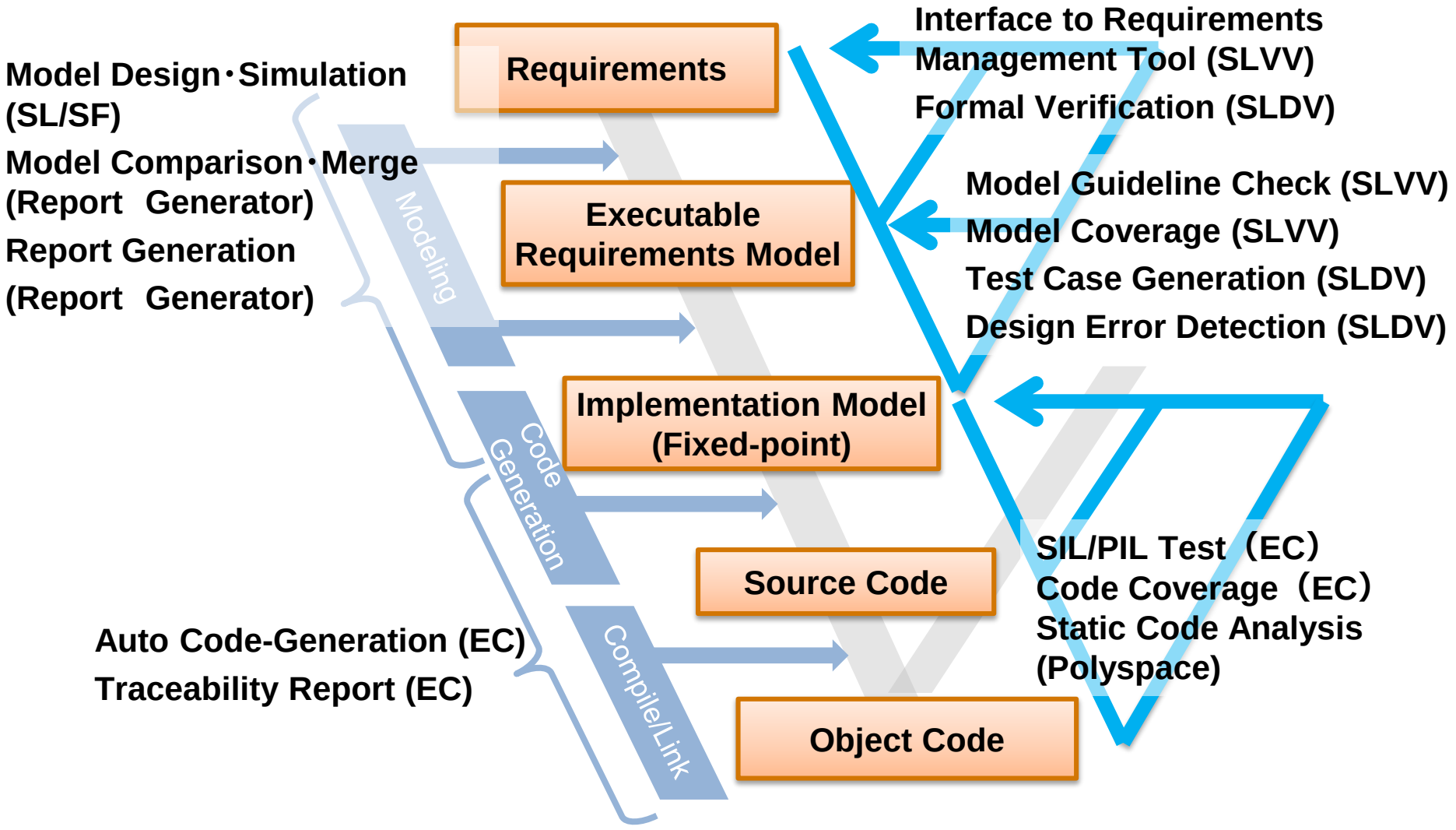
Verification and Validation Solutions for High Integrity Systems

Tiffany Liang
Application Engineer
MathWorks

Recommended Workflow

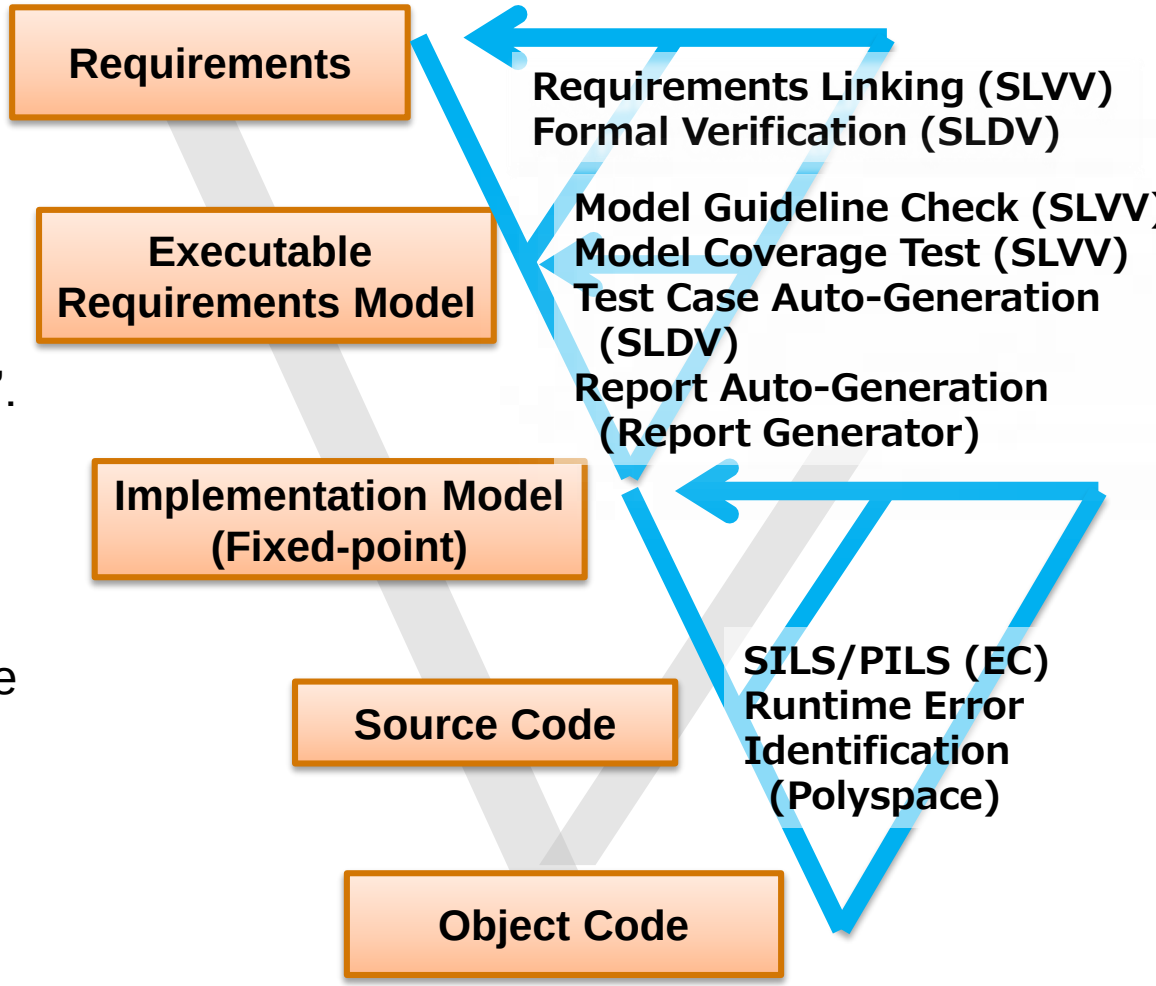
Detecting errors early in the development cycle



MathWorks benefits
Early verification and Validation

・Able to form small V-loops
・Able to detect errors early in the development cycle

- Model ⇔ Code consistency allows for Simulink simulation results to be considered “truth”.
- Early model verification is possible due to the ability to investigate floating-point models
- Large team development made easy through highly customizable tool chain
- Errors in object code detected easily through synchronization between simulations and SILS/PILS



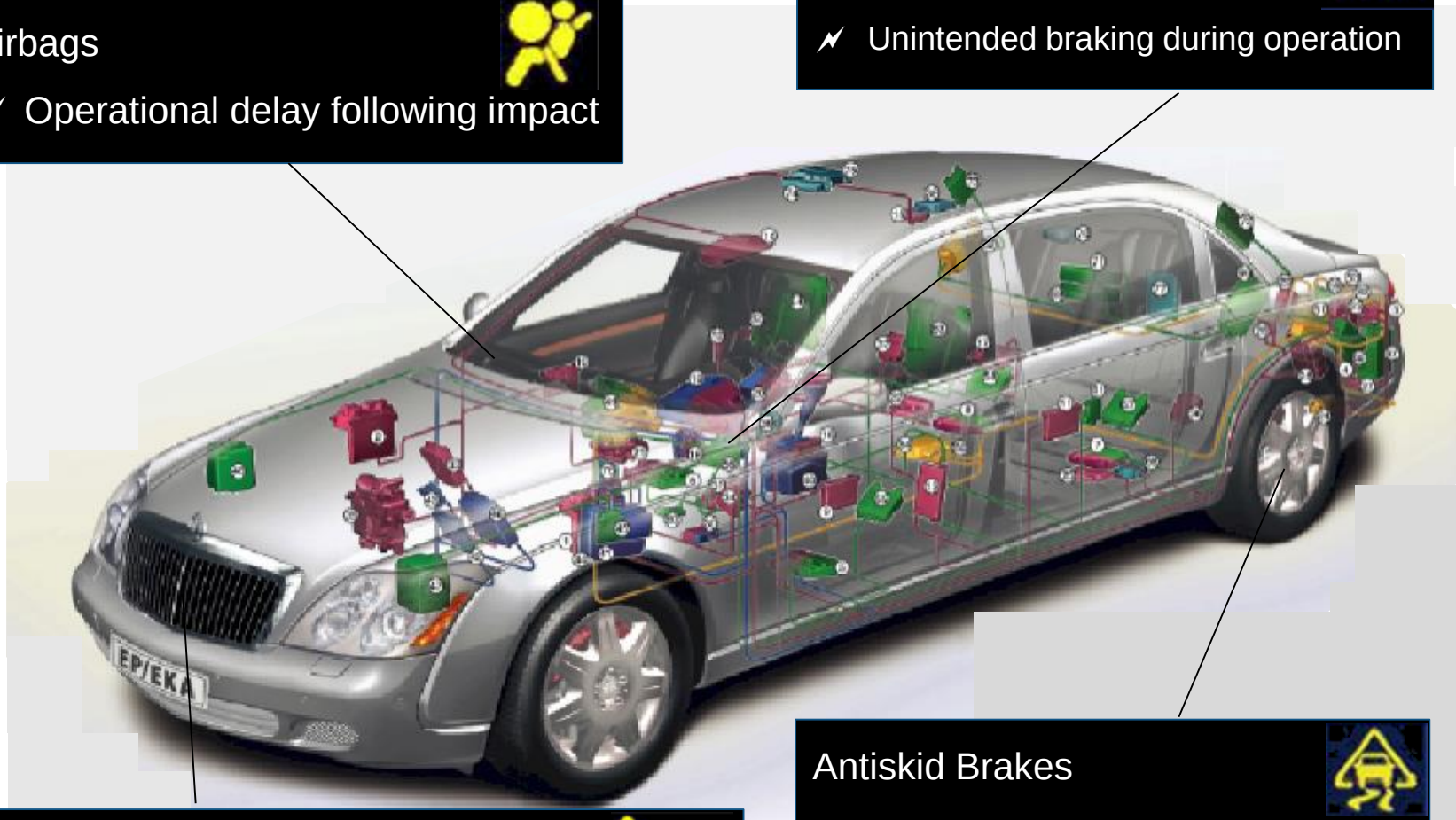
Examples of High Reliability Applications

Airbags

- Operational delay following impact

Electronic Parking Brake

- Unintended braking during operation



Antiskid Brakes

- Unintended asymmetrical braking

Vehicle-to-vehicle distance control

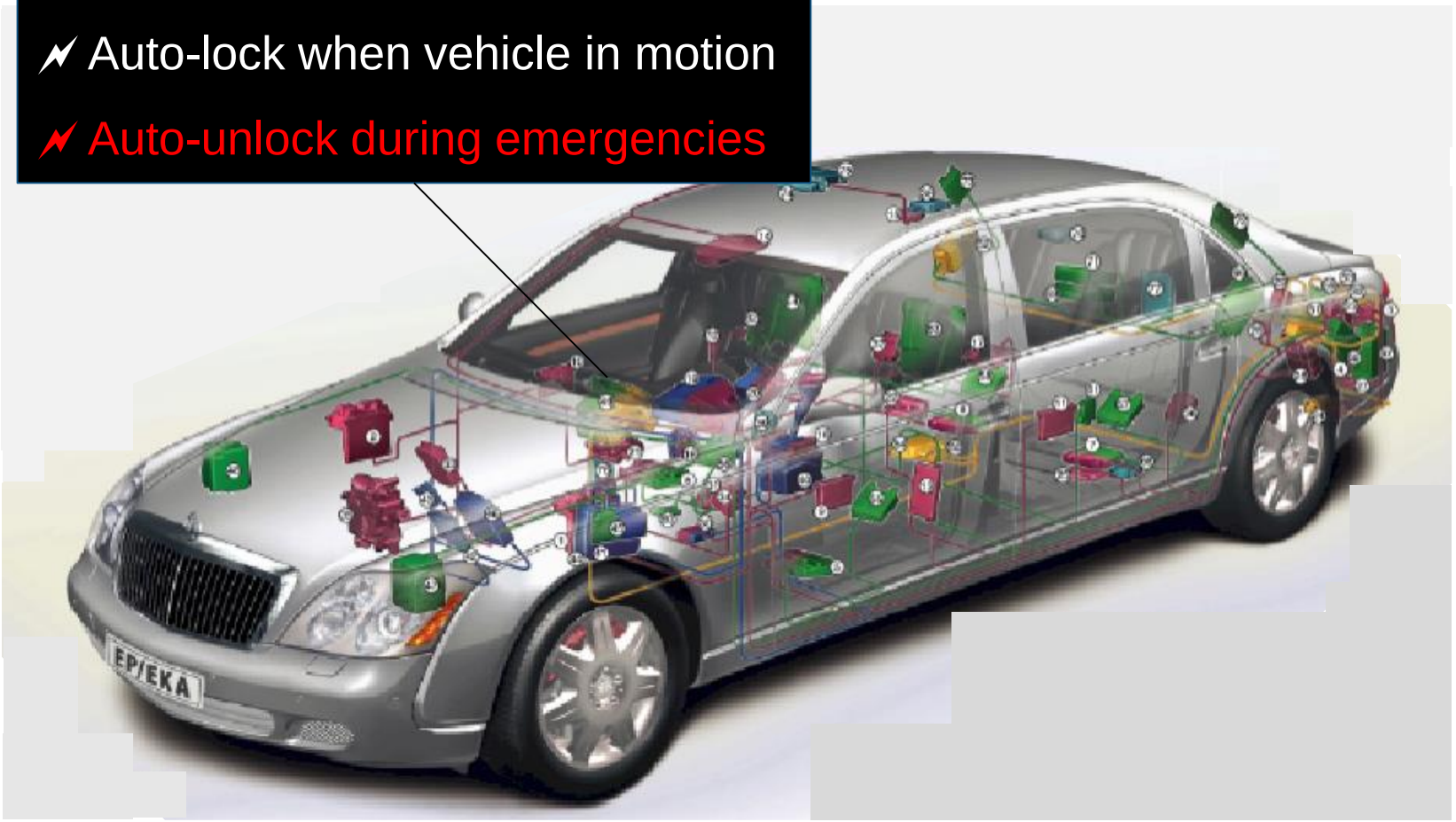
- Insufficient deceleration within required time

Source: K. Grimm: Software-Technologie in der Automoboilindustrie. 19. STEV-OESTERREICH-FACHTAGUNG IT-/Softwarequalitätsmanagment in der Praxis, Wien, Austria, May 2004

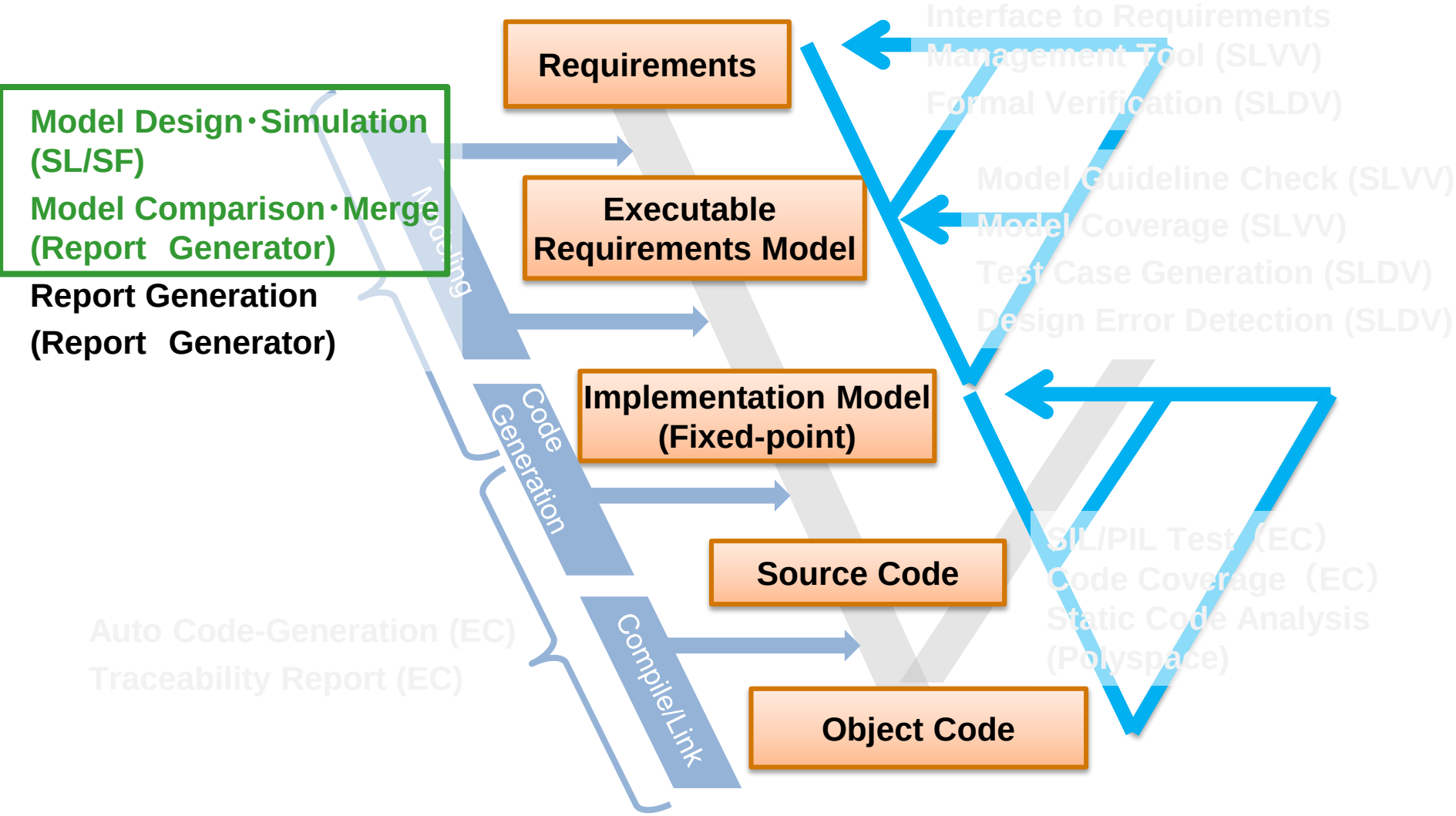
Example: Door Lock Control System

Door Lock Control

- Auto-lock when vehicle in motion
- Auto-unlock during emergencies



Our First Topic



Door Lock Control Software Requirements

1. Task Rate Requirements

REQ101 – The software shall execute as a 100ms task rate.

2. Initialization Requirements

REQ201 – The software shall initialize controls in the Unlock state.

3. Diagnosis Requirements 

REQ301 – The software shall determine the lock state of each door based on the lock positions.

- Lock position is under 1mm：Unlock state
- Lock position is over 4mm：Lock state
- Otherwise：Neutral state

REQ302 – The software shall determine the overall vehicle lock state based on all door lock positions.

- All doors in lock position: Lock state
- All doors in unlock position: Unlock state

REQ303 – The software shall determine the overall vehicle lock state to be in failure state due to lock failure in the case where there is no response to a door lock request in under 2 seconds .

4. Door Lock Request Requirements 

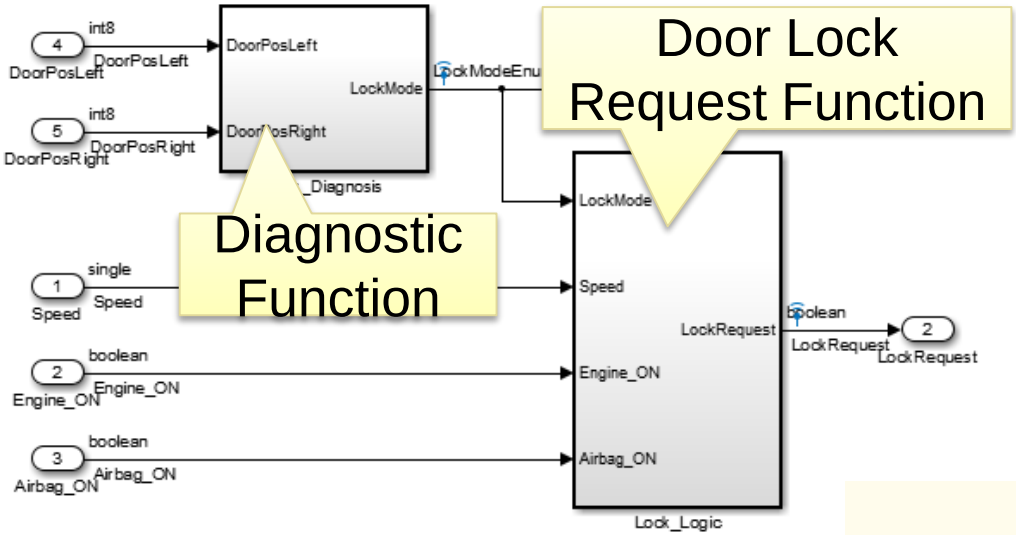
REQ401 – The doors shall automatically lock when the vehicle speed is above 5km/h for over 2 seconds and the engine is operating

REQ402 – The door locks shall automatically release after the airbags deploy.

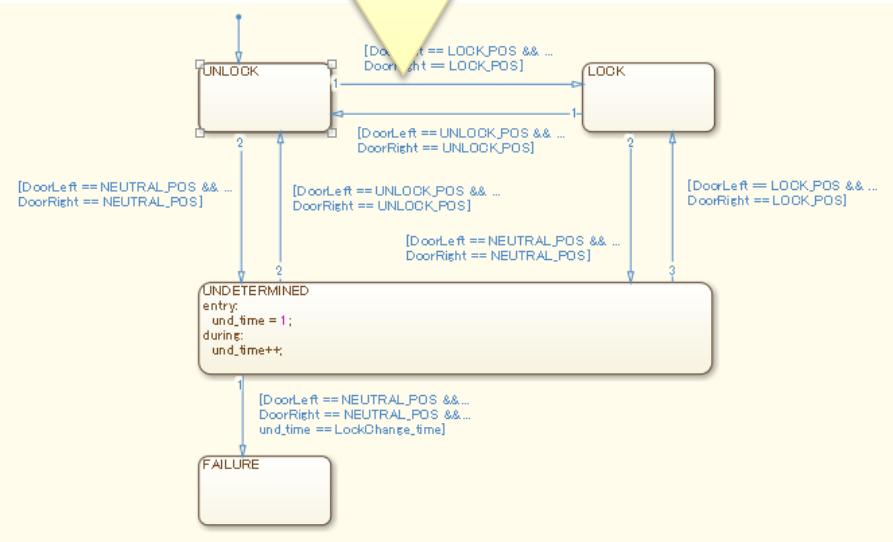
Door Lock Model

Simulink / Stateflow

Increased Readability / Productivity through Graphical Modeling



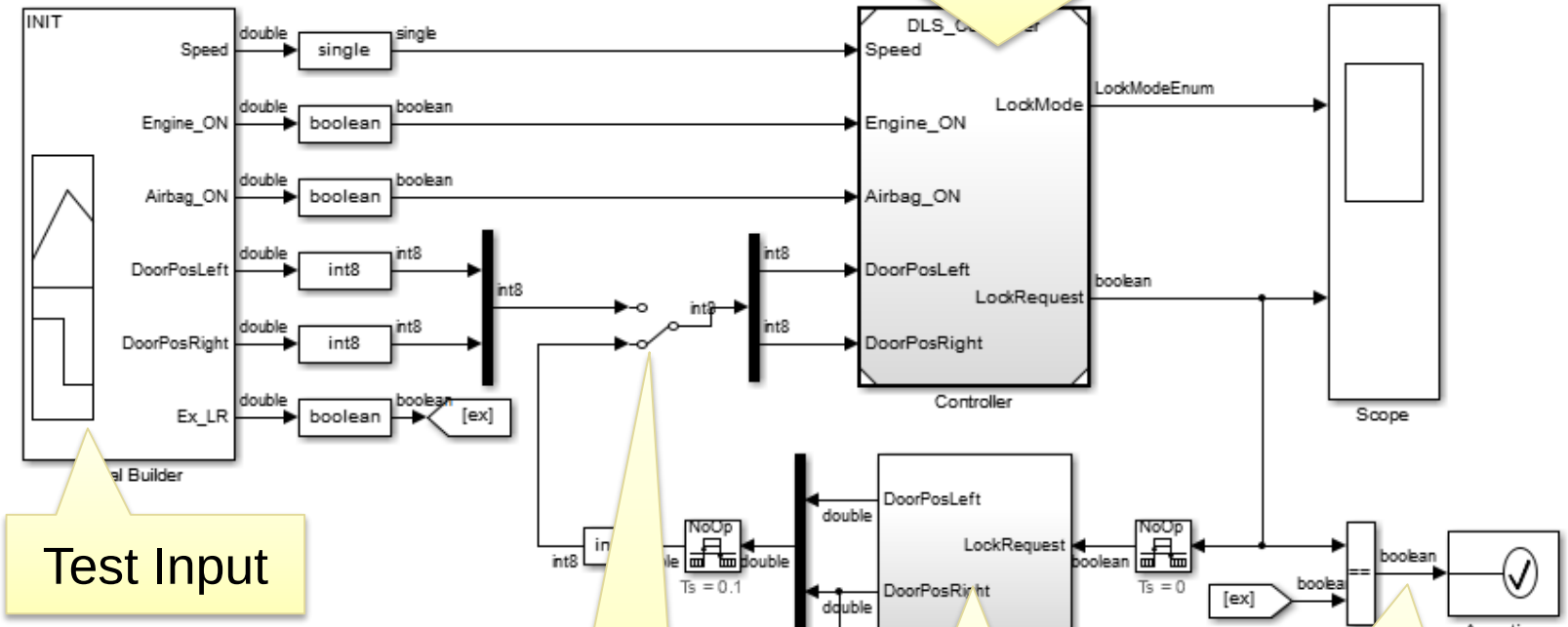
Diagnostic Function State Transition Diagram



Door Lock Test Model

Simulink / Simscape

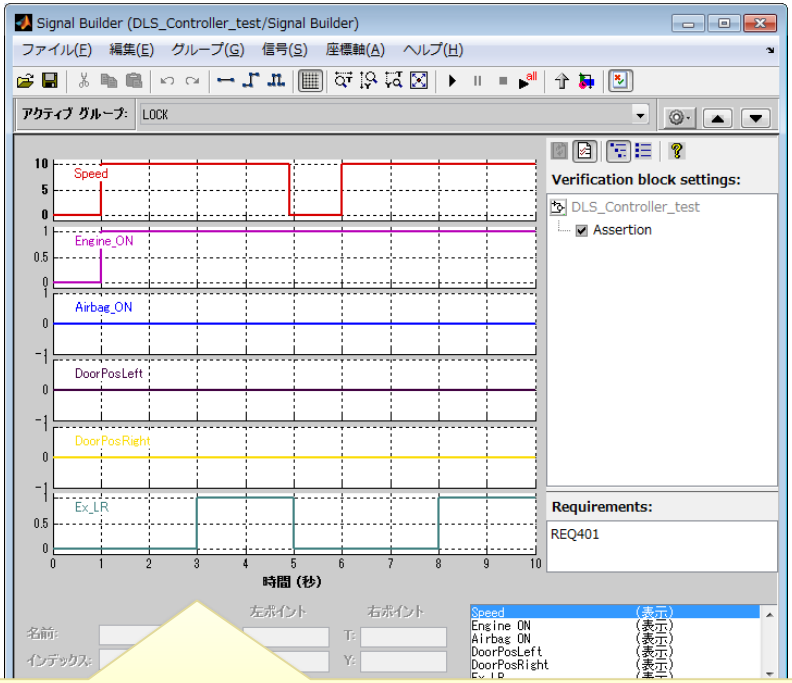
Able to execute various tests using the control model



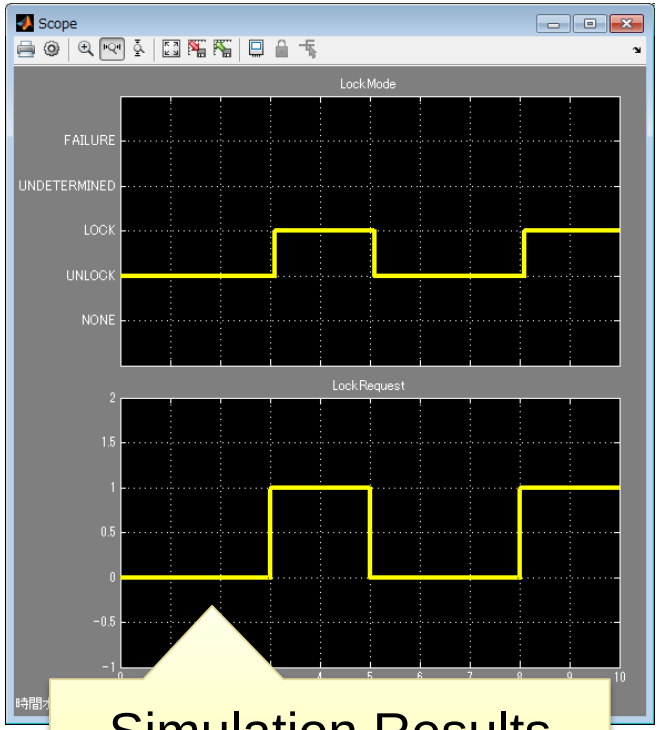
Requirements & Logic Testing through Simulation

Simulink / Stateflow

- Early verification of entire system incl. plant behavior
- Investigation of failure/anomaly modes (difficult on H/W)



Test data definition in Signal Builder

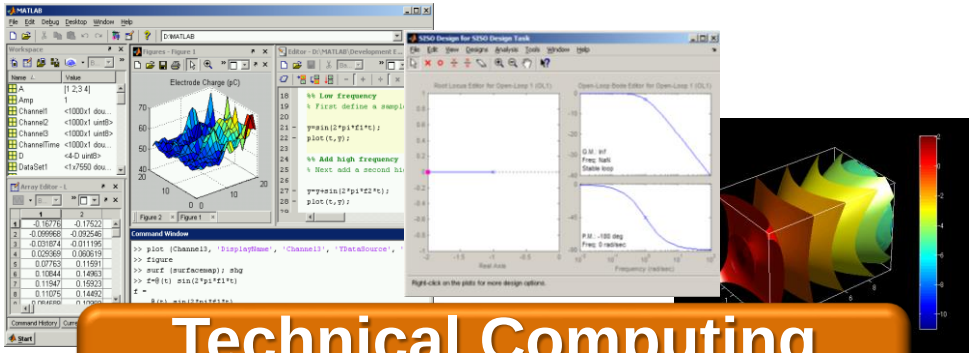


Simulation Results

MATLAB/Simulink Products

MATLAB

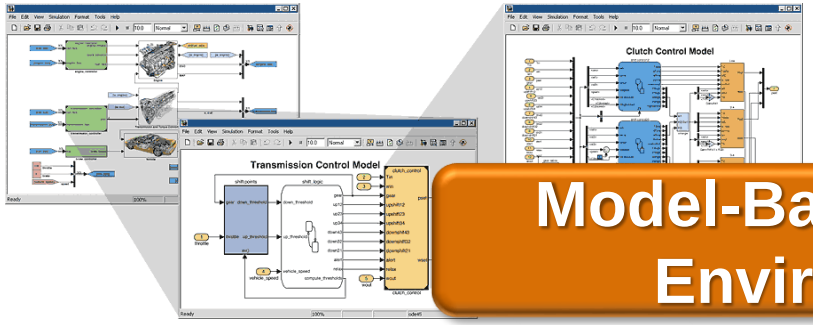
- Easy data processing
- Concise programming language
- Abundant mathematical functions・file I/O
- 2-D/3-D visualization functionality



Technical Computing Environment

Simulink

- Block diagram modeling
- Abundant block library
- High-precision time simulation



Stateflow

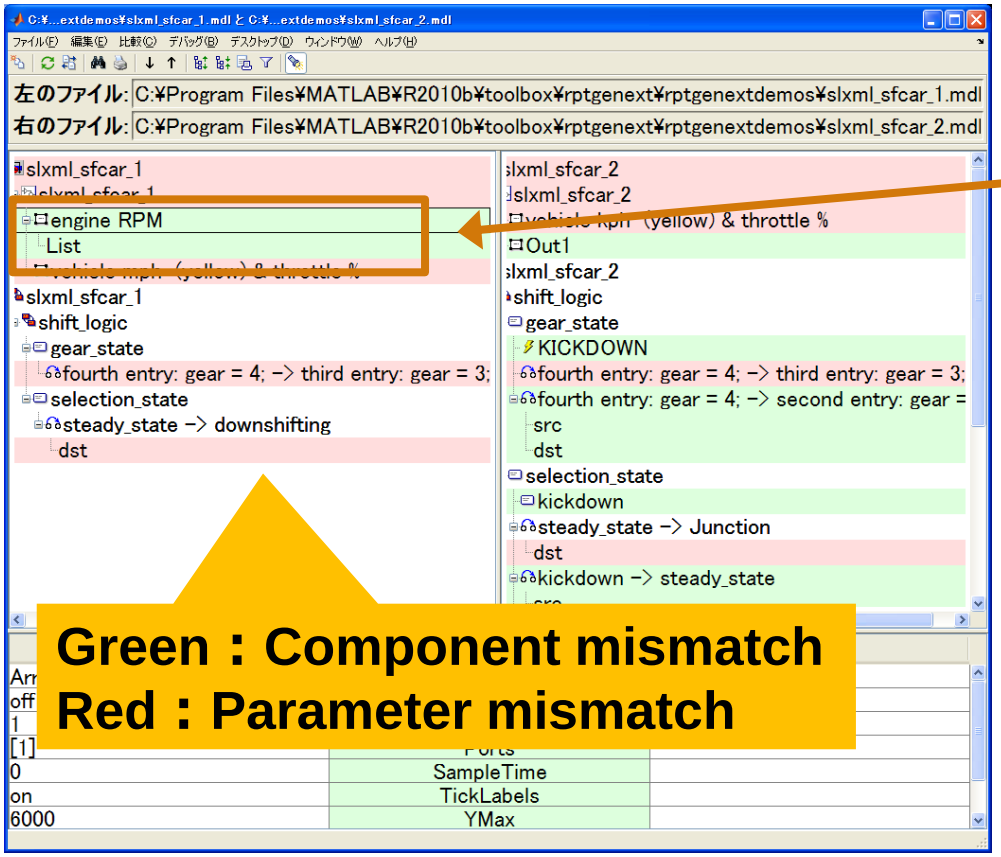
- Flowcharts, State Diagrams, State Transition Tables

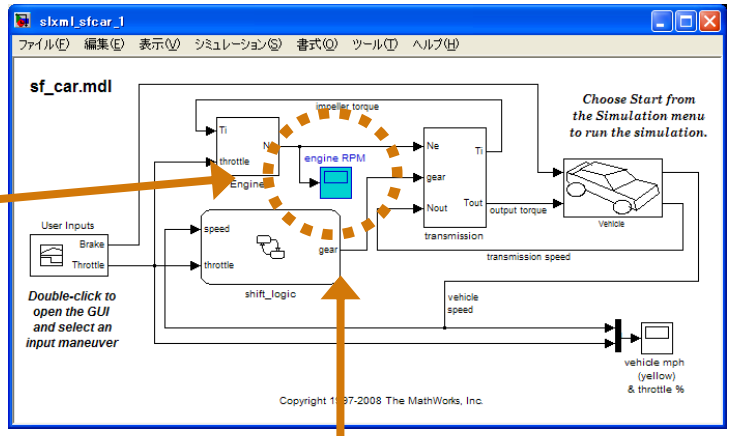
Model-Based Design Environment

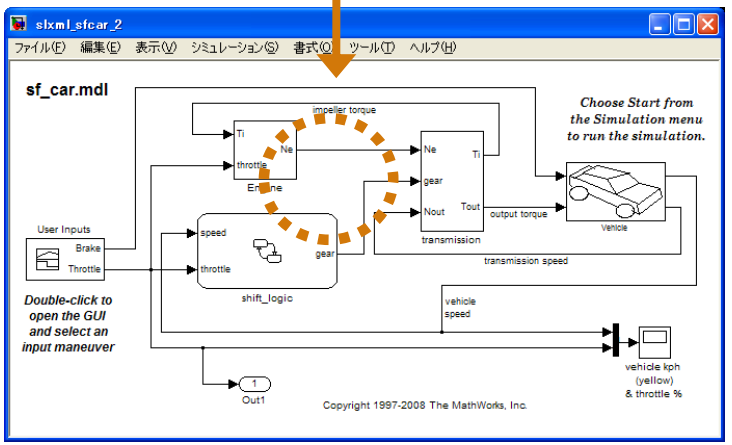
Model Difference Comparisons

Simulink Report Generator

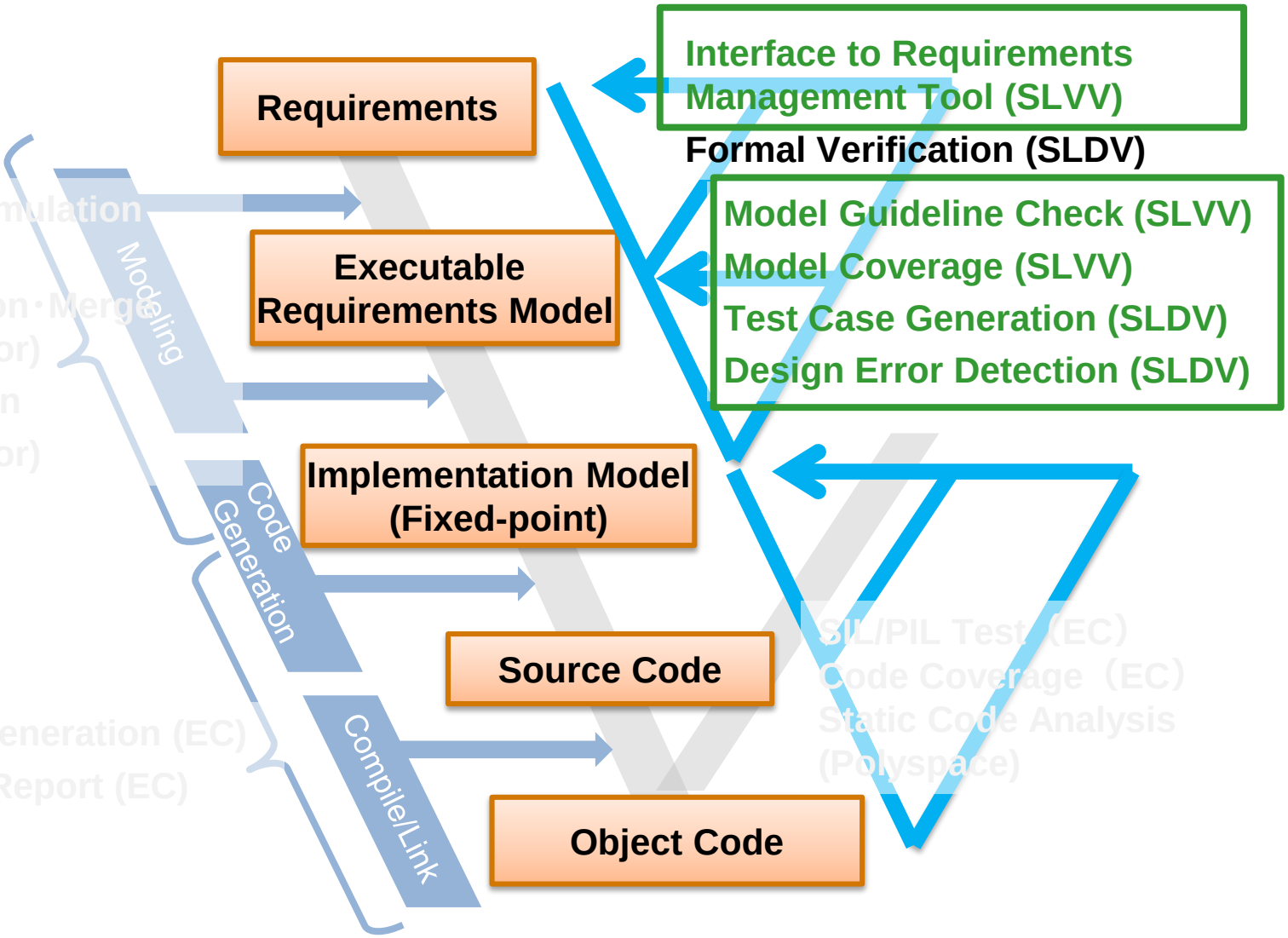
- Generate reports on difference comparisons between 2 models
  - Compatible with Simulink Project and version management software (i.e. Subversion)







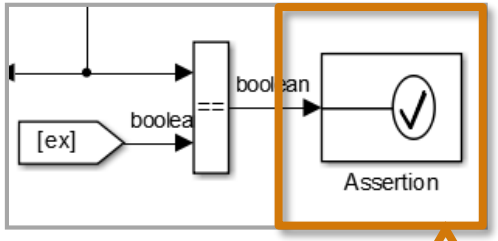
The Next Topic

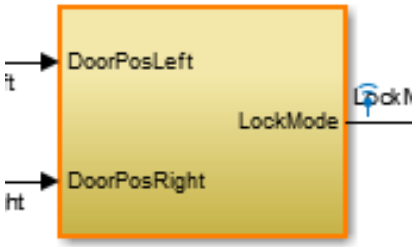


Ensure Traceability
Requirement⇔Model⇔Test

Clarification of effects of requirement changes

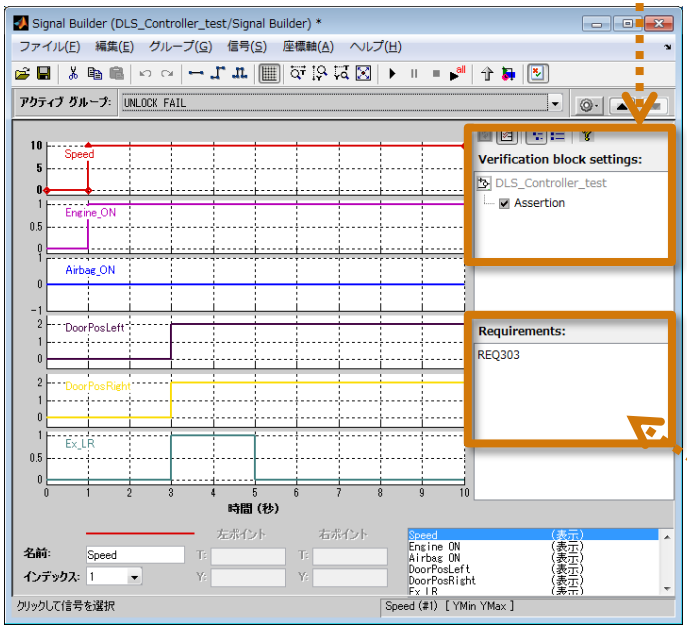
Simulink Verification & Validation





What is being checked?

What is being modeled?



3. Diagnosis Requirements

REQ301 – The software shall determine the lock state of each door based on the lock positions.
- Lock position is under 1mm : Unlock state
- Lock position is over 4mm : Lock state
- Otherwise : Neutral state

REQ302 – The software shall determine the overall vehicle lock state based on all door lock positions.
- All doors in lock position: Lock state
- All doors in unlock position: Unlock state

REQ303 – The software shall determine the overall vehicle lock state to be in failure state due to lock failure in the case where there is no response to a door lock request in under 2 seconds .

What is being tested?

Model Coverage for Measuring Test Completeness Level

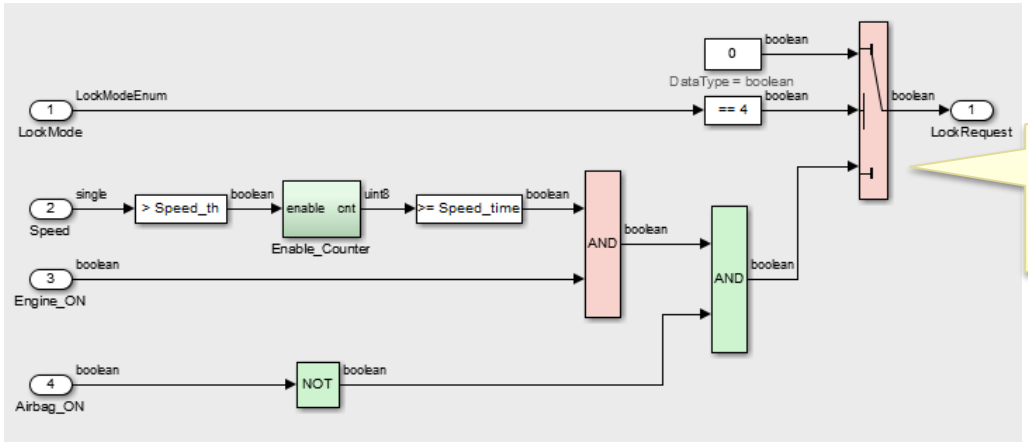
Simulink Verification & Validation

Check for insufficient testing

Model Hierarchy/Complexity:

		D1	C1	MCDC
1. DLS Controller	28	63%	45%	26%
2. . . . Lock Diagnosis	24	62%	27%	13%
3. . . . . Supervisory	19	44%	27%	13%
4. . . . . SF: Supervisory	18	44%	27%	13%
5. . . . Lock Logic	3	75%	100%	75%
6. . . . . Enable Counter	1	100%	NA	NA

Cumulative coverage results on multiple tests



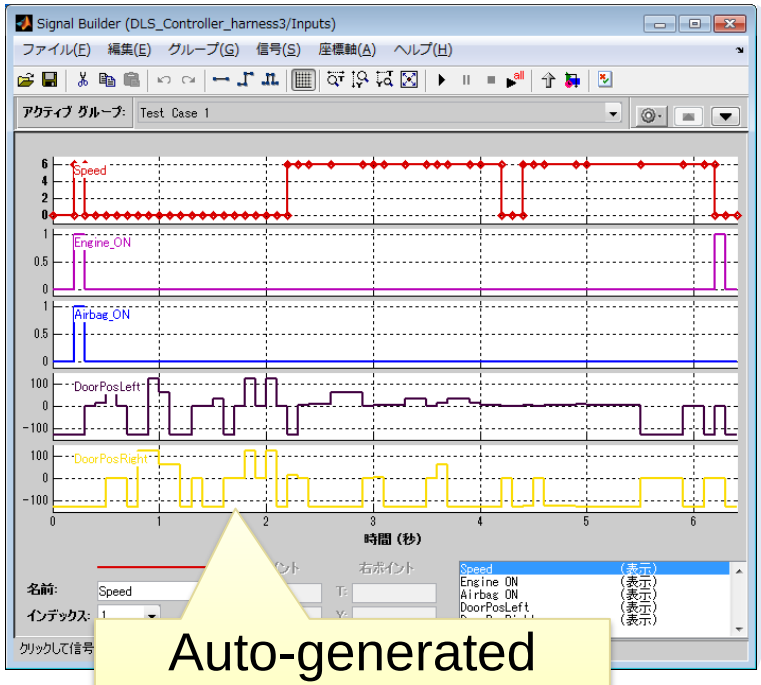
Identify areas of missing coverage

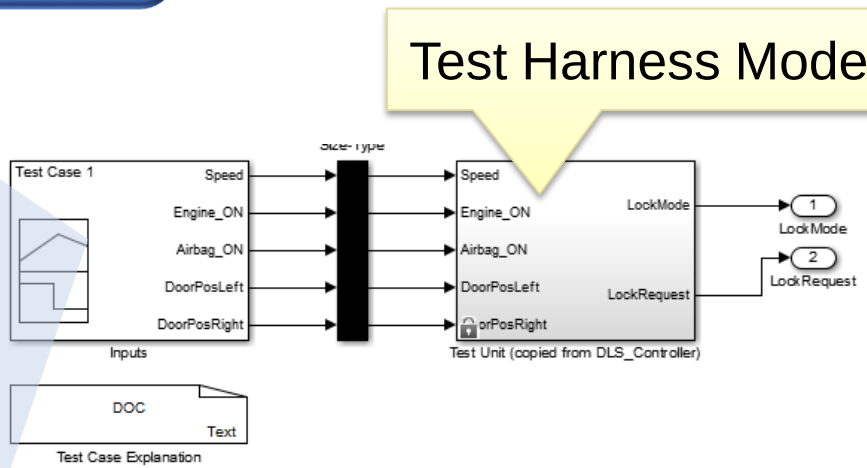
Generate Tests for Full Model Coverage

Simulink Design Verifier

- **Automatic test generation**
- **Suitable for equality tests**

※ Able to generate missing tests based on user-defined tests





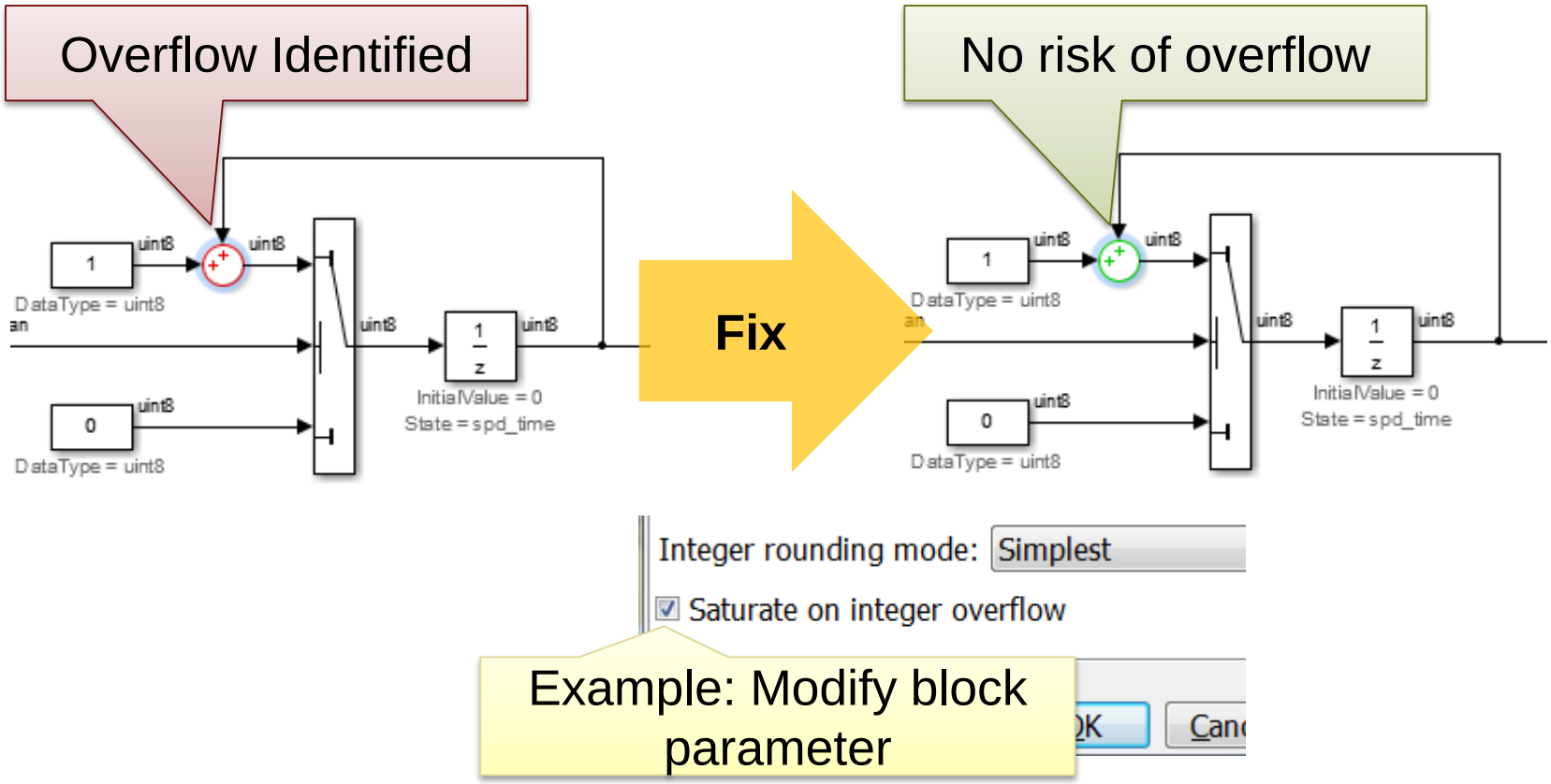
Model Hierarchy/Complexity:

		D1	C1	MCDC
1. DLS_Controller	28	100%	100%	100%
2. . . . Lock_Diagnosis	24	100%	100%	100%
3. . . . . Supervisory	19	100%	100%	100%
4. . . . . SF: Supervisory	18	100%	100%	100%
5. . . . Lock_Logic	3	100%	100%	100%
6. . . . . Enable_Counter	1	100%	NA	NA

Identification of Software Design Errors

Simulink Design Verifier

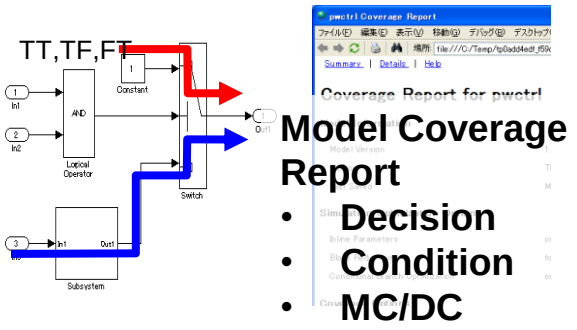
- **Check for risks of software design errors prior to implementation**
  Integer overflow, division by zero, range violations, dead logic



Model Verification & Validation Products

Simulink Verification and Validation™ (SLVnV)

Measure Model Coverage

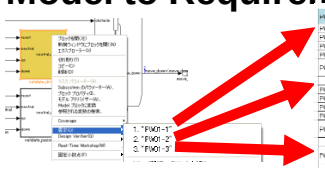


- Model Coverage Report
  - Decision
  - Condition
  - MC/DC

Test Data Sufficiency Check

Traceability

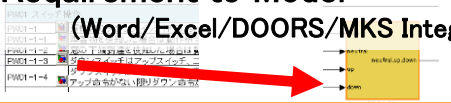
Model to Requirement



- Word
- Excel
- DOORS
- MKS Integrity

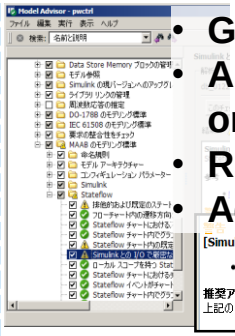
Requirement to Model

(Word/Excel/DOORS/MKS Integrity)

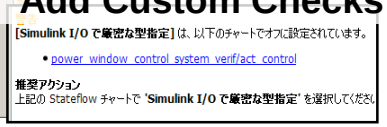


Requirement Sufficiency Check

Model Checker (Model Advisor)



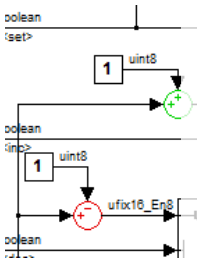
- GUI for Model Checks
- Automate corrections on warnings
- Report Generation
- Add Custom Checks



Automate Model Checking

Simulink Design Verifier™ (SLDV)

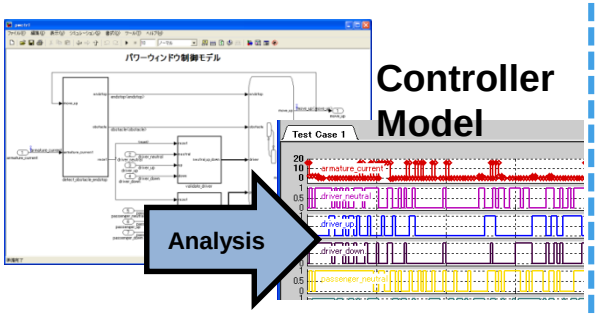
Design Error Detection



- Auto-detect design errors
  - Division by zero
  - Range overflow
  - Deadl Logic
  - Saturation overflow
  - Out of bounds access

Automate Error Detection

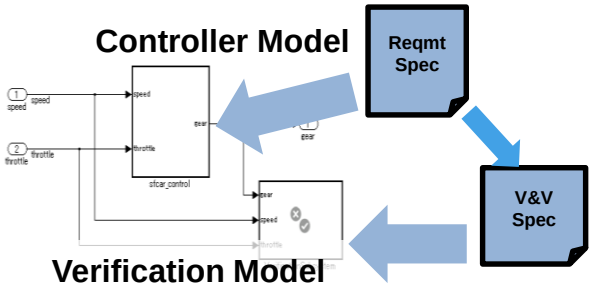
Auto-Generate Test Cases



Controller Model

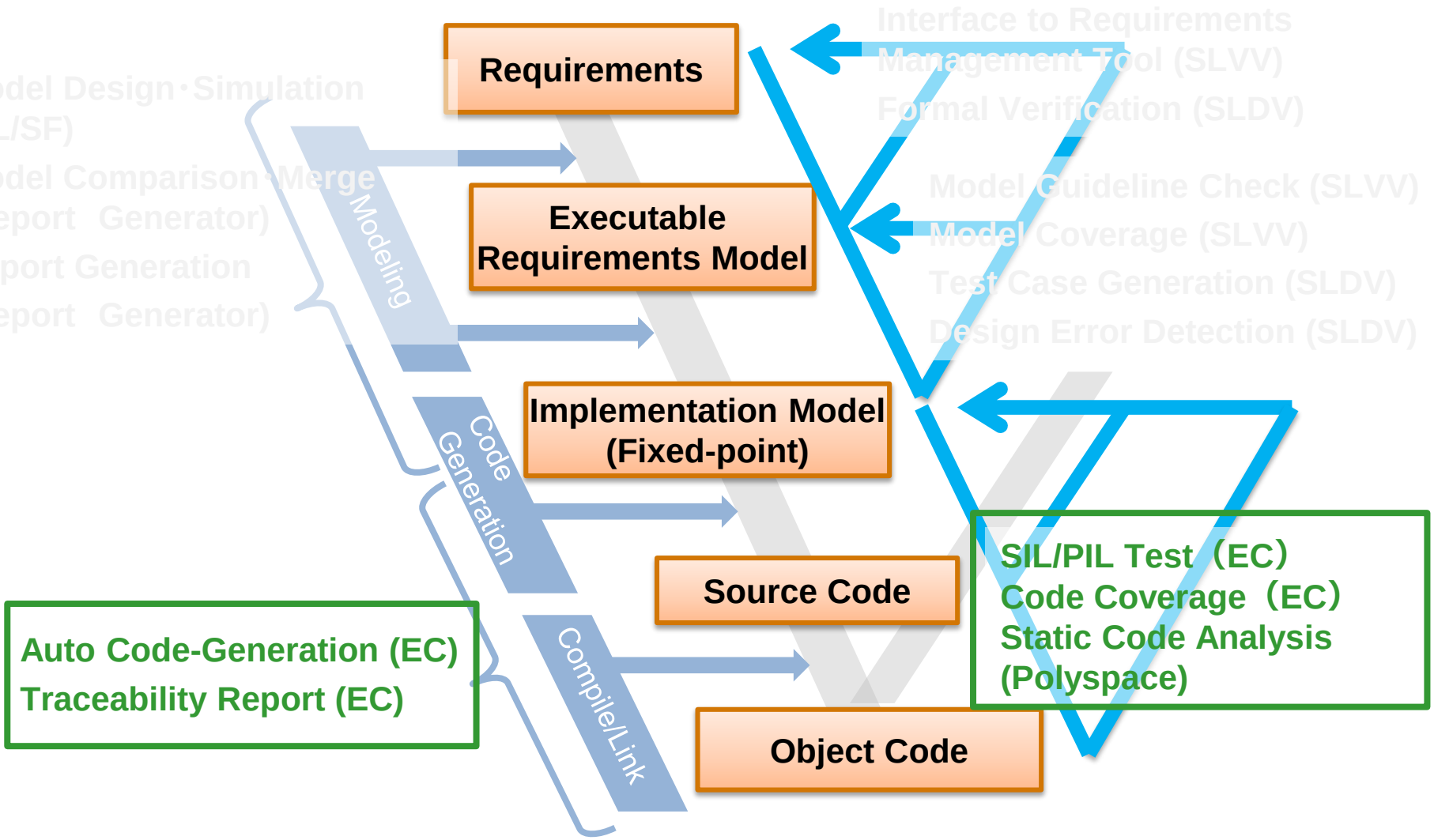
100% Coverage Test Data

Property Proving (Formal Methods)



Certify Correct Behavior

The Final Topic

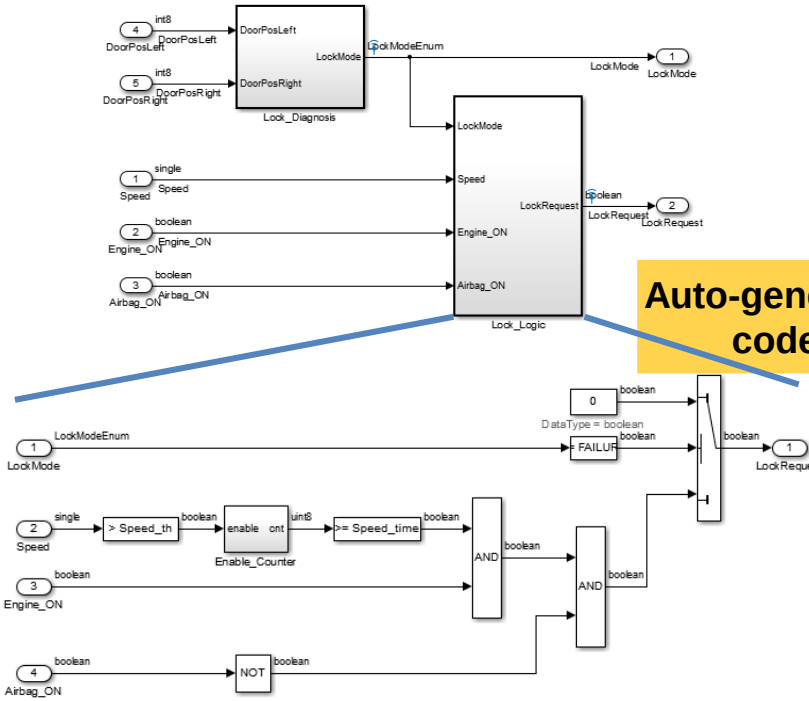


Configuration Management (Simulink Project)

Generate Code from Controller Model

Embedded Coder

- Auto-generate C-code of high readability/efficiency
- Option settings for variable attributes, function settings, code style, etc.
- Auto-generate scaling for fixed-point design

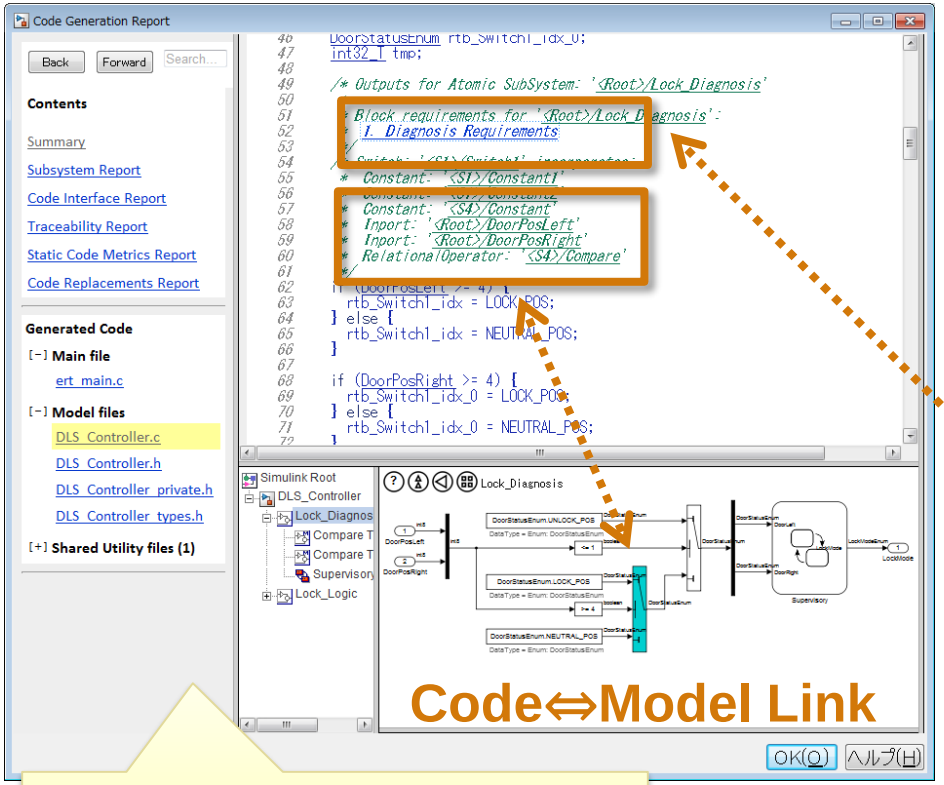



```
if (LockMode == FAILURE) {
  LockRequest = FALSE;
} else {
  LockRequest =
   ((spd_time >= Speed_time) &&
   Engine_ON && (!Airbag_ON));
}
```

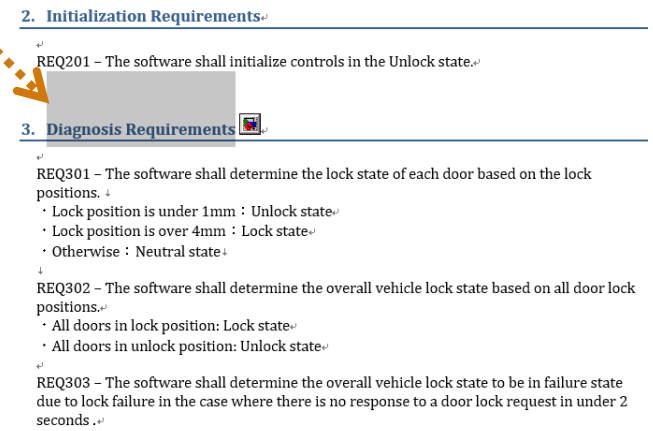
Ensuring Traceability between Requirements, Models, and Code

Embedded Coder / Simulink Report Generator

- Reflect model specifications in generated code
- Distribute reports with model views (html)



Code⇔Document Link



2. Initialization Requirements

REQ201 – The software shall initialize controls in the Unlock state.

3. Diagnosis Requirements

REQ301 – The software shall determine the lock state of each door based on the lock positions.
- Lock position is under 1mm : Unlock state
- Lock position is over 4mm : Lock state
- Otherwise : Neutral state

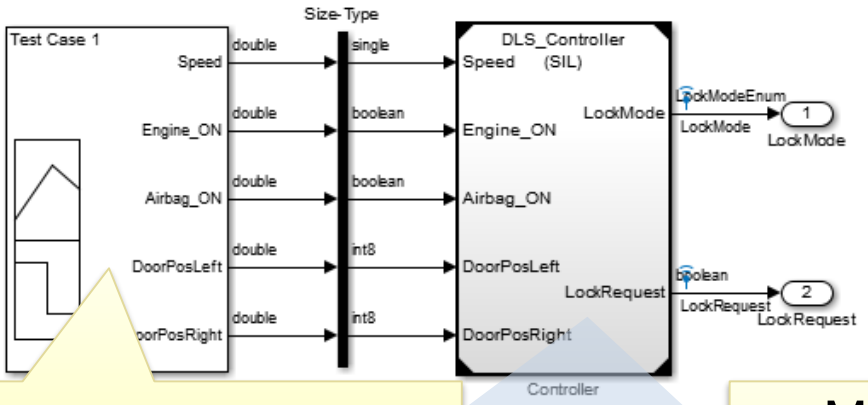
REQ302 – The software shall determine the overall vehicle lock state based on all door lock positions.
- All doors in lock position: Lock state
- All doors in unlock position: Unlock state

REQ303 – The software shall determine the overall vehicle lock state to be in failure state due to lock failure in the case where there is no response to a door lock request in under 2 seconds .

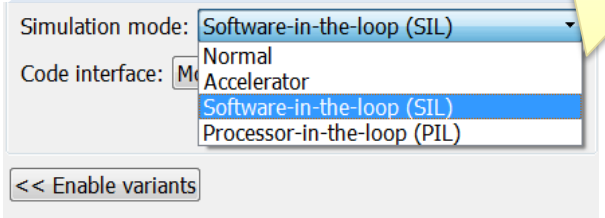
Model⇔Code Equality Checks （SIL/PIL, Back 2 Back Test）

Embedded Coder

Efficient testing by reuse of model verification test data

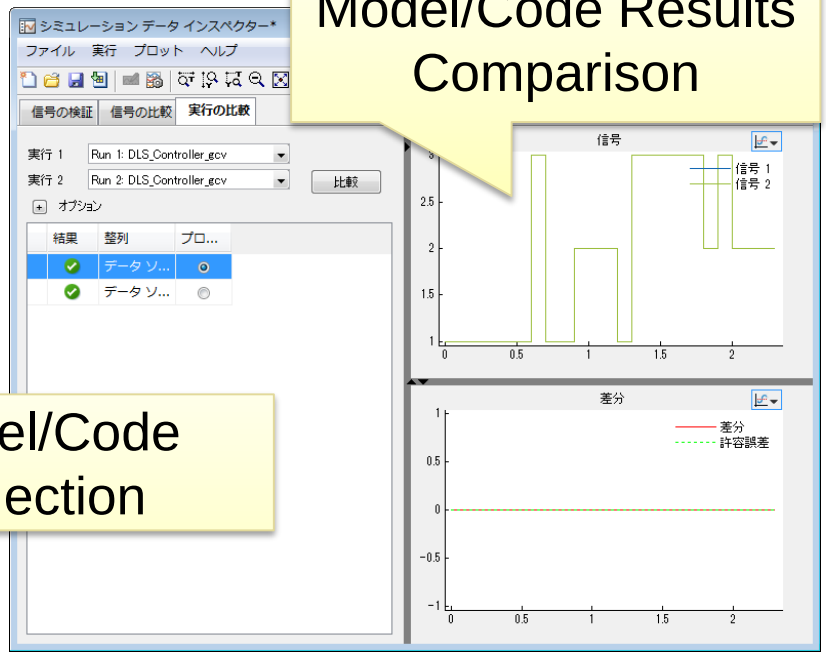


Existing data/SLDV generated test data



Model/Code Selection

Model/Code Results Comparison



※ Test automation through Simulink Test.

Tool Chain Example: Product List

Product	Functionality	Usage
Simulink	Modeling: Controller Block	Modeling Module/Integration Test
Stateflow	Modeling: State Transitions, Flow Charts	Modeling
Fixed-Point Designer	Modeling: Fixed-Point Processing	Modeling
Simulink Verification and Validation	Model Coverage Requirements Interface Model Advisor	Module/Integration Test Review and Static Analysis
Simulink Design Verifier	Property Proving Test Generation Design Error Detection	Review and Static Analysis
Embedded Coder	Code Generation PIL Test/CGV Bullseye/LDRA Integration Traceability Report	Code Generation Equality Testing Code Coverage Measurement
IEC Certification Kit	Traceability Matrix Generation Templates for Certification	ISO26262 Support
Simulink Report Generator	Report Editing and Generation	Report Generation Model Comparison/Merge

Proving Source Code Correctness

Polyspace Code Prover: Static Code Verification

- **Quality**
  - Prove absence of runtime errors (RTEs)
  - Measure, Improve, Manage
- **Usage**
  - No need to compile, execute, or generate test cases
  - Supports：C/C++/Ada
- **Process**
  - Early detection of RTEs
  - Analyze both hand-code and auto-generated code
  - Measure code reliability

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]

Analyze all executable paths to detect errors and prove the absence of errors

ISO26262 Functional Safety Standard

- Functional safety standard for automotive equipment
- Based on IEC61508
- Description of purpose and requirements for development
  - Activities for development process （Software safety life cycle）
  - Development and verification tools （Tool qualification）
- Description of new software engineering concepts
  - **Model-based development**
  - **Early verification and validity checks**
  - **Automatic code generation**

Model-Based Design Benefits（ISO26262 excerpt）

Annex B
(informative)

ISO/DIS 26262-6

Model-based development

B.1 Objectives

This Annex describes the concept of model-based development of in-vehicle software and outlines its implications on the product development at the software level.

The seamless utilization of models facilitates a highly consistent and efficient development.

ISO/DIS 26262-1

1.74 model-based development

development that uses models to describe the functional behavior of the elements which are to be developed

NOTE Depending on the level of abstraction used for such a model it can be used for simulation or code generation or both.

MathWorks Solution: Summary

Using Models to Detect Errors Early and Increase Efficiency

・Able to form small V-loops
・Able to detect errors early in the development cycle

- Mode⇔Code consistency allows for Simulink simulation results to be considered “truth”.
- Early model verification is possible due to the ability to investigate floating-point models
- Large team development made easy through highly customizable tool chain
- Errors in object code detected easily through synchronization between simulations and SILS/PILS