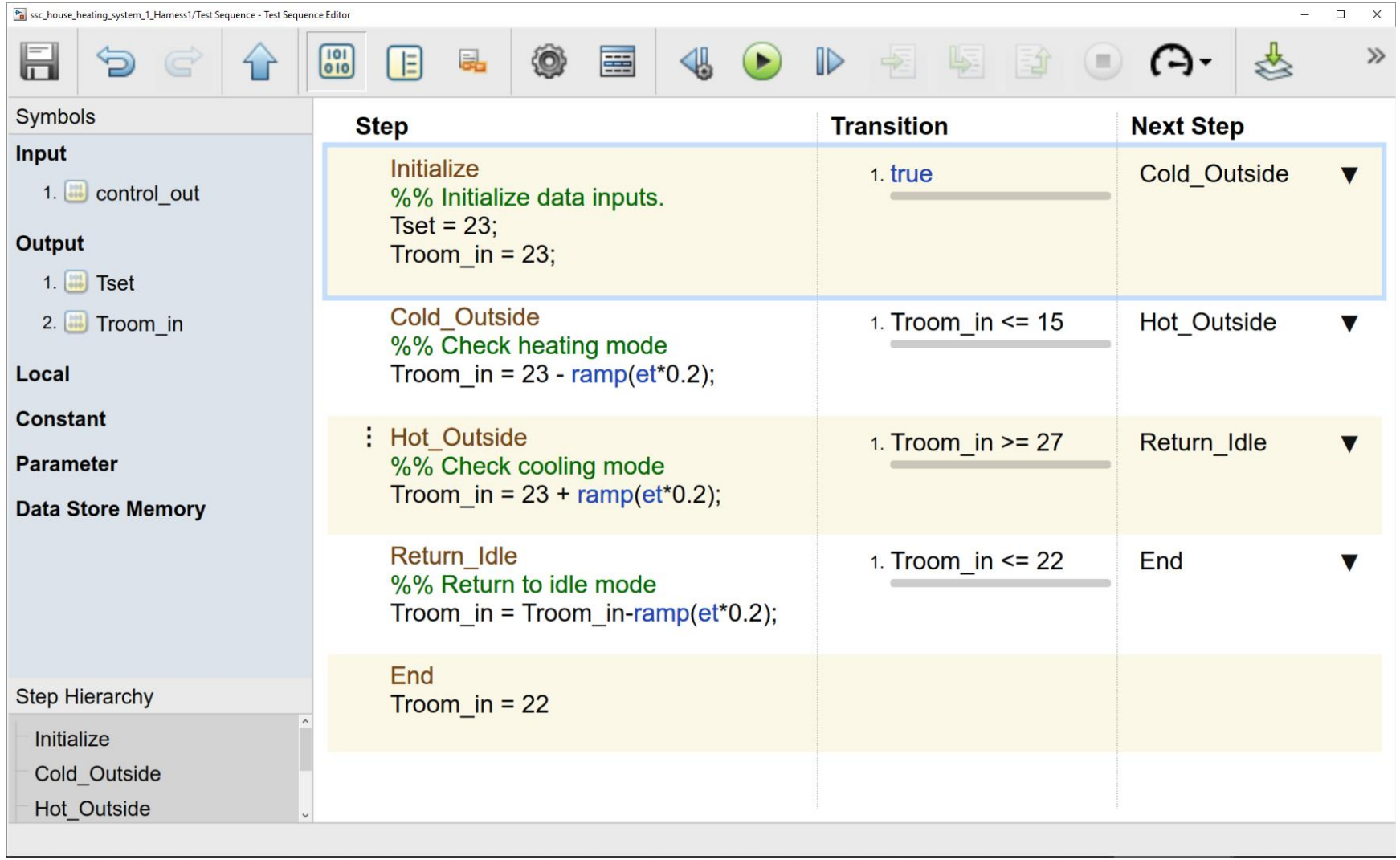
Work with Model and Requirements with Requirements Perspective



Isolate Component Under Test with Test Harness



Test Sequence Block: Step-based and temporal test sequences



Test Assessments: Formalize and execute requirements

R2019a

Activate Heat Pump

If the temperature difference exceeds 2 degrees for more than 2 seconds, then the pump shall activate for at least 2 seconds

When <condition 1> **is true,**
Then <condition 2> **must be true for some time**

Simple concept

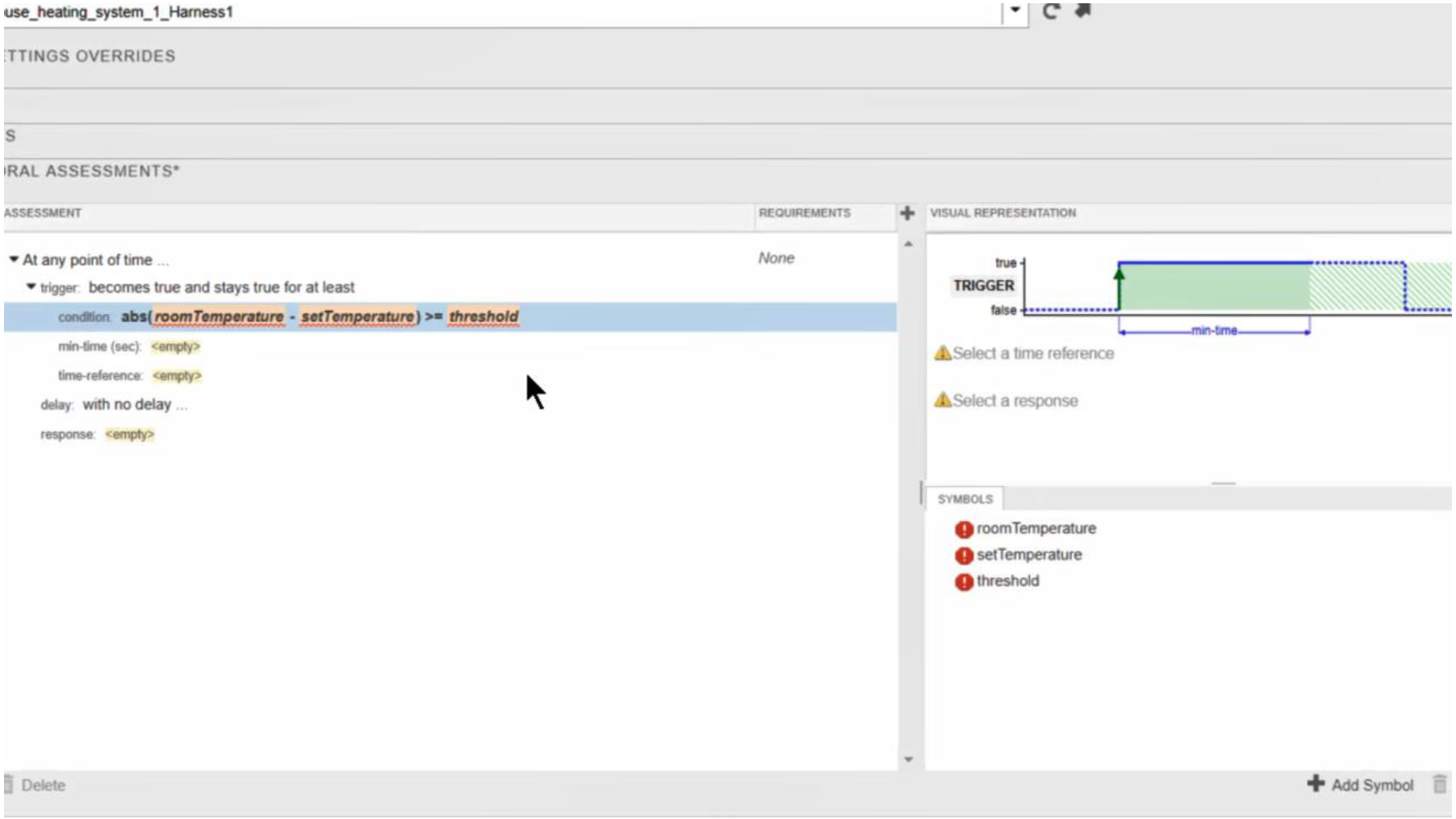
$$(|x_1 - x_2| \geq x_3)^{\overset{\varepsilon}{\leftarrow}} \wedge \square_{[0,t_1)}(|x_1 - x_2| \geq x_3) \rightarrow \square_{[0,t_2)} x_4$$

Hard to formalize

MTL logic

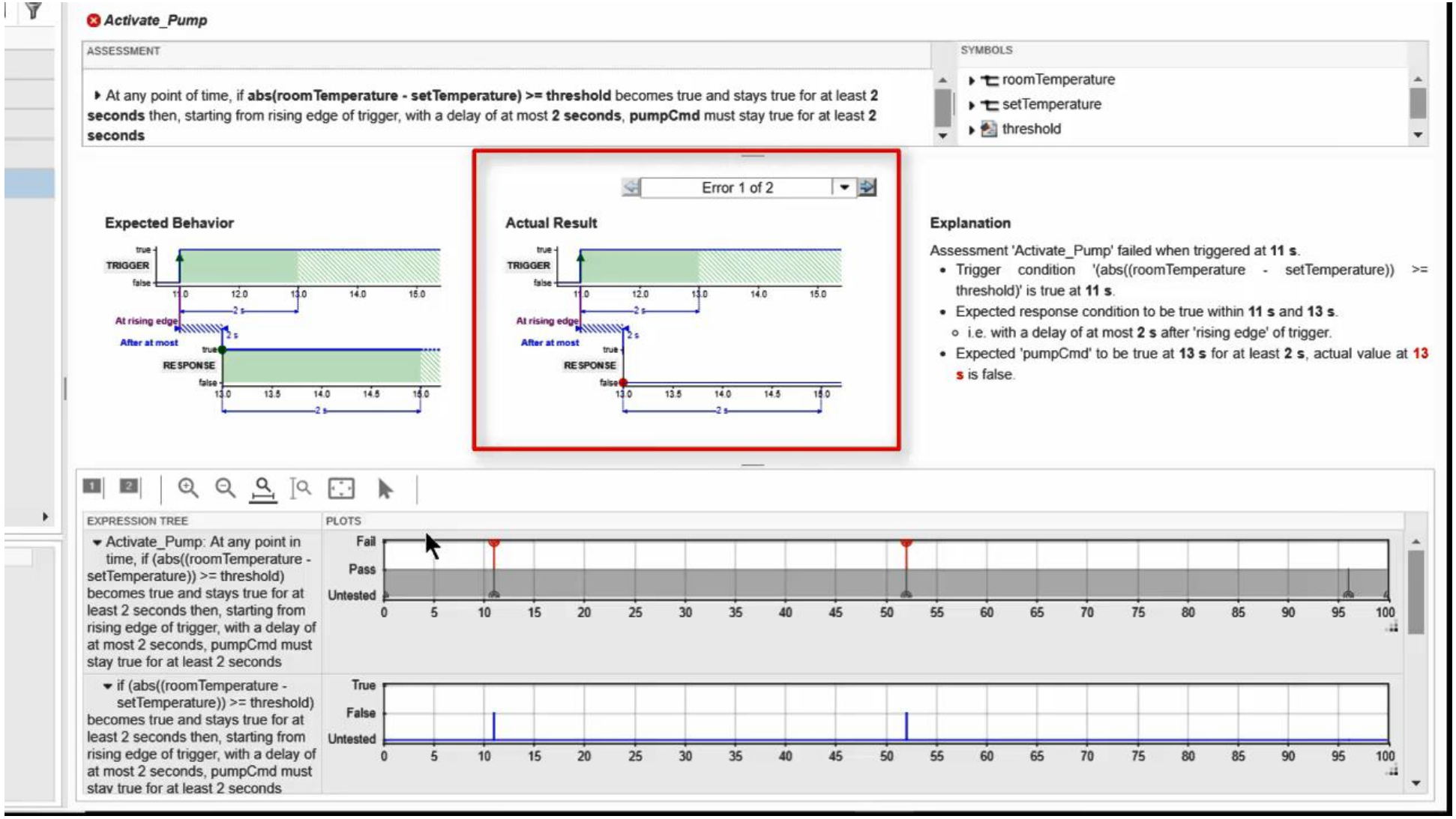
Author temporal assessments using form based editor

R2019a

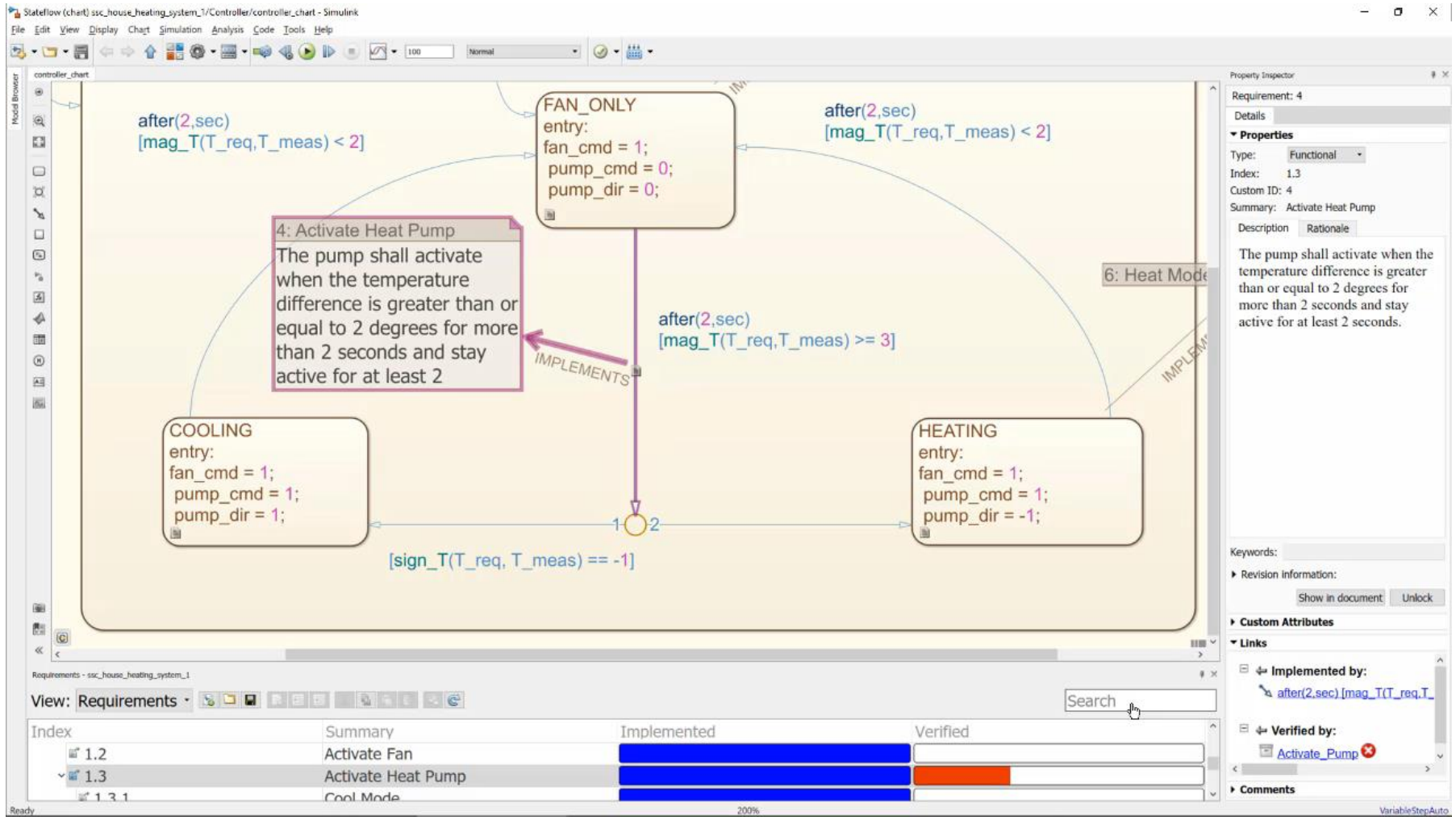


Execute assessments to verify requirements

R2019a



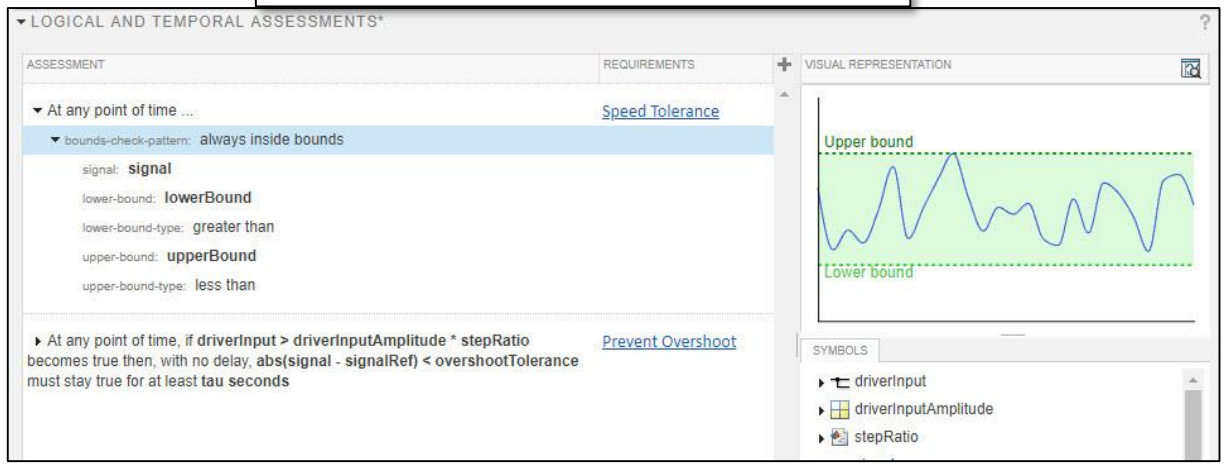
Locate implementation of requirement using link



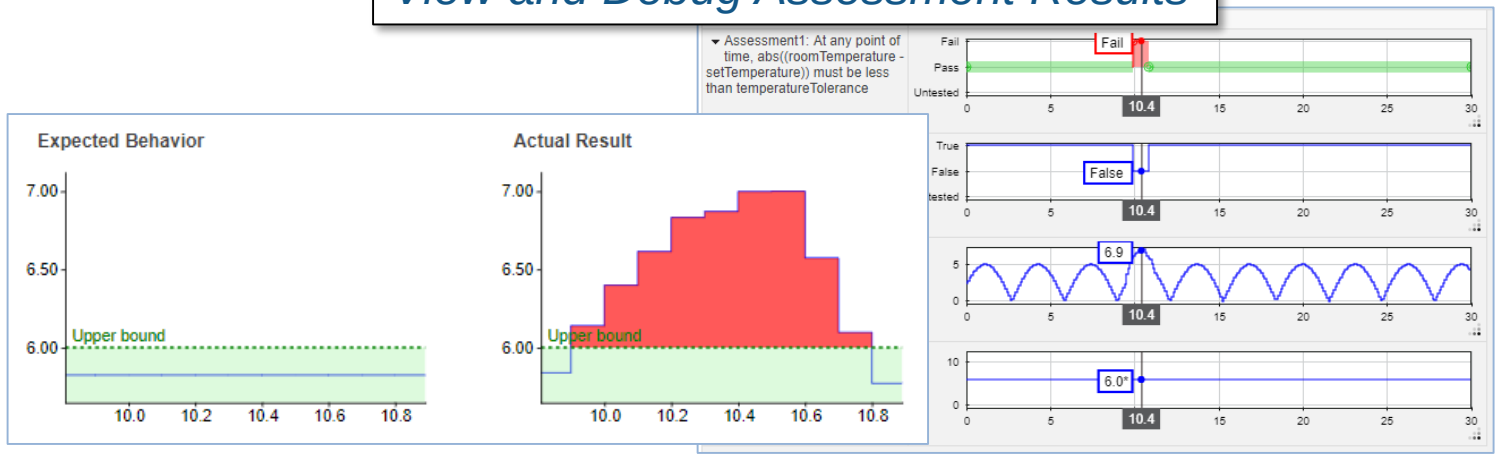
Translate textual requirements into unambiguous Temporal Assessments

- Compose assessments using form based editor
- View assessments as English-like sentence
- Review and debug temporal assessment results
- Link to requirements

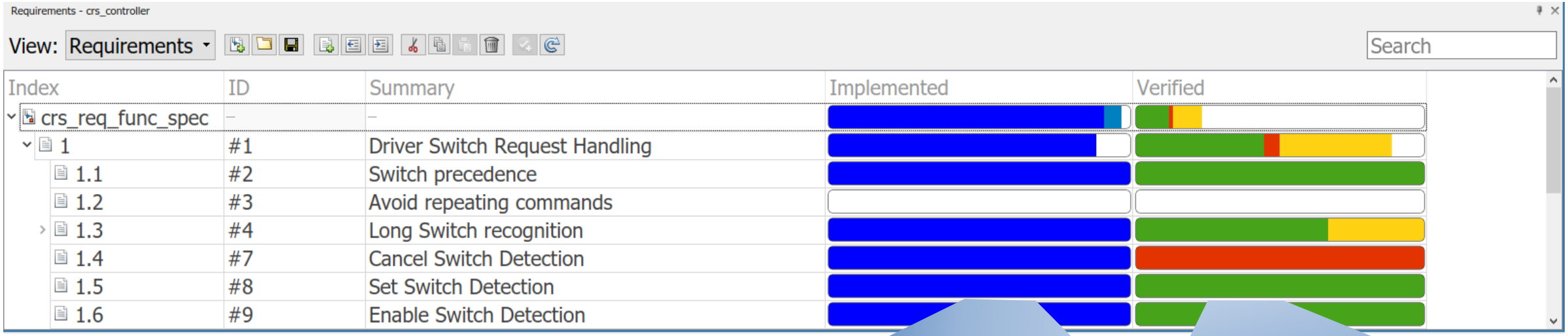
Temporal Assessment Editor



View and Debug Assessment Results



Track Implementation and Verification



Implementation Status

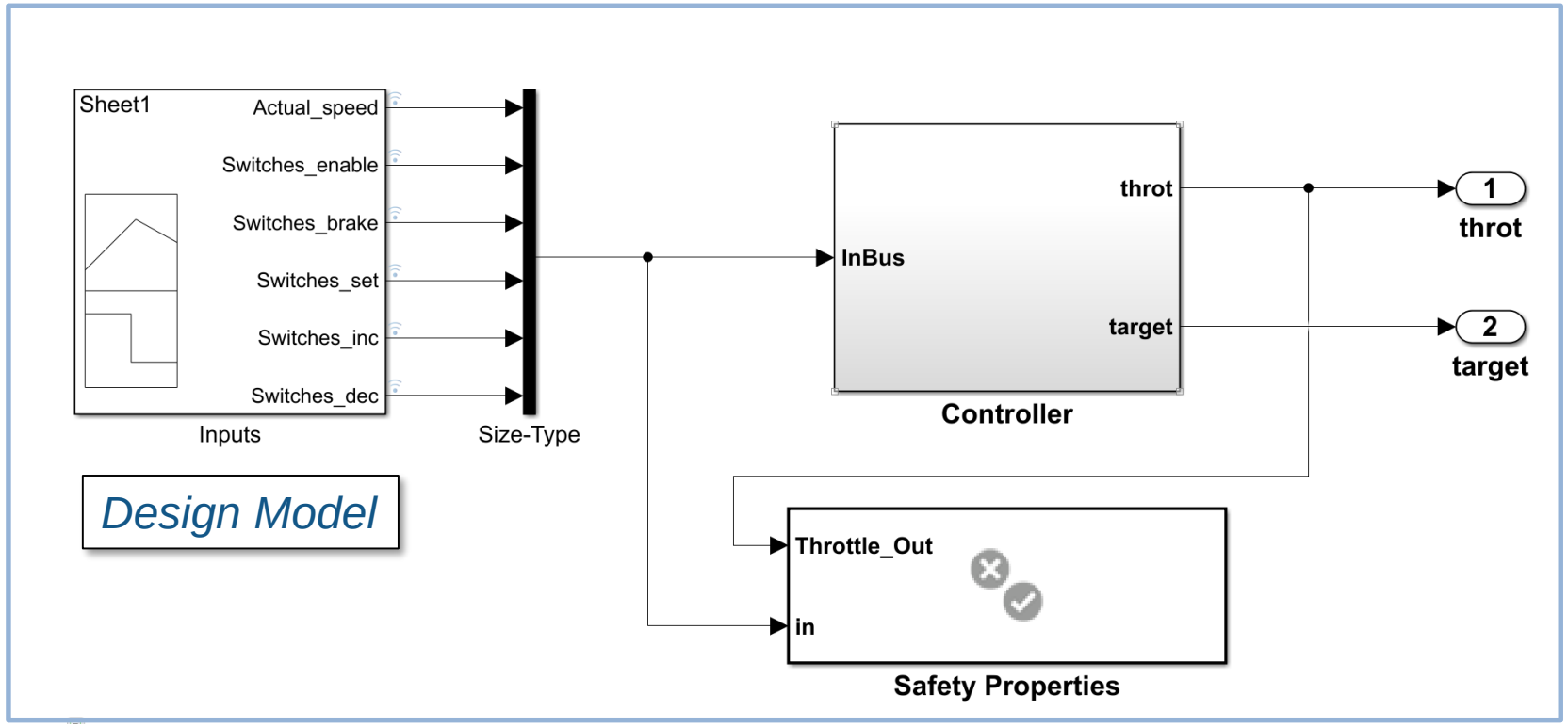
- Implemented
- Justified
- Missing

Verification Status

- Passed
- Failed
- Unexecuted
- Missing

Observers: Separate test/verification logic from design

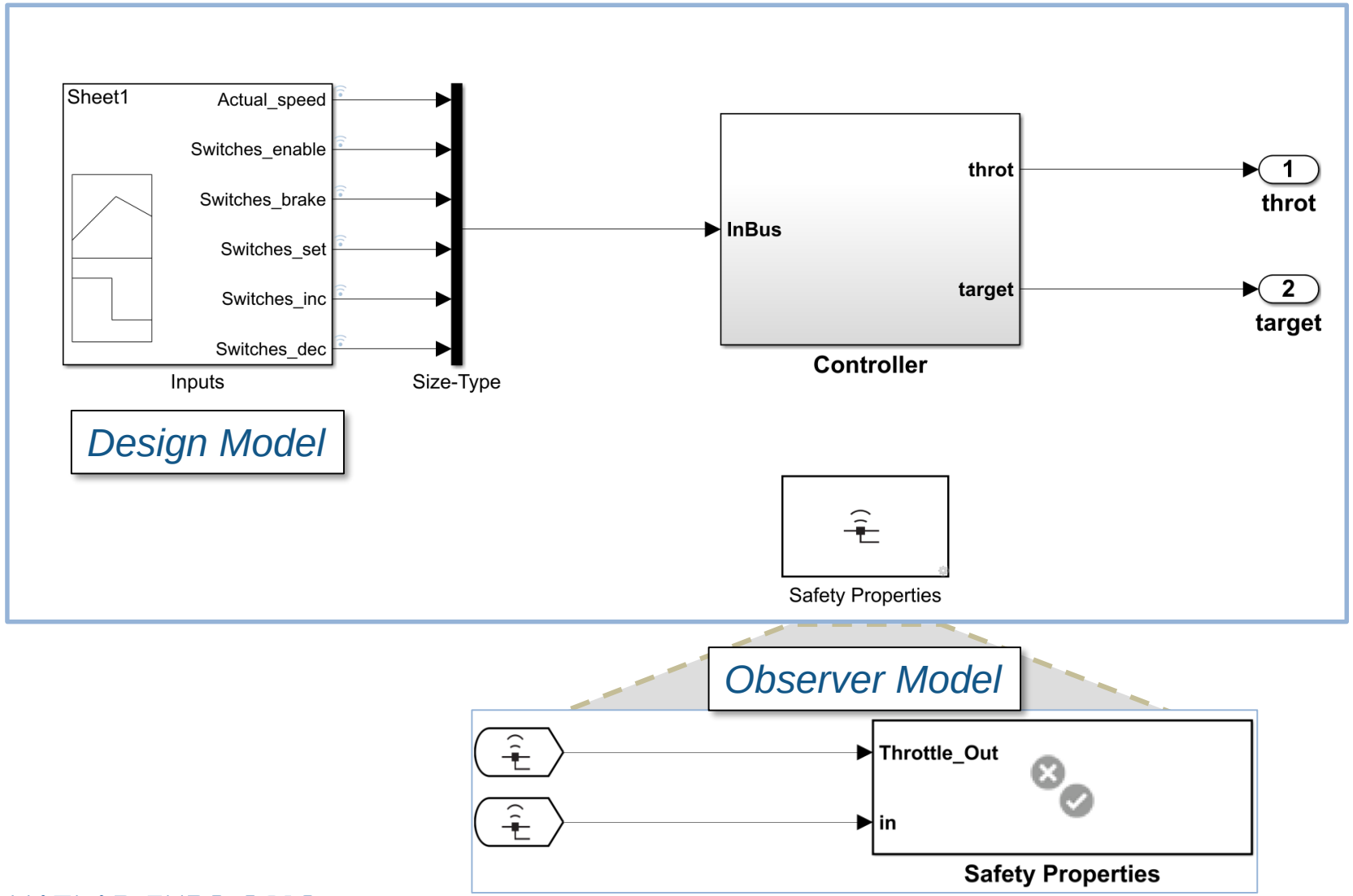
R2019a



- Access nested signals without signal lines or changing dynamic response
- Avoid modifying interface for testing
- Simplify design and test by avoiding additional signal lines

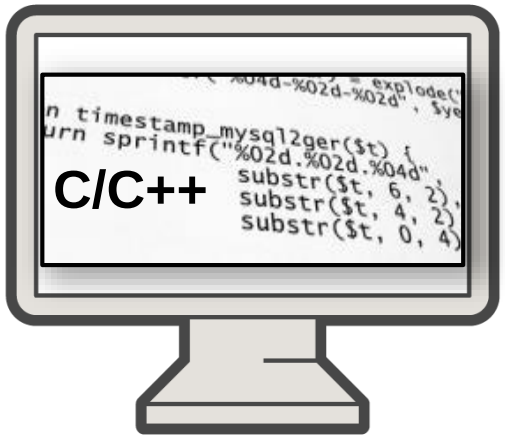
Observers: Separate test/verification logic from design

R2019a



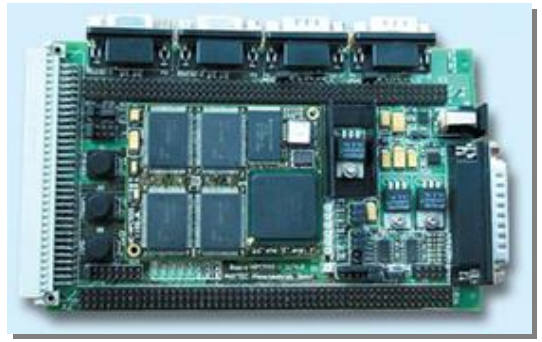
- Access nested signals without signal lines or changing dynamic response
- Avoid modifying interface for testing
- Simplify design and test by avoiding additional signal lines

Re-use tests developed for model to test code



Software in the Loop (SIL)

- Show functional equivalence, model to code
- Execute on desktop



Processor in the Loop (PIL)

- Numerical equivalence, model to target code
- Execute on target board

Hardware in the Loop (HIL)

- Check real-time behavior of the design and code.
- Execute on Speedgoat target computer using Simulink Real-Time

IDNEO Accelerates Development of AUTOSAR Software Components and Complex Device Drivers with Model-Based Design

Challenge

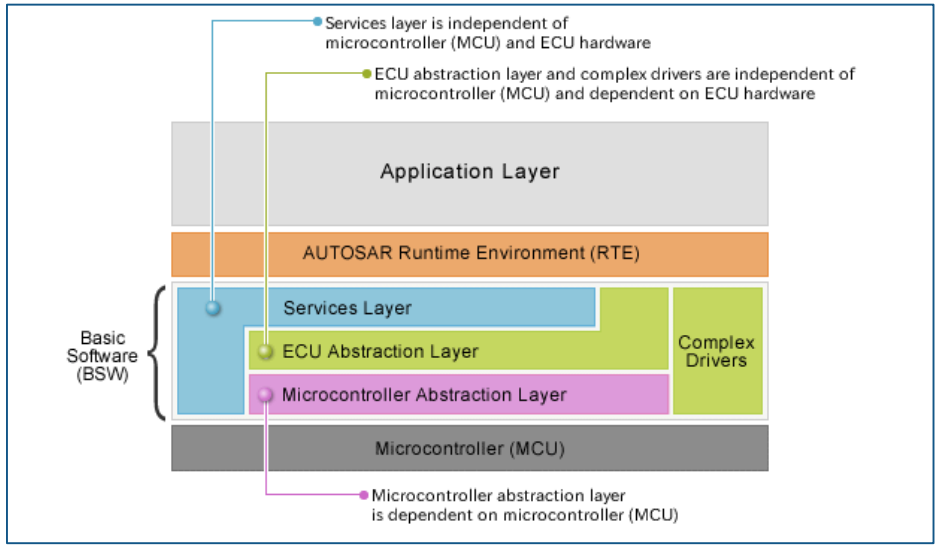
Reduce development time for embedded software for automotive applications

Solution

Use MATLAB and Simulink to model AUTOSAR software components and complex device drivers, run simulation-based tests, and generate embedded C code

Results

- Development time cut by at least 50%
- 80% of errors detected before hardware testing
- Test harnesses and MISRA-compliant C code generated from models



AUTOSAR software architecture.

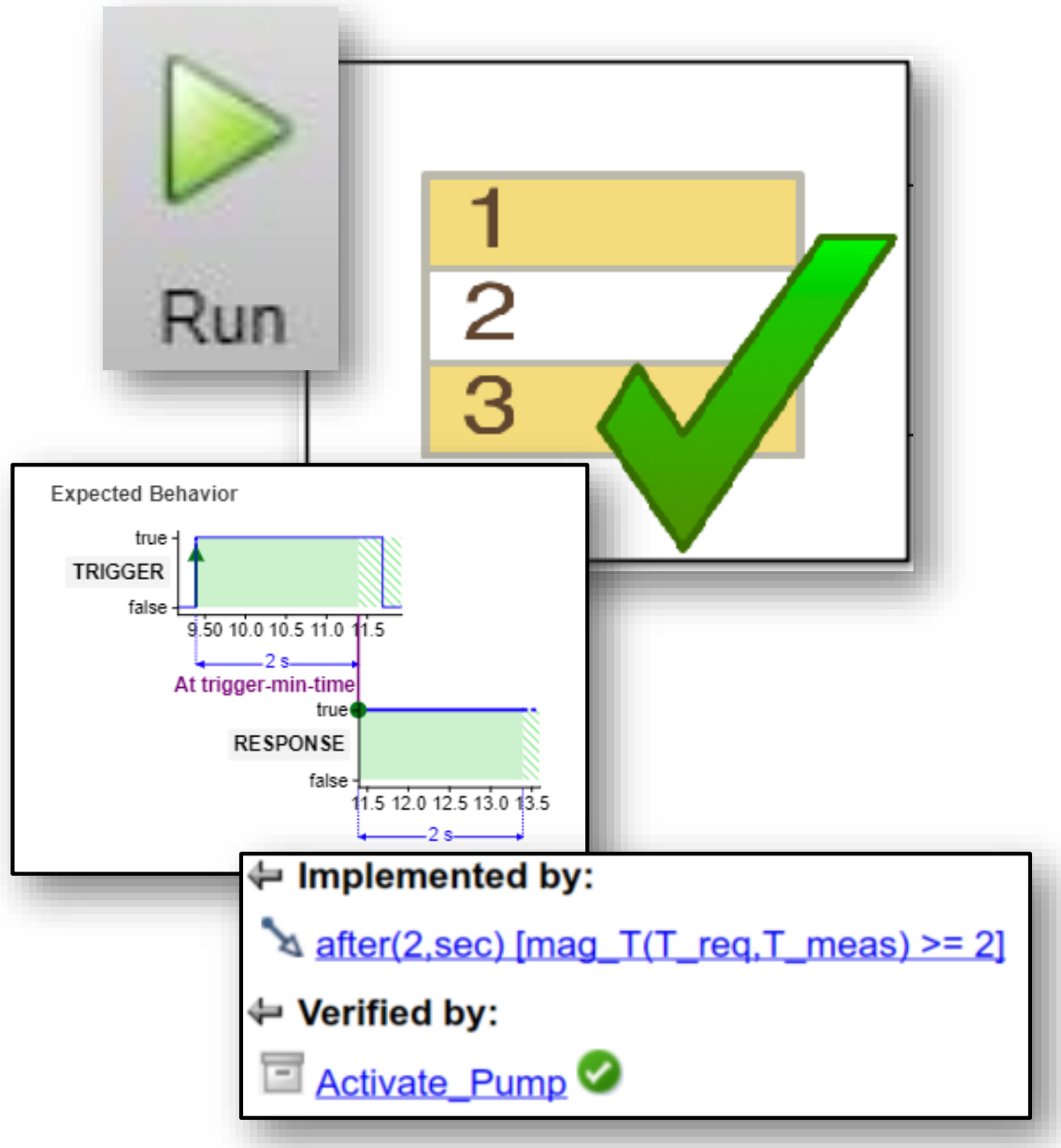
"By using Model-Based Design for all AUTOSAR projects, we have cut development time by at least 50 percent while increasing the number of defects identified early in the design phase and reducing the number of defects found in hardware tests and beyond."

- Joan Albesa, IDNEO

Link to article

Summary

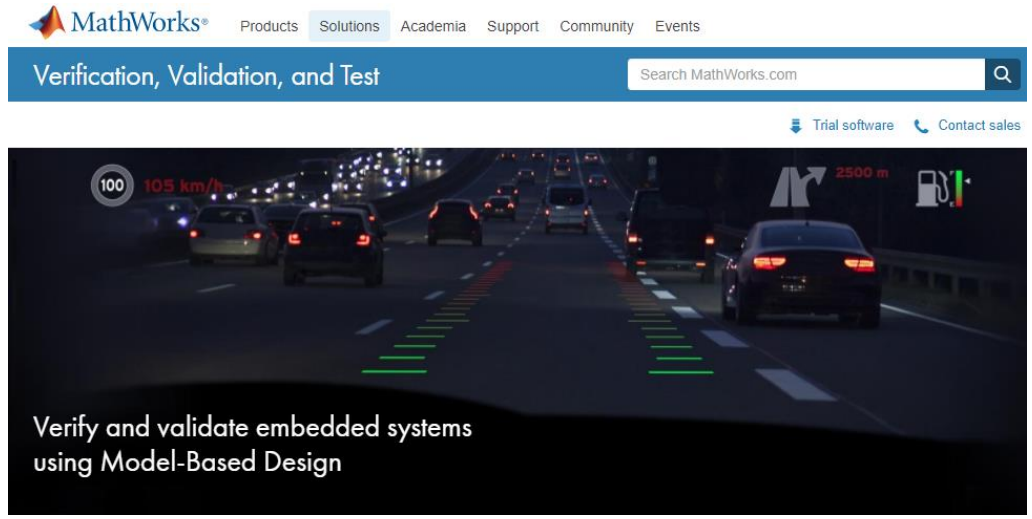
- Verify and validate requirements earlier
- Identify inconsistencies in requirements by using unambiguous assessments
- Traceability from requirements to design and test



Learn More

Key products covered in this presentation:

- Simulink Requirements
- Simulink Test
- Embedded Coder
- Simulink Real-Time



Learn more at Verification, Validation and Test Solution Page:
mathworks.com/solutions/verification-validation.html

MathWorks Training Services:
mathworks.com/training-schedule/simulation-based-testing-with-Simulink

Visit our Demo Booth