

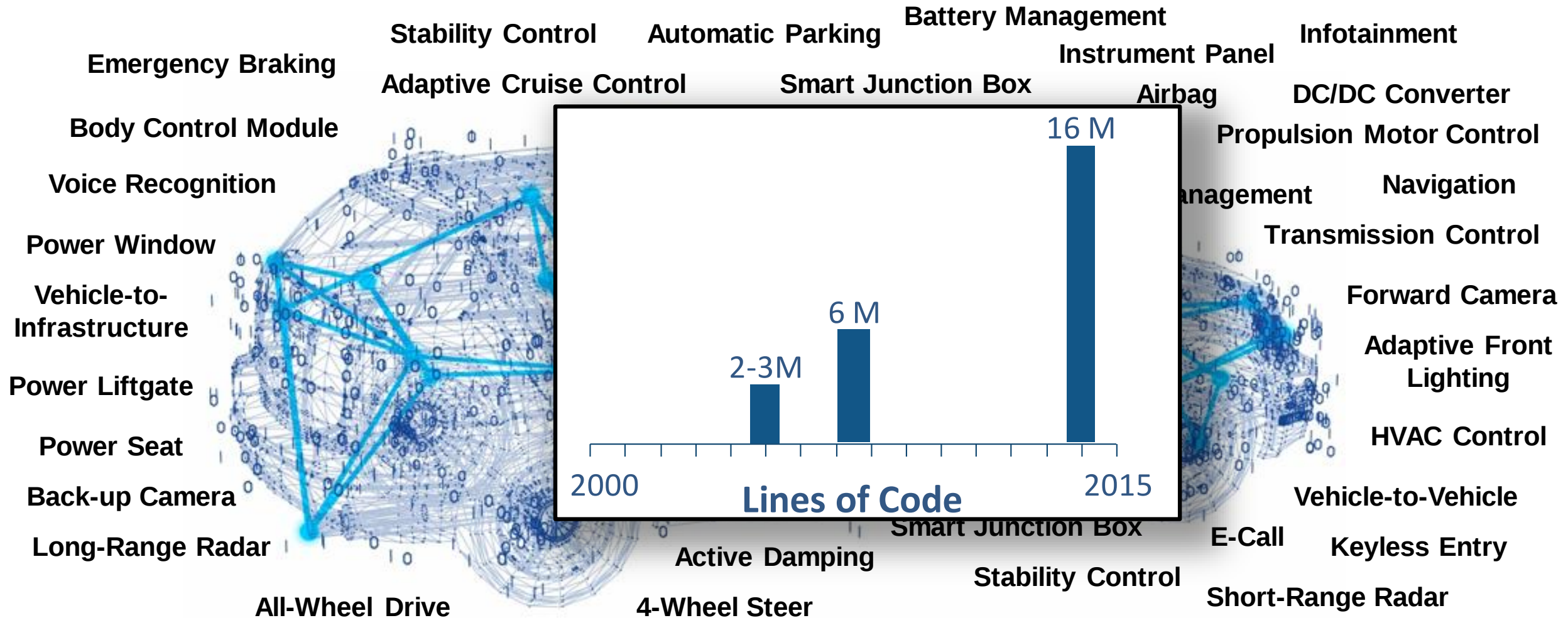
MATLAB EXPO 2018

Automatización de Métodos y Procesos
para Mejorar la Calidad del Diseño

Luis López



Growing Complexity of Embedded Systems



Siemens, "[Ford Motor Company Case Study](#)," Siemens PLM Software, 2014

McKendrick, J. "[Cars become 'datacenters on wheels', carmakers become software companies.](#)" ZDJNet, 2013

Why do 71% of Embedded Projects Fail?

Poor Requirements Management

Sources: Christopher Lindquist, Fixing the Requirements Mess, CIO Magazine, Nov 2005

Key Takeaways

- Author, manage requirements in Simulink
- Early verification to find defects sooner
- Automate manual verification tasks
- Workflow that conforms to safety standards

System Requirements

maximum machine velocity, left track
maximum machine acceleration, left track
maximum machine jolt, left track
motor speed for 50% rise time, left track
10% rise time, left track
motor speed for 95% rise time, left track
10% rise time, left track
maximum machine velocity, right track
maximum machine acceleration, right track
maximum machine jolt, right track
motor speed for 50% rise time, right track

Verified & Validated System



High Level Design

Detailed Design

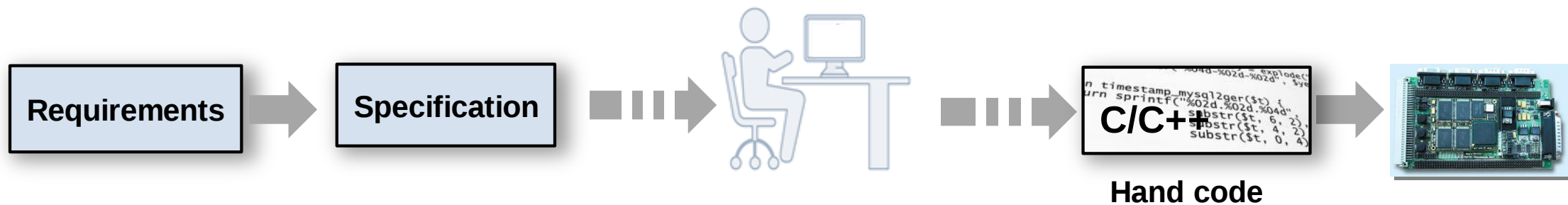
Unit Testing

Integration Testing

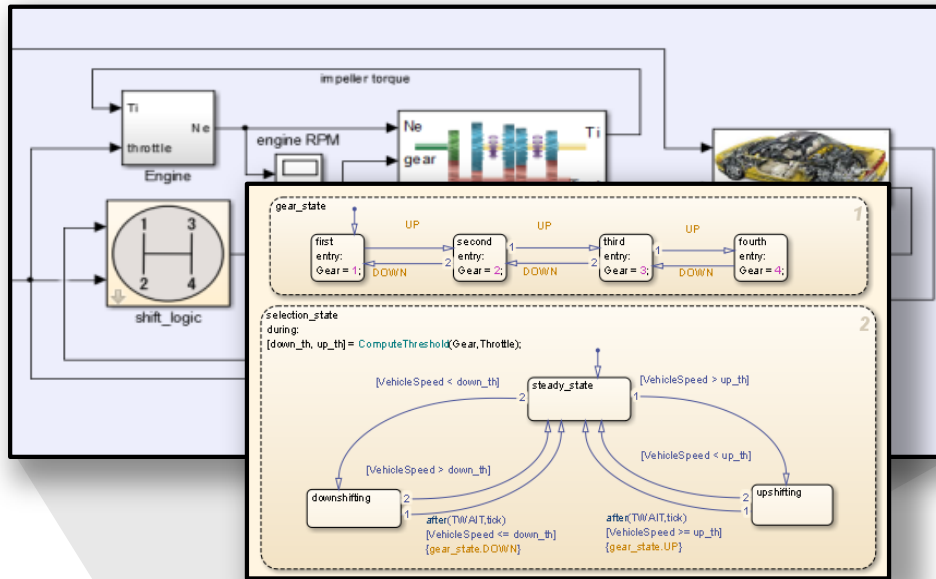
Coding

“Reduce costs and project risk through early verification, shorten time to market on a certified system, and deliver high-quality production code that was first-time right” Michael Schwarz, ITK Engineering

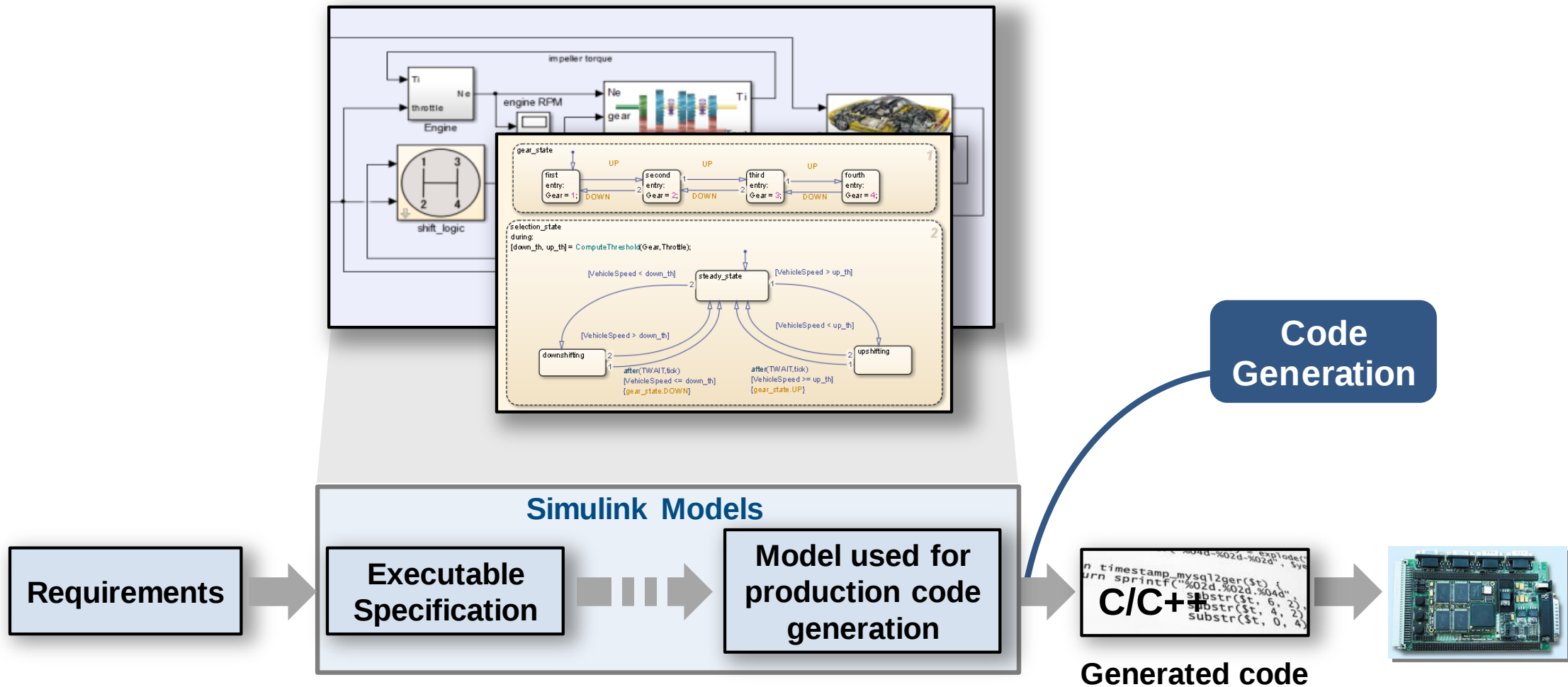
Challenge with Traditional Development Process



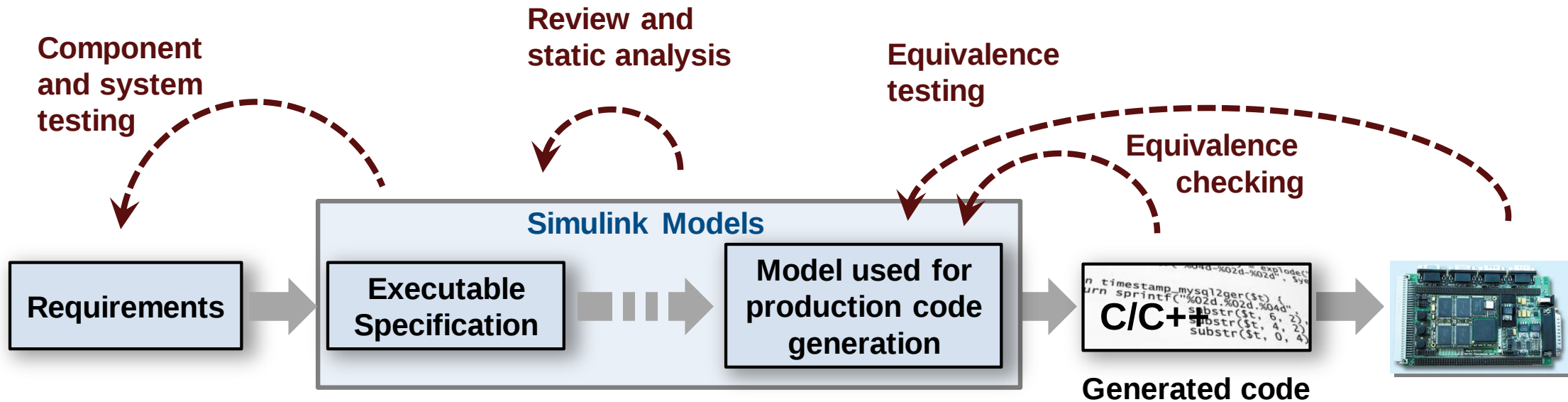
Simulink Models for Specification



Complete Model Based Design



Model Based Design Verification Workflow

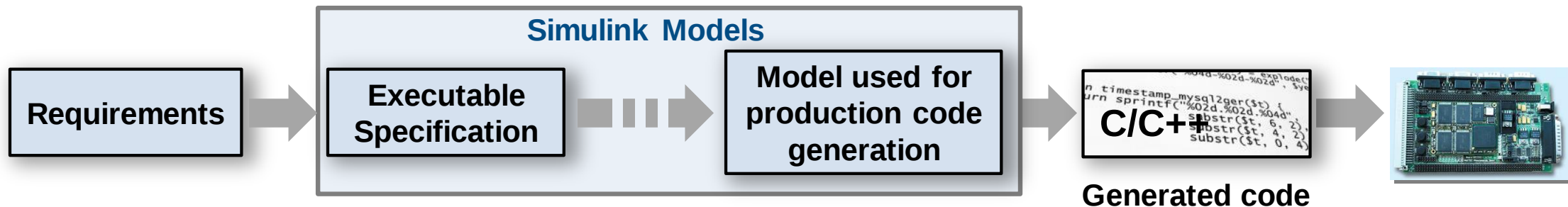


Challenges with Requirements

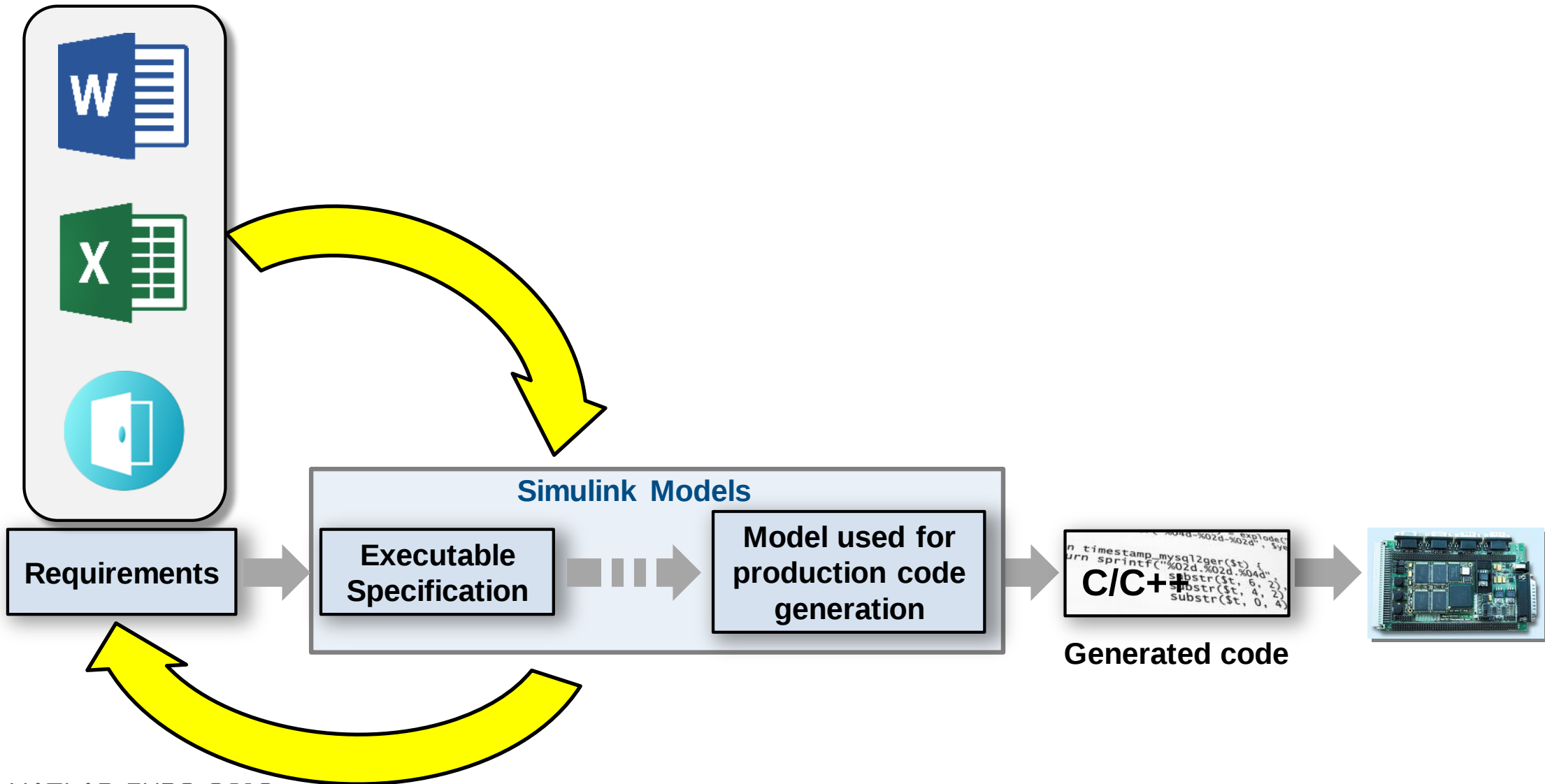
Where are requirements implemented?

Is design and requirements consistent?

How are they tested?



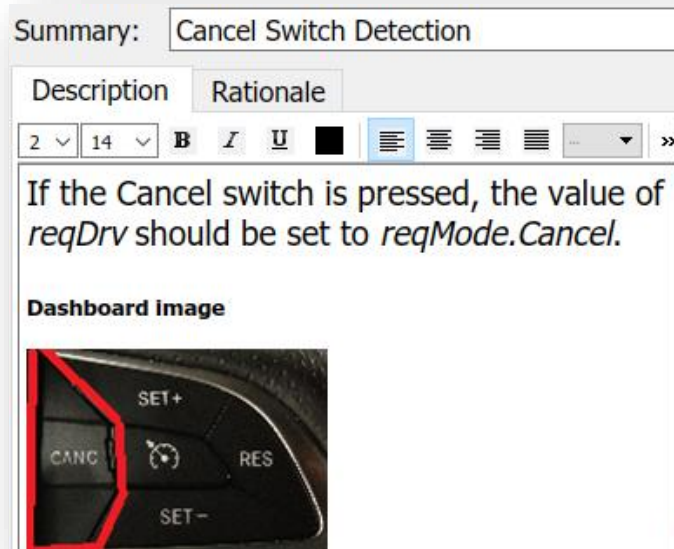
Gap Between Requirements and Design



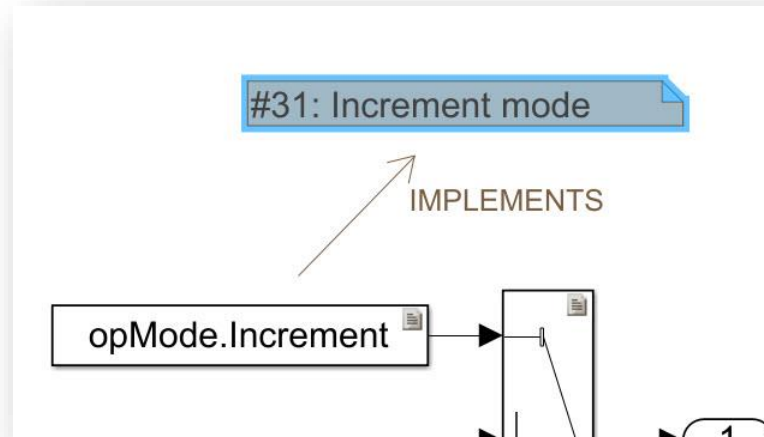
Simulink Requirements

R2017b

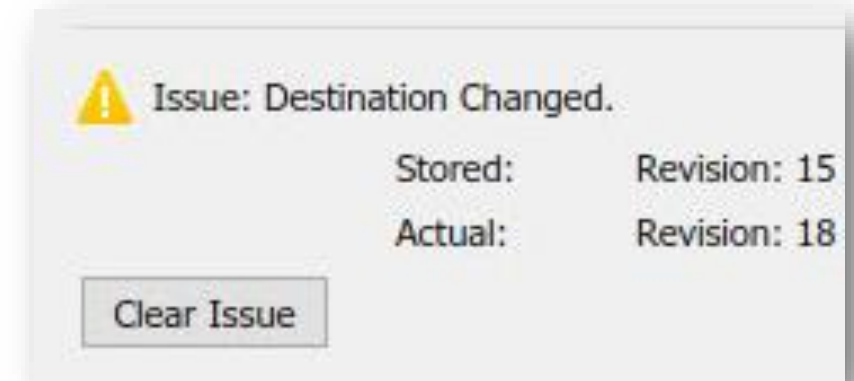
Author



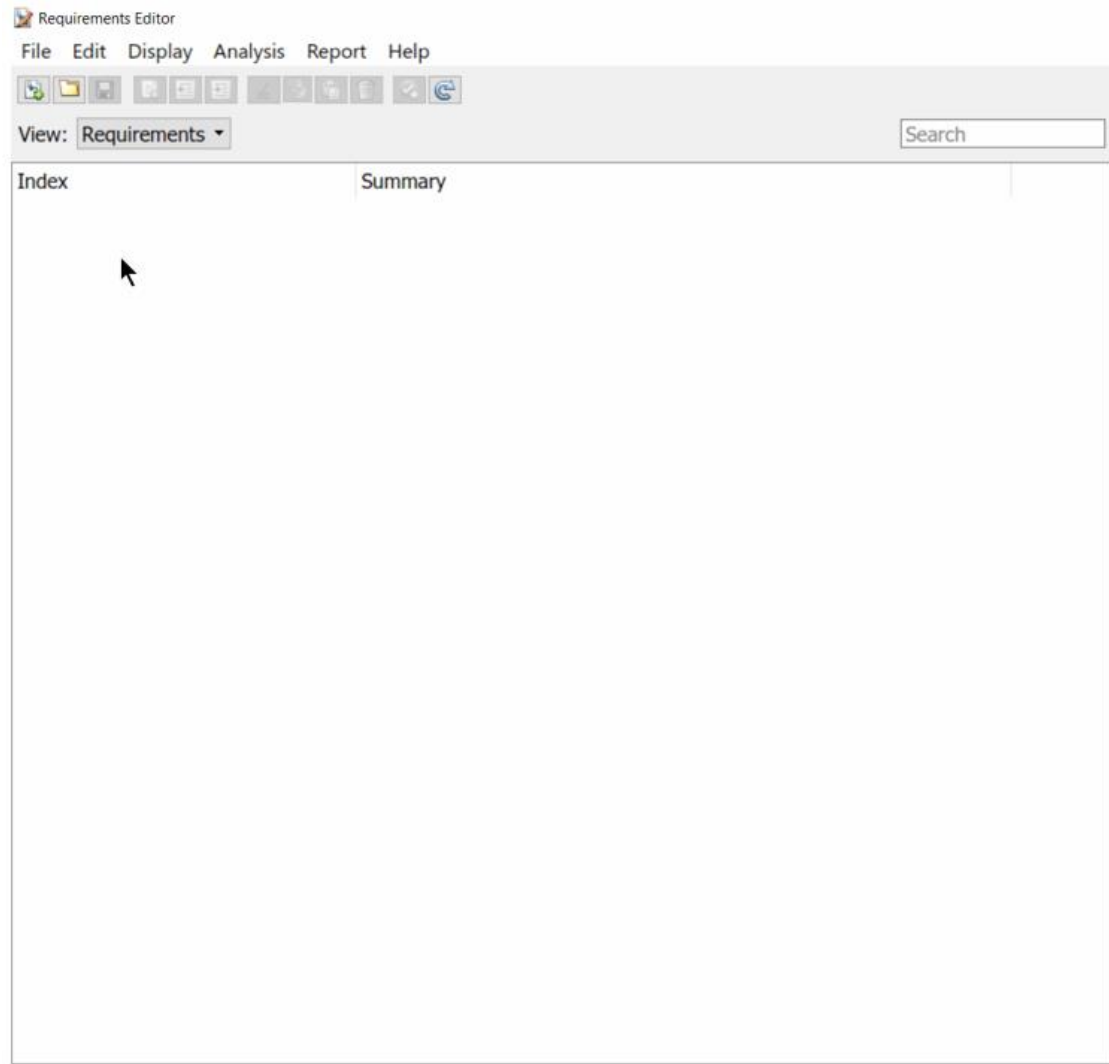
Track



Manage



Requirements Editor



The screenshot shows the Requirements Editor window. At the top is a menu bar with File, Edit, Display, Analysis, Report, and Help. Below the menu is a toolbar with icons for creating, saving, opening, and deleting requirement sets and requirements. A 'View:' dropdown menu is set to 'Requirements', and a search bar is to its right. The main workspace is divided into two panes: 'Index' on the left and 'Summary' on the right. The 'Index' pane is currently empty, with a mouse cursor hovering over it. The 'Summary' pane is also empty.

To create a new requirement set to store requirements, click **New Requirement Set**. Save the requirement set to assign a name.

To add a requirement to a requirement set, select the requirement set and click **Add Requirement**. In the **Properties** pane, enter details for the requirement.

To add a child requirement, right-click a requirement and select **Add Child Requirement**.

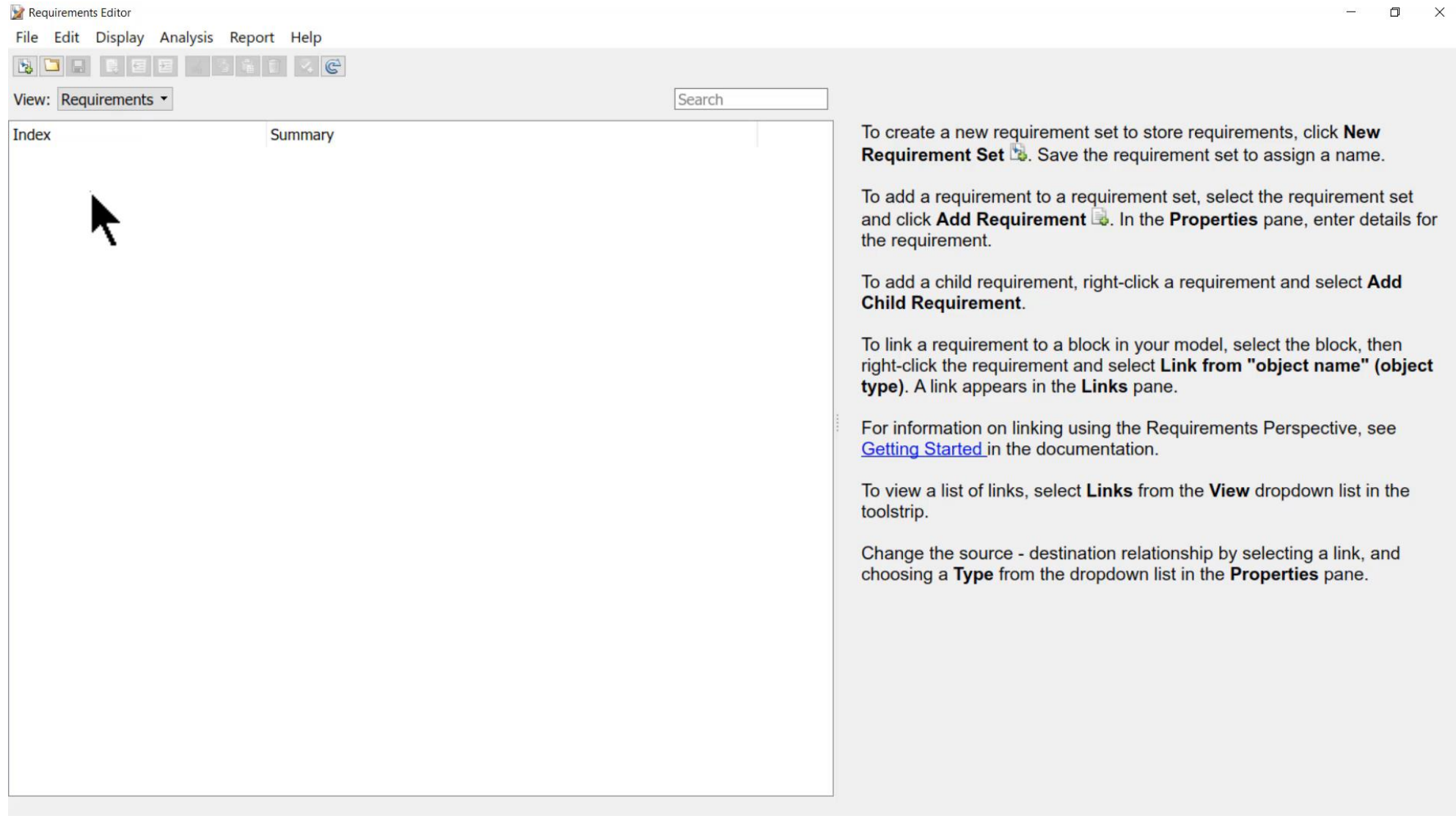
To link a requirement to a block in your model, select the block, then right-click the requirement and select **Link from "object name" (object type)**. A link appears in the **Links** pane.

For information on linking using the Requirements Perspective, see [Getting Started](#) in the documentation.

To view a list of links, select **Links** from the **View** dropdown list in the toolbar.

Change the source - destination relationship by selecting a link, and choosing a **Type** from the dropdown list in the **Properties** pane.

Requirements Editor



Requirements Editor

File Edit Display Analysis Report Help

View: Requirements Search

Index Summary

To create a new requirement set to store requirements, click **New Requirement Set**. Save the requirement set to assign a name.

To add a requirement to a requirement set, select the requirement set and click **Add Requirement**. In the **Properties** pane, enter details for the requirement.

To add a child requirement, right-click a requirement and select **Add Child Requirement**.

To link a requirement to a block in your model, select the block, then right-click the requirement and select **Link from "object name" (object type)**. A link appears in the **Links** pane.

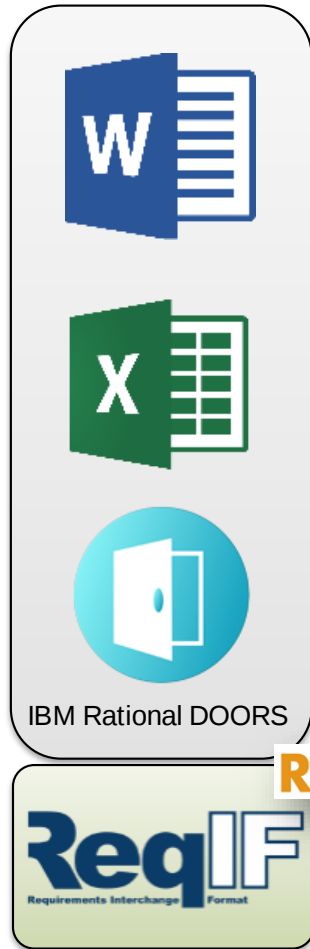
For information on linking using the Requirements Perspective, see [Getting Started](#) in the documentation.

To view a list of links, select **Links** from the **View** dropdown list in the toolbar.

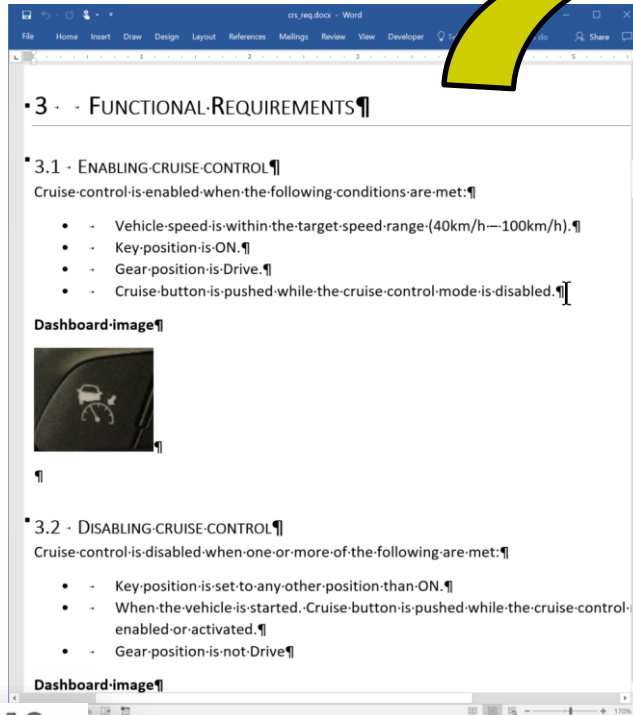
Change the source - destination relationship by selecting a link, and choosing a **Type** from the dropdown list in the **Properties** pane.

Import Requirements from External Sources

Import

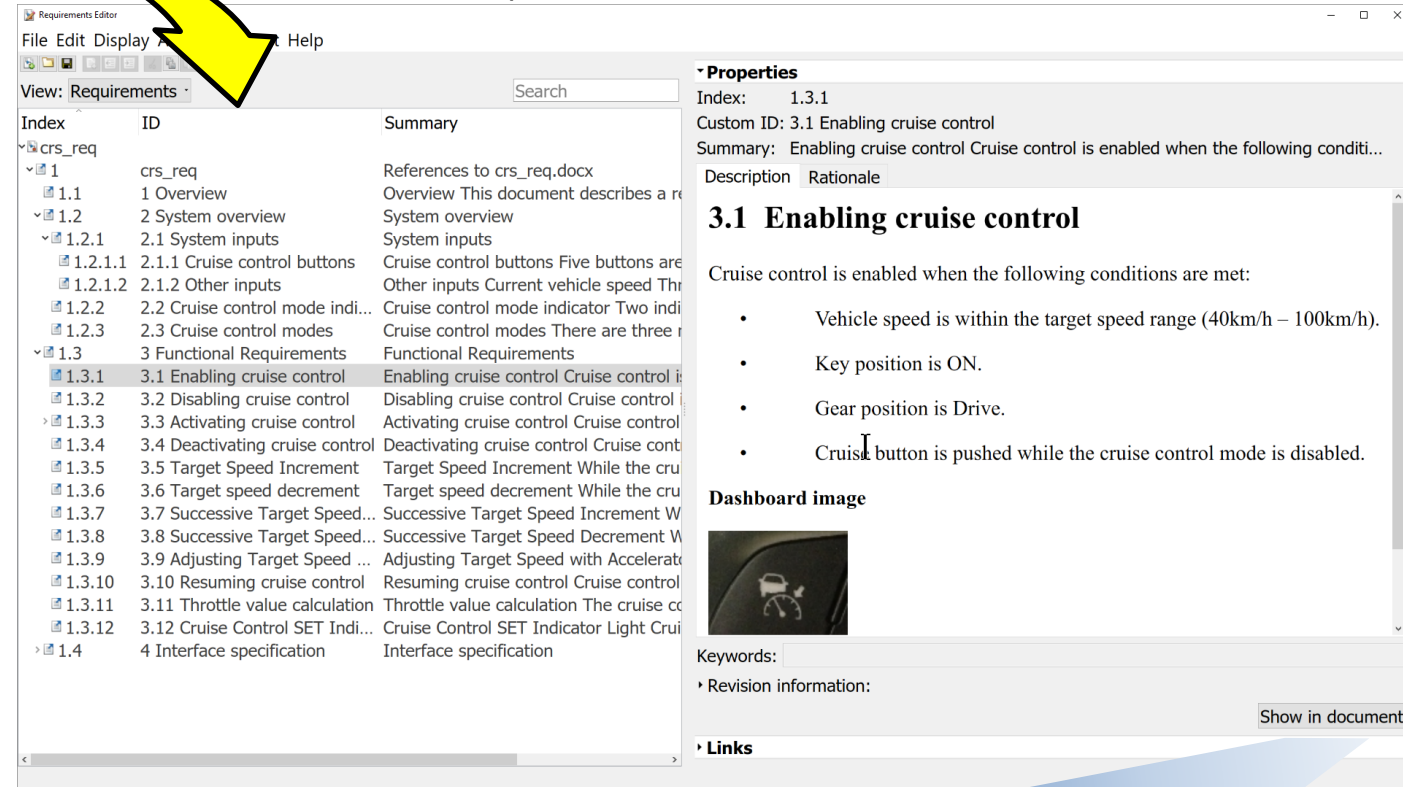


Microsoft Word



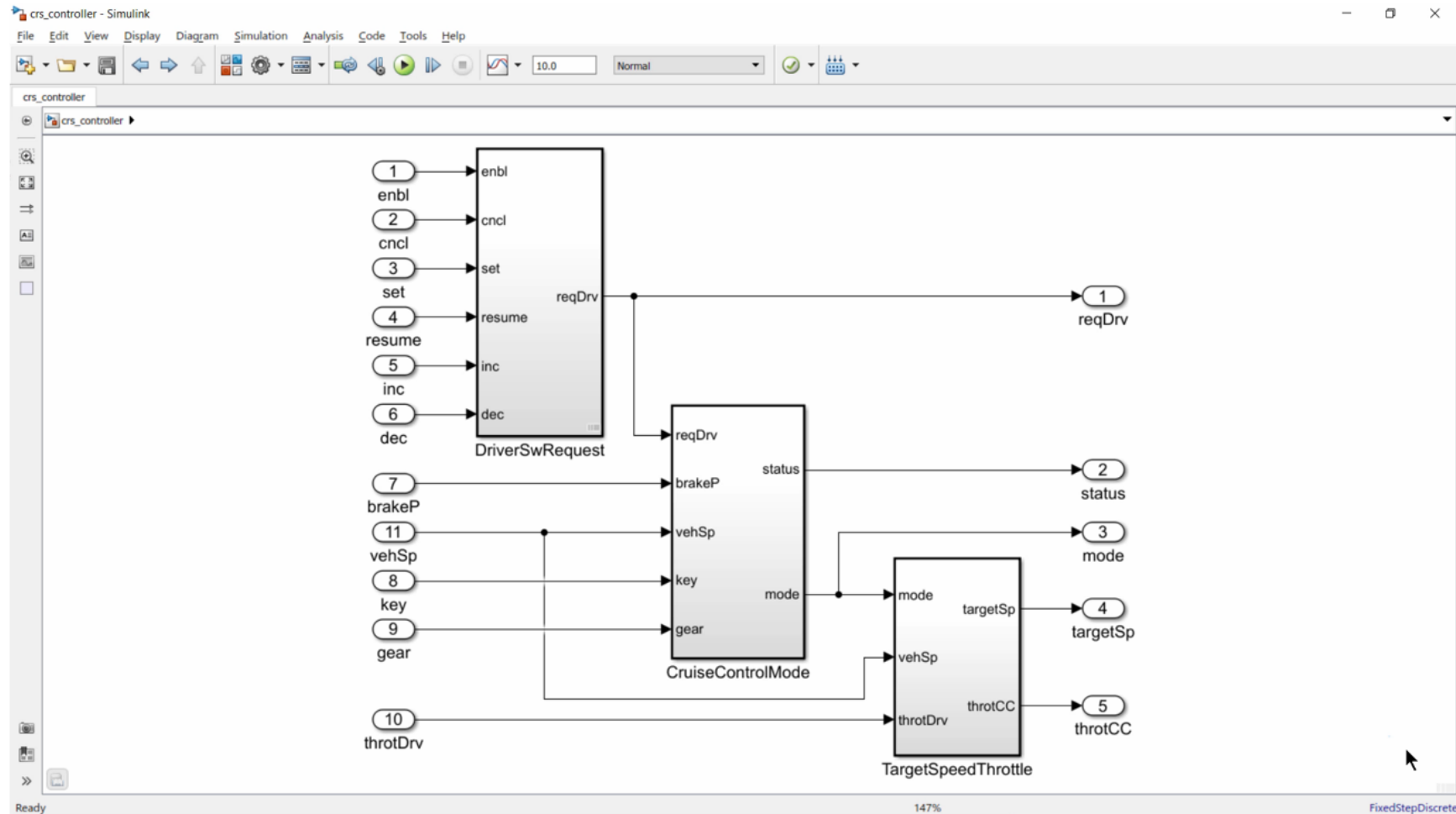
R2018a

Simulink Requirements Editor

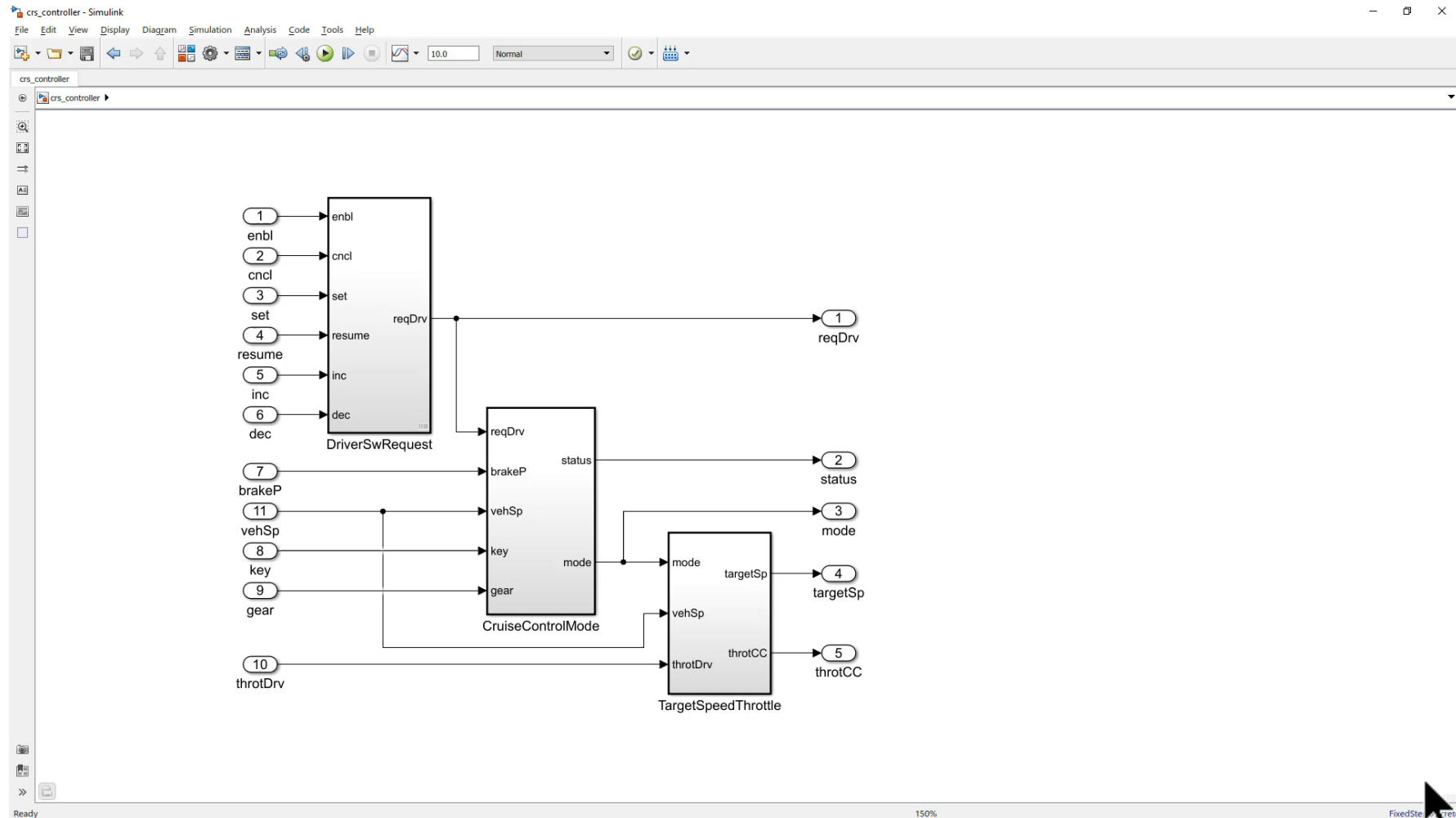


Show in document

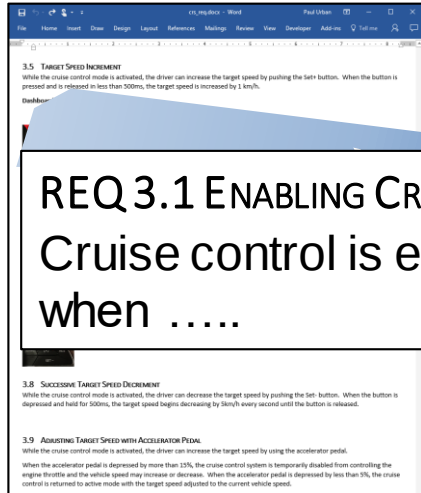
Requirements Perspective



Requirements Perspective

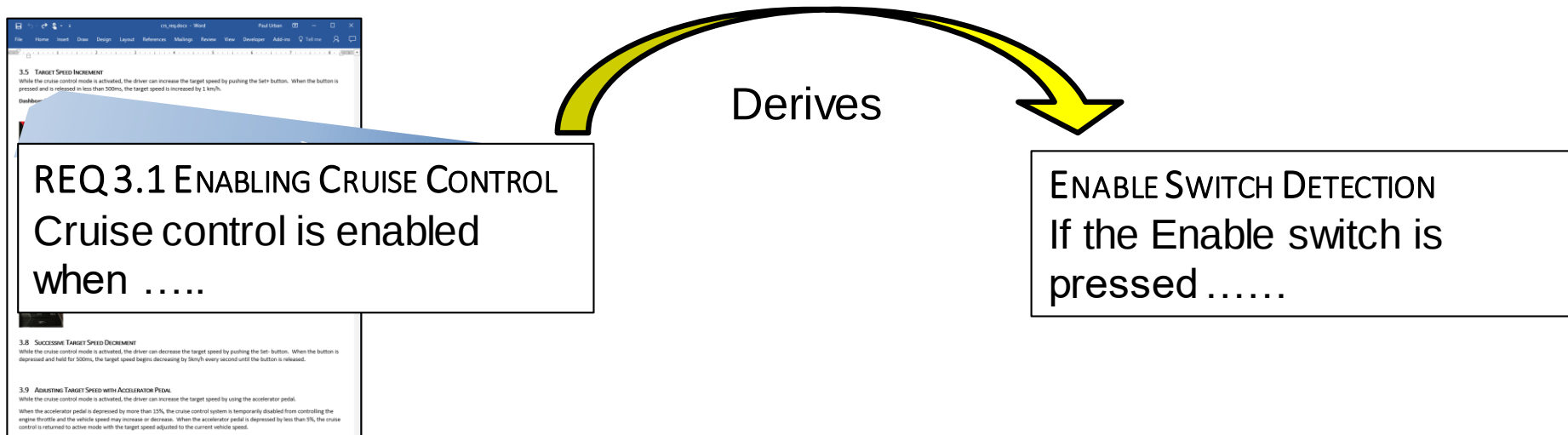


Link Requirements, Designs and Tests

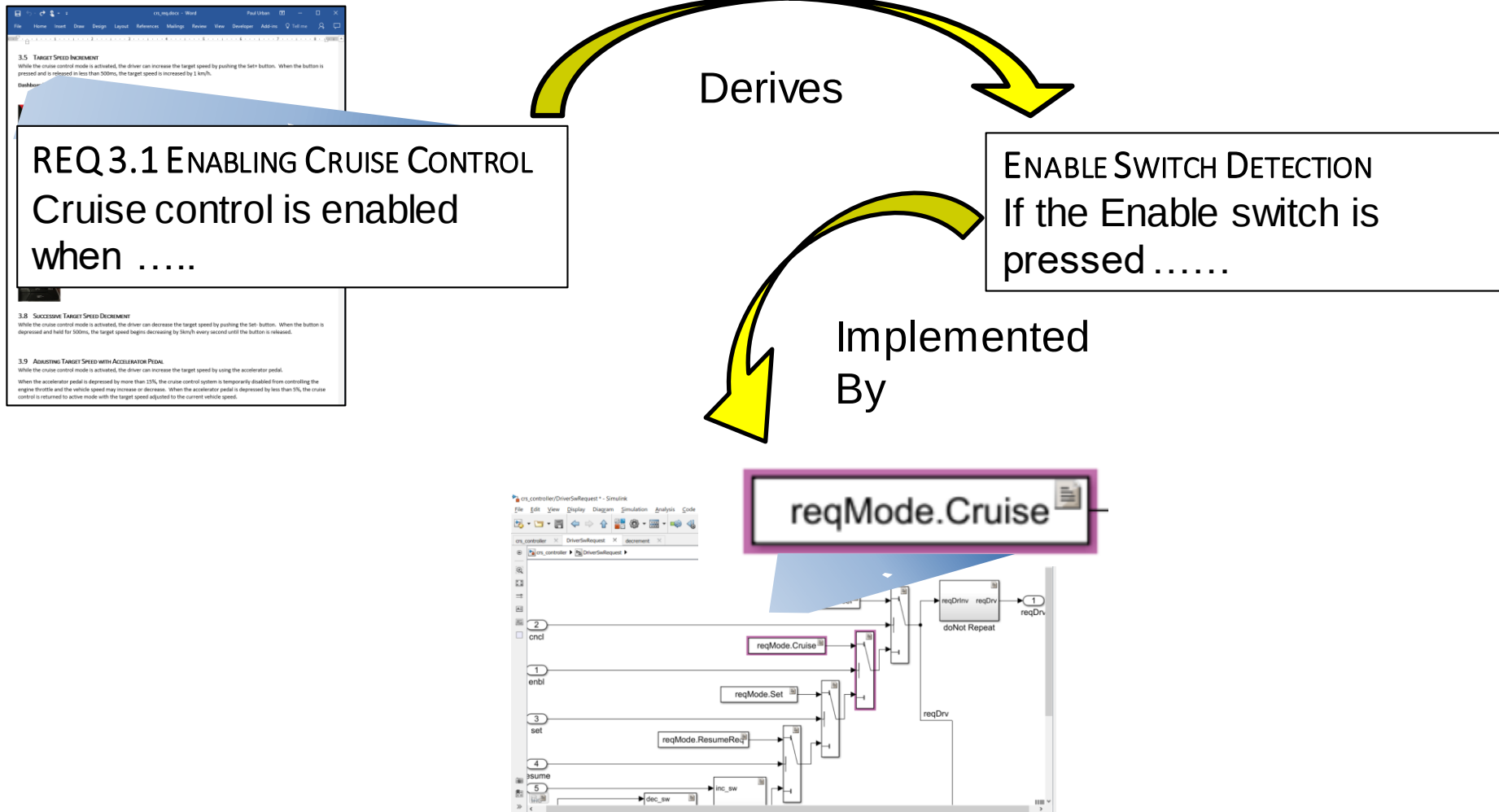


REQ 3.1 ENABLING CRUISE CONTROL
Cruise control is enabled when

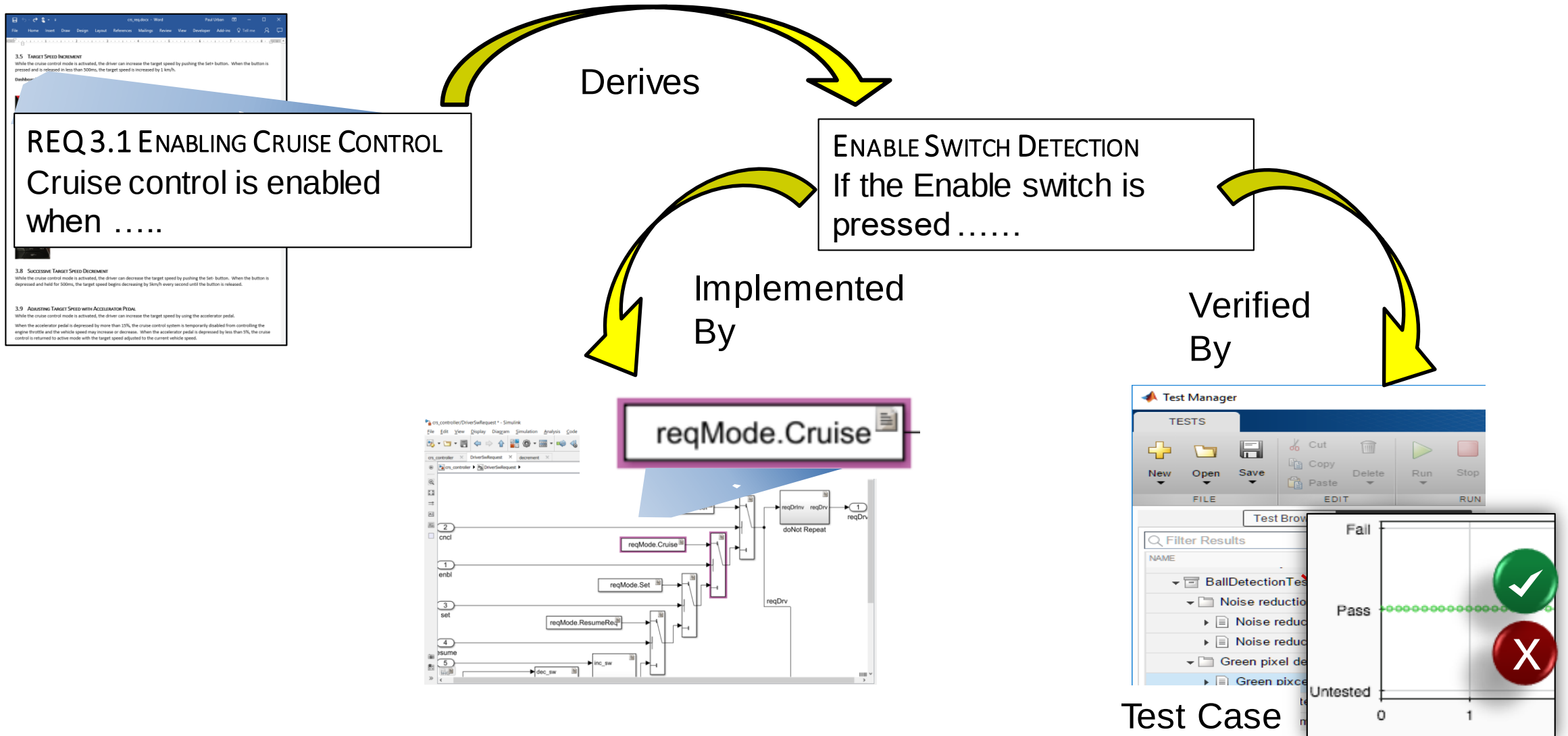
Link Requirements, Designs and Tests



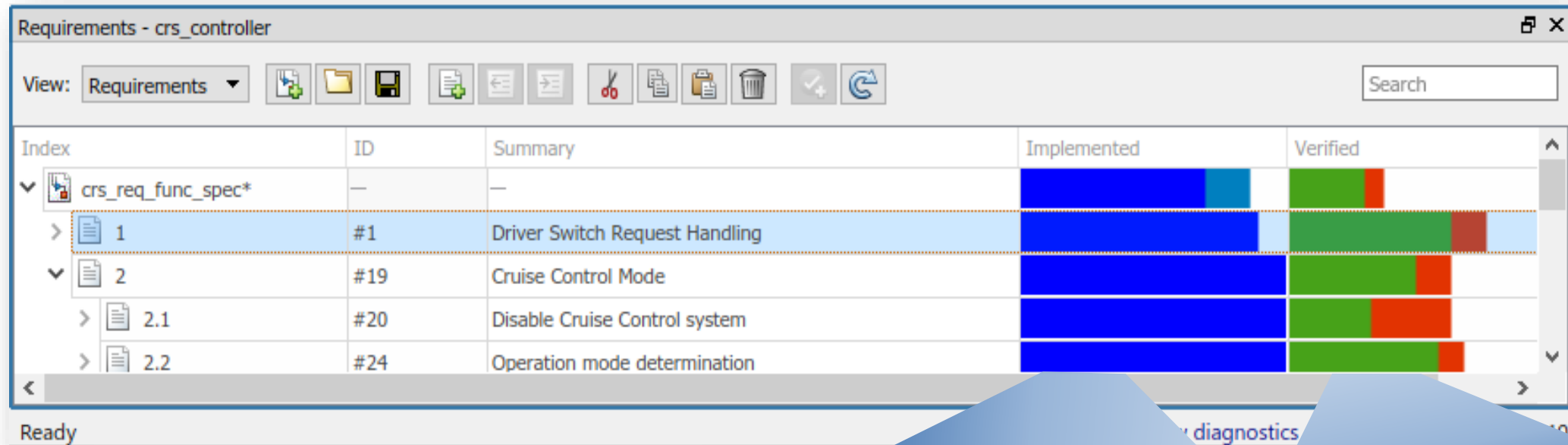
Link Requirements, Designs and Tests



Link Requirements, Designs and Tests



Track Implementation and Verification



The screenshot shows the 'Requirements - crs_controller' window. It features a toolbar with icons for file operations and a search bar. The main table lists requirements with columns for Index, ID, Summary, Implemented, and Verified. The 'Implemented' column uses blue bars to show the percentage of implemented requirements, while the 'Verified' column uses green, red, and orange bars to show verification results. Two callout boxes provide legends for the colors used in the bars.

Index	ID	Summary	Implemented	Verified
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

Verification Status

-  Passed
-  Failed
-  No Result
-  Missing

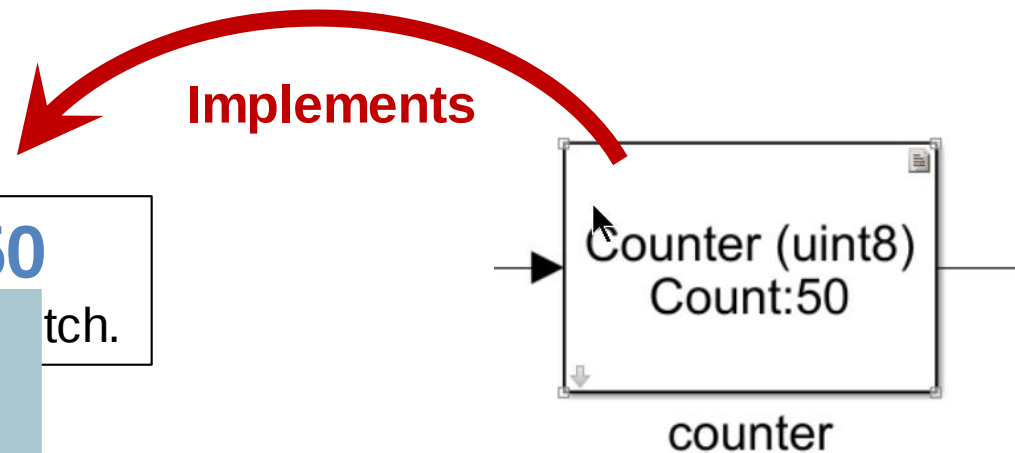
Respond to Change

Original Requirement

If the switch is pressed and the counter reaches **50** then it shall be recognized as a long press of the switch.

Updated Requirement

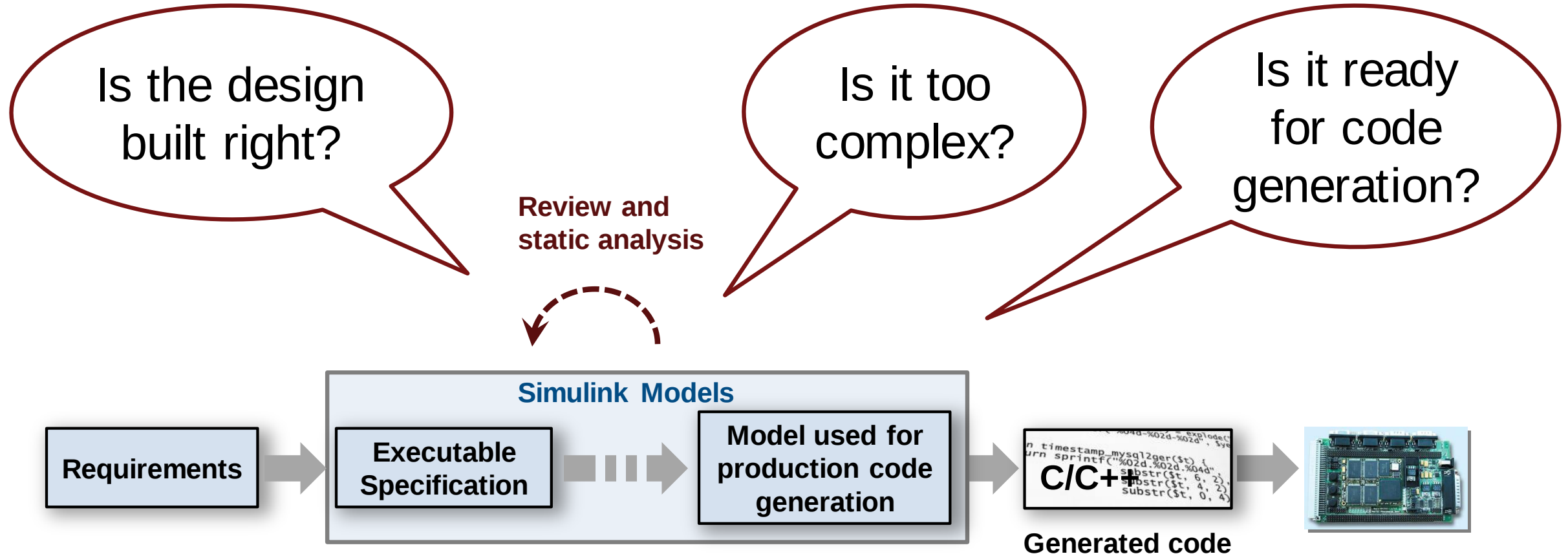
If the switch is pressed and the counter reaches **75** then it shall be recognized as a long press of the switch.



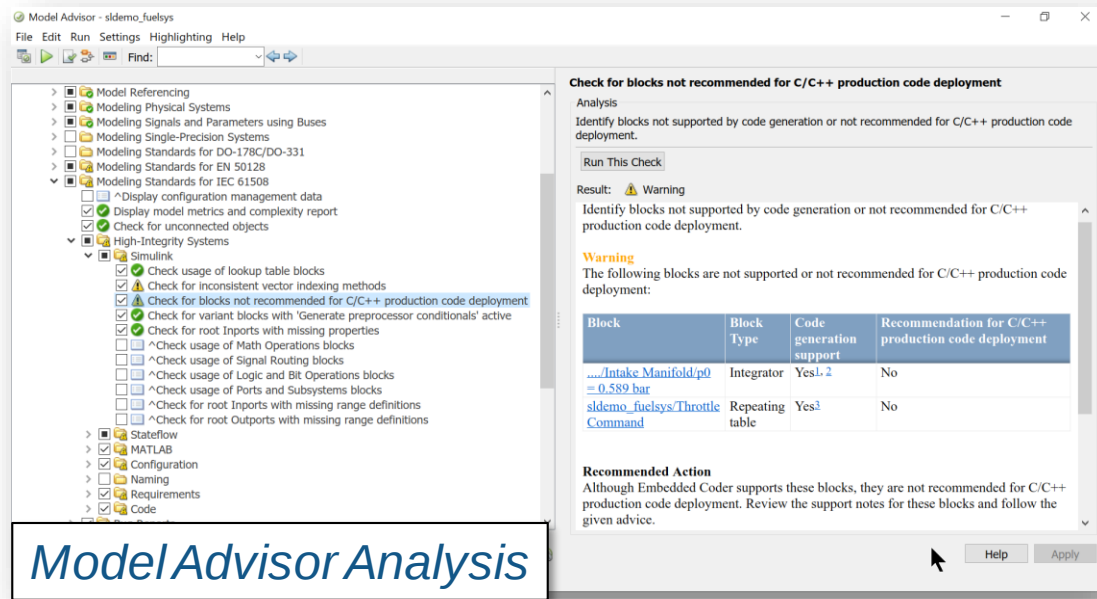
← **Implemented by:**
 counter

 **Issue: Destination Changed.**

Verify Design to Guidelines and Standards

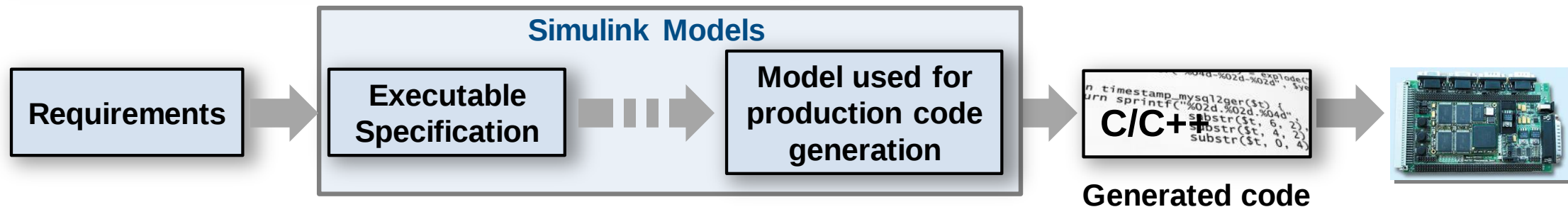


Automate verification with static analysis



Check for:

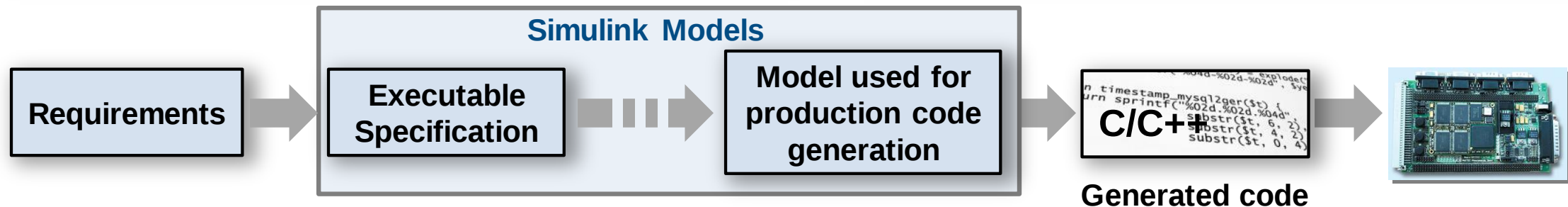
- Readability and Semantics
- Performance and Efficiency
- Clones
- And more.....



Generate reports for reviews and documentation

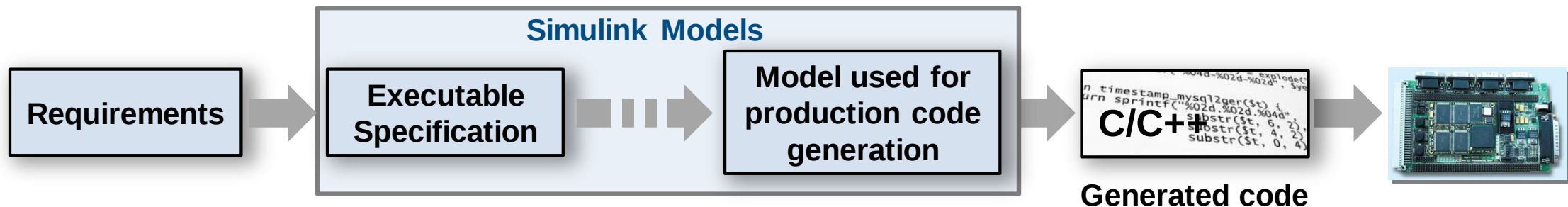
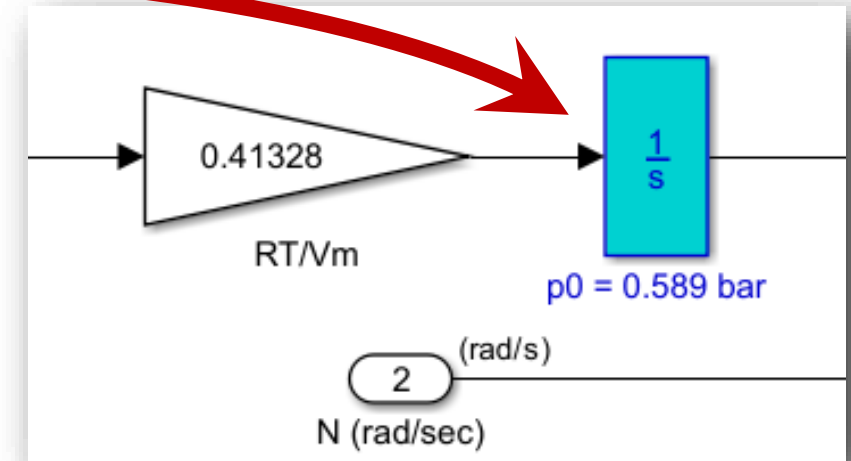
Model Advisor Analysis

Model Advisor Reports



Navigate to Problematic Blocks

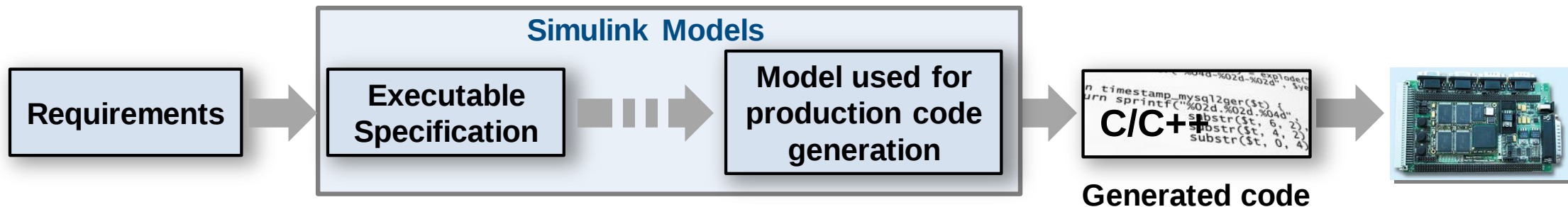
Block	Block Type	Code generation support	Recommendation for C/C++ production code deployment
.../Intake Manifold/p0 = 0.589 bar	Integrator	Yes ^{1, 2}	No
sldemo_fuelsys/Throttle Command	Repeating table	Yes ³	No



Guidance Provided to Address Issues or Automatically Correct

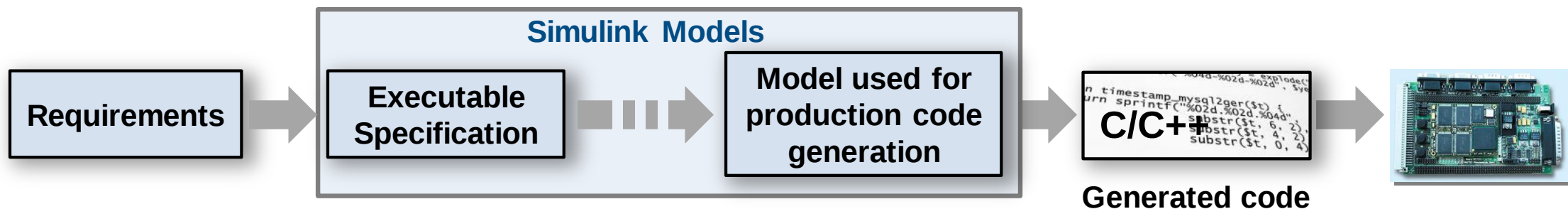
Recommended Action

Although Embedded Coder supports these blocks, they are not recommended for C/C++ production code deployment. Review the support notes for these blocks and follow the given advice.

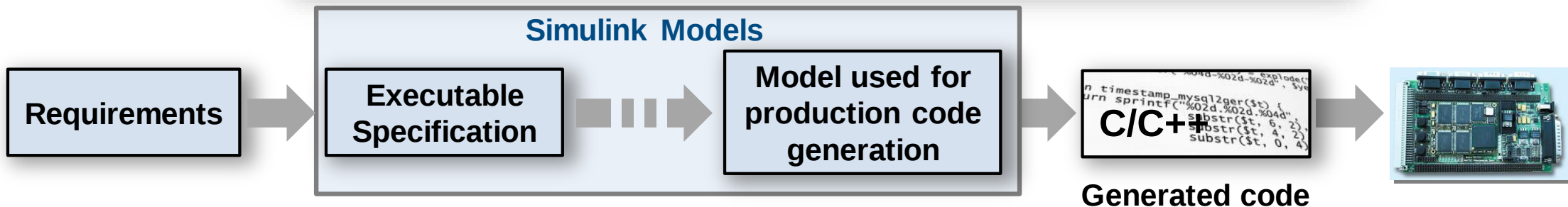


Built in checks for industry standards and guidelines

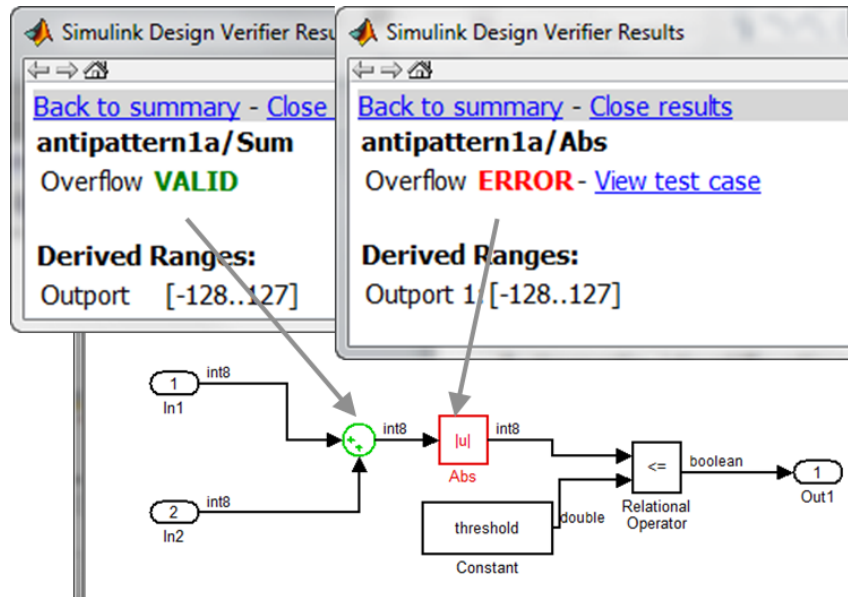
- DO-178/DO-331
- MISRA C:2012
- ISO 26262
- CERT C, CWE, ISO/IEC TS 17961
- IEC 61508
- MAAB (MathWorks Automotive Advisory Board)
- IEC 62304
- JMAAB (Japan MATLAB Automotive Advisory Board)
- EN 50128



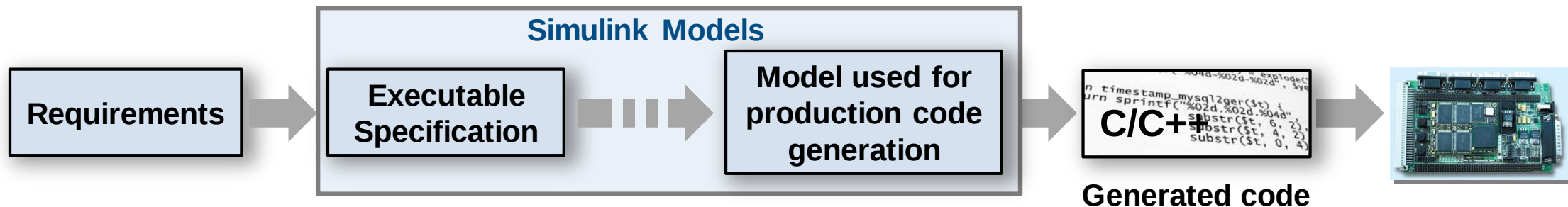
Configure and customize analysis



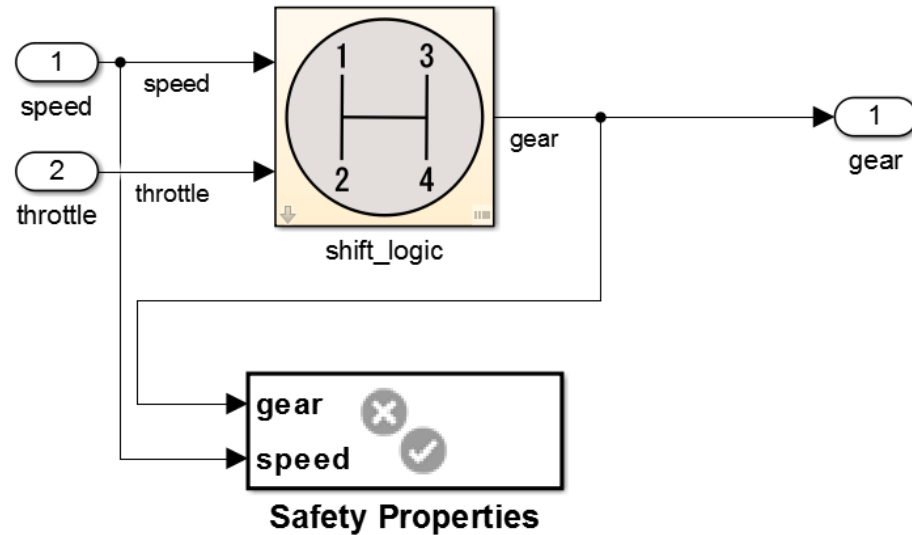
Detect Design Errors with Formal Methods



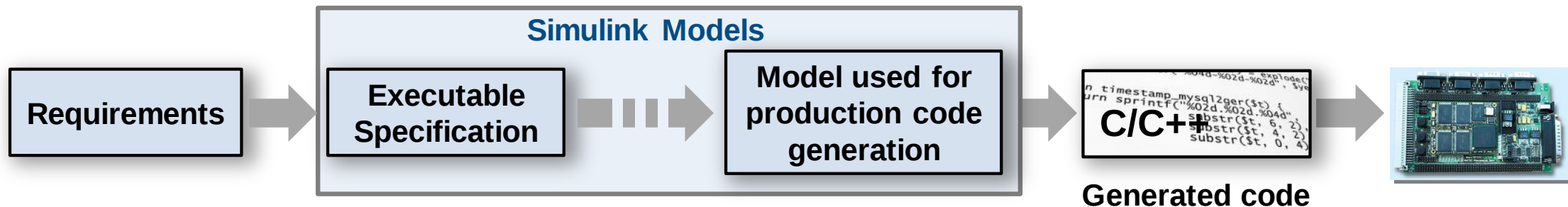
- Find run-time design errors:
 - Integer overflow
 - Dead Logic
 - Division by zero
 - Array out-of-bounds
 - Range violations
- Generate counter example to reproduce error



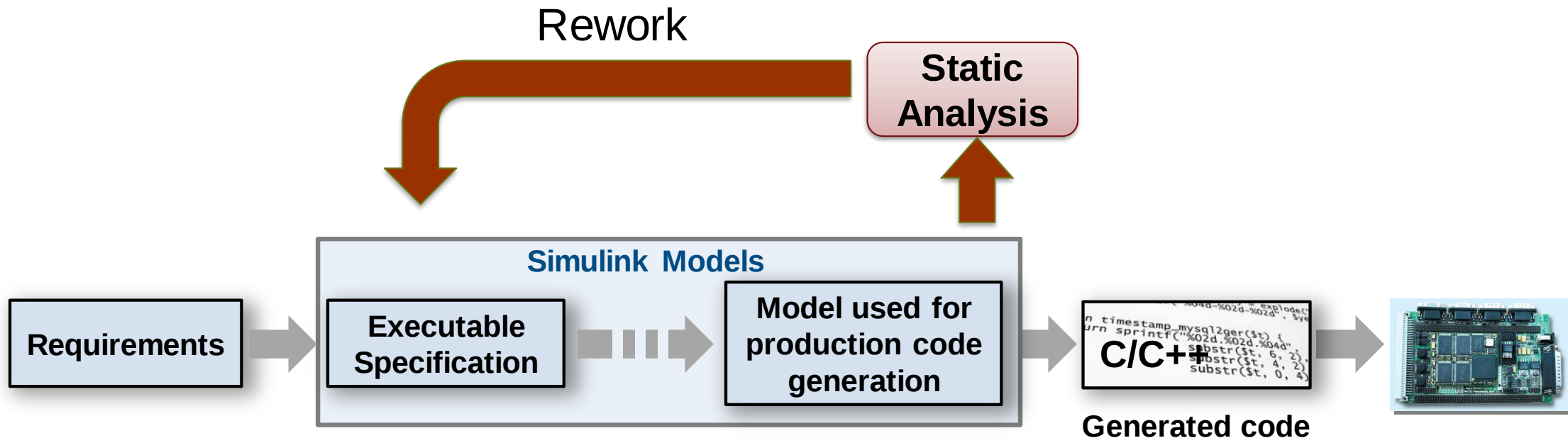
Prove That Design Meets Requirements



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

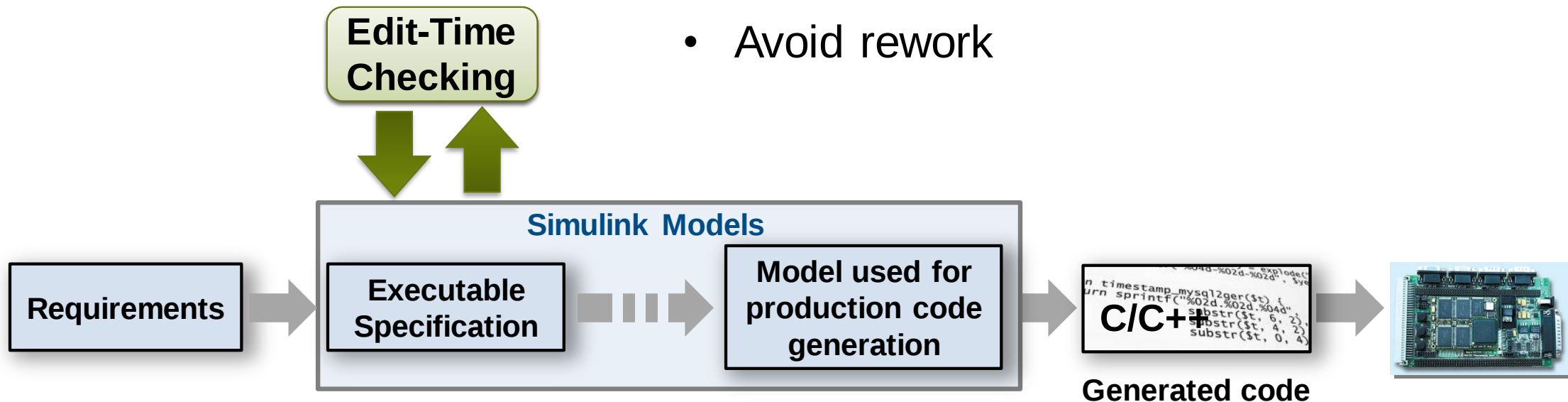


Checks for standards and guidelines are often performed late



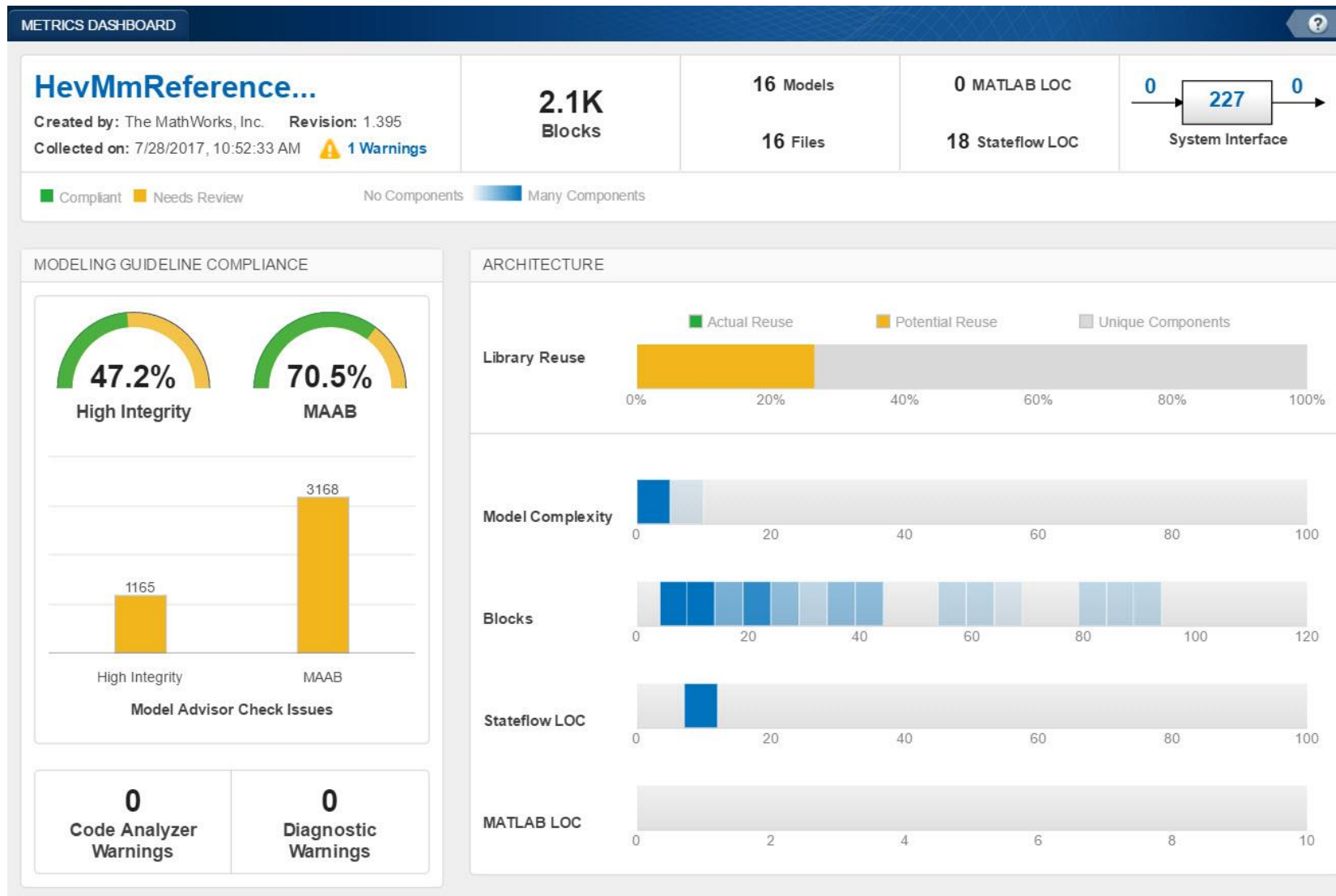
Shift Verification Earlier With Edit-Time Checking

- Highlight violations as you edit
- Fix issues earlier
- Avoid rework





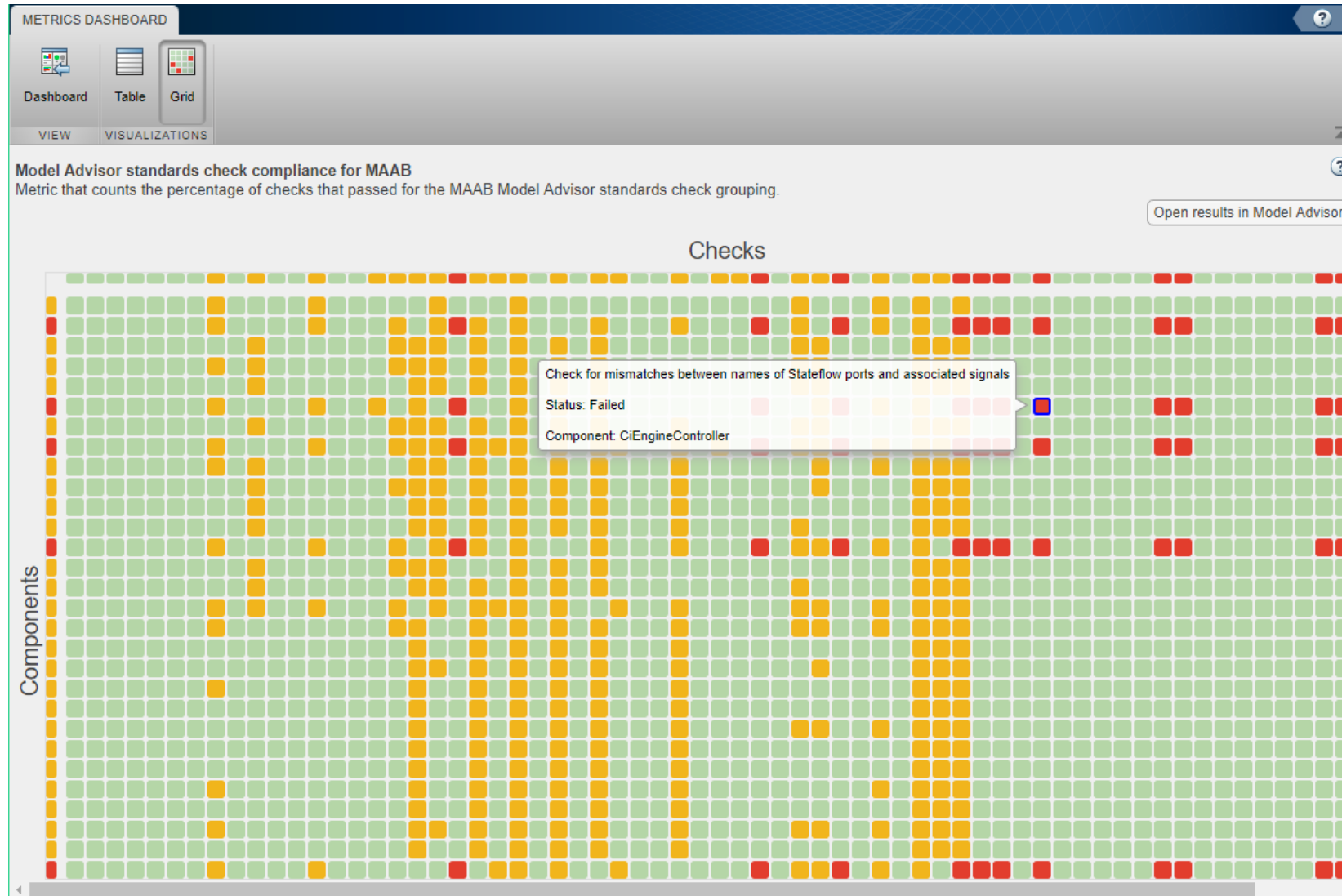
Assess Quality with Metrics Dashboard



- Consolidated view of metrics
 - Size
 - Compliance
 - Complexity
- Identify where problem areas may be

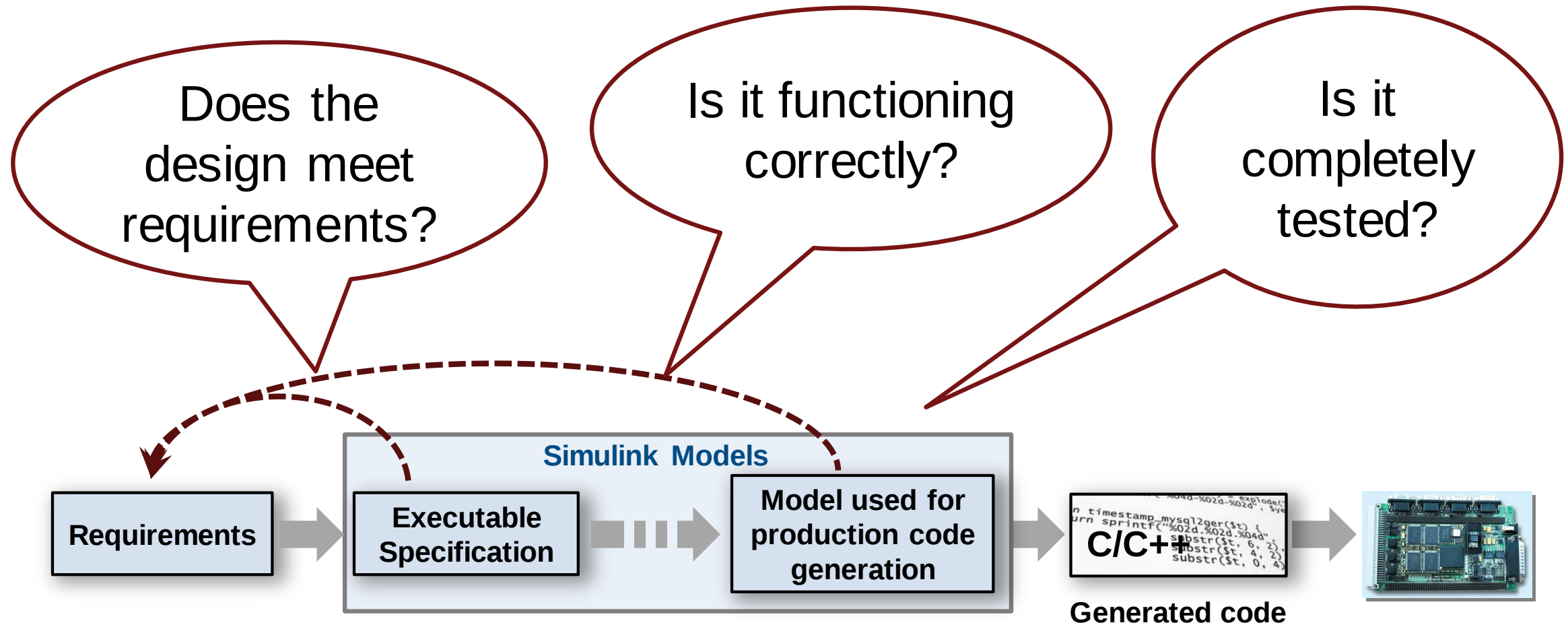
Grid Visualization for Metrics

R2018a



- Visualize Standards Check Compliance
 - Find Issues
 - Identify patterns
 - See hot spots

Functional Testing



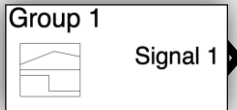
Systematic Functional Testing

Test Case

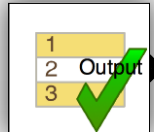
Inputs



MAT file (input)



Signal Builder



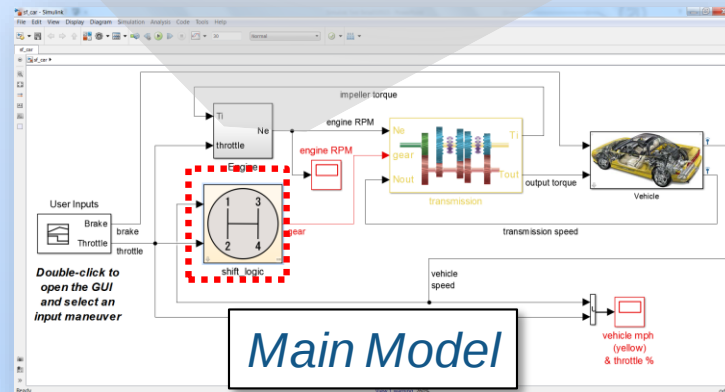
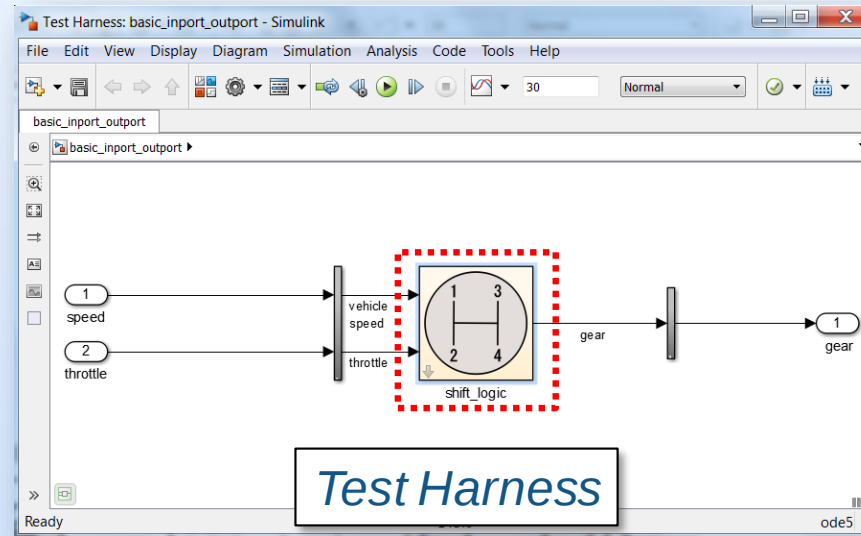
Test Sequence

and more!



Excel file (input)

R2017b



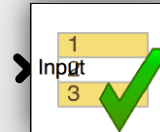
Assessments



MAT file (baseline)

```
function customCriteria
Perform custom criteria
1 test.verifyThat(test.sl
```

MATLAB Unit Test



Test Assessment

and more!



Excel file (baseline)

R2017b

Manage Testing and Test Results

Test Manager

TESTS

FILE EDIT RUN RESULTS RESOURCES

Test Browser Results and Artifacts

Start Page x Slow Accel x

Filter Tests

- ComponentTesting
 - General Performance Test
 - Functional and Regression tests
 - Signal Builder Baseline examples
 - Slow Accel
 - Fast Accel
 - Decel
 - ExcelDrivenExamples
 - Software-in-the-loop Testing
 - SystemTesting
 - ExampleBaselineTesting

Slow Accel

ComponentTesting > Functional and Regression tests > Signal Builder Baseline examples > Slow Accel

Baseline Test

DESCRIPTION

REQUIREMENTS

SYSTEM UNDER TEST

PARAMETER OVERRIDES

CALLBACKS

INPUTS

OUTPUTS

CONFIGURATION SETTINGS OVERRIDES

BASLINE CRITERIA

SIGNAL NAME	ABS TOL	REL TOL
✓ SlowAccelbaselineCheckpoint1.mat	0	0.00 %

PROPERTY VALUE

Name	Slow Accel
Type	Baseline Test
Location	C:\Users\monell\Desktop\...
Enabled	✓
Hierarchy	ComponentTesting > Fu...
Model	st_car
Simulation Mode	[Model Settings]
Harness Name	SigBdriven

Test Manager

TESTS VISUALIZE FORMAT

Clear Plot Data Cursors Highlight in Model Send to Figure

EDIT ZOOM & PAN MEASURE & TRACE SHARE

Test Browser Results and Artifacts

Start Page x Slow Accel x Comparison x

Filter Results

NAME	STATUS
Results : 2015-Jan-12 17:35:31	2 ✓ 1 ✗
Signal Builder Baseline examples	2 ✓ 1 ✗
Slow Accel	✓
Fast Accel	✗
Baseline Criteria Result	✗
gear	✗
throttle	✗
vehicle speed	✗
Sim Output (sf_car : normal)	
Decel	✓

PROPERTY VALUE

Name	gear
Status	✗
Absolute Tolerance	0
Relative Tolerance	0.00 %
Block Path	SigBdriven/shift_logic

fourth
third
second
first
None

Baseline Compare To

0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30

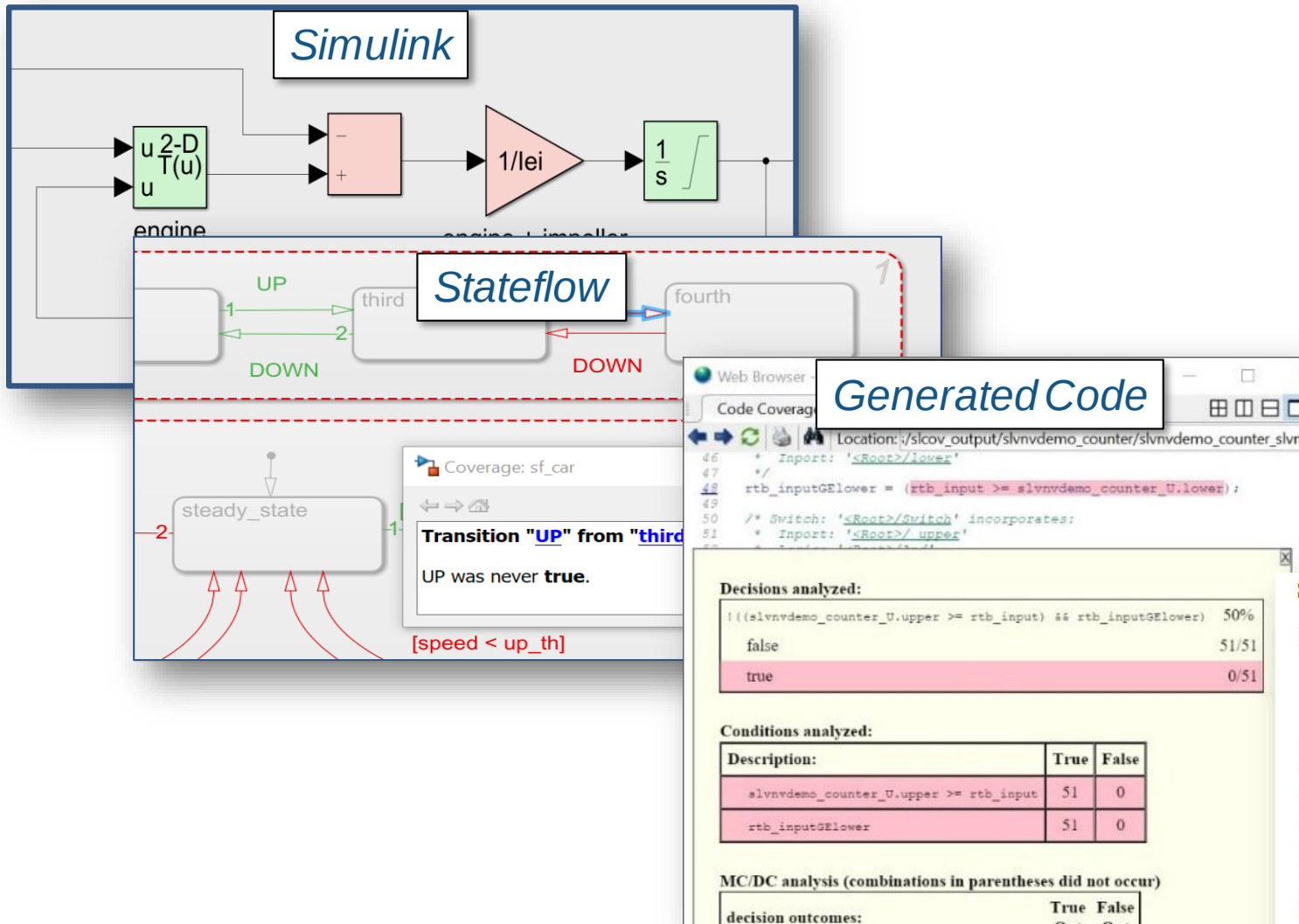
1.0
0.8
0.6
0.4
0.2
0

Tolerance Difference

0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30

Coverage Analysis to Measure Testing

- Identify testing gaps
- Missing requirements
- Unintended Functionality
- Design Errors

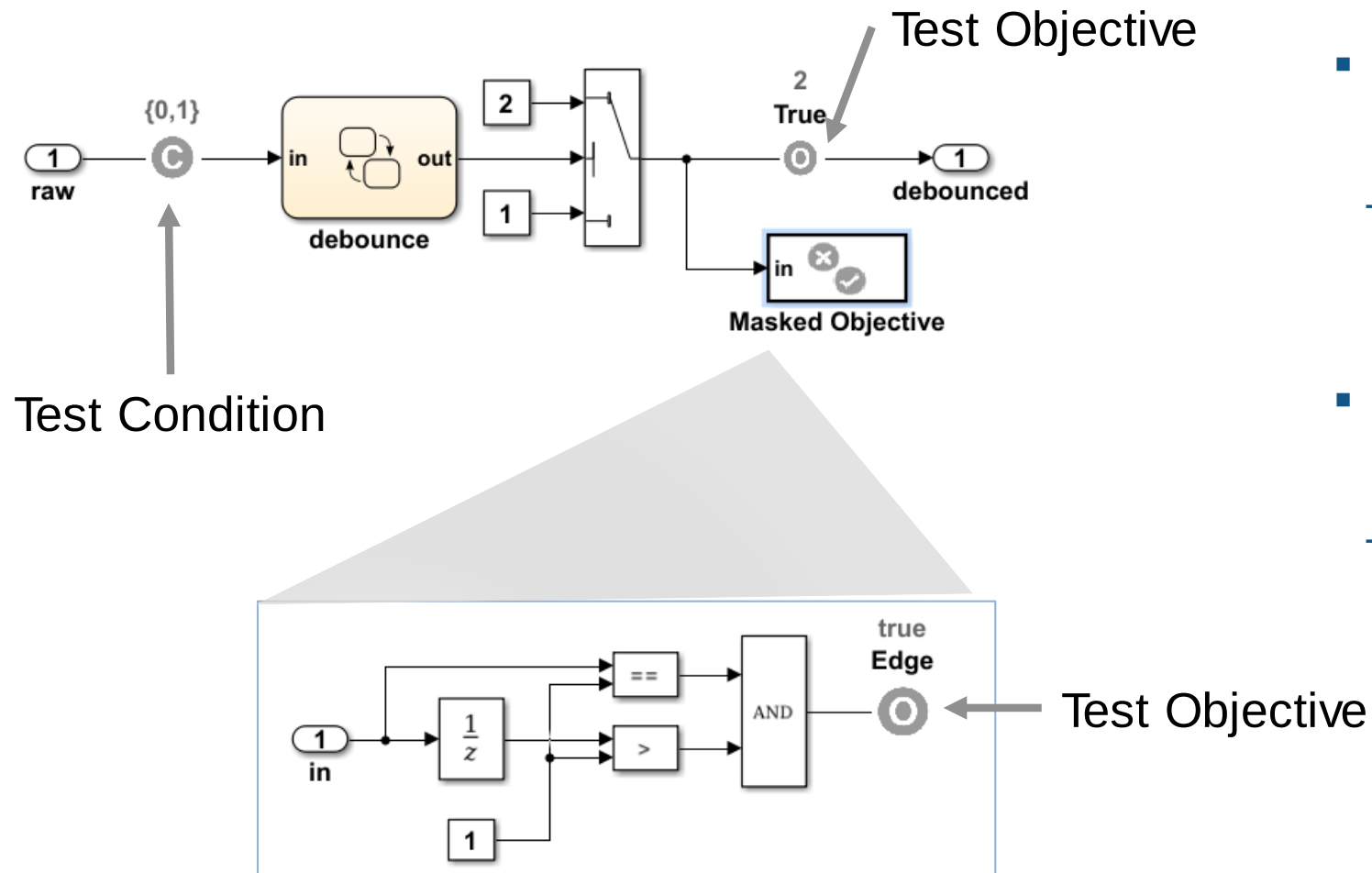


Summary

Coverage Reports

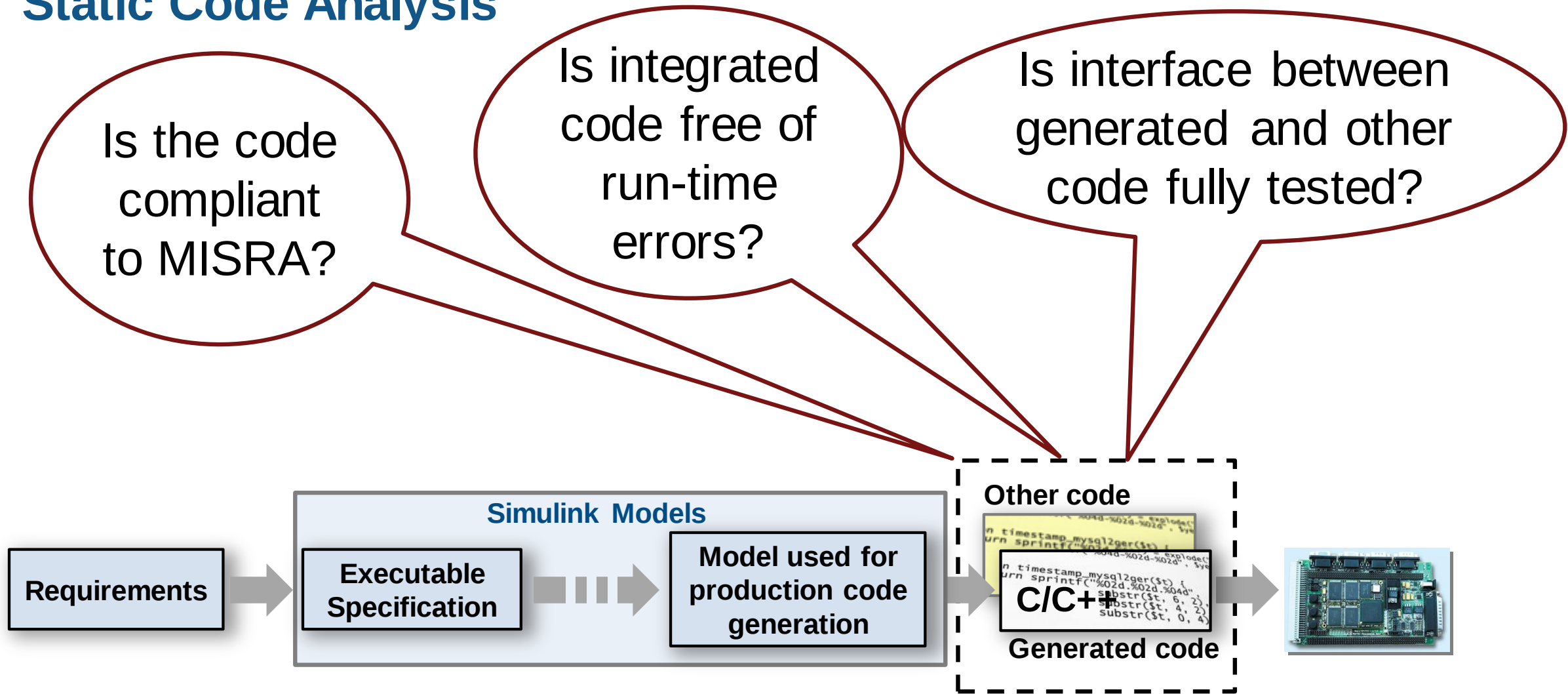
Model Hierarchy/Complexity	Test 1	Decision	Condition	MCDC	Execution	Relational Boundary	Saturation on integer overflow
1. slidemo_fuelsys	80	34%	34%	7%	90%	10%	50%
2. Engine Gas Dynamics	13	71%	NA	NA	100%	50%	50%
3. Mixing & Combustion	3	67%	NA	NA	100%	NA	50%
4. EGO Sensor	2	100%	NA	NA	NA	NA	NA
5. System Lag	NA	NA	NA	NA	100%	NA	NA
6. Throttle & Manifold	10	73%	NA	NA	100%	50%	50%
7. Intake Manifold	2	100%	NA	NA	100%	NA	50%
8. MATLAB Function	2	100%	NA	NA	NA	NA	NA
9. Throttle	6	83%	NA	NA	100%	100%	50%

Test Case Generation for Functional Testing



- Specify functional test objectives
 - Define custom objectives that signals must satisfy in test cases
- Specify functional test conditions
 - Define constraints on signal values to constrain test generator

Static Code Analysis



The Generated Code is integrated with Other Code (Handwritten)

Static Code Analysis with Polyspace

- Code metrics and standards
 - Comment density, cyclomatic complexity,...
 - MISRA and Cybersecurity standards
 - Support for DO-178, ISO 26262,
- Bug finding and code proving
 - Check data and control flow of software
 - Detect bugs and security vulnerabilities
 - Prove absence of runtime errors

Green: reliable
safe pointer access

Red: faulty
out of bounds error

Gray: dead
unreachable code

Orange: unproven
may be unsafe for some conditions

Purple: violation
MISRA-C/C++ or JSF++
code rules

Range data
tool tip

```
static void pointer_arithmetic (void) {
    int array[100];
    int *p = array;
    int i;

    for (i = 0; i < 100; i++) {
        *p = 0;
        p++;
    }

    if (get_bus_status() > 0) {
        if (get_oil_pressure() > 0) {
            *p = 5;
        } else {
            i++;
        }
    }

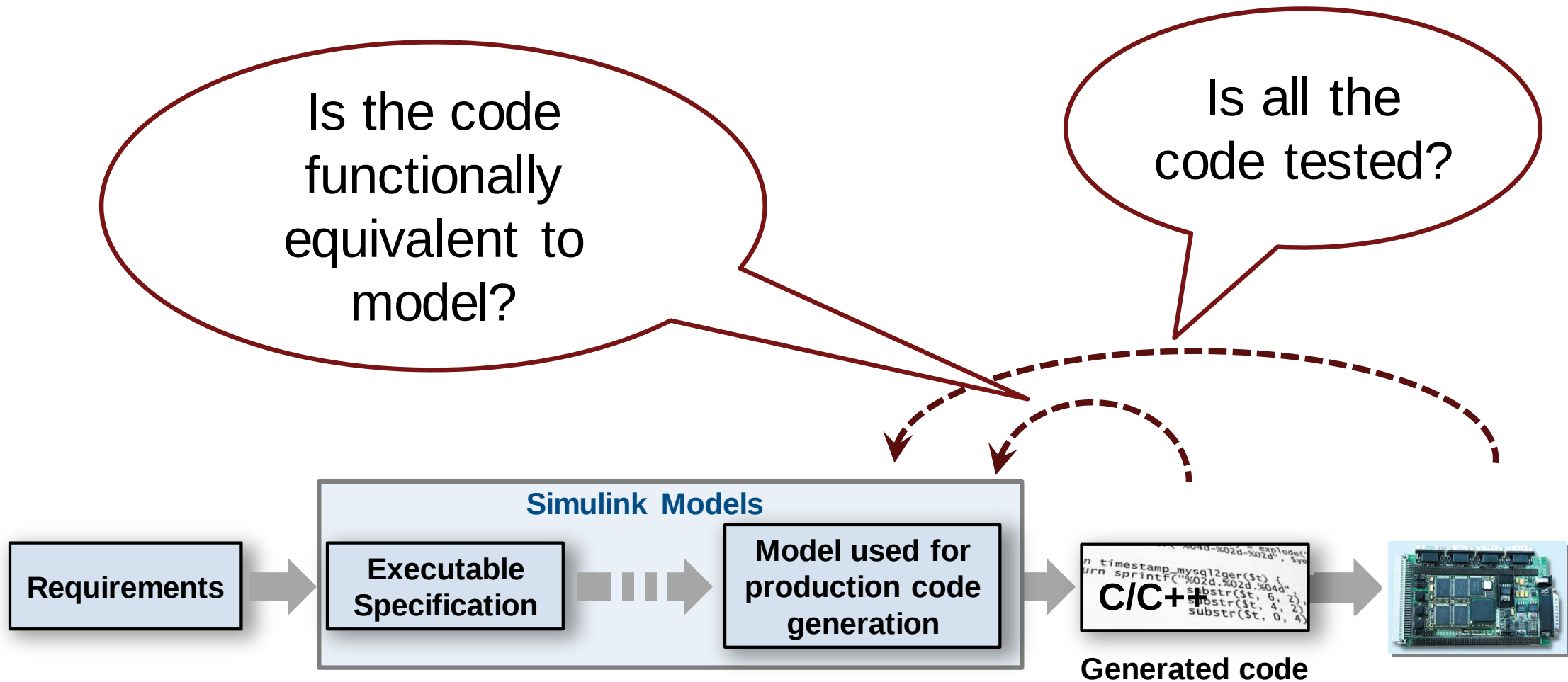
    i = get_bus_status();

    if (i >= 0) {
        *(p - i) = 10;
    }
}
```

variable 'i' (int32): [0 .. 99]
assignment of 'i' (int32): [1 .. 100]

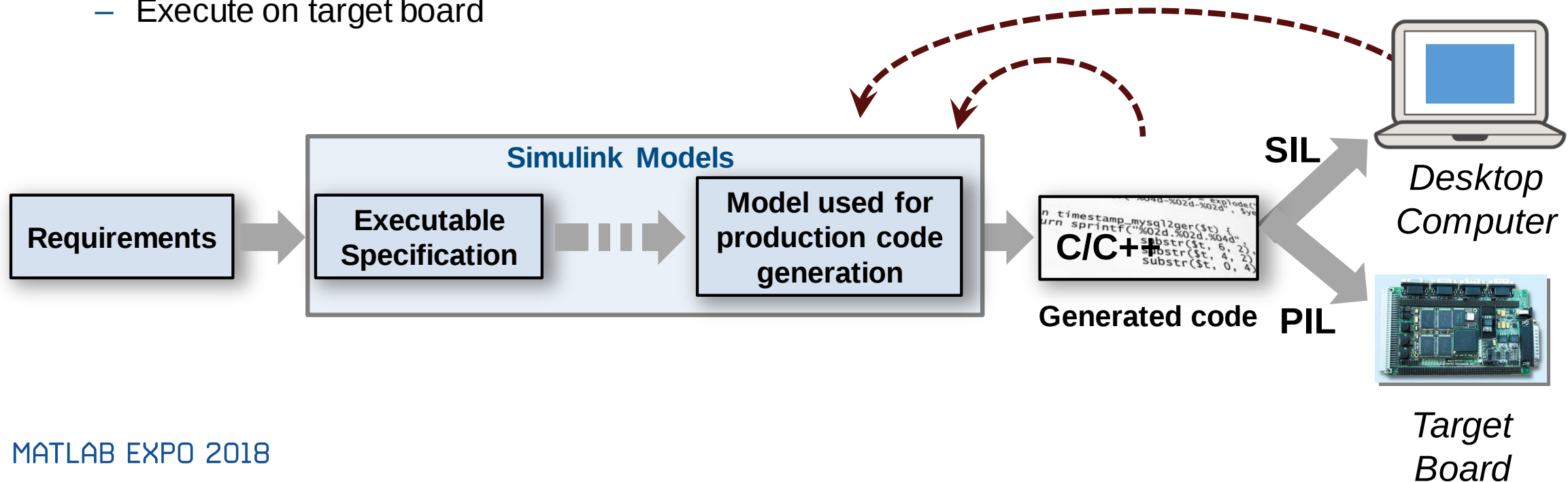
Results from Polyspace Code Prover

Equivalence Testing



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



Qualify tools with IEC Certification Kit and DO Qualification Kit

- Qualify code generation and verification products
- Includes documentation, test cases and procedures

KOSTAL Asia R&D Center Receives ISO 26262 ASIL D Certification for Automotive Software Developed with Model-Based Design



Kostal's electronic steering column lock module.

BAE Systems Delivers DO-178B Level A Flight Software on Schedule with Model-Based Design



Primary flight control computers from BAE Systems.

Lear Delivers Quality Body Control Electronics Faster Using Model-Based Design

Challenge

Design, verify, and implement high-quality automotive body control electronics

Solution

Use Model-Based Design to enable early and continuous verification via simulation, SIL, and HIL testing

Results

- Requirements validated early. Over 95% of issues fixed before implementation, versus 30% previously
- Development time cut by 40%. 700,000 lines of code generated and test cases reused throughout the development cycle
- Zero warranty issues reported

MATLAB EXPO 2018

[Link to user story](#)



Lear automotive body electronic control unit.

"We adopted Model-Based Design not only to deliver better-quality systems faster, but because we believe it is a smart choice. Recently we won a project that several of our competitors declined to bid on because of its tight time constraints. Using Model-Based Design, we met the original delivery date with no problem."

- Jason Bauman, Lear Corporation

Customer References and Applications



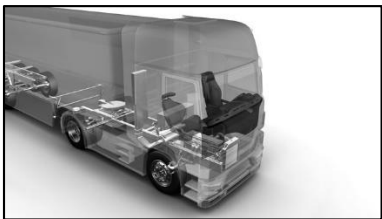
Airbus Helicopters Accelerates Development of DO-178B Certified Software with Model-Based Design

Software testing time cut by two-thirds



LS Automotive Reduces Development Time for Automotive Component Software with Model-Based Design

Specification errors detected early



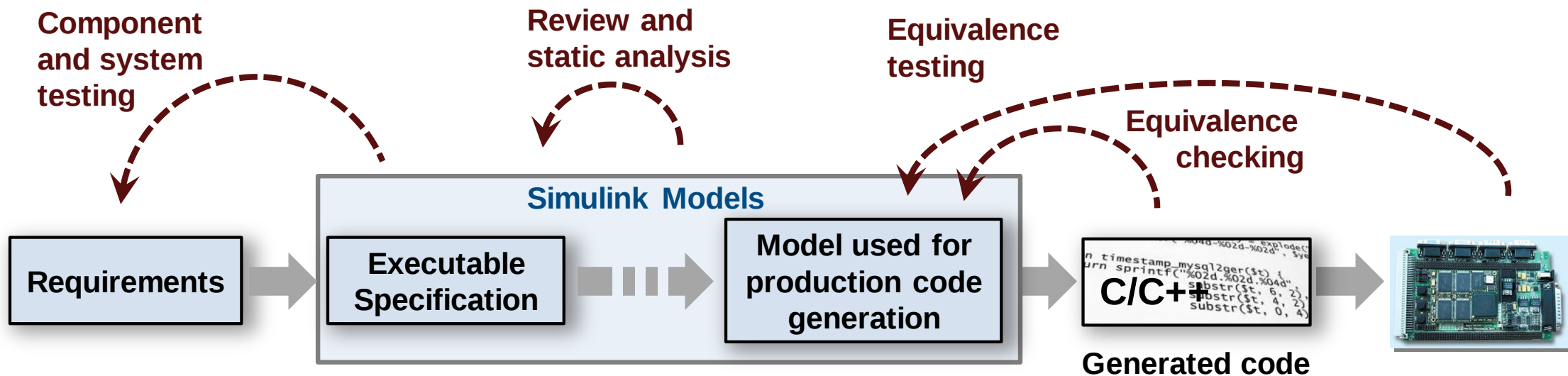
Continental Develops Electronically Controlled Air Suspension for Heavy-Duty Trucks

Verification time cut by up to 50 percent

More User Stories: www.mathworks.com/company/user_stories.html

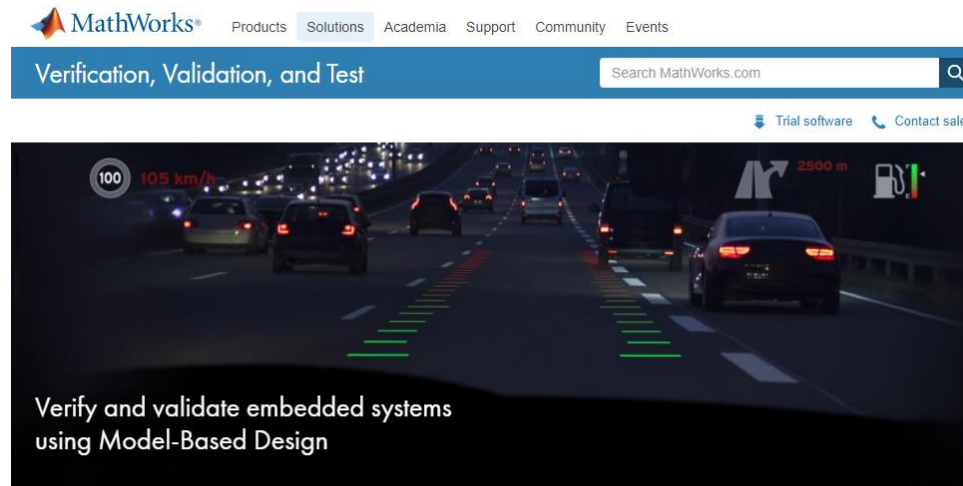
Summary

1. Author and manage requirements within Simulink
2. Find defects earlier
3. Automate manual verification tasks
4. Reference workflow that conforms to safety standards



Learn More

Visit MathWorks Verification, Validation and Test Solution Page:
mathworks.com/solutions/verification-validation.html



Engineering teams use [Model-Based Design](#) with MATLAB® and Simulink® to verify and validate embedded systems. Teams author requirements directly in their models and can then use those models to generate production code for certification.

- **Author requirements in your model**, and verify and trace them to the design, tests, and code.
- Prove that your design **meets requirements**, and **automatically generate tests**.
- **Check compliance** of models and code using static analysis and formal methods.
- Find bugs, security vulnerabilities, and **prove the absence of critical run-time errors**.
- Produce reports and artifacts, and **certify to standards** (such as DO-178 and ISO 26262).

Thank You!