2024 MathWorks
中国汽车年会

基于模型的系统工程应用于需求开发和管理

龚小平 kgong@mathworks.com

MathWorks

基于模型的系统工程 – MBSE

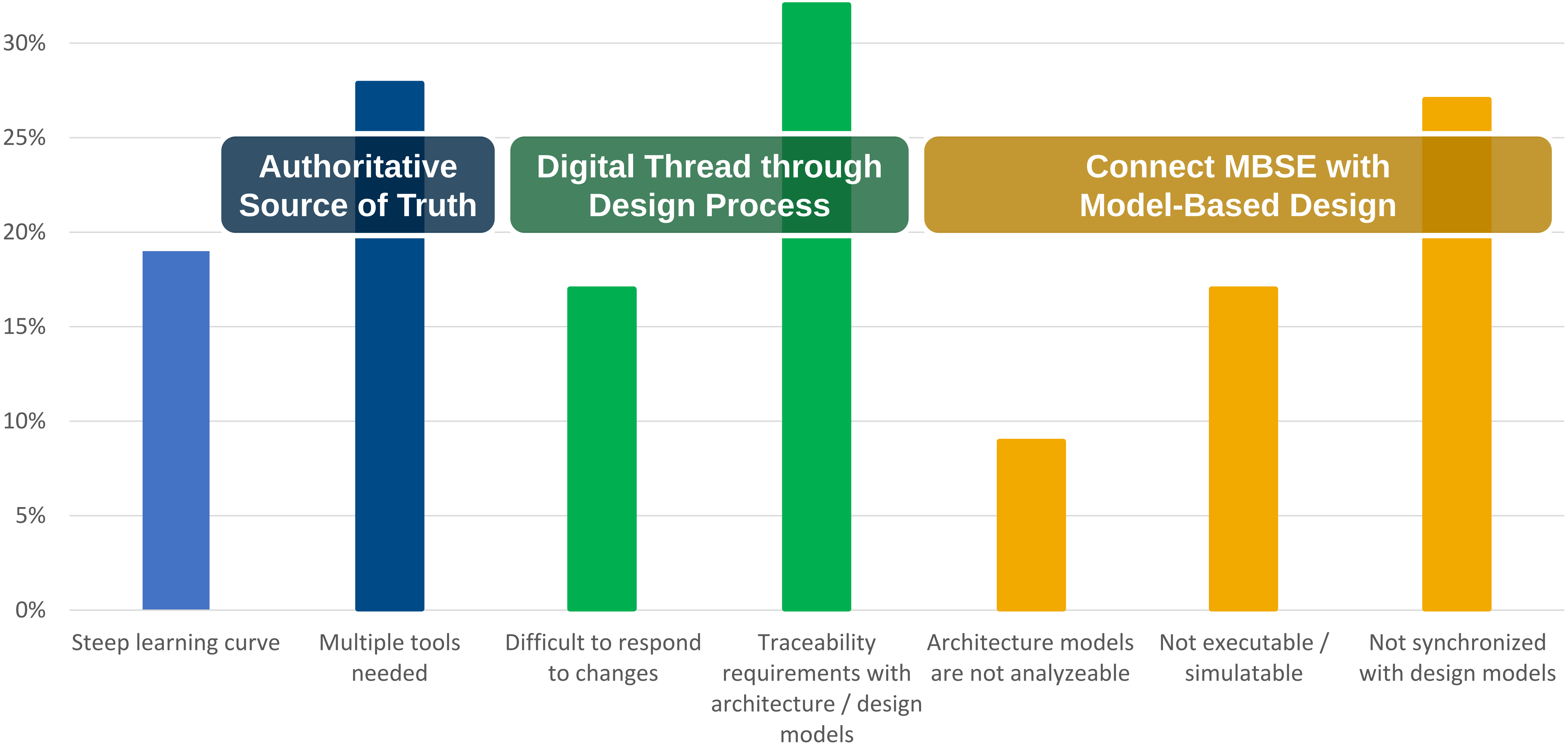
“ Systems Engineering is a **transdisciplinary and integrative approach** to enable the successful realization, use, and retirement of engineered systems, using systems principles and concepts, and scientific, technological, and management methods.

“ Model-based systems engineering (MBSE) is the **formalized application of modeling** to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases.

INCOSE: What is Systems Engineering?
INCOSE: Systems Engineering Glossary

INCOSE Model Based Systems Engineering (MBSE) Initiative
Sanford Friedenthal, Regina Griego, Mark Sampson

MBSE的应用挑战



Source: MathWorks webinar, Model-Based Systems Engineering - Practical Use and Applications, December 9th, 2021 (173 responses)

MBSE开发和管理需求

Maintain **requirements as an authoritative source of truth** throughout the product development process, by using (simulation) models to:

Manage Requirements

- Transform stakeholder requirements/needs into design requirements using models, simulation and code generation
- Establish traceability between requirements, models and testcases

Authoritative Source of Truth

Digital Thread through Design Process

Connect MBSE with Model-Based Design

Manage Complexity

- Explore the design space through (reusable) trade-off studies
- Through views and traceable architecture models

Digital Thread through Design Process

Connect MBSE with Model-Based Design

Manage Interfaces

- Connect system architecture with software architecture, component implementation, FMEA (fault injection models)

Digital Thread through Design Process

Connect MBSE with Model-Based Design

MBSE开发和管理需求

Maintain requirements as an authoritative source of truth throughout the product development process, by using (simulation) models to:

Manage Requirements

- Transform stakeholder requirements/needs into design requirements using models, simulation and code generation
- Establish traceability between requirements, models and testcases

Authoritative Source of Truth

Digital Thread through Design Process

Connect MBSE with Model-Based Design

Manage Complexity

- Explore the design space through (reusable) trade-off studies
- Through views and traceable architecture models

Digital Thread through Design Process

Connect MBSE with Model-Based Design

Manage Interfaces

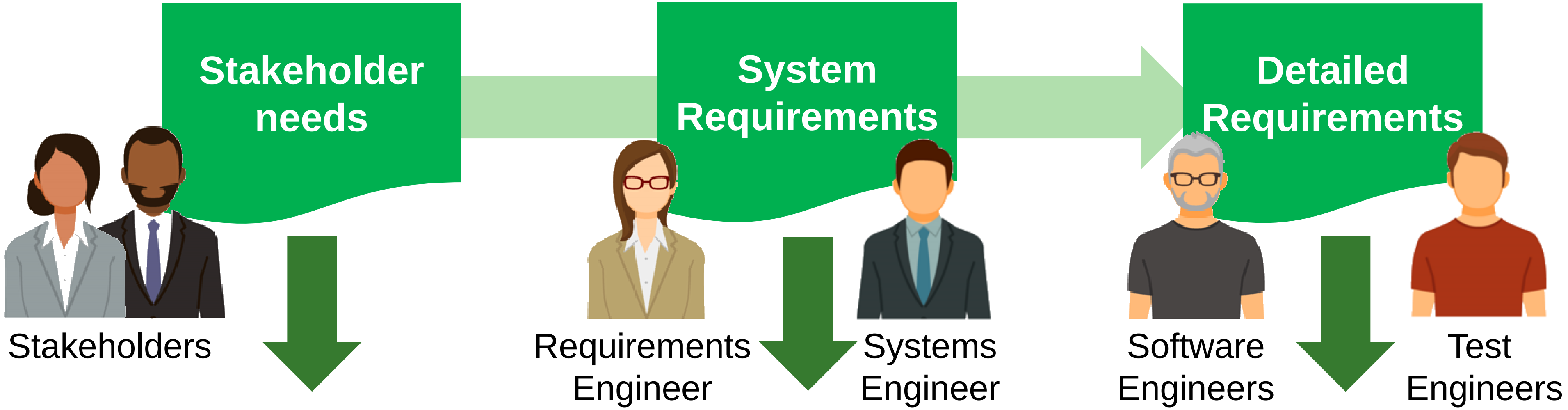
- Connect system architecture with software architecture, component implementation, FMEA (fault injection models)

Digital Thread through Design Process

Connect MBSE with Model-Based Design

需求开发流程 – 传统方式

Requirements flow



Produced artifacts

Requirements set (textual)

Architecture models (descriptive)
Requirements set (textual)
Traceability

Requirements set (textual)
Design models
Code (C/C++/HDL)
Test harnesses
Traceability

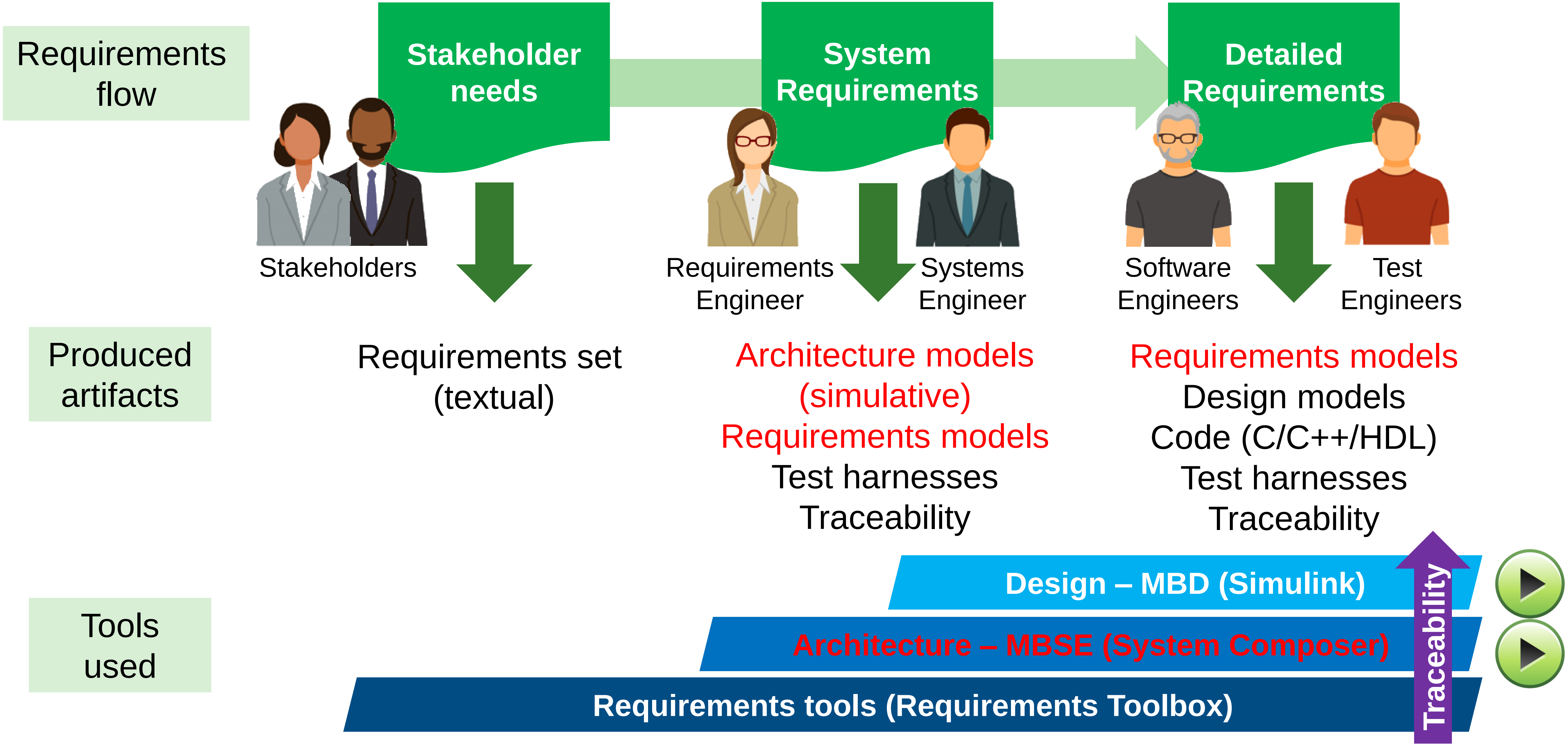
Tools used

Design – MBD (Simulink)

Architecture – MBSE

Requirements tools (Requirements Toolbox, ….)

需求开发流程 – 同平台方式



Requirements flow

Stakeholder needs

System Requirements

Detailed Requirements

Stakeholders

Requirements Engineer

Systems Engineer

Software Engineers

Test Engineers

Produced artifacts

Requirements set (textual)

Architecture models (simulative)
Requirements models
Test harnesses
Traceability

Requirements models
Design models
Code (C/C++/HDL)
Test harnesses
Traceability

Tools used

Design – MBD (Simulink)

Architecture – MBSE (System Composer)

Requirements tools (Requirements Toolbox)

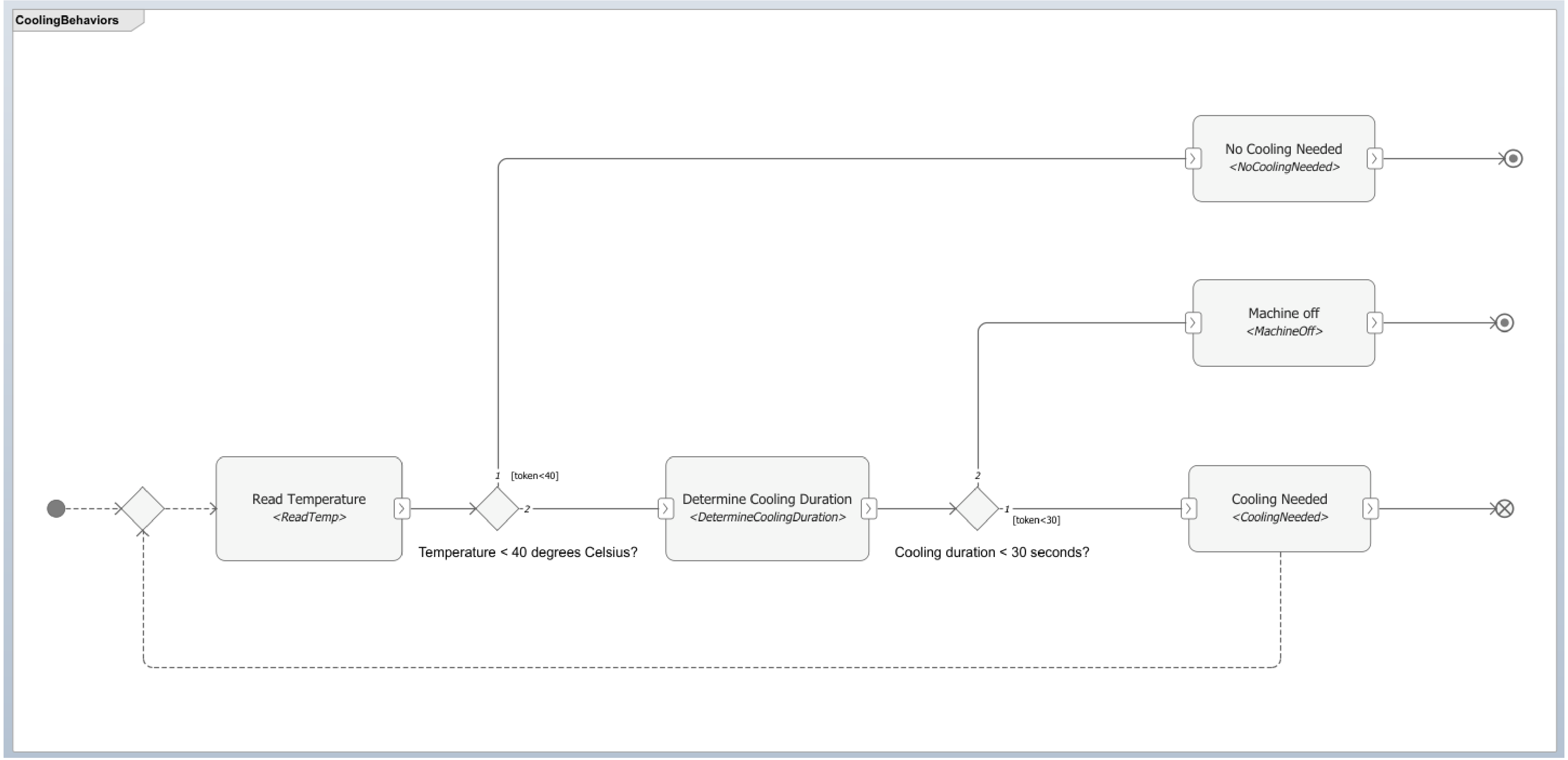
Traceability

案例 – 冷却系统需求开发

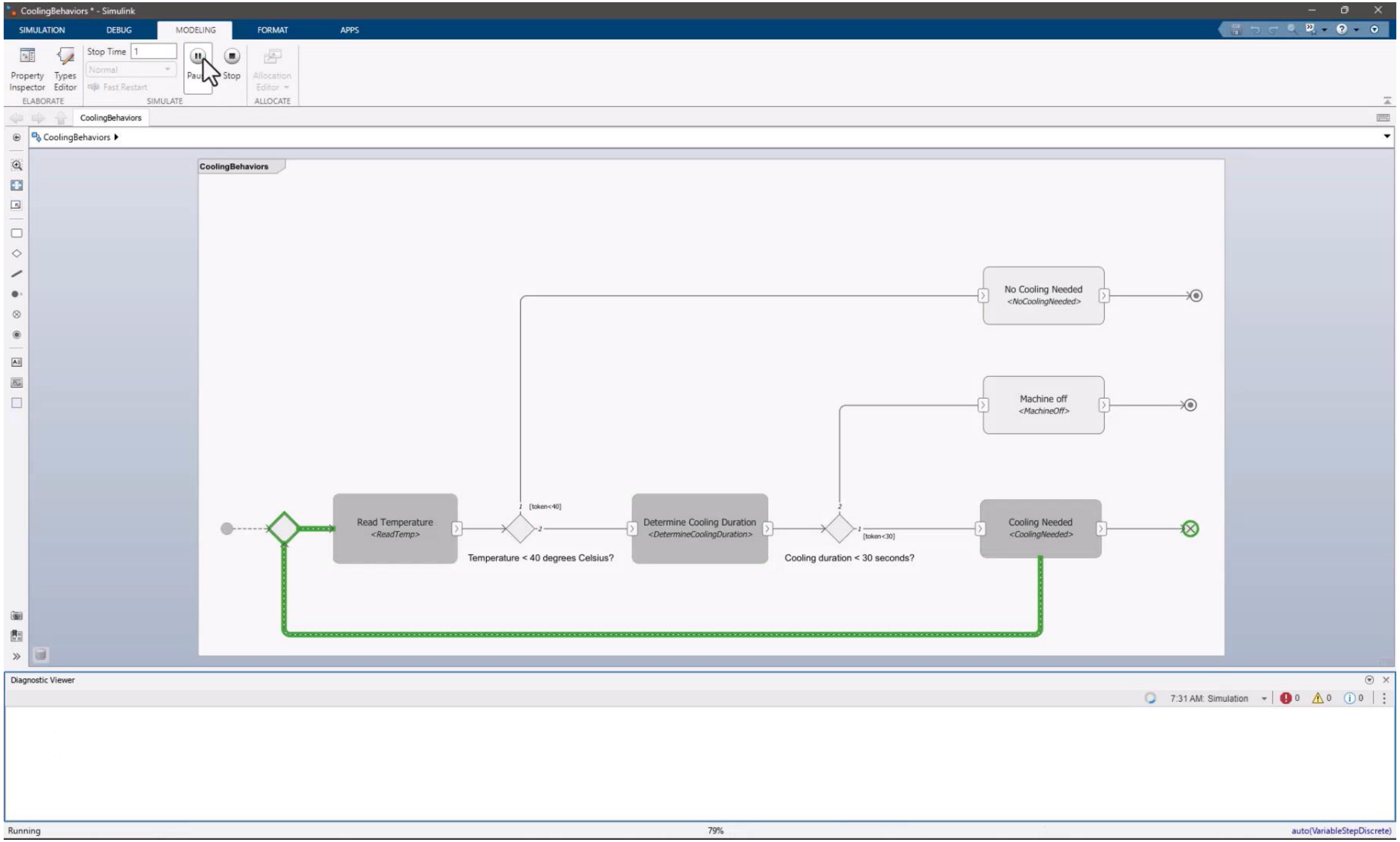
Provide a system which maintains the operating temperature of a machine, avoiding damage to the machine because of overheating.

- **[constraint]** Cooling system needs to maintain operating temperature.
- **[constraint]** Cooling needs to be effective within a predetermined time.
- **[assumption]** Environmental temperature greater than -10 degrees and smaller than 80 degrees.

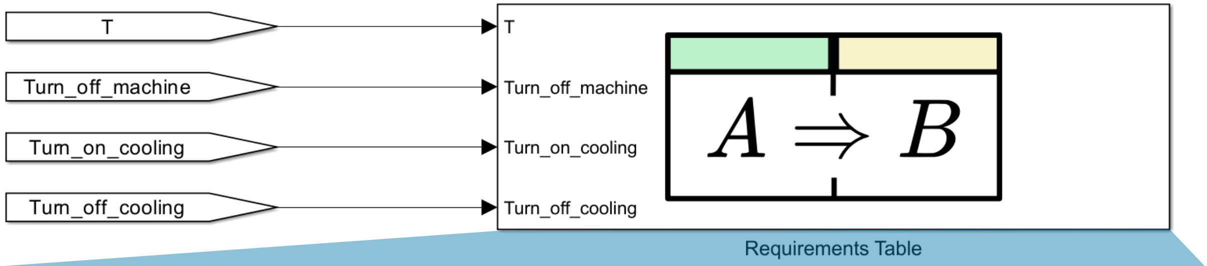
理解和确认用例场景 – 活动图



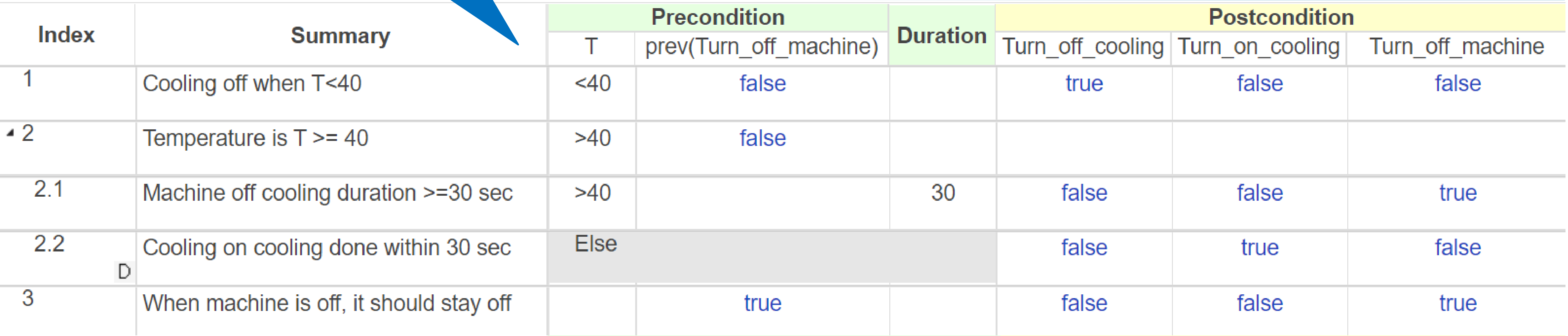
理解和确认用例场景 – 活动图仿真



理解和确认用例场景 – 需求表



Formal description of requirements

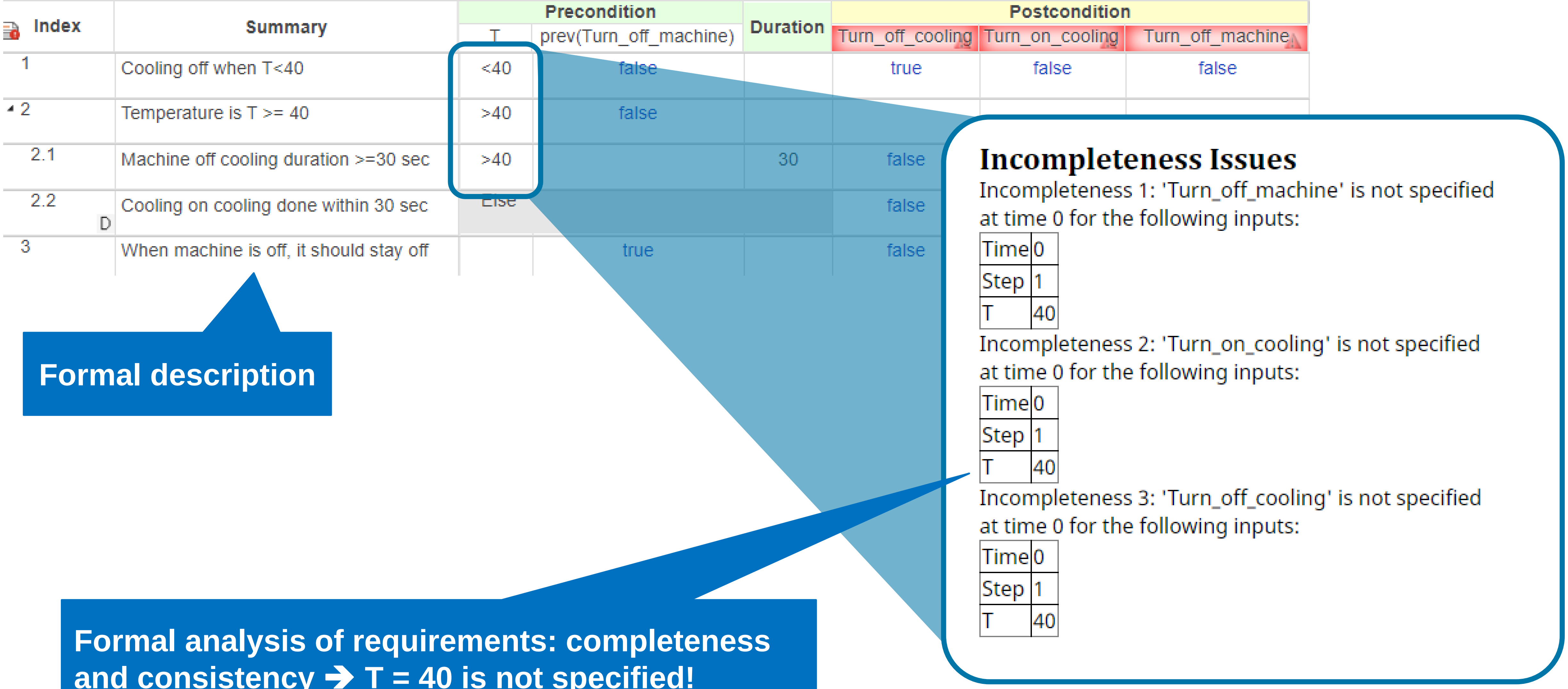


Index	Summary	Precondition		Duration	Postcondition		
		T	prev(Turn_off_machine)		Turn_off_cooling	Turn_on_cooling	Turn_off_machine
1	Cooling off when T<40	<40	false		true	false	false
2	Temperature is T >= 40	>40	false				
2.1	Machine off cooling duration >=30 sec	>40		30	false	false	true
2.2 D	Cooling on cooling done within 30 sec	Else			false	true	false
3	When machine is off, it should stay off		true		false	false	true

Input condition to activate a requirement

Expected outcome of a requirement

理解和确认用例场景 – 需求表分析



Index	Summary	Precondition		Duration	Postcondition		
		T	prev(Turn_off_machine)		Turn_off_cooling	Turn_on_cooling	Turn_off_machine
1	Cooling off when T<40	<40	false		true	false	false
2	Temperature is T >= 40	>40	false				
2.1	Machine off cooling duration >=30 sec	>40		30	false		
2.2 D	Cooling on cooling done within 30 sec	Else			false		
3	When machine is off, it should stay off		true		false		

Incompleteness Issues

Incompleteness 1: 'Turn_off_machine' is not specified at time 0 for the following inputs:

Time	0
Step	1
T	40

Incompleteness 2: 'Turn_on_cooling' is not specified at time 0 for the following inputs:

Time	0
Step	1
T	40

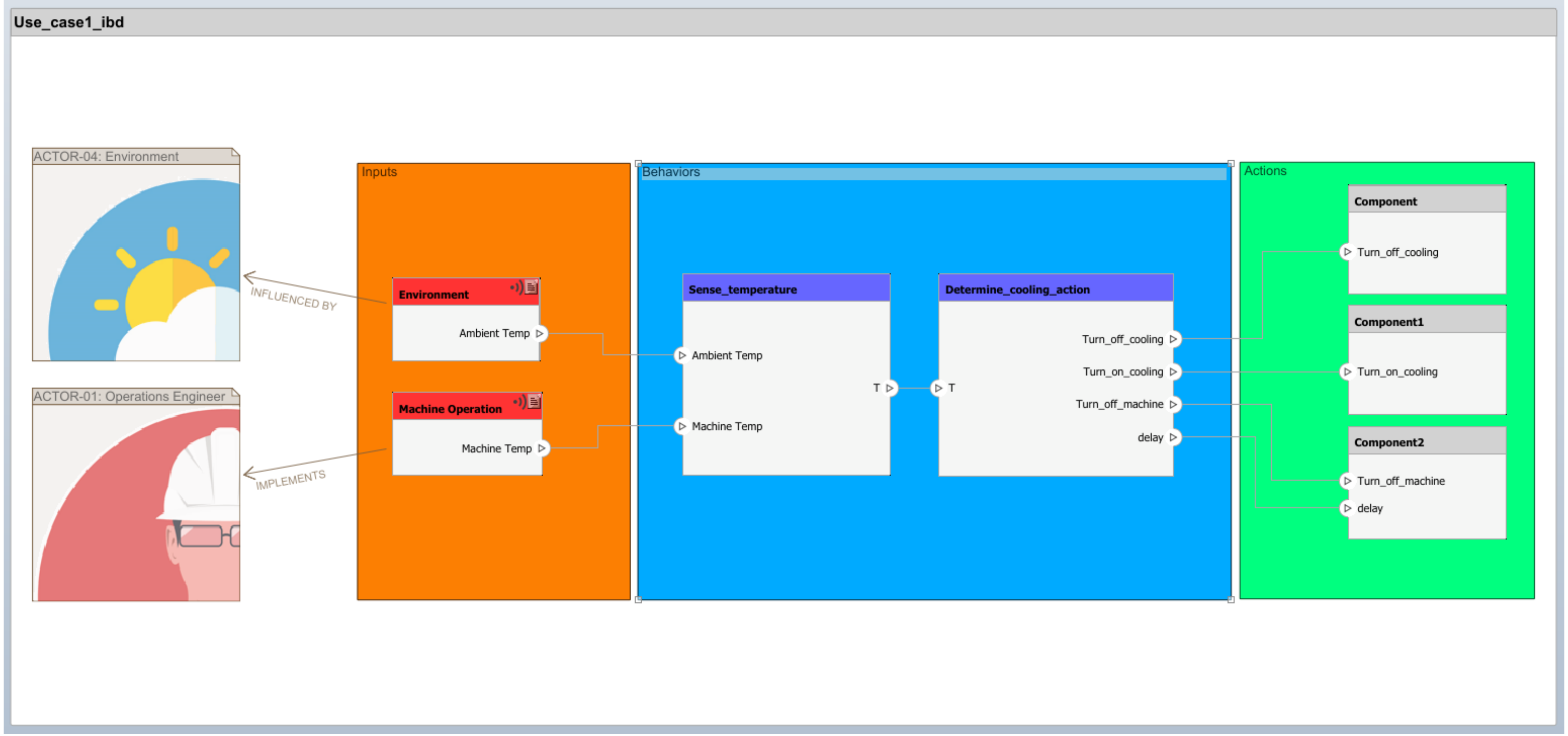
Incompleteness 3: 'Turn_off_cooling' is not specified at time 0 for the following inputs:

Time	0
Step	1
T	40

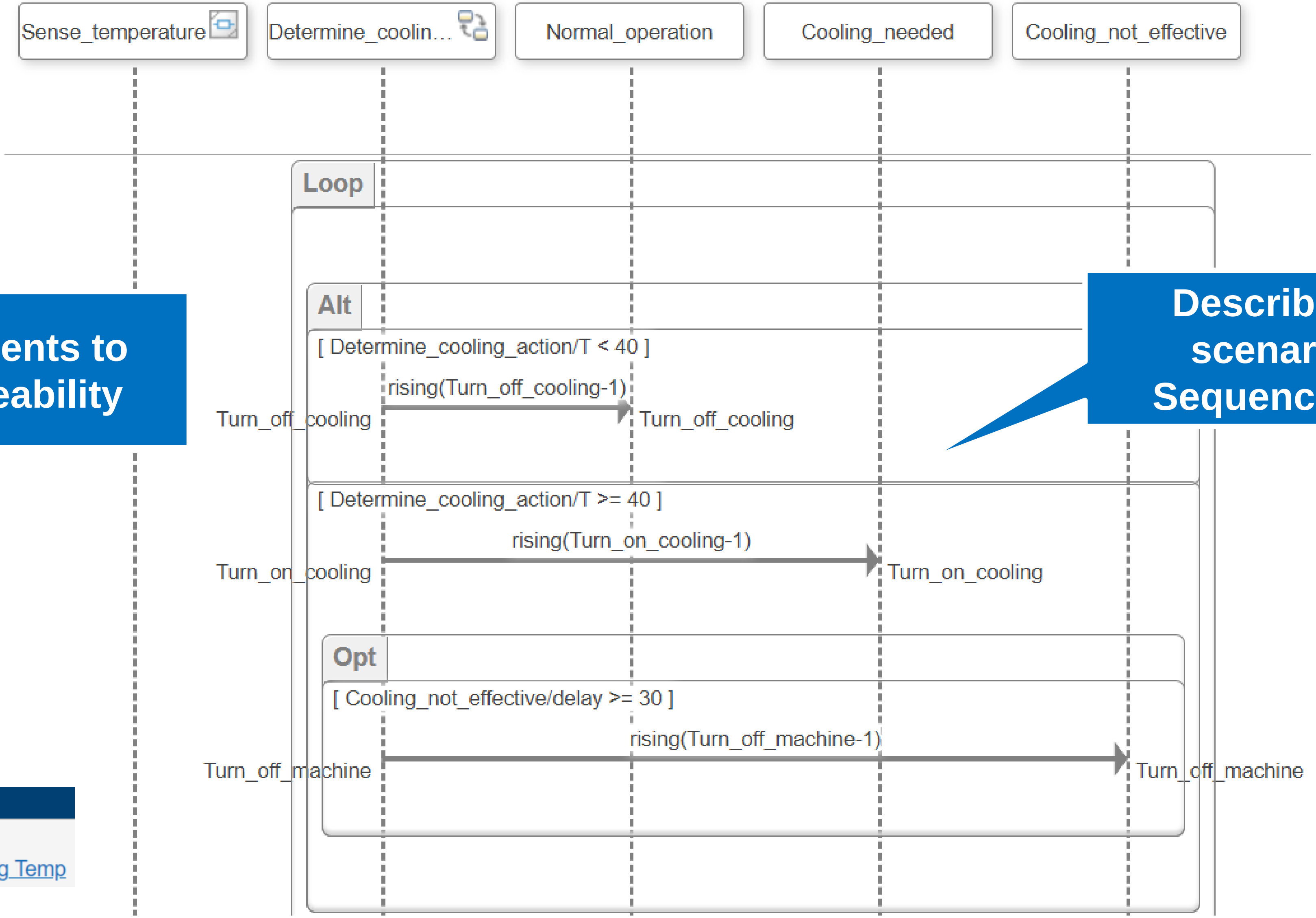
Formal description

Formal analysis of requirements: completeness and consistency ➔ T = 40 is not specified!

描述系统结构和接口 – 结构图

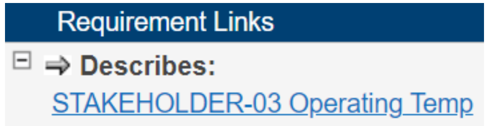


描述系统组件交互 – 顺序图



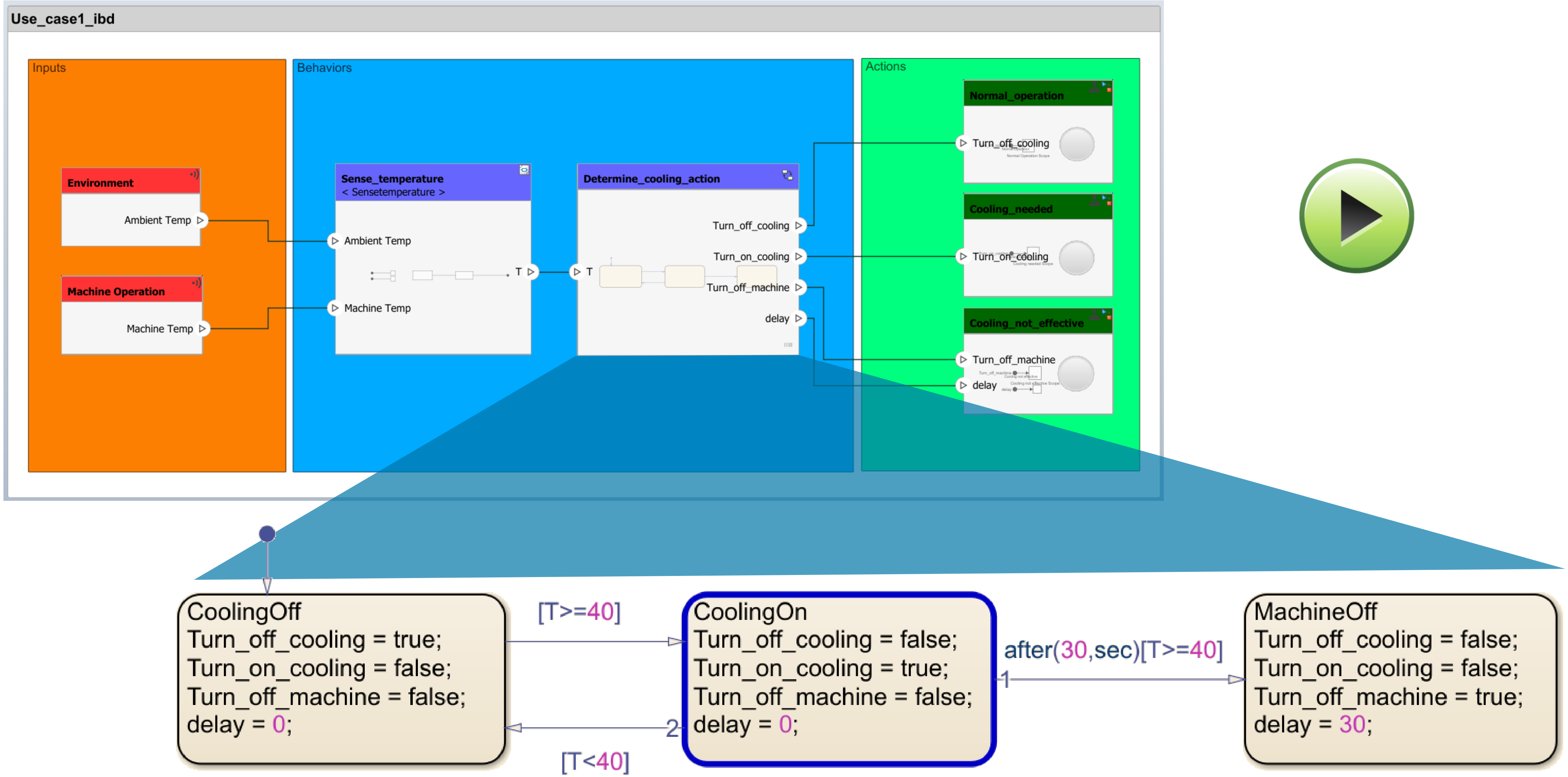
Describe complex scenarios using Sequence Diagrams

Link requirements to facilitate traceability


Requirement Links

⊟ ⇒ **Describes:**
STAKEHOLDER-03 Operating Temp

开发系统详细设计模型 – 状态图



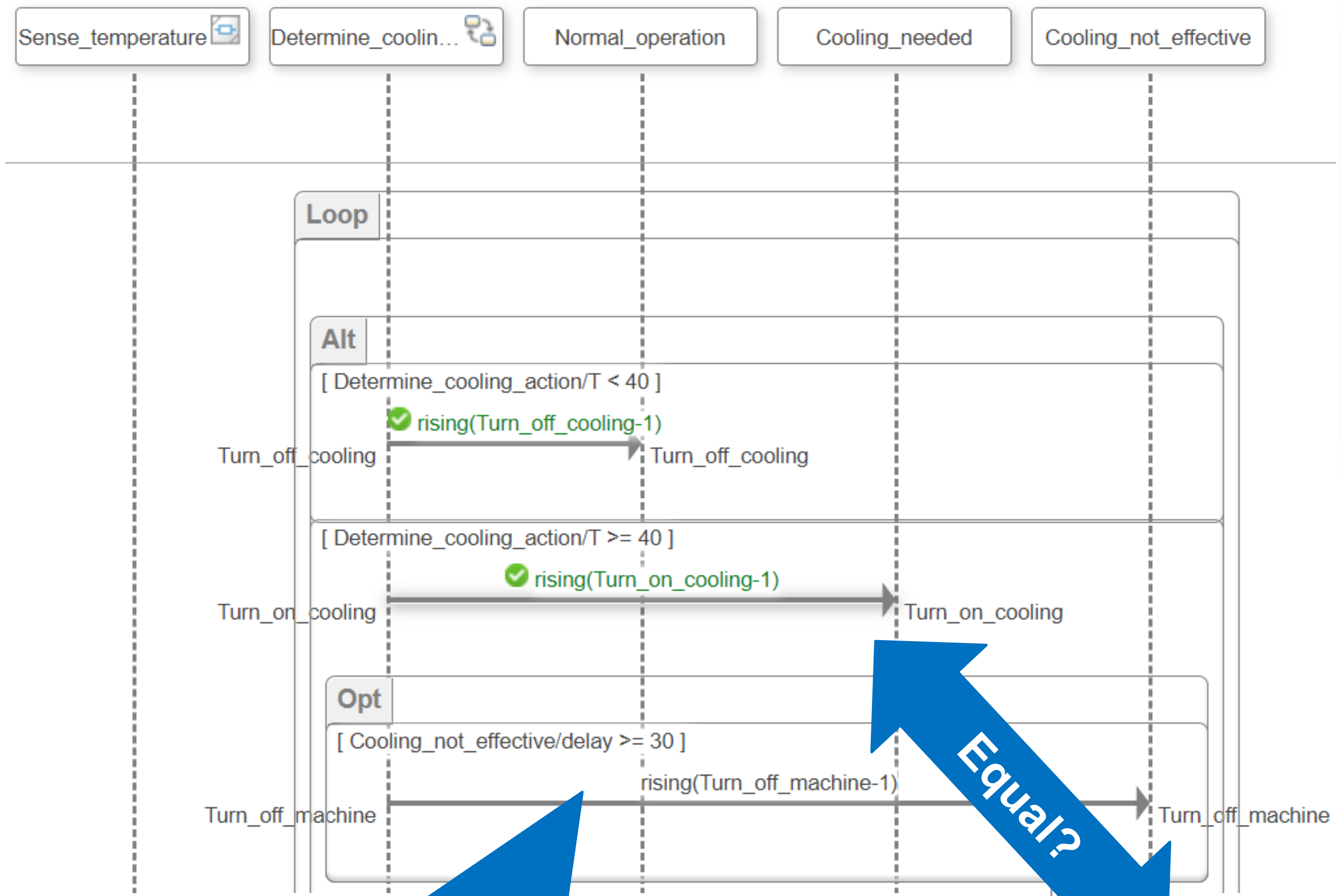
```
CoolingOff
Turn_off_cooling = true;
Turn_on_cooling = false;
Turn_off_machine = false;
delay = 0;
```

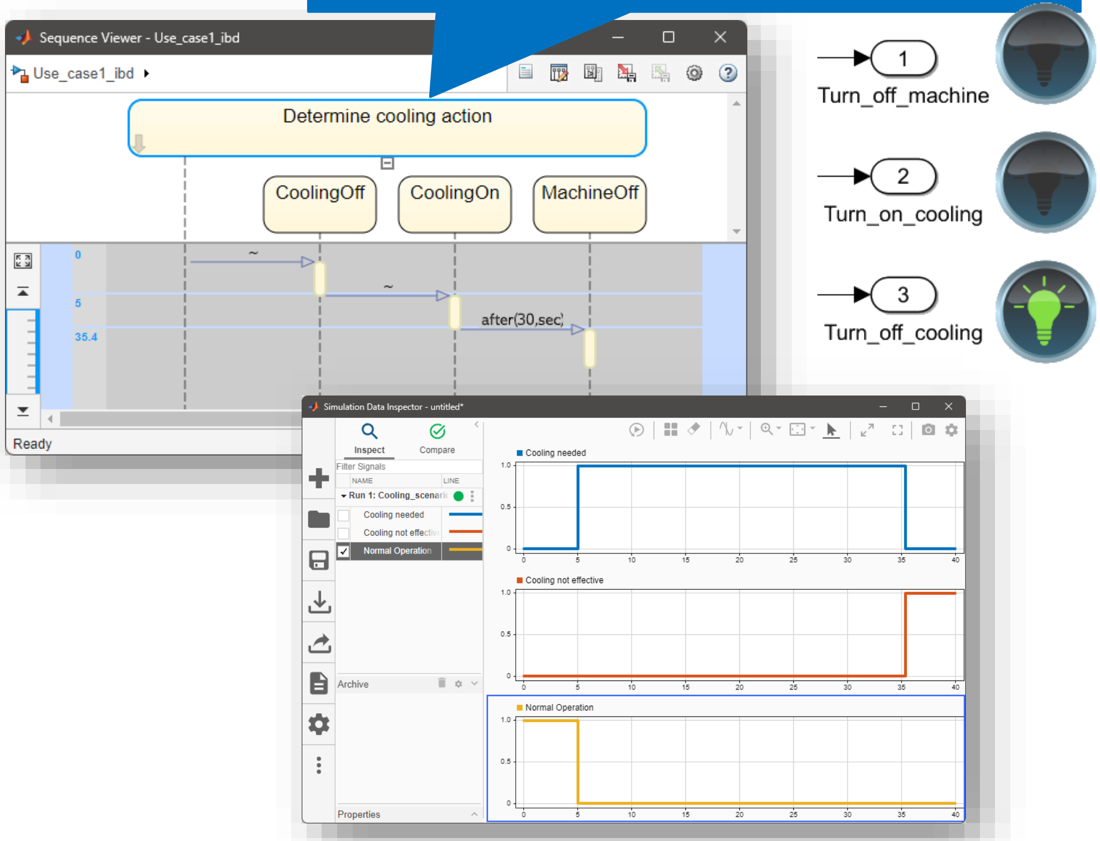
```
CoolingOn
Turn_off_cooling = false;
Turn_on_cooling = true;
Turn_off_machine = false;
delay = 0;
```

```
MachineOff
Turn_off_cooling = false;
Turn_on_cooling = false;
Turn_off_machine = true;
delay = 30;
```

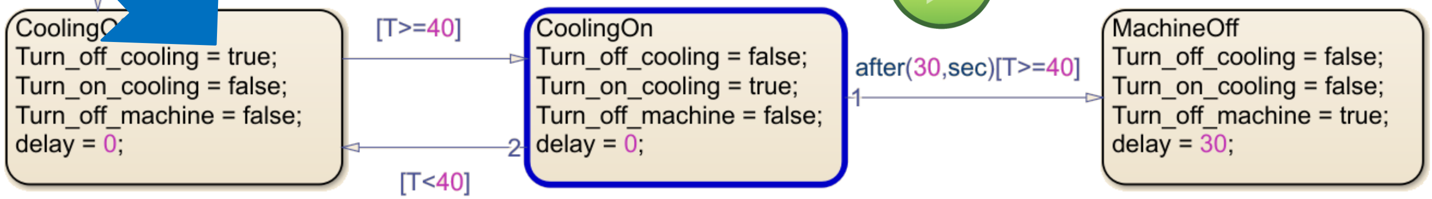
验证详细设计模型行为 – 需求仿真

Visualize simulation results



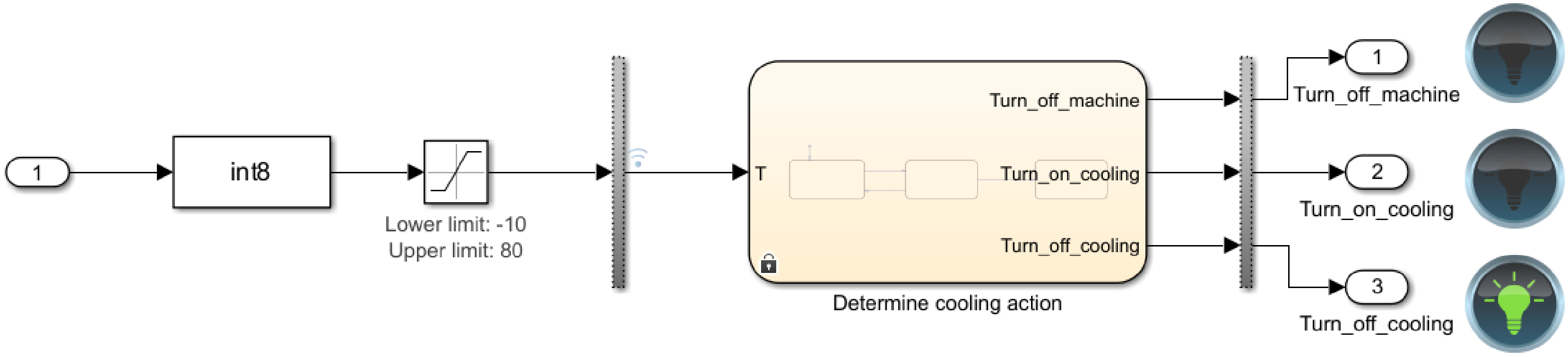


Equal?

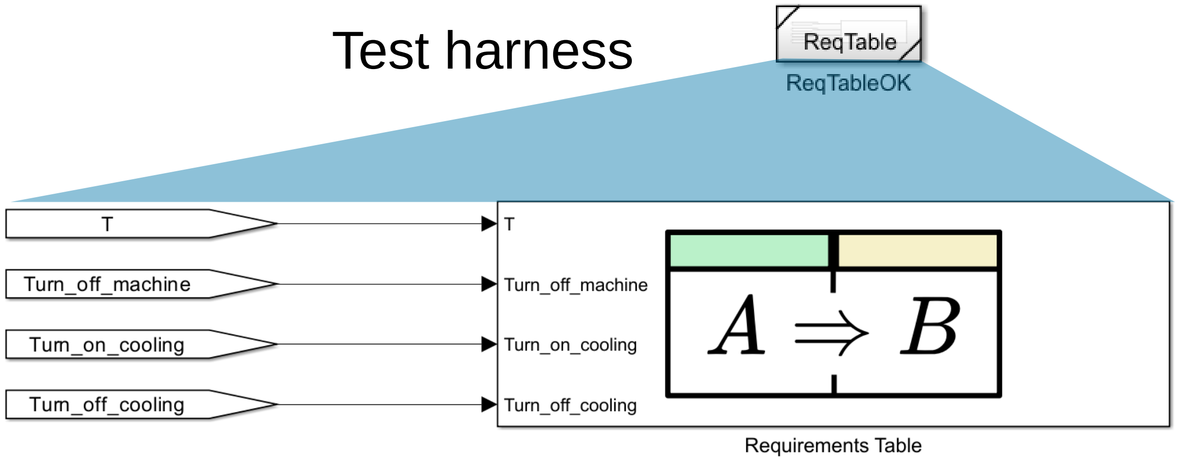


Verify expected behavior through simulation

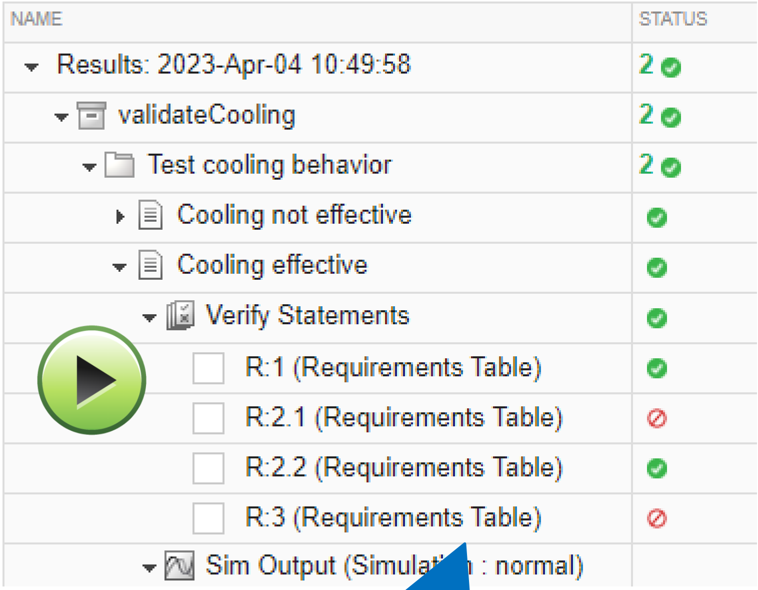
验证详细设计模型行为 – 需求确认



Test harness



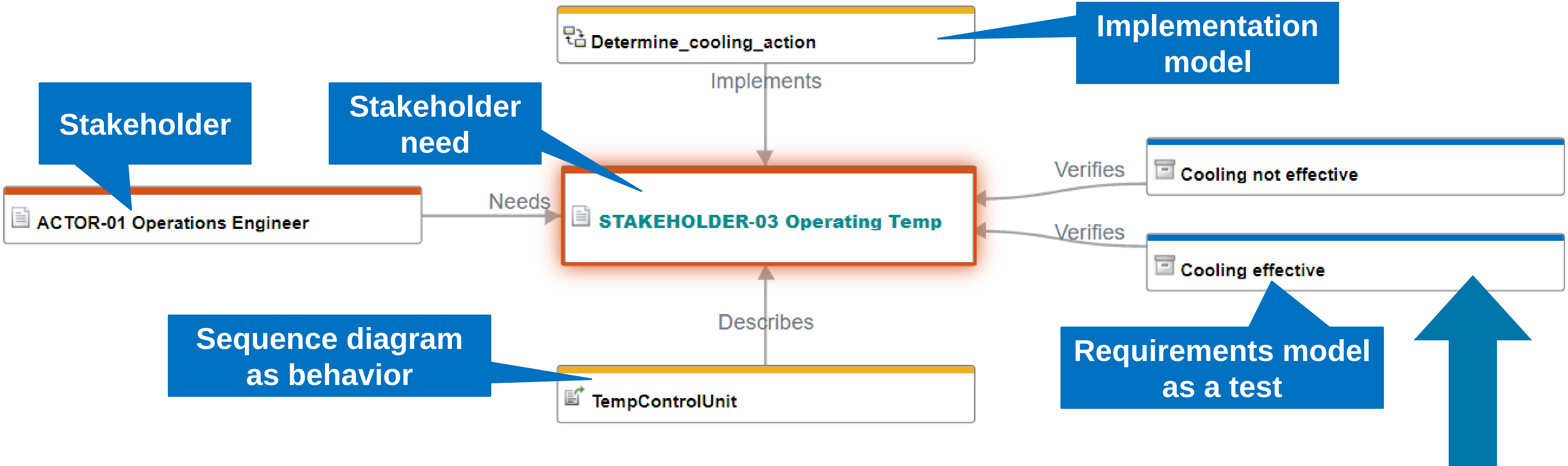
Test definition

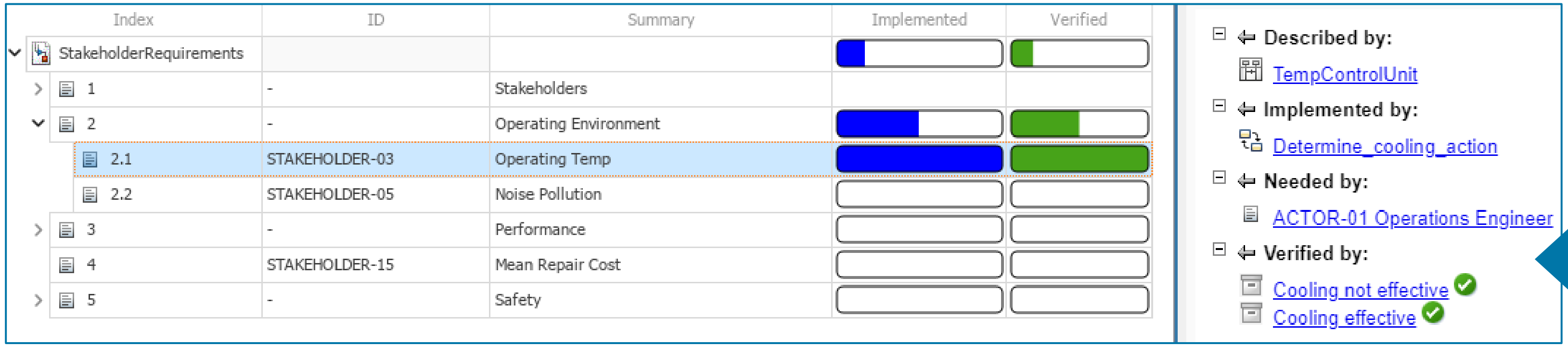


NAME	STATUS
Results: 2023-Apr-04 10:49:58	2 ✓
validateCooling	2 ✓
Test cooling behavior	2 ✓
Cooling not effective	✓
Cooling effective	✓
Verify Statements	✓
□ R:1 (Requirements Table)	✓
□ R:2.1 (Requirements Table)	⊘
□ R:2.2 (Requirements Table)	✓
□ R:3 (Requirements Table)	⊘
Sim Output (Simulati n : normal)	

Validate compliance to requirements through simulation

构建数字化线索 – 需求/设计/测试





Index	ID	Summary	Implemented	Verified
StakeholderRequirements			🟦	🟩
1	-	Stakeholders		
2	-	Operating Environment	🟦	🟩
2.1	STAKEHOLDER-03	Operating Temp	🟦	🟩
2.2	STAKEHOLDER-05	Noise Pollution	□	□
3	-	Performance	□	□
4	STAKEHOLDER-15	Mean Repair Cost	□	□
5	-	Safety	□	□

⊟ **Described by:**
TempControlUnit

⊟ **Implemented by:**
Determine_cooling_action

⊟ **Needed by:**
ACTOR-01 Operations Engineer

⊟ **Verified by:**
Cooling not effective ✓
Cooling effective ✓

Status

MBSE开发和管理需求

Maintain requirements as an authoritative source of truth throughout the product development process, by using (simulation) models to:

Manage Requirements

- Transform stakeholder requirements/needs into design requirements using models, simulation and code generation
- Establish traceability between requirements, models and testcases

Authoritative Source of Truth

Digital Thread through Design Process

Connect MBSE with Model-Based Design

Manage Complexity

- Explore the design space through (reusable) trade-off studies
- Through views and traceable architecture models

Digital Thread through Design Process

Connect MBSE with Model-Based Design

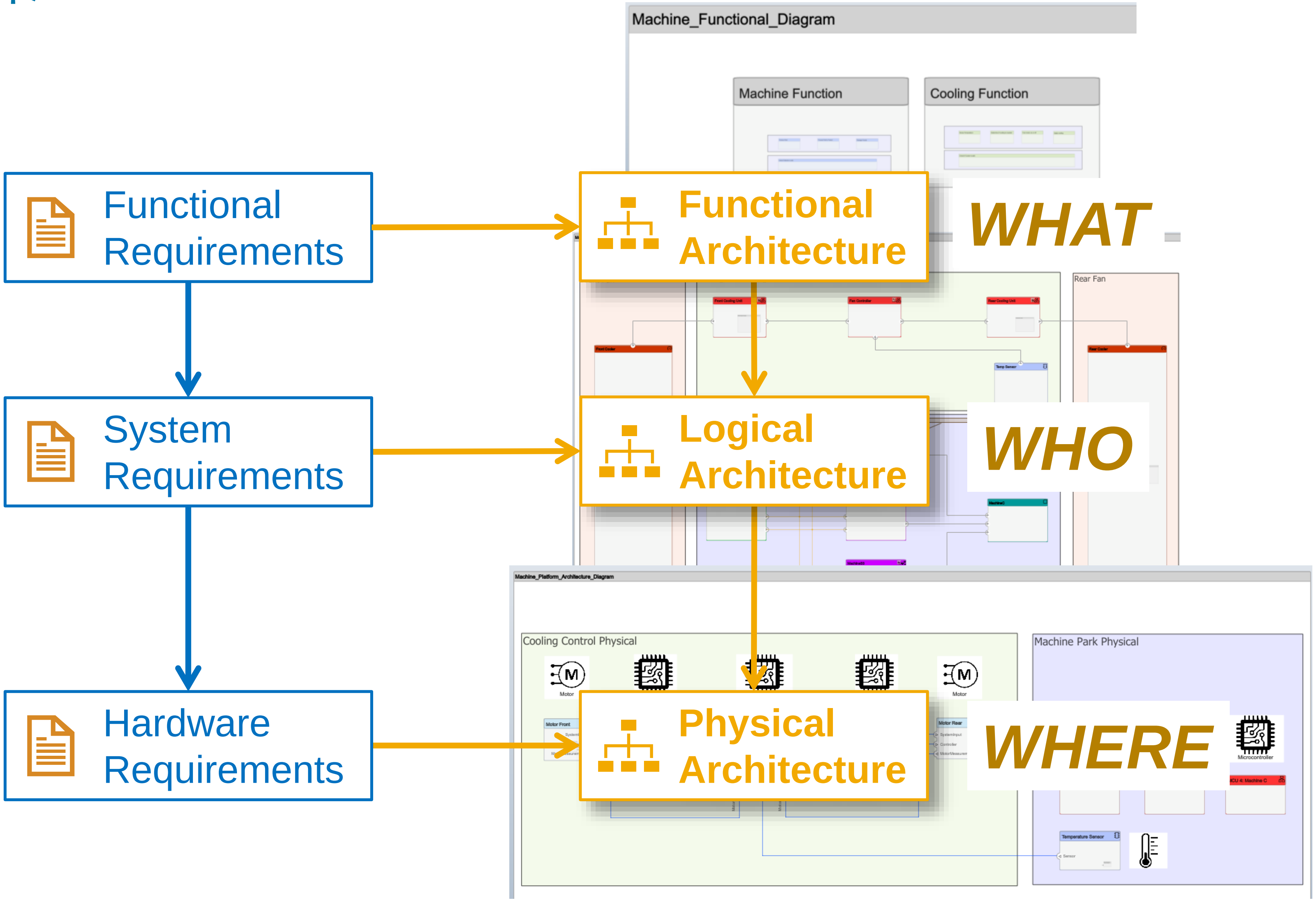
Manage Interfaces

- Connect system architecture with software architecture, component implementation, FMEA (fault injection models)

Digital Thread through Design Process

Connect MBSE with Model-Based Design

架构框架



Functional Requirements

System Requirements

Hardware Requirements

Functional Architecture

Logical Architecture

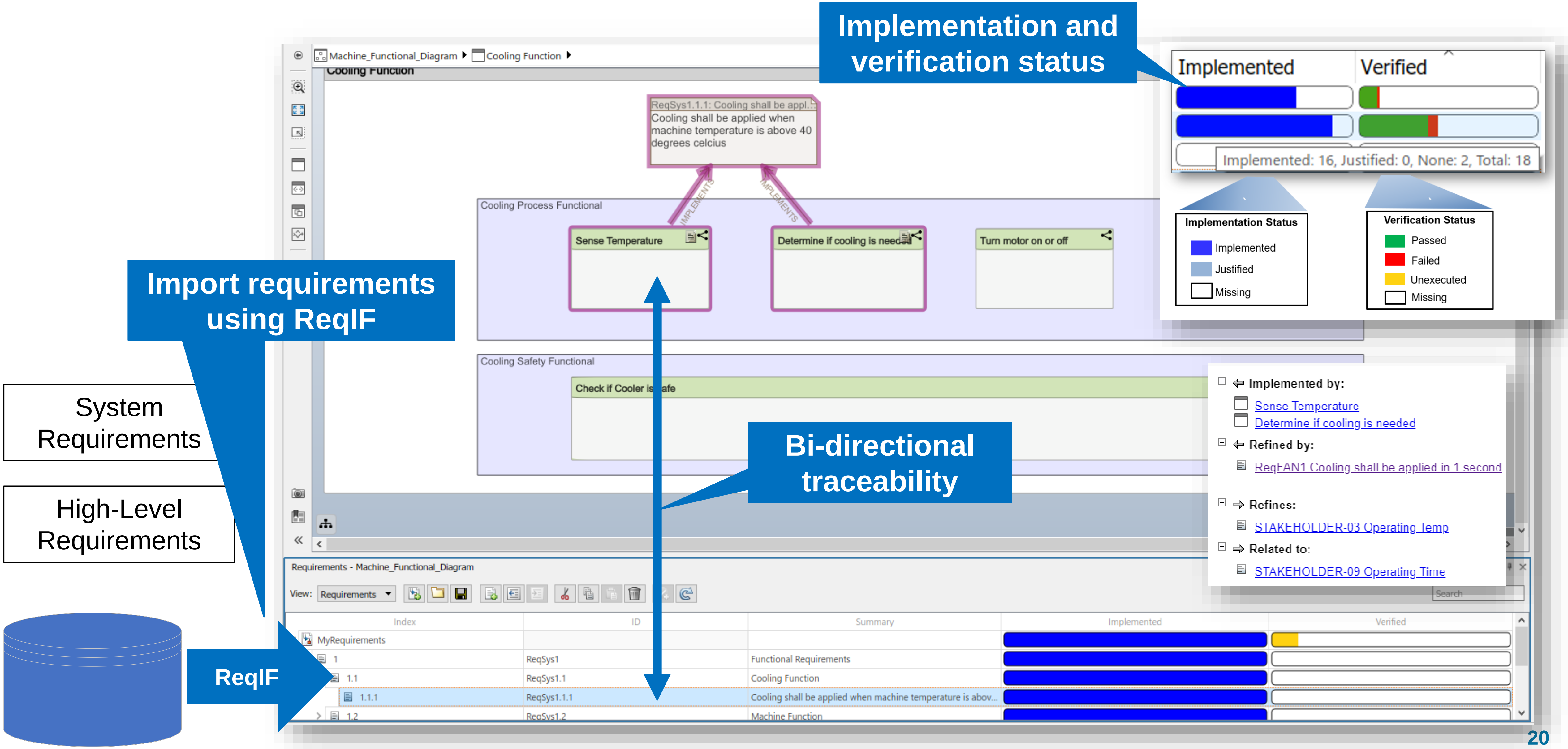
Physical Architecture

WHAT

WHO

WHERE

功能分解和需求追溯 – 功能架构



Implementation and verification status

Import requirements using ReqIF

Bi-directional traceability

System Requirements

High-Level Requirements

ReqIF

Machine_Functional_Diagram ▸ Cooling Function ▸

Cooling Function

ReqSys1.1.1: Cooling shall be appl...
Cooling shall be applied when machine temperature is above 40 degrees celcius

Cooling Process Functional

Sense Temperature

Determine if cooling is needed

Turn motor on or off

Cooling Safety Functional

Check if Cooler is safe

Implemented	Verified

Implemented: 16, Justified: 0, None: 2, Total: 18

Implementation Status

- Implemented
- Justified
- Missing

Verification Status

- Passed
- Failed
- Unexecuted
- Missing

⊟ ⇐ Implemented by:
- Sense Temperature
- Determine if cooling is needed

⊟ ⇐ Refined by:
- ReqFAN1 Cooling shall be applied in 1 second

⊟ ⇒ Refines:
- STAKEHOLDER-03 Operating Temp

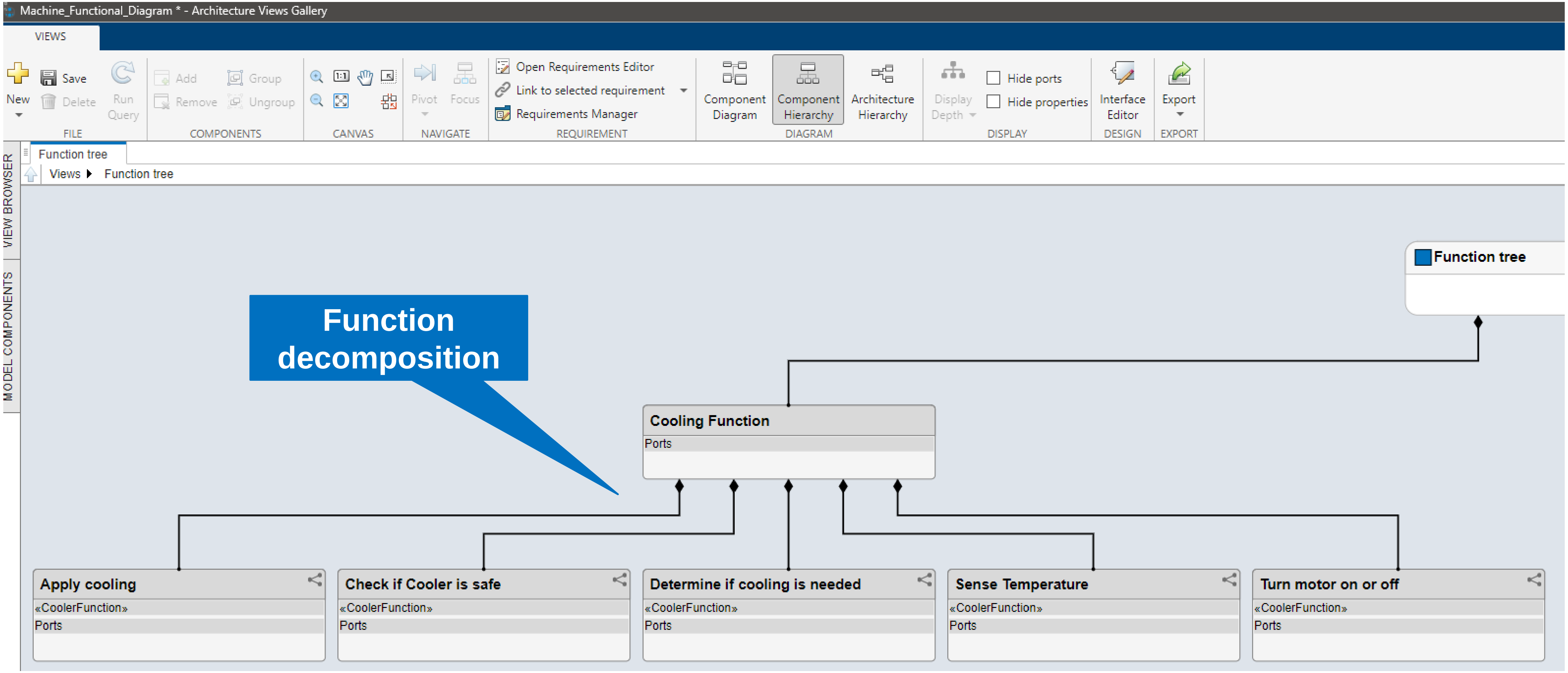
⊟ ⇒ Related to:
- STAKEHOLDER-09 Operating Time

Requirements - Machine_Functional_Diagram

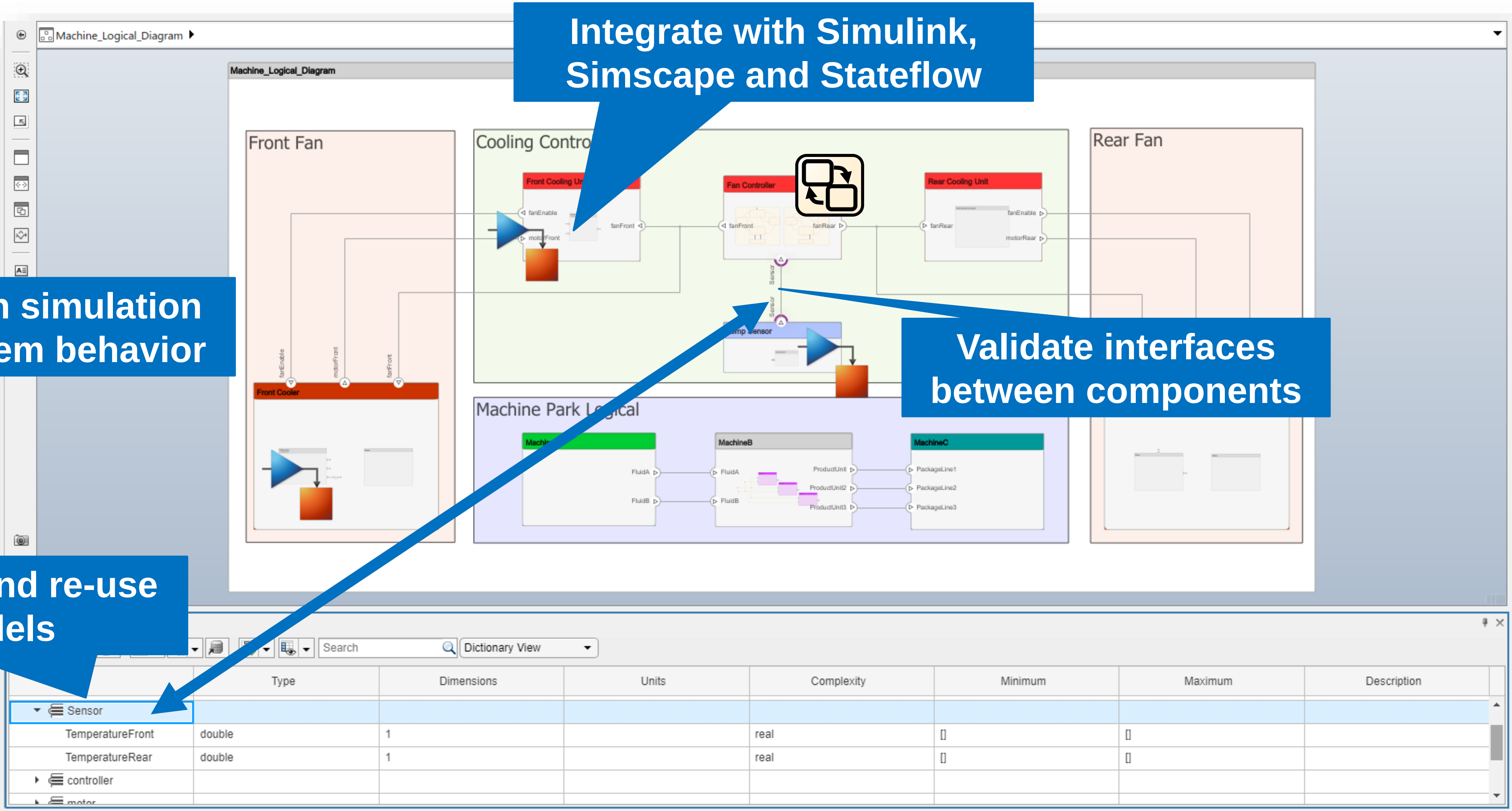
View: Requirements

Index	ID	Summary	Implemented	Verified
MyRequirements				
1	ReqSys1	Functional Requirements		
1.1	ReqSys1.1	Cooling Function		
1.1.1	ReqSys1.1.1	Cooling shall be applied when machine temperature is abov...		
1.2	ReqSys1.2	Machine Function		

理解功能架构 – 层级视图


Machine_Functional_Diagram * - Architecture Views Gallery
VIEWS
New
Save
Delete
Run Query
FILE
Add
Remove
Group
Ungroup
COMPONENTS
CANVAS
Pivot
Focus
NAVIGATE
Open Requirements Editor
Link to selected requirement
Requirements Manager
REQUIREMENT
Component Diagram
Component Hierarchy
Architecture Hierarchy
DIAGRAM
Display Depth
□ Hide ports
□ Hide properties
DISPLAY
Interface Editor
DESIGN
Export
EXPORT
VIEW BROWSER
MODEL COMPONENTS
Function tree
Views ▸ Function tree
Function tree
Function decomposition
Cooling Function
Ports
Apply cooling
«CoolerFunction»
Ports
Check if Cooler is safe
«CoolerFunction»
Ports
Determine if cooling is needed
«CoolerFunction»
Ports
Sense Temperature
«CoolerFunction»
Ports
Turn motor on or off
«CoolerFunction»
Ports

确认组件接口及系统行为 – 逻辑架构



	Type	Dimensions	Units	Complexity	Minimum	Maximum	Description
Sensor							
TemperatureFront	double	1		real	[]	[]	
TemperatureRear	double	1		real	[]	[]	
controller							
motor							

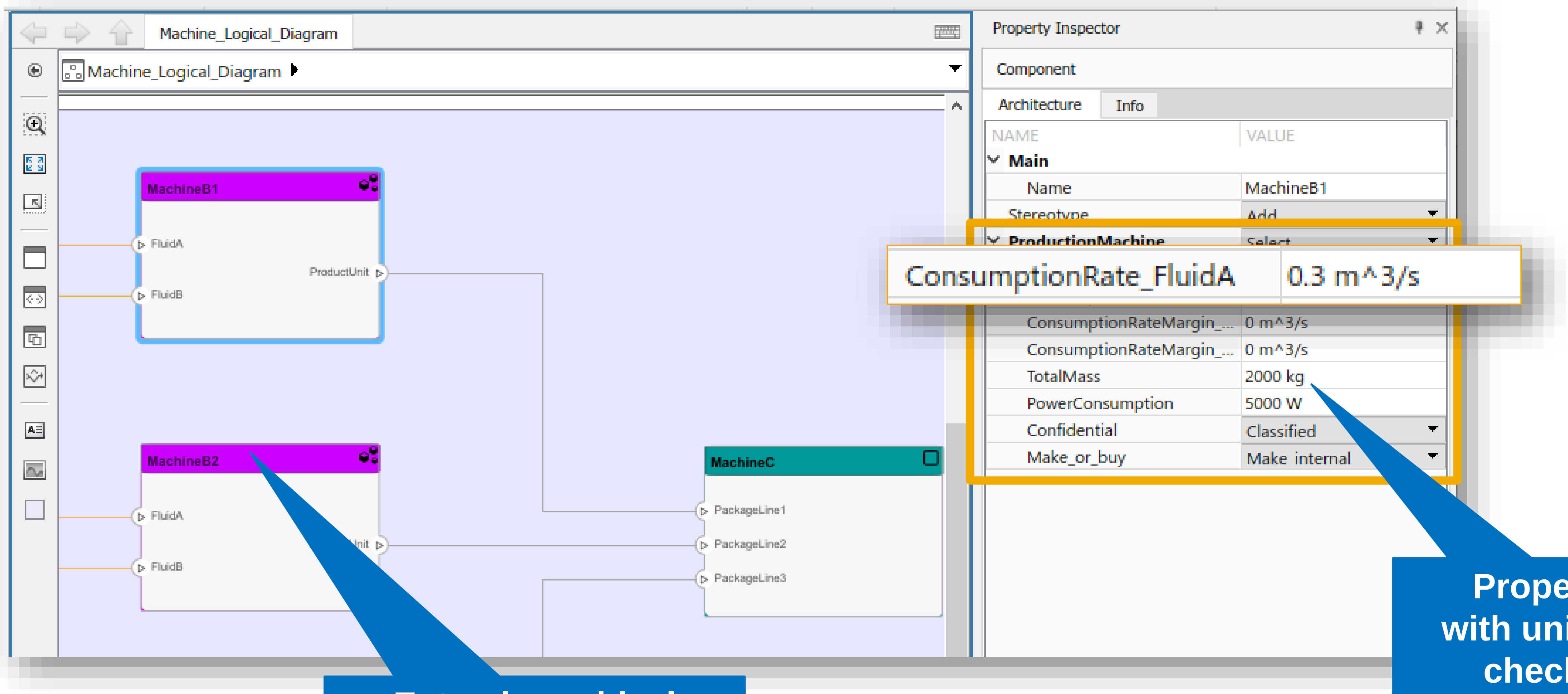
Integrate with Simulink, Simscape and Stateflow

Enable models with simulation to understand system behavior

Validate interfaces between components

Define interfaces and re-use between models

扩展架构属性 – 构型模板

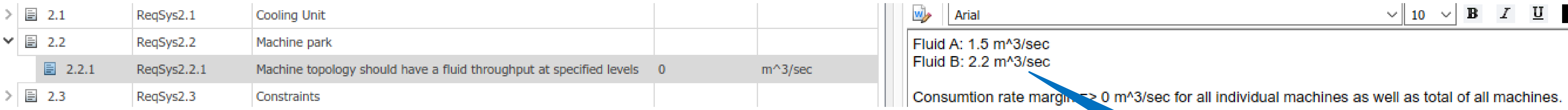


NAME	VALUE
Main	
Name	MachineB1
Stereotype	Add
ProductionMachine	Select
ConsumptionRate_FluidA	0.3 m^3/s
ConsumptionRateMargin_...	0 m^3/s
ConsumptionRateMargin_...	0 m^3/s
TotalMass	2000 kg
PowerConsumption	5000 W
Confidential	Classified
Make_or_buy	Make internal

Properties with units and checking

Extend graphical language with domain specific elements

基于属性的需求验证



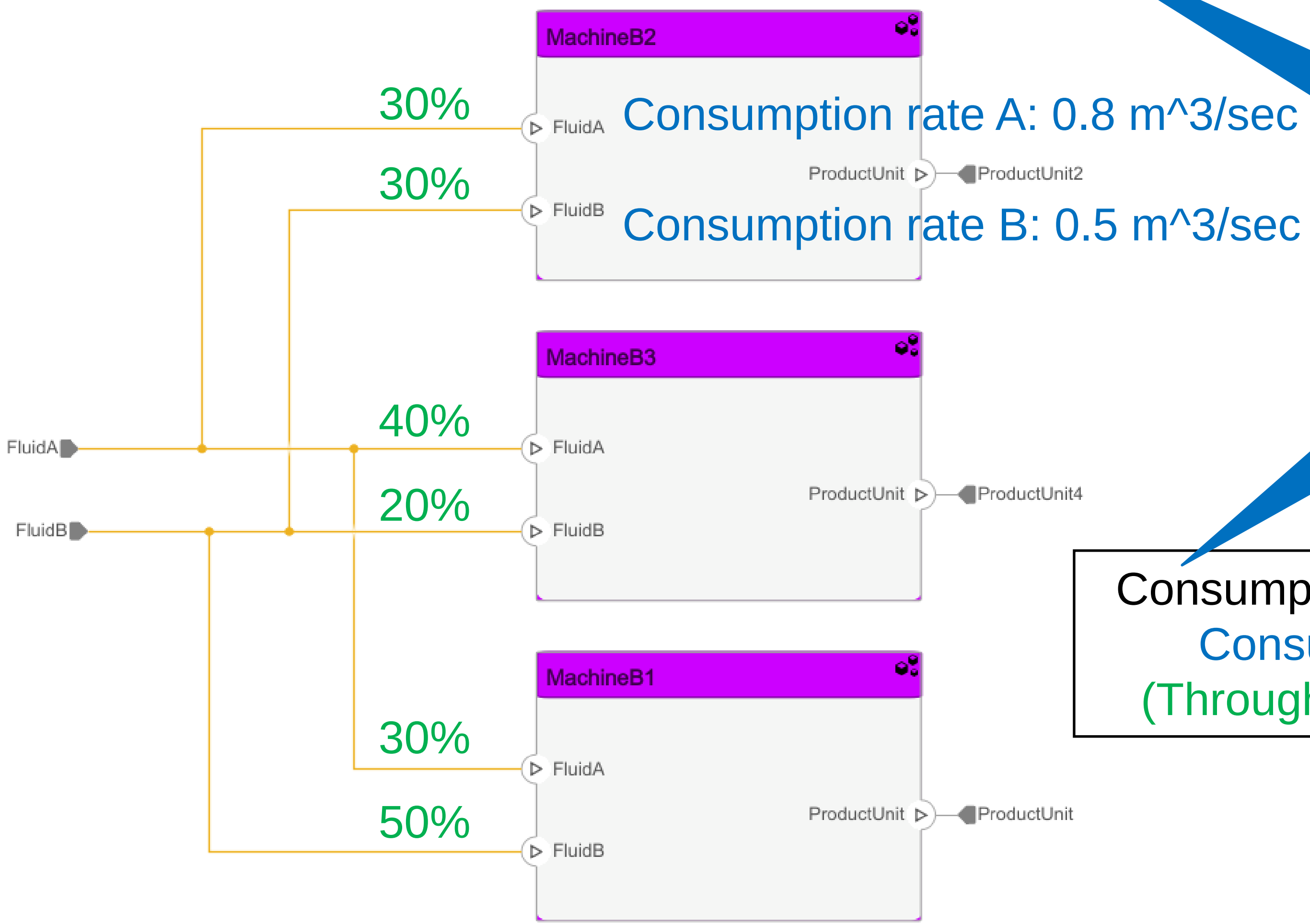
2.1	ReqSys2.1	Cooling Unit		
2.2	ReqSys2.2	Machine park		
2.2.1	ReqSys2.2.1	Machine topology should have a fluid throughput at specified levels	0	m^3/sec
2.3	ReqSys2.3	Constraints		

Fluid A: 1.5 m^3/sec
Fluid B: 2.2 m^3/sec

Consumtion rate margin => 0 m^3/sec for all individual machines as well as total of all machines.

Fluid A – 1.5 m^3/sec

Fluid B – 2.2 m^3/sec



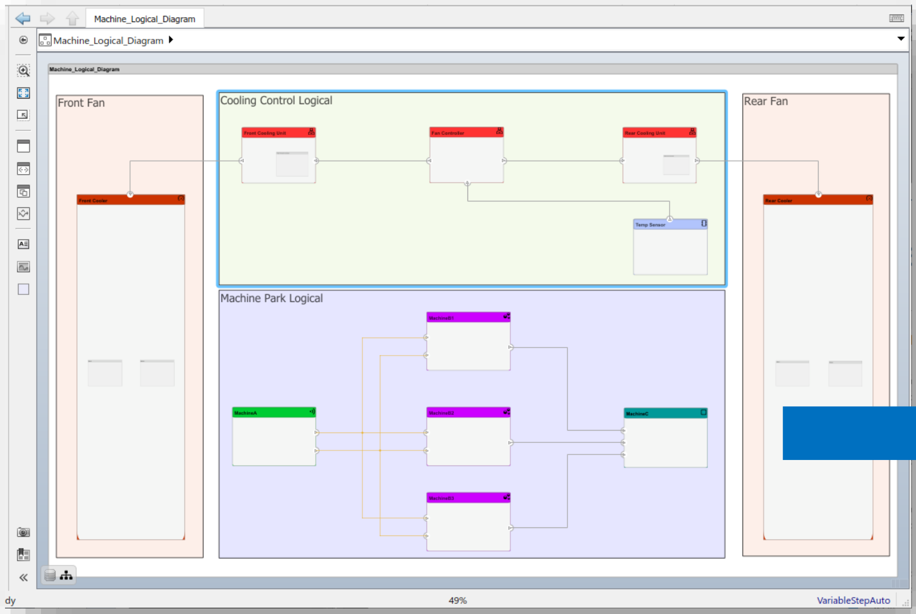
Consumption rate A: 0.8 m^3/sec

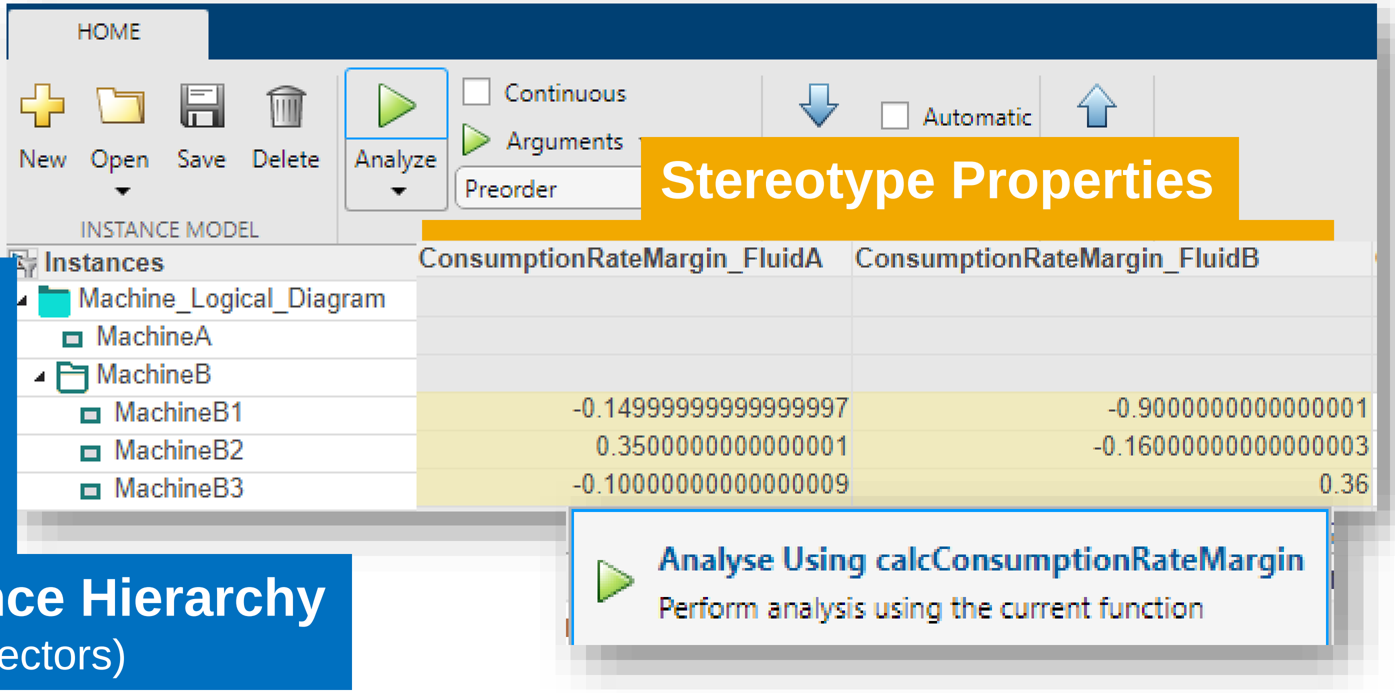
Consumption rate B: 0.5 m^3/sec

Validate whether requirements are met through static analysis

Consumption rate margin = Consumption rate - (Throughput x Fluid rate)

基于属性的设计权衡





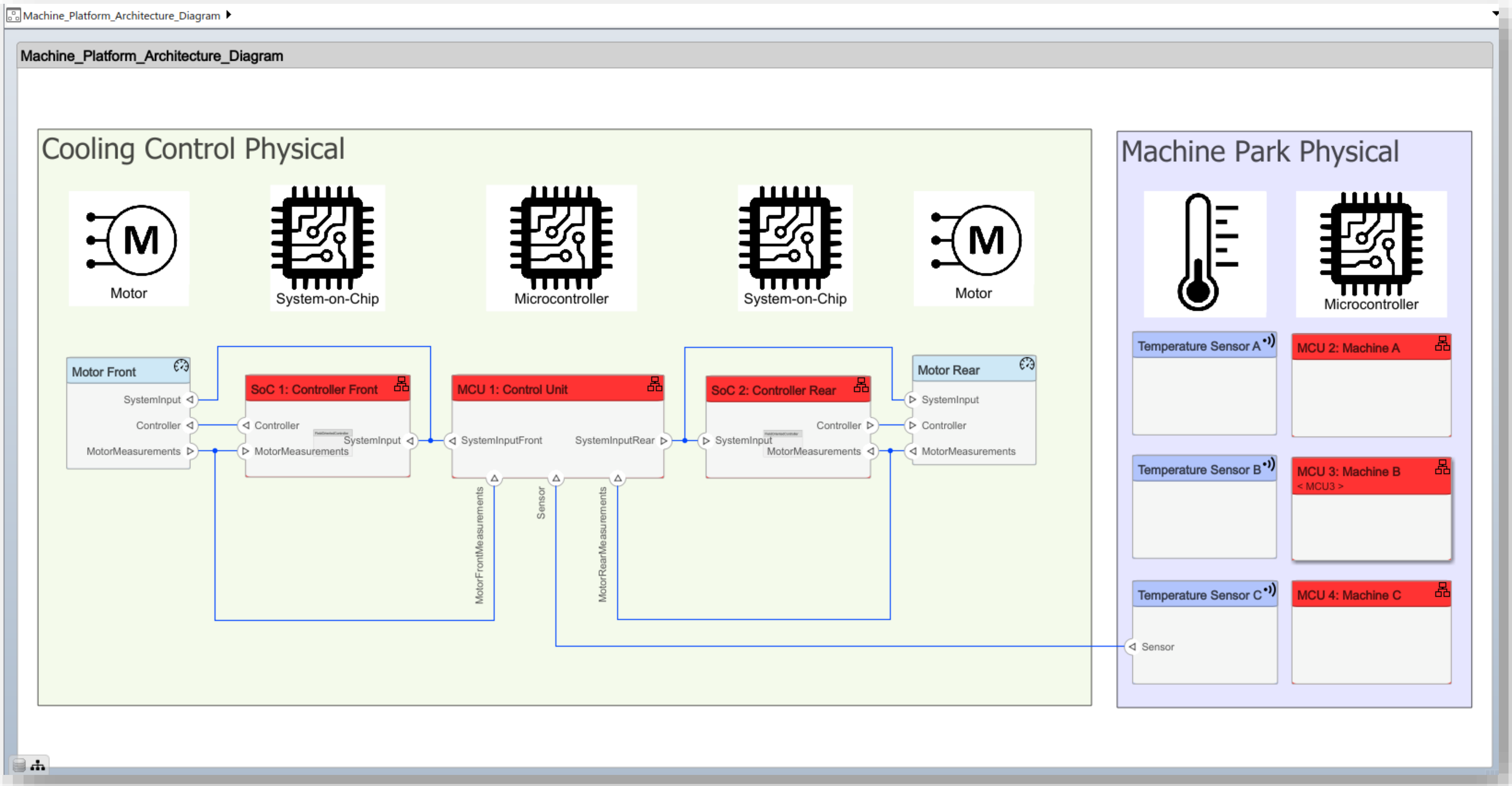
Stereotype Properties

Instances	ConsumptionRateMargin_FluidA	ConsumptionRateMargin_FluidB
Machine_Logical_Diagram		
MachineA		
MachineB		
MachineB1	-0.14999999999999997	-0.9000000000000001
MachineB2	0.3500000000000001	-0.16000000000000003
MachineB3	-0.10000000000000009	0.36

Analyse Using calcConsumptionRateMargin
Perform analysis using the current function

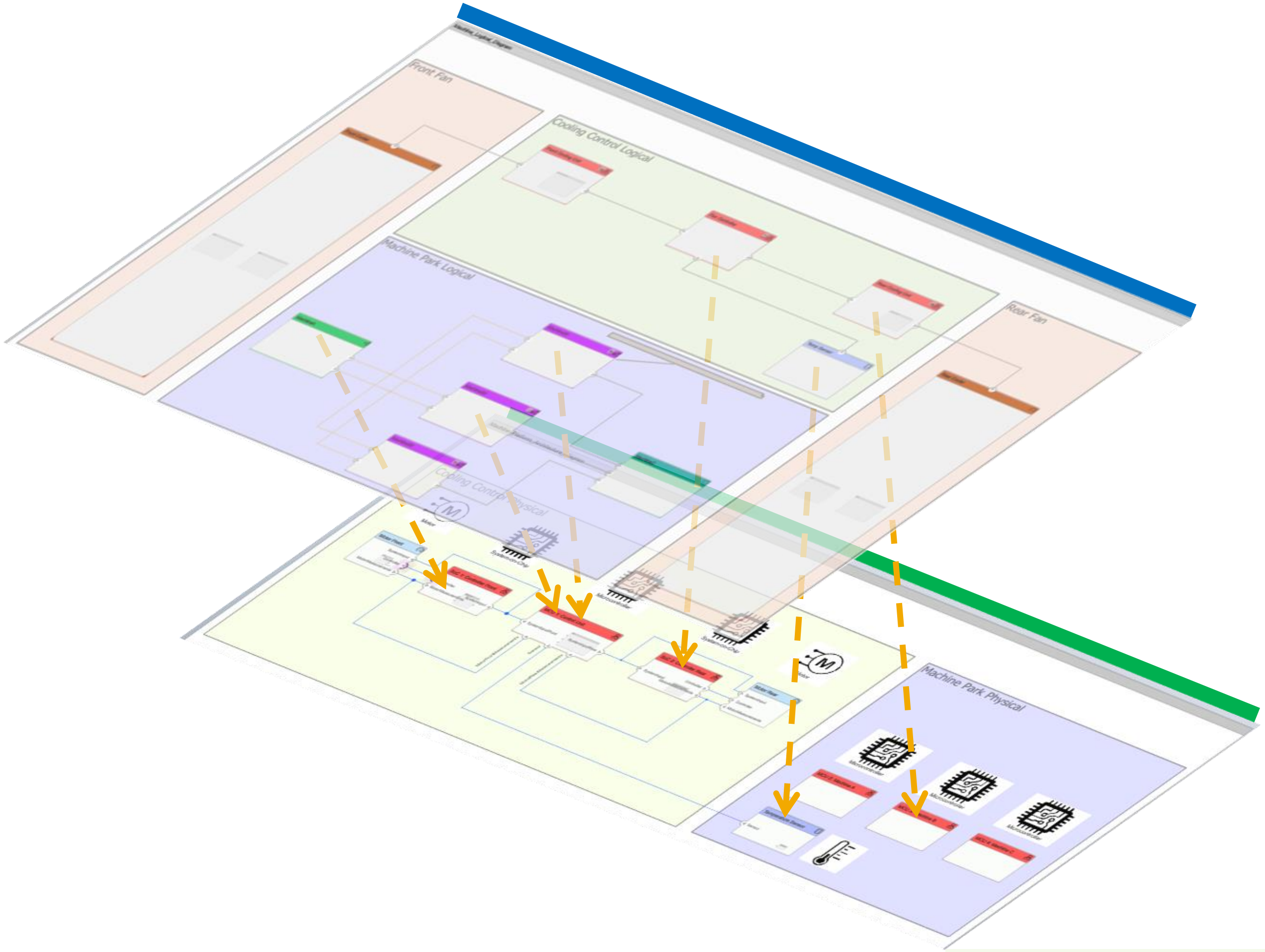
Model Instance Hierarchy
(incl Ports, Connectors)

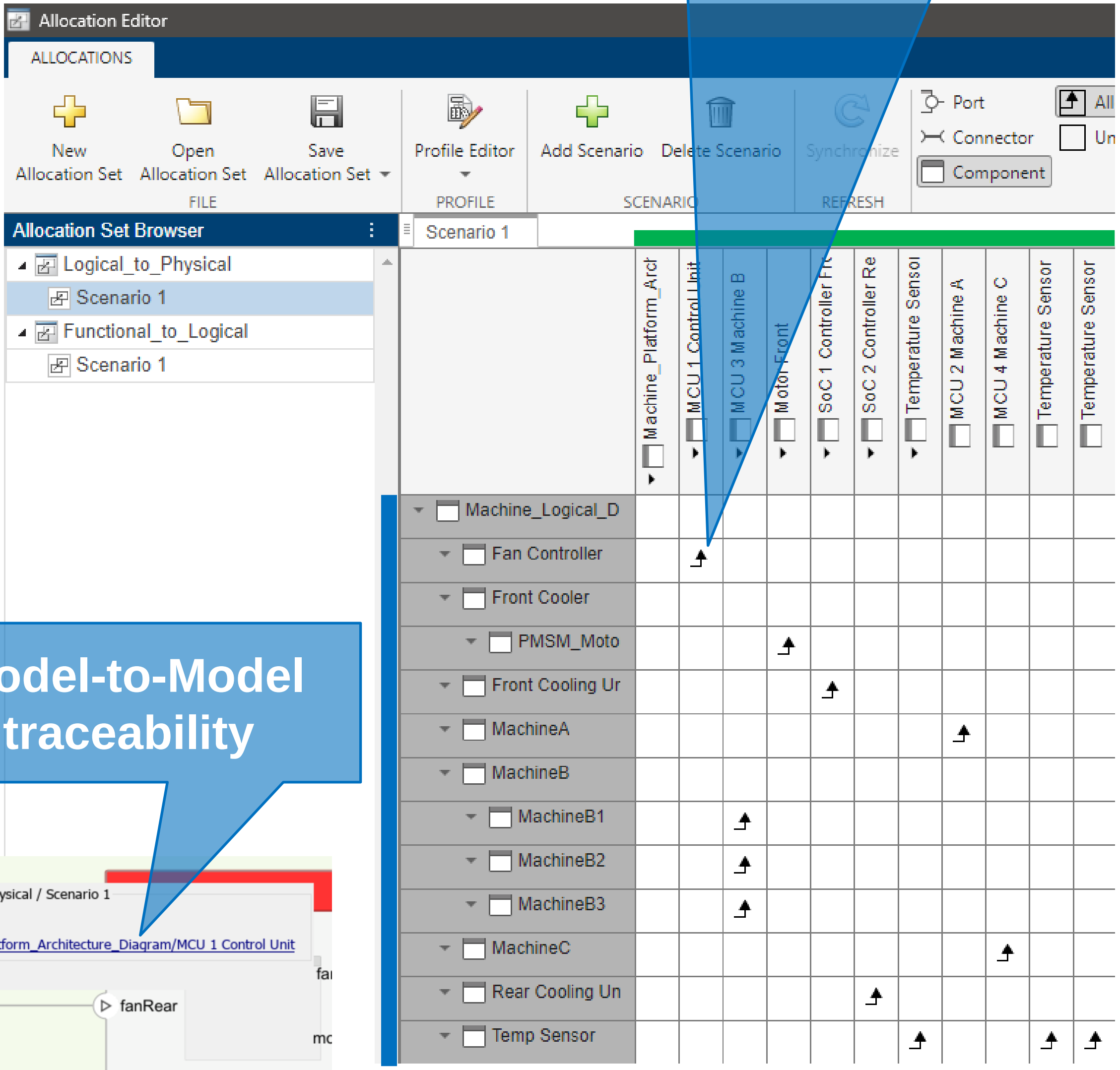
描述部署平台 – 物理架构


Machine_Platform_Architecture_Diagram
Machine_Platform_Architecture_Diagram
Cooling Control Physical
Motor
System-on-Chip
Microcontroller
System-on-Chip
Motor
Motor Front
SystemInput
Controller
MotorMeasurements
SoC 1: Controller Front
Controller
SystemInput
MotorMeasurements
MCU 1: Control Unit
SystemInputFront
SystemInputRear
MotorFrontMeasurements
Sensor
MotorRearMeasurements
SoC 2: Controller Rear
SystemInput
Controller
MotorMeasurements
Motor Rear
SystemInput
Controller
MotorMeasurements
Machine Park Physical
Microcontroller
Temperature Sensor A
MCU 2: Machine A
Temperature Sensor B
MCU 3: Machine B
< MCU3 >
Temperature Sensor C
Sensor
MCU 4: Machine C

架构模型分配和追溯

Allocation between functional, logical and physical architecture models





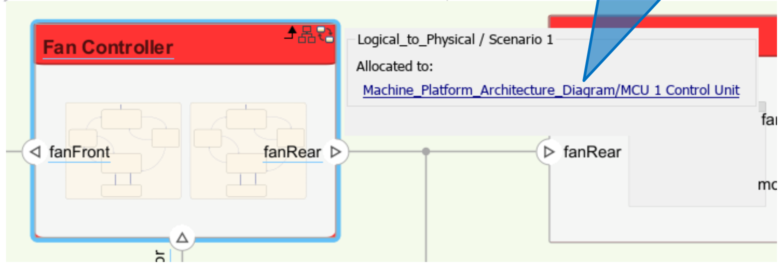
Allocation Set Browser

- Logical_to_Physical
  - Scenario 1
- Functional_to_Logical
  - Scenario 1

Scenario 1

	Machine_Platform_Arct	MCU 1 Control Unit	MCU 3 Machine B	Motor Front	SoC 1 Controller Frt	SoC 2 Controller Re	Temperature Sensor	MCU 2 Machine A	MCU 4 Machine C	Temperature Sensor	Temperature Sensor
Machine_Logical_D											
Fan Controller		↑									
Front Cooler											
PMSM_Moto				↑							
Front Cooling Ur					↑						
MachineA								↑			
MachineB											
MachineB1			↑								
MachineB2			↑								
MachineB3			↑								
MachineC									↑		
Rear Cooling Un						↑					
Temp Sensor							↑			↑	↑

Model-to-Model traceability



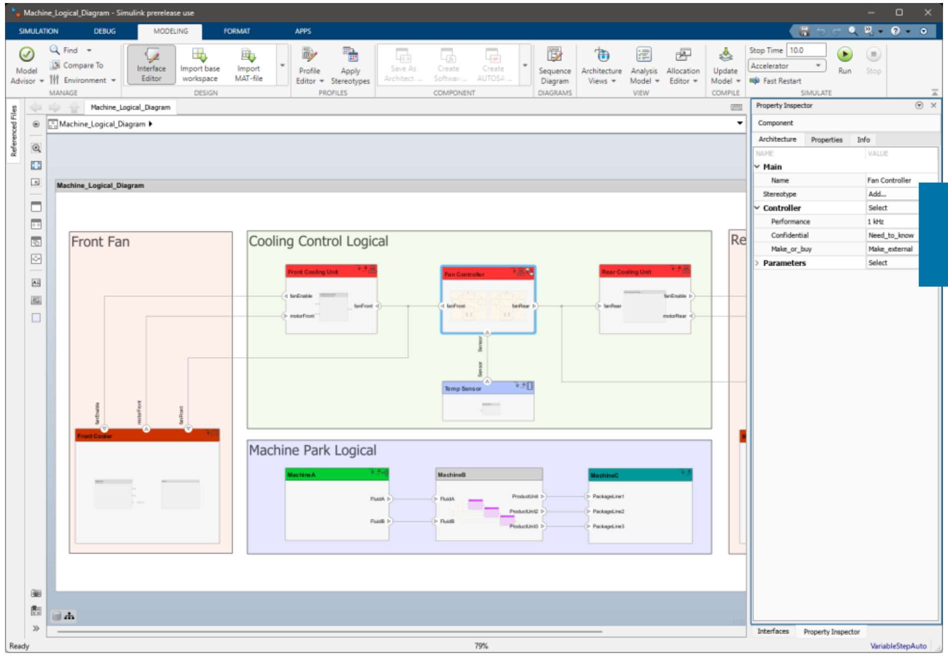
Logical_to_Physical / Scenario 1

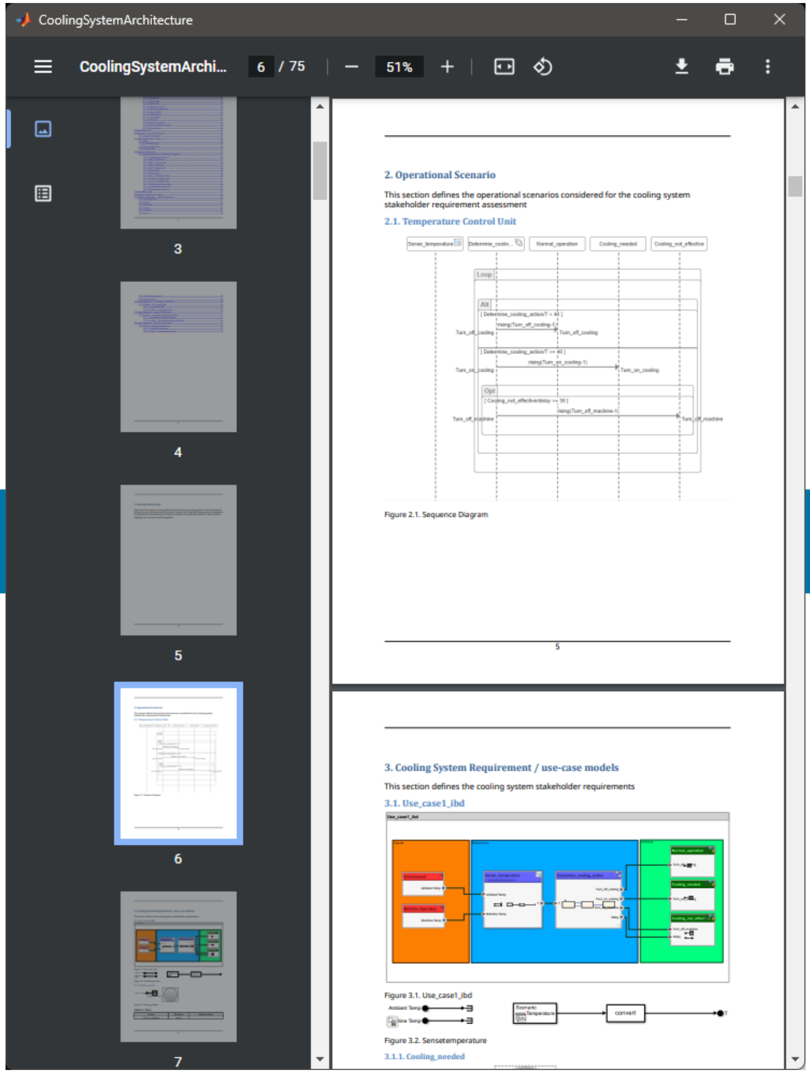
Allocated to:

Machine_Platform_Architecture_Diagram/MCU 1 Control Unit

协同团队开发 – 架构报告

Systems Engineer





2. Operational Scenario

This section defines the operational scenarios considered for the cooling system stakeholder requirement assessment

2.1. Temperature Control Unit

Figure 2.1. Sequence Diagram

5

3. Cooling System Requirement / use-case models

This section defines the cooling system stakeholder requirements

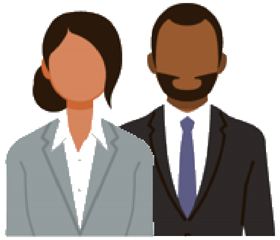
3.1. Use_case1_ibd

Figure 3.1. Use_case1_ibd

Figure 3.2. Sensetemperature

3.1.1. Cooling_needed

Software Engineers



Stakeholders

MBSE开发和管理需求

Maintain requirements as an authoritative source of truth throughout the product development process, by using (simulation) models to:

Manage Requirements

- Transform stakeholder requirements/needs into design requirements using models, simulation and code generation
- Establish traceability between requirements, models and testcases

Authoritative Source of Truth

Digital Thread through Design Process

Connect MBSE with Model-Based Design

Manage Complexity

- Explore the design space through (reusable) trade-off studies
- Through views and traceable architecture models

Digital Thread through Design Process

Connect MBSE with Model-Based Design

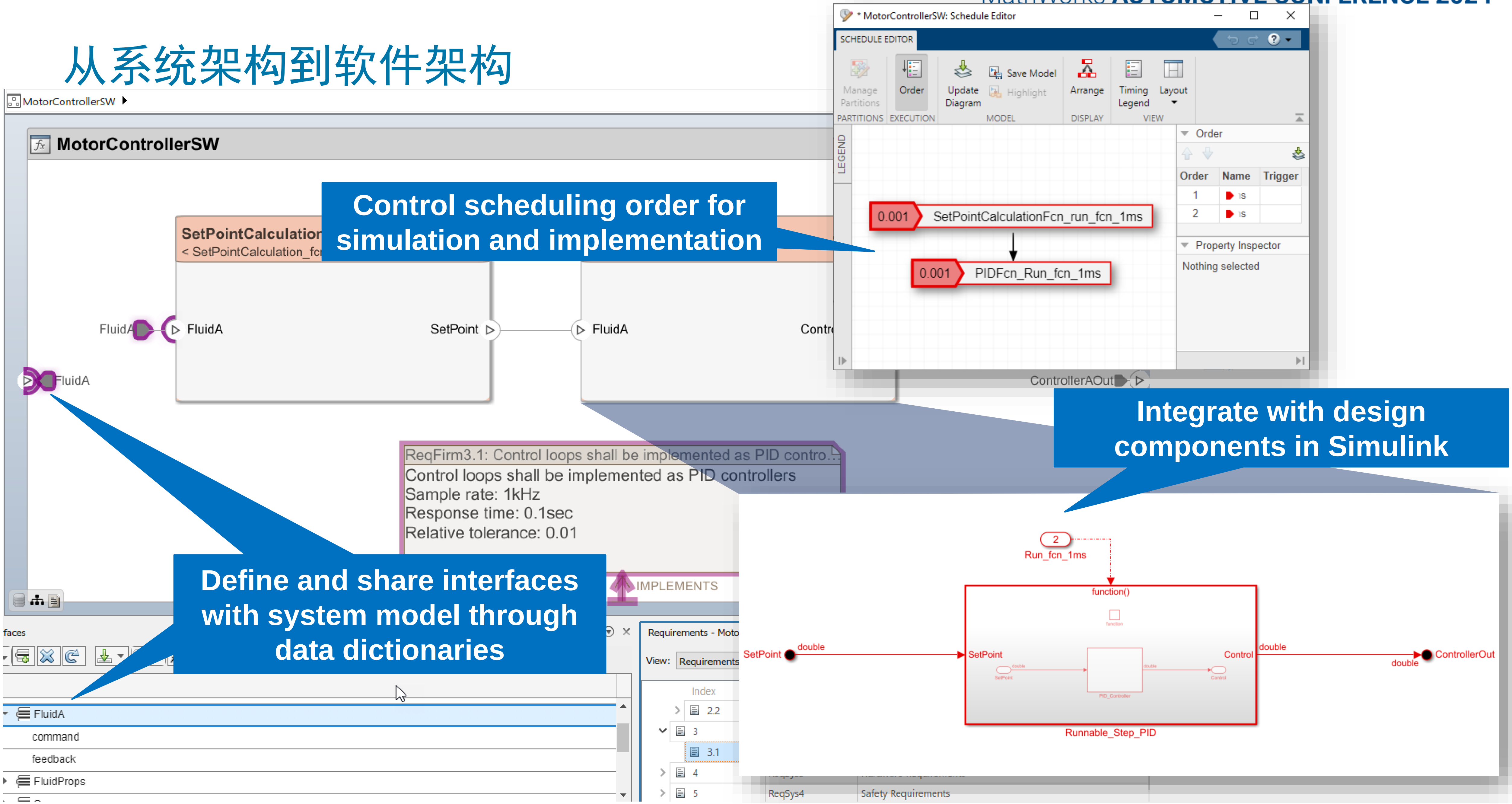
Manage Interfaces

- Connect system architecture with software architecture, component implementation, FMEA (fault injection models)

Digital Thread through Design Process

Connect MBSE with Model-Based Design

从系统架构到软件架构



ReqFirm3.1: Control loops shall be implemented as PID contro...
Control loops shall be implemented as PID controllers
Sample rate: 1kHz
Response time: 0.1sec
Relative tolerance: 0.01

Control scheduling order for simulation and implementation

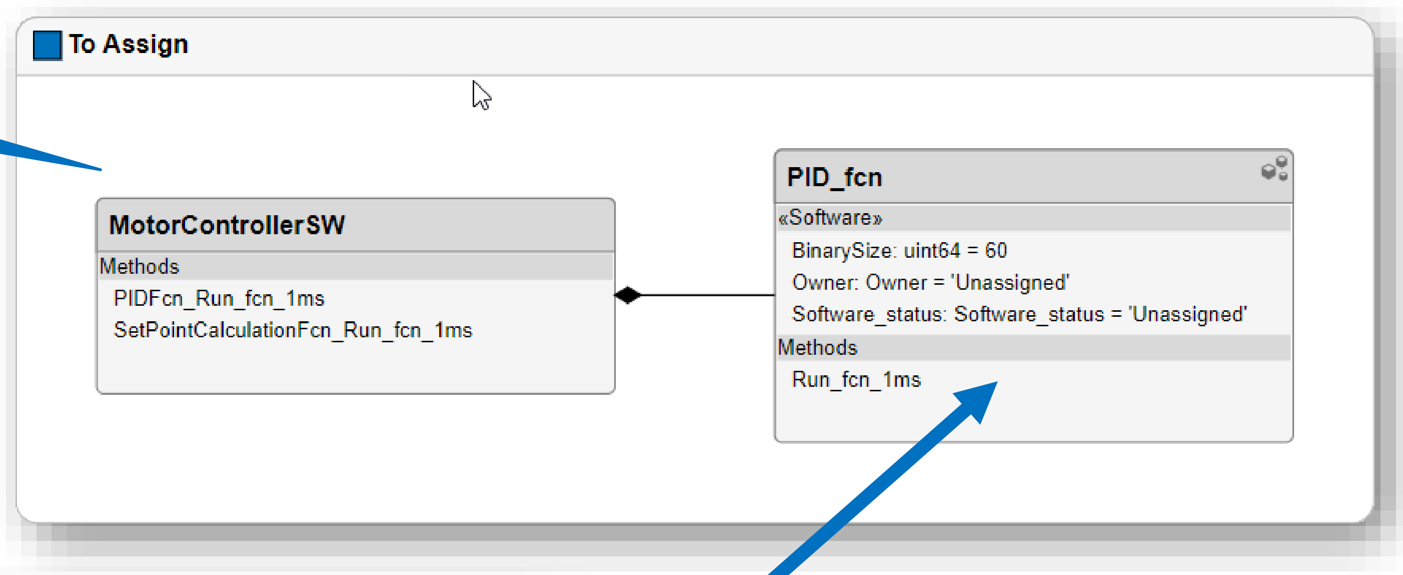
Integrate with design components in Simulink

Define and share interfaces with system model through data dictionaries

管理复杂度 – 软件视图

Views filtered by software status as a class diagram

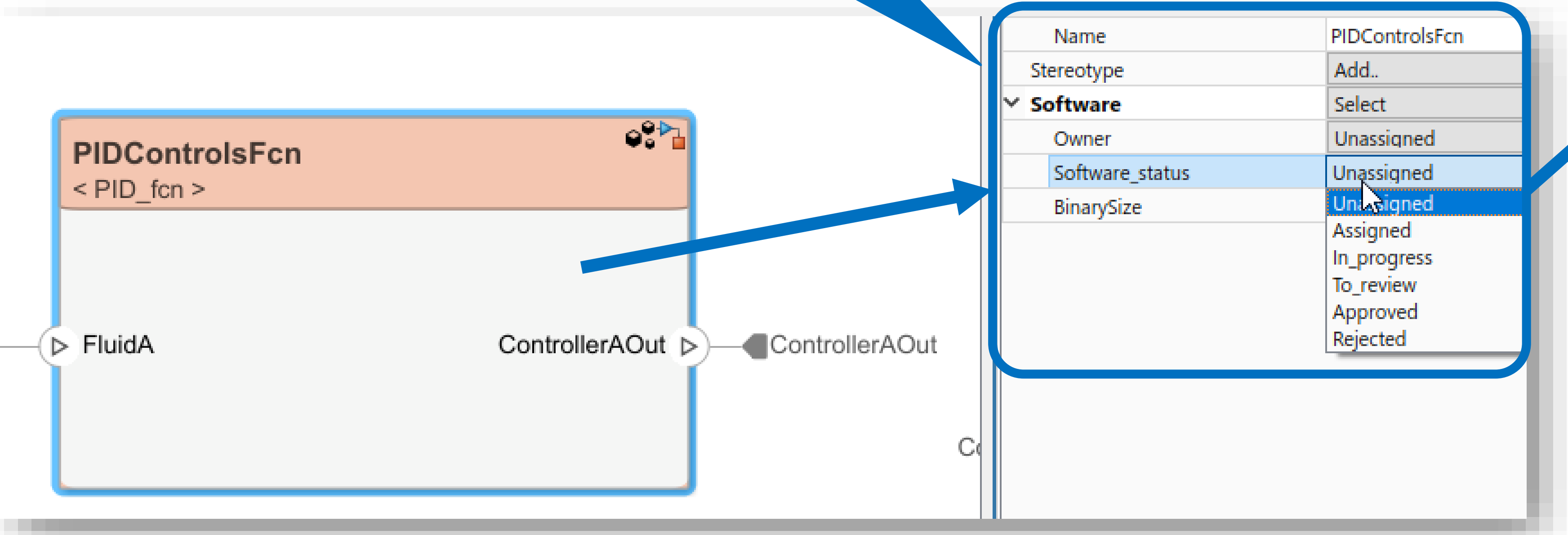
Stereotypes to specify custom properties



To Assign

MotorControllerSW
Methods
PIDFcn_Run_fcn_1ms
SetPointCalculationFcn_Run_fcn_1ms

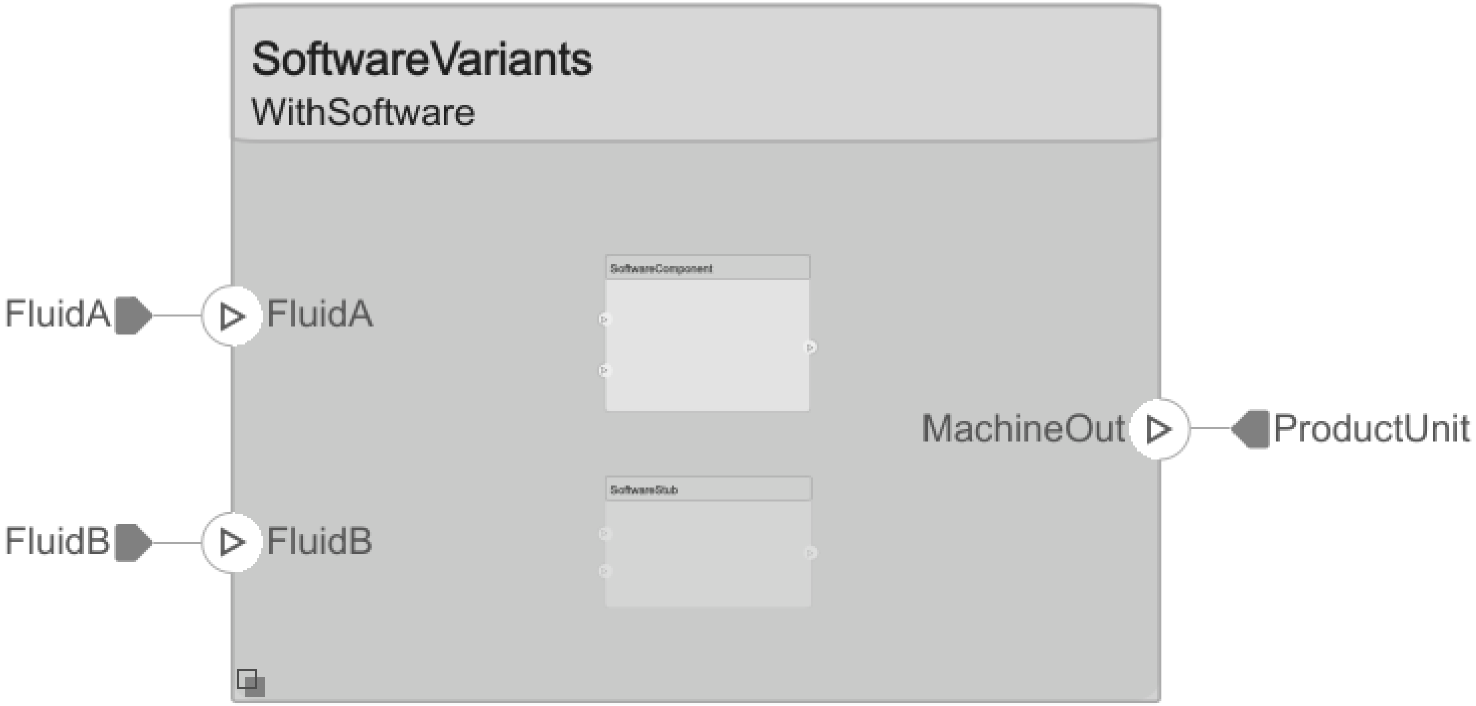
PID_fcn
«Software»
BinarySize: uint64 = 60
Owner: Owner = 'Unassigned'
Software_status: Software_status = 'Unassigned'
Methods
Run_fcn_1ms

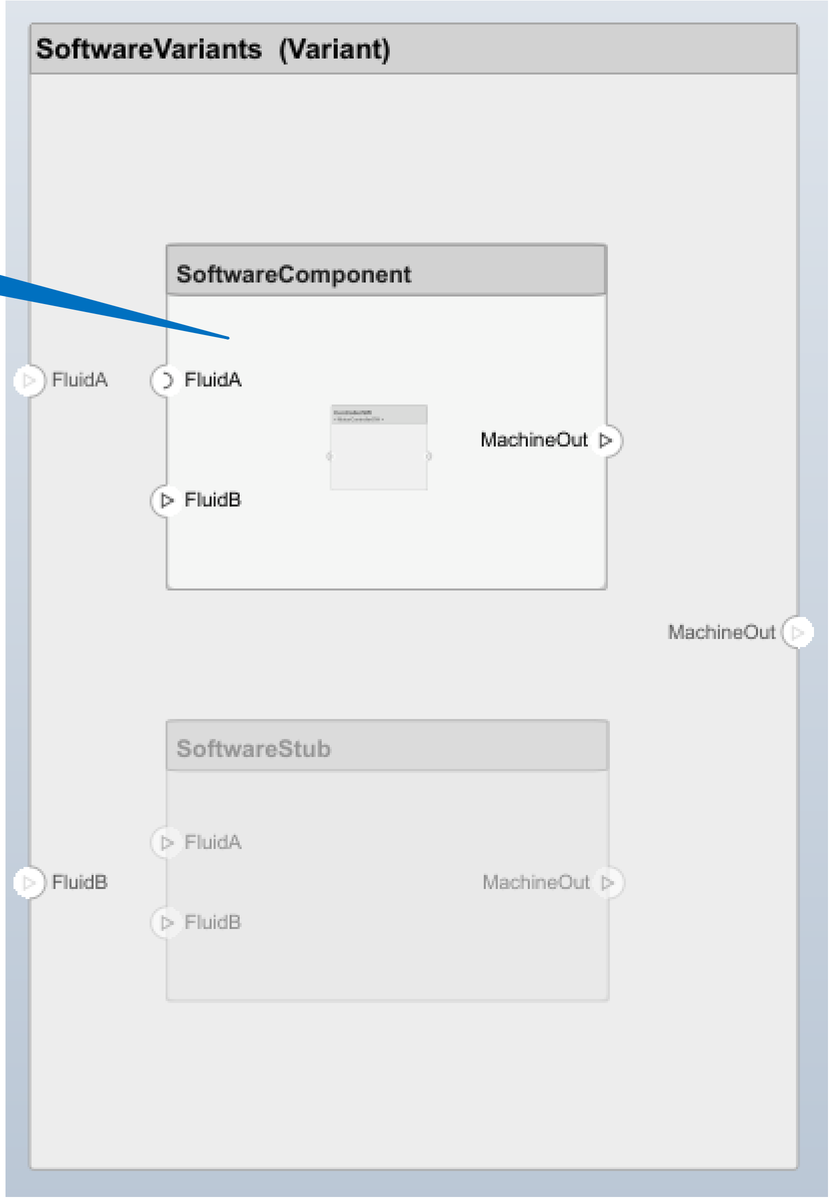


Name	PIDControlsFcn
Stereotype	Add..
Software	Select
Owner	Unassigned
Software_status	Unassigned
BinarySize	Unassigned Assigned In_progress To_review Approved Rejected

管理架构配置 – 变型设计

Active software variant



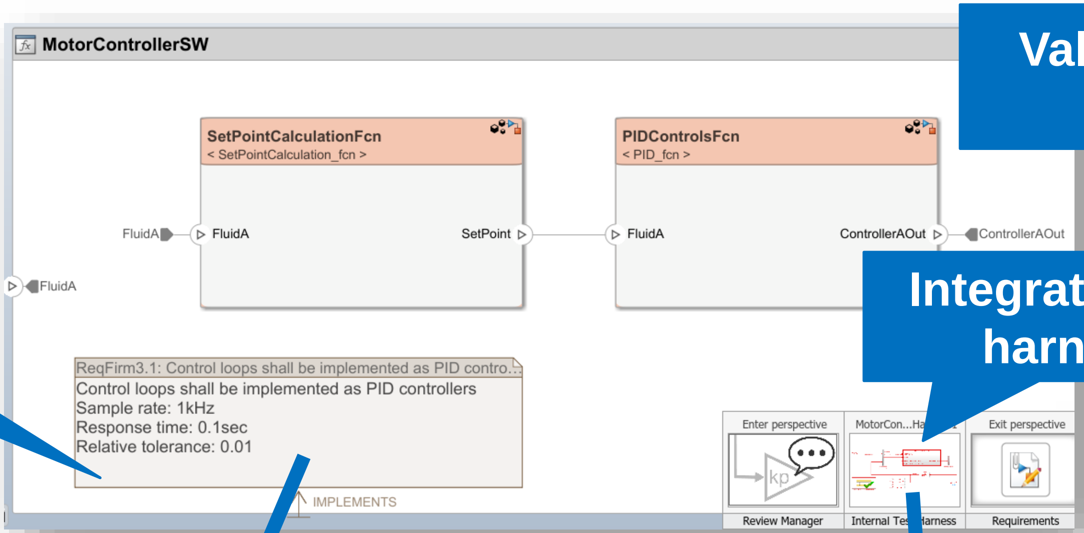


测试和验证软件组件

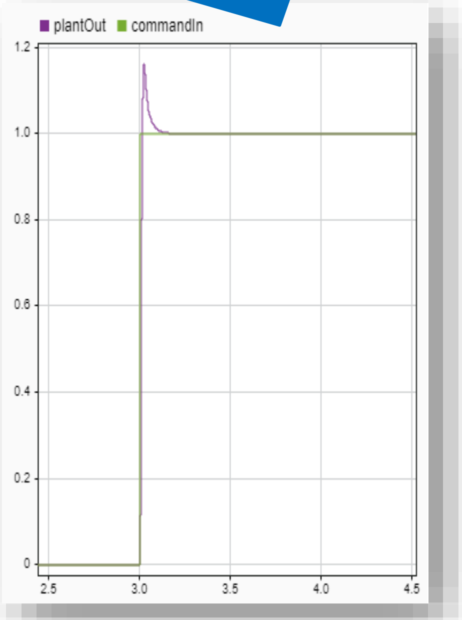
Capture design requirements through unit tests

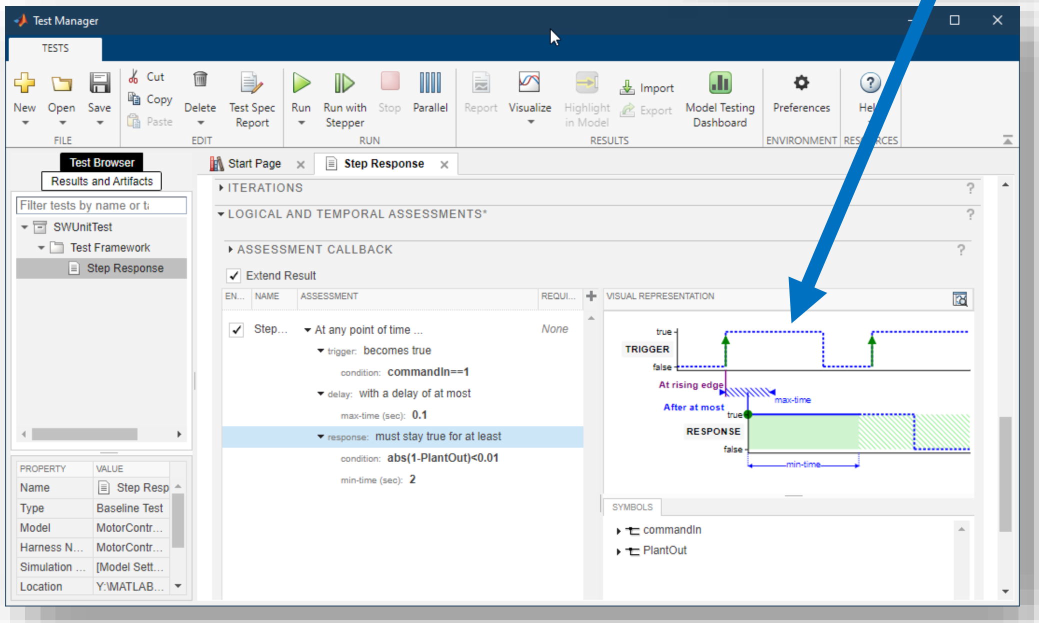
Validate and analyze test results through simulation

Integrated test harness



ReqFirm3.1: Control loops shall be implemented as PID contro...
Control loops shall be implemented as PID controllers
Sample rate: 1kHz
Response time: 0.1sec
Relative tolerance: 0.01





- SWUnitTest
  - Test Framework
    - Step Response

PROPERTY	VALUE
Name	Step Resp
Type	Baseline Test
Model	MotorContr...
Harness N...	MotorContr...
Simulation ...	[Model Sett...
Location	Y:\MATLAB...

ITERATIONS

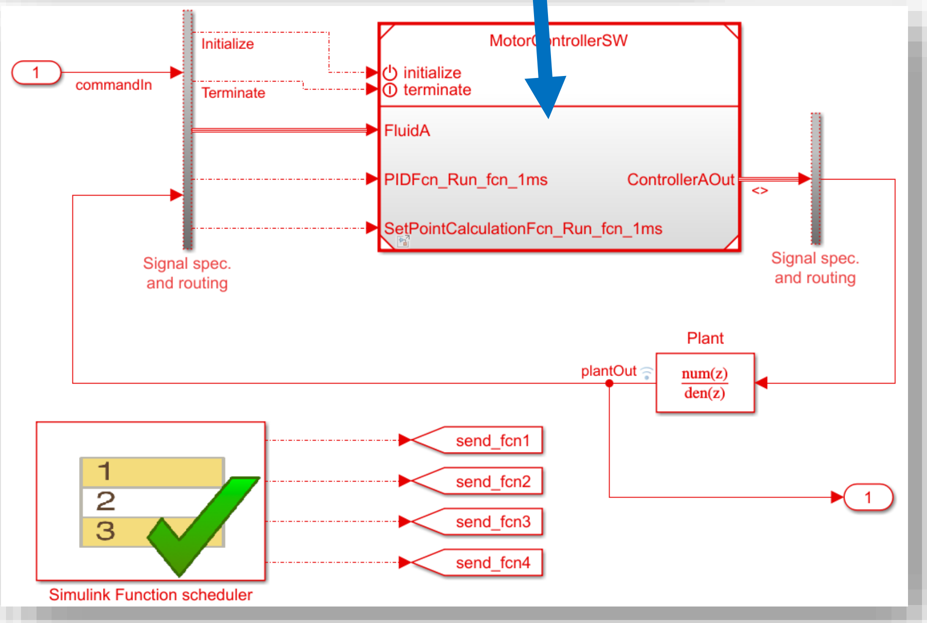
LOGICAL AND TEMPORAL ASSESSMENTS*

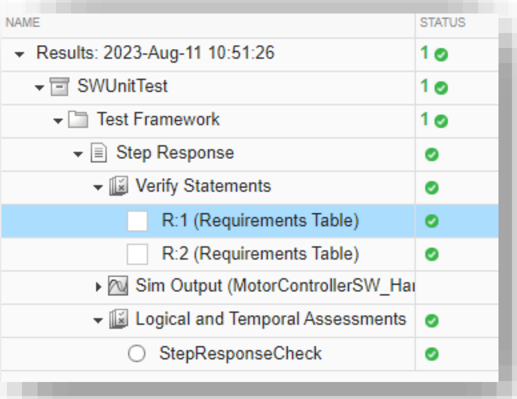
ASSESSMENT CALLBACK

☑ Extend Result

EN...	NAME	ASSESSMENT	REQUI...
☑	Step...	At any point of time ...	None
		trigger: becomes true	
		condition: commandIn==1	
		delay: with a delay of at most	
		max-time (sec): 0.1	
		response: must stay true for at least	
		condition: abs(1-PlantOut)<0.01	
		min-time (sec): 2	

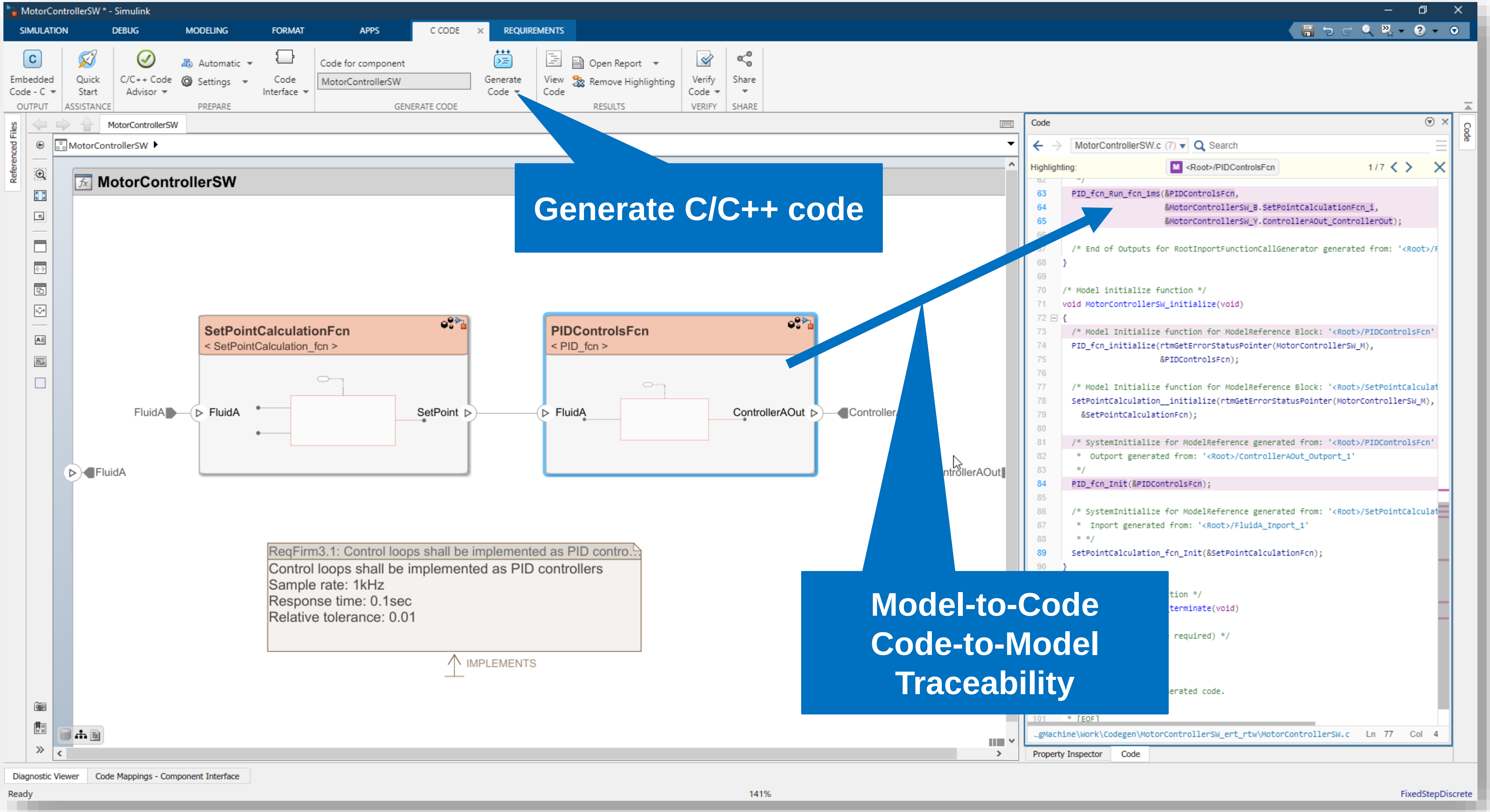
commandIn
PlantOut





NAME	STATUS
Results: 2023-Aug-11 10:51:26	1 ✓
SWUnitTest	1 ✓
Test Framework	1 ✓
Step Response	✓
Verify Statements	✓
R:1 (Requirements Table)	✓
R:2 (Requirements Table)	✓
Sim Output (MotorControllerSW_Ha	
Logical and Temporal Assessments	✓
StepResponseCheck	✓

生成可追溯的代码



ReqFirm3.1: Control loops shall be implemented as PID contro...
Control loops shall be implemented as PID controllers
Sample rate: 1kHz
Response time: 0.1sec
Relative tolerance: 0.01

```
  */
  PID_fcn_Run_fcn_1ms(&PIDControlsFcn,
                      &MotorControllerSW_B.SetPointCalculationFcn_i,
                      &MotorControllerSW_Y.ControllerAOut_ControllerOut);

  /* End of Outputs for RootInportFunctionCallGenerator generated from: '<Root>/F
}

/* Model initialize function */
void MotorControllerSW_initialize(void)
{
  /* Model Initialize function for ModelReference Block: '<Root>/PIDControlsFcn'
  PID_fcn_initialize(rtmGetErrorStatusPointer(MotorControllerSW_M),
                     &PIDControlsFcn);

  /* Model Initialize function for ModelReference Block: '<Root>/SetPointCalculat
  SetPointCalculation__initialize(rtmGetErrorStatusPointer(MotorControllerSW_M),
    &SetPointCalculationFcn);

  /* SystemInitialize for ModelReference generated from: '<Root>/PIDControlsFcn'
   *  Outport generated from: '<Root>/ControllerAOut_Outport_1'
   */
  PID_fcn_Init(&PIDControlsFcn);

  /* SystemInitialize for ModelReference generated from: '<Root>/SetPointCalculat
   *  Inport generated from: '<Root>/FluidA_Inport_1'
   * */
  SetPointCalculation_fcn_Init(&SetPointCalculationFcn);
}

tion */
terminate(void)

required) */

erated code.

 * [EOF]
```

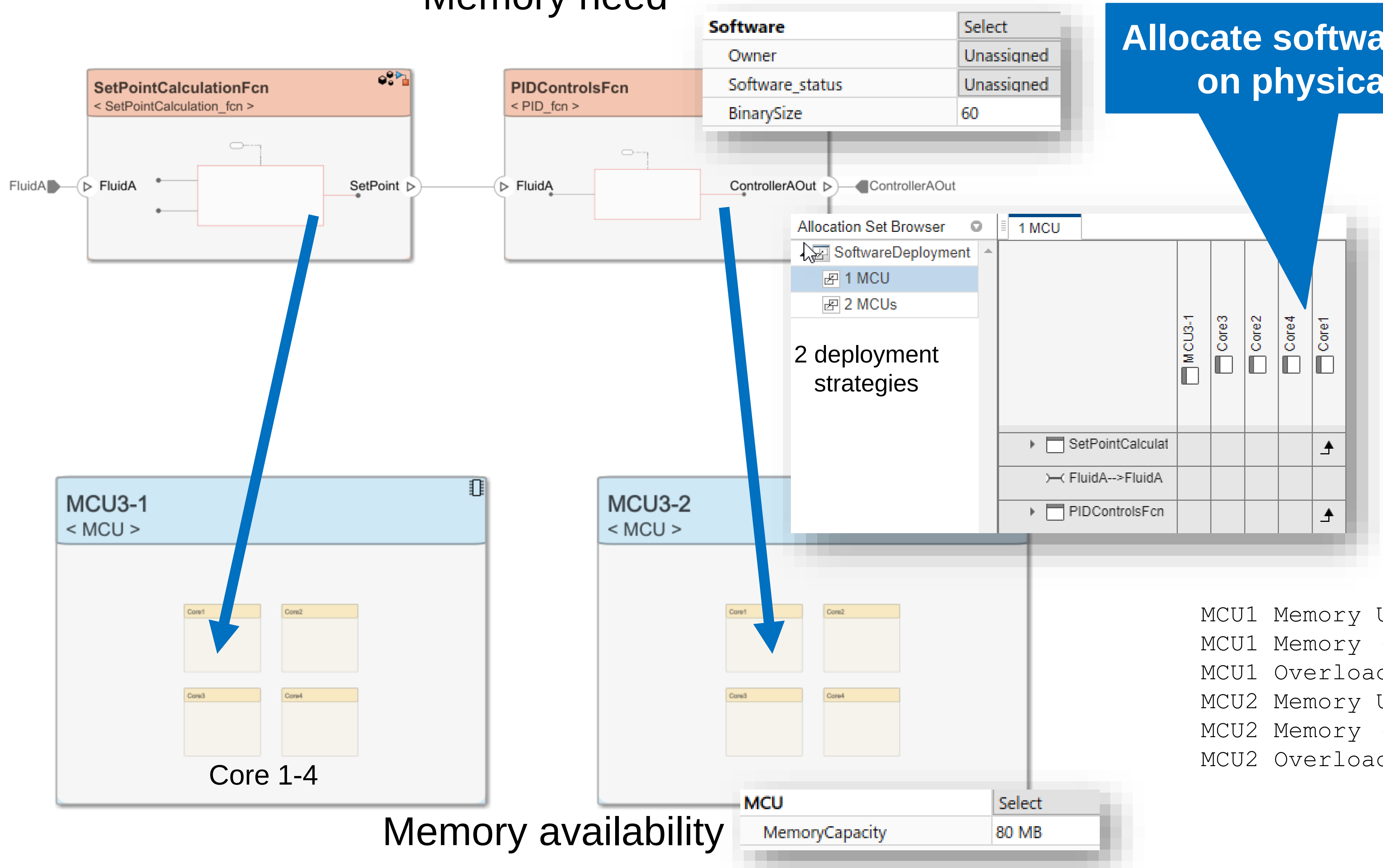
…gMachine\Work\Codegen\MotorControllerSW_ert_rtw\MotorControllerSW.c   Ln 77   Col 4

Generate C/C++ code

Model-to-Code
Code-to-Model
Traceability

分配软件组件到硬件平台

Memory need



Software	Select
Owner	Unassigned
Software_status	Unassigned
BinarySize	60

2 deployment strategies

Core 1-4

Memory availability

MCU	Select
MemoryCapacity	80 MB

Allocate software components on physical platforms

Analyze deployment impact on memory

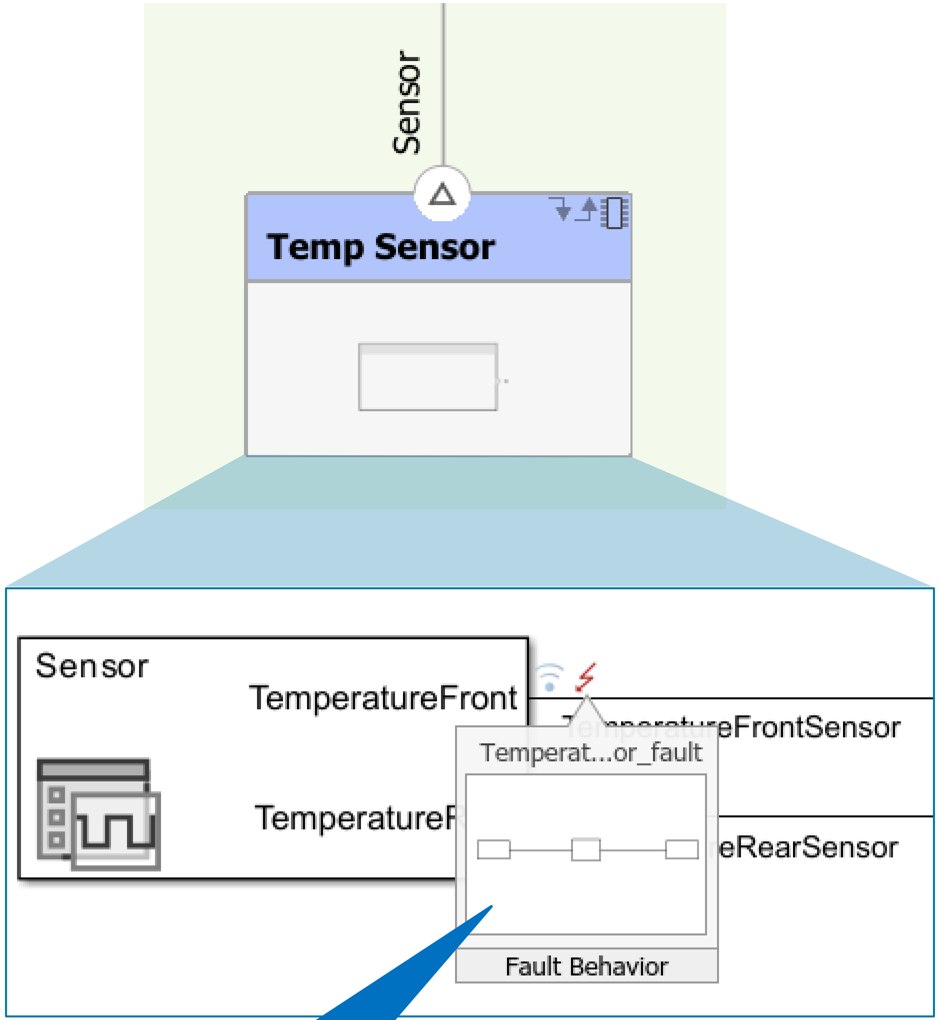
```
                           1 MCU    2 MCUs
                           _____    ______

MCU1 Memory Used (MB)       100       60
MCU1 Memory (MB)             80       80
MCU1 Overloaded               1        0
MCU2 Memory Used (MB)         0       40
MCU2 Memory (MB)             80       80
MCU2 Overloaded               0        0
```

Memory overloaded

Memory not overloaded

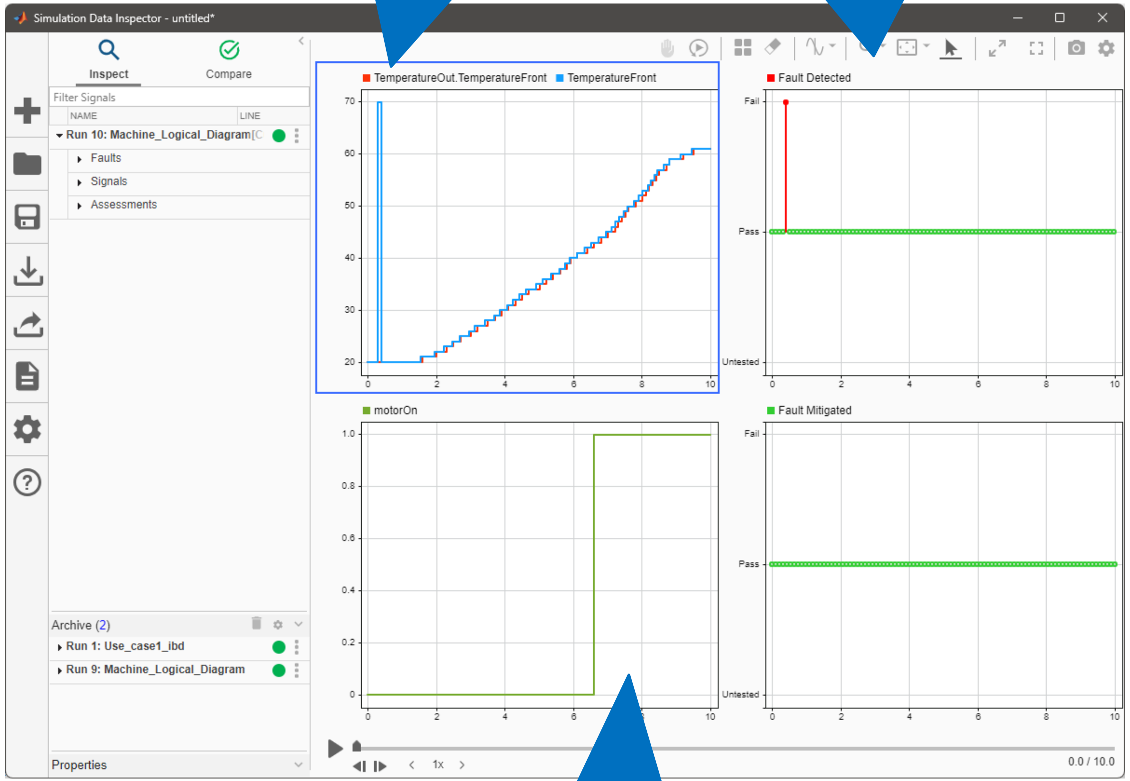
故障注入和仿真



Sensor fault model

Sensor fault injected

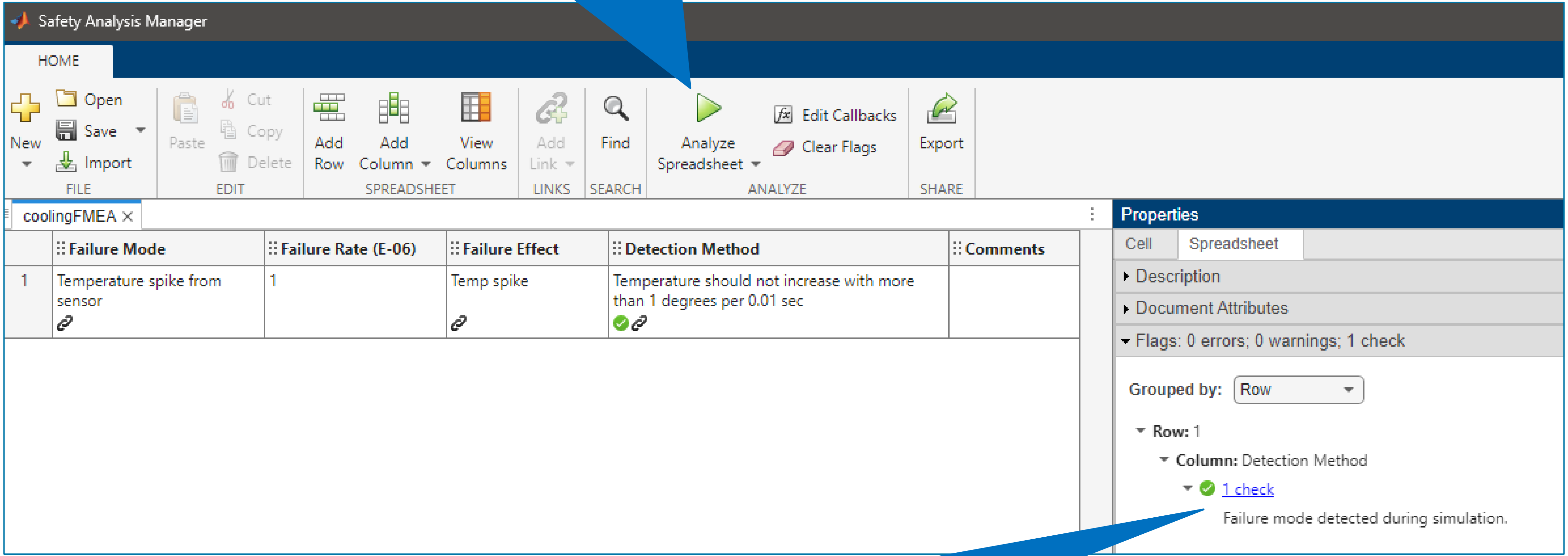
Sensor fault detected & mitigated



Cooling enabled

基于模型的安全分析

Analyze, simulate and report FMEA analyses



	Failure Mode	Failure Rate (E-06)	Failure Effect	Detection Method	Comments
1	Temperature spike from sensor	1	Temp spike	Temperature should not increase with more than 1 degrees per 0.01 sec ✓	

Flags: 0 errors; 0 warnings; 1 check

Grouped by: Row

Row: 1

Column: Detection Method

✓ 1 check

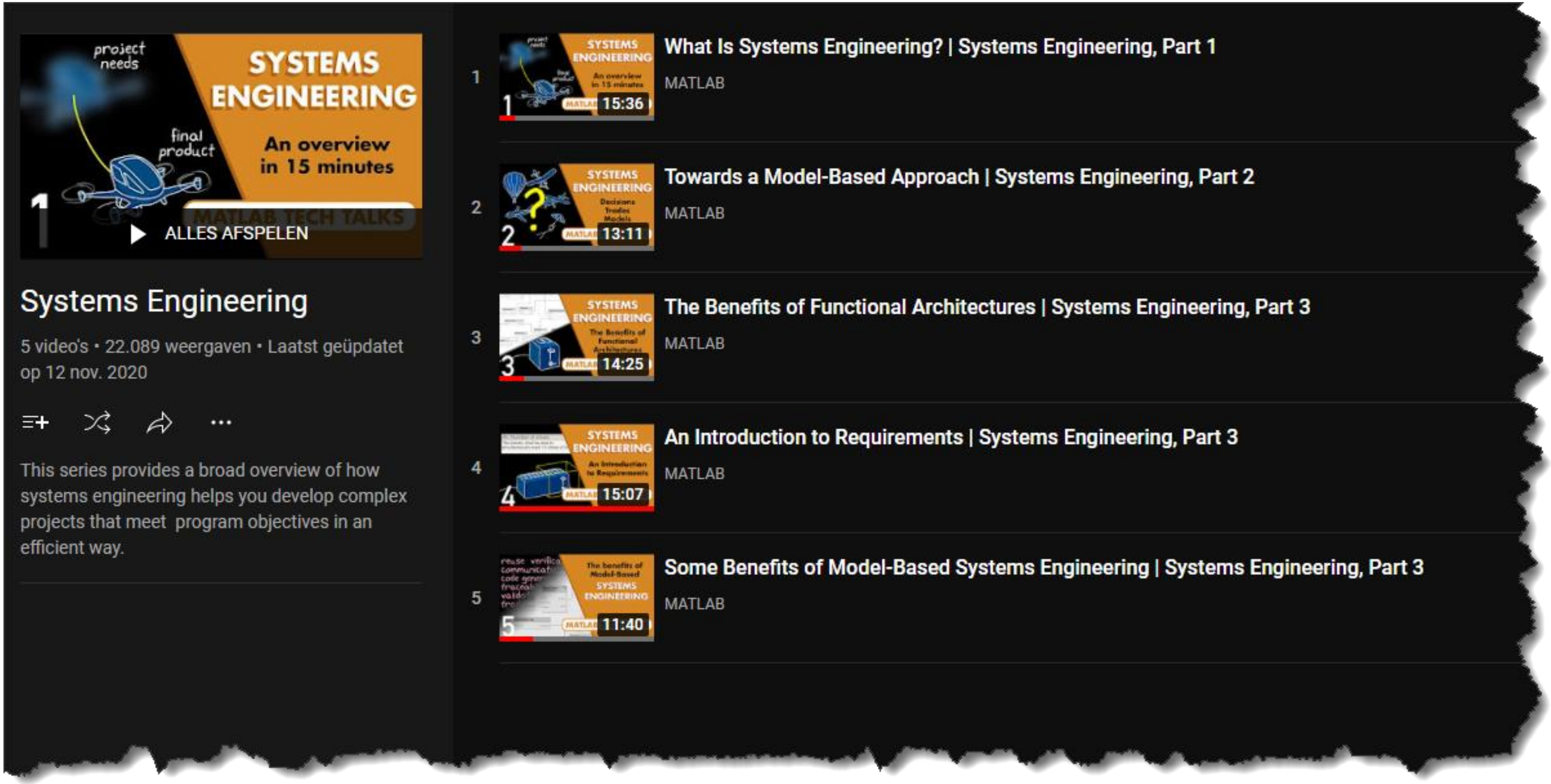
Failure mode detected during simulation.

Automated verification if known failure modes are detected and mitigated

总结

- 通过建立追溯性来维护需求作为权威的设计来源是MBSE的关键之一。
- RFLP的架构框架和视图、模板功能有助于MBSE对复杂度进行管理。
- 基于同平台的架构模型与设计模型无缝对接最大程度上保证了接口的一致性。
- 充分利用分析和仿真为MBSE过程中的创建的模型提供了更高的附加值。

MathWorks的MBSE解决方案



Systems Engineering

5 video's • 22.089 weergaven • Laatst geüpdatet op 12 nov. 2020

This series provides a broad overview of how systems engineering helps you develop complex projects that meet program objectives in an efficient way.

1 What Is Systems Engineering? | Systems Engineering, Part 1
MATLAB
15:36

2 Towards a Model-Based Approach | Systems Engineering, Part 2
MATLAB
13:11

3 The Benefits of Functional Architectures | Systems Engineering, Part 3
MATLAB
14:25

4 An Introduction to Requirements | Systems Engineering, Part 3
MATLAB
15:07

5 Some Benefits of Model-Based Systems Engineering | Systems Engineering, Part 3
MATLAB
11:40

Systems Engineering: Managing System Complexity - MATLAB & Simulink (mathworks.cn)

2024 MathWorks
中国汽车年会

Thank you

MathWorks®