MATLAB EXPO 2021

系统架构划分和需求分配的工作流

吴菁

MathWorks®

投票时间!

- 在系统工程过程中，你面临着哪些挑战?
  a) 需要多种工具
  b) 工具很难学习和掌握使用
  c) 响应变更
  d) 工件之间的追溯
  e) 无法运行和分析
  f) 和设计的同步

我们从你那里听到的

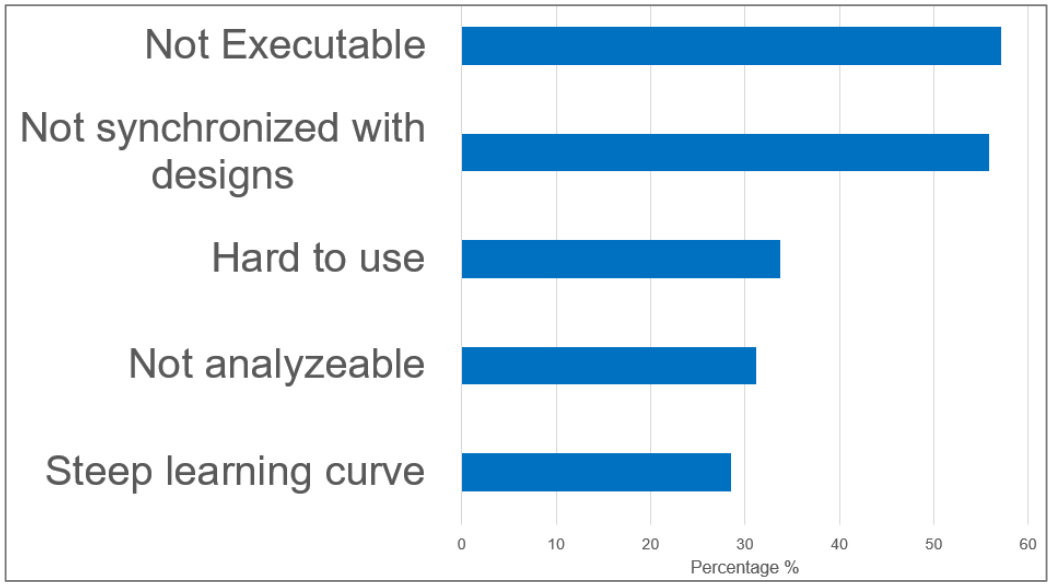
- 基于模型的系统工程是对基于文档的方法的巨大改进

我们从你那里听到的

- 基于模型的系统工程是对基于文档的方法的巨大改进
- 现有工具通常缺少关键功能

我们从你那里听到的

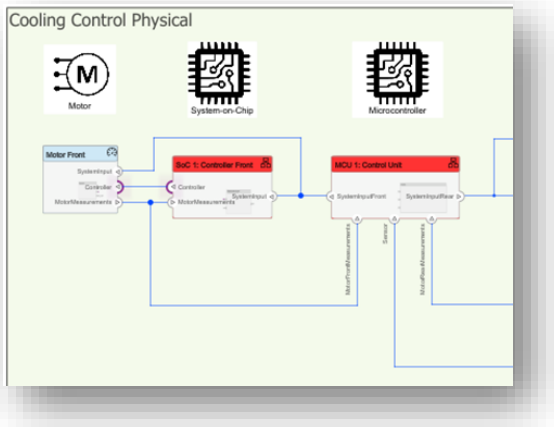
- 基于模型的系统工程是对基于文档的方法的巨大改进
- 现有工具通常缺少关键功能



为什么要使用MathWorks工具进行MBSE?

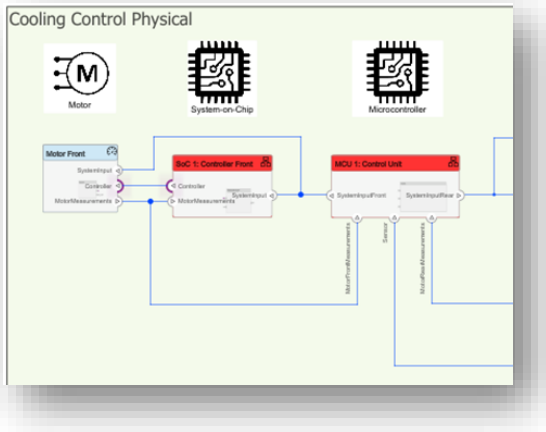
为什么要使用MathWorks工具进行MBSE?

直观

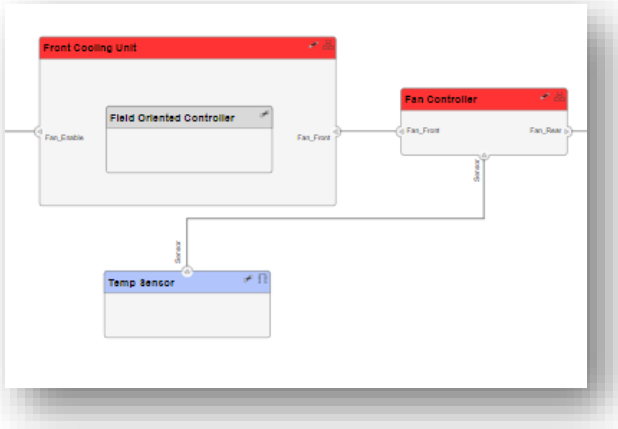


为什么要使用MathWorks工具进行MBSE?

直观

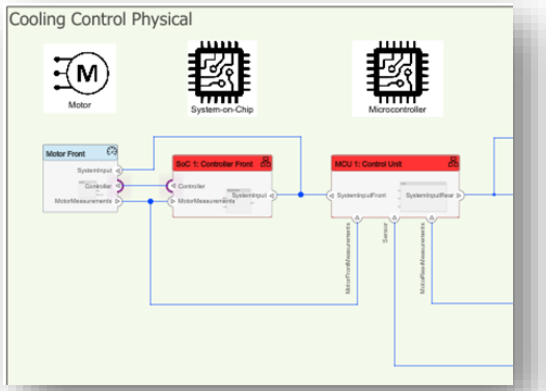


应对复杂性

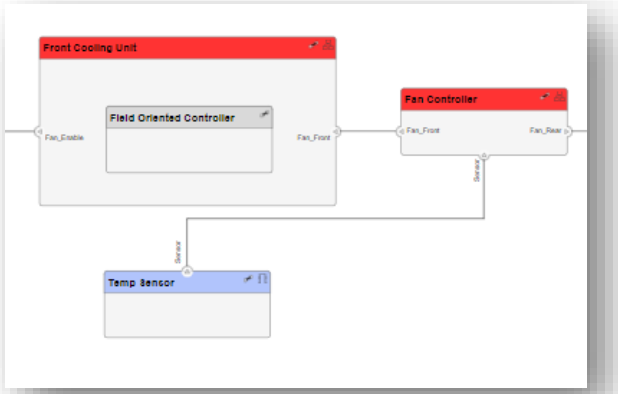


为什么要使用MathWorks工具进行MBSE?

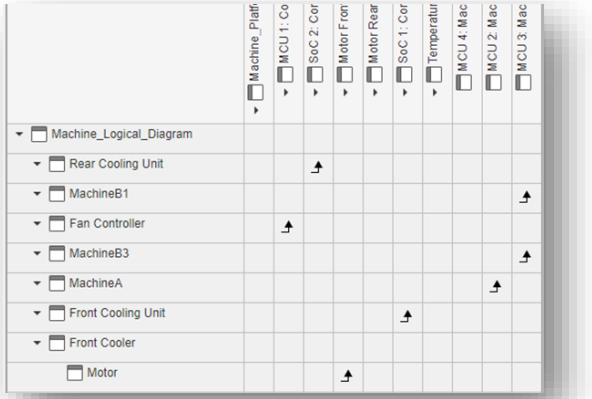
直观



应对复杂性

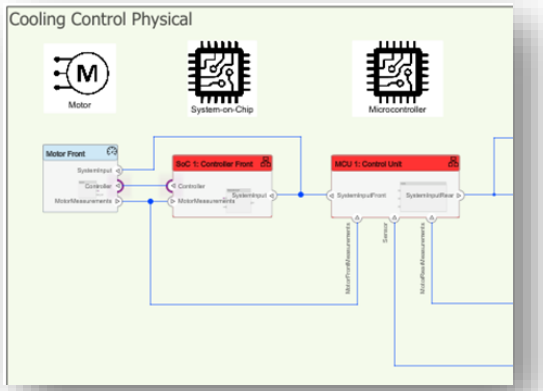


促进可追溯性

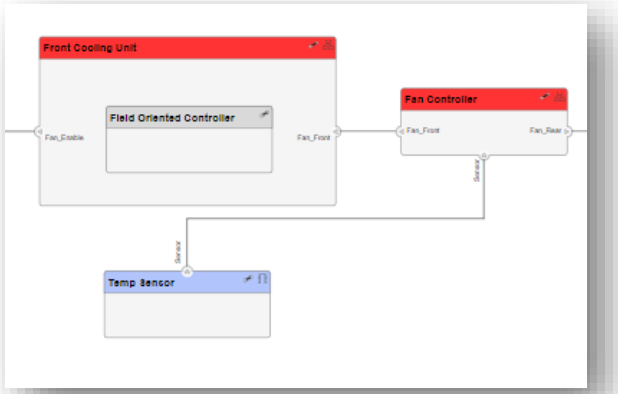


为什么要使用MathWorks工具进行MBSE?

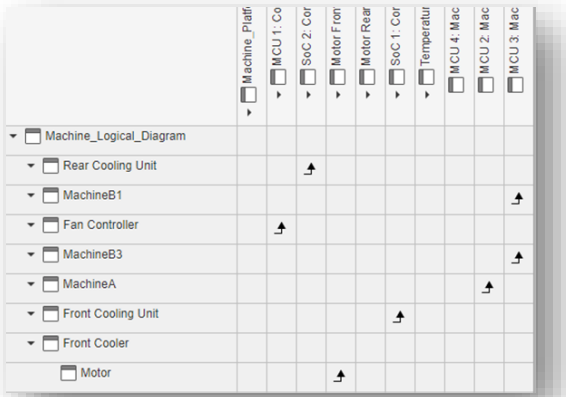
直观



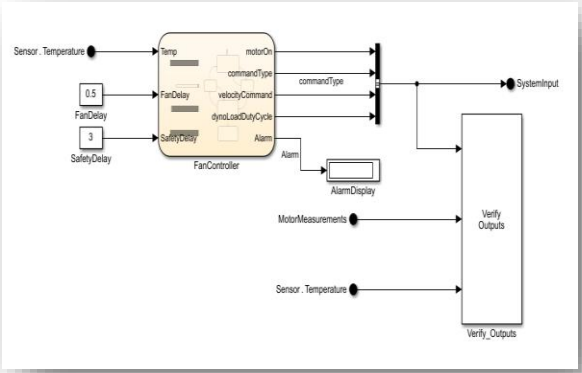
应对复杂性



促进可追溯性

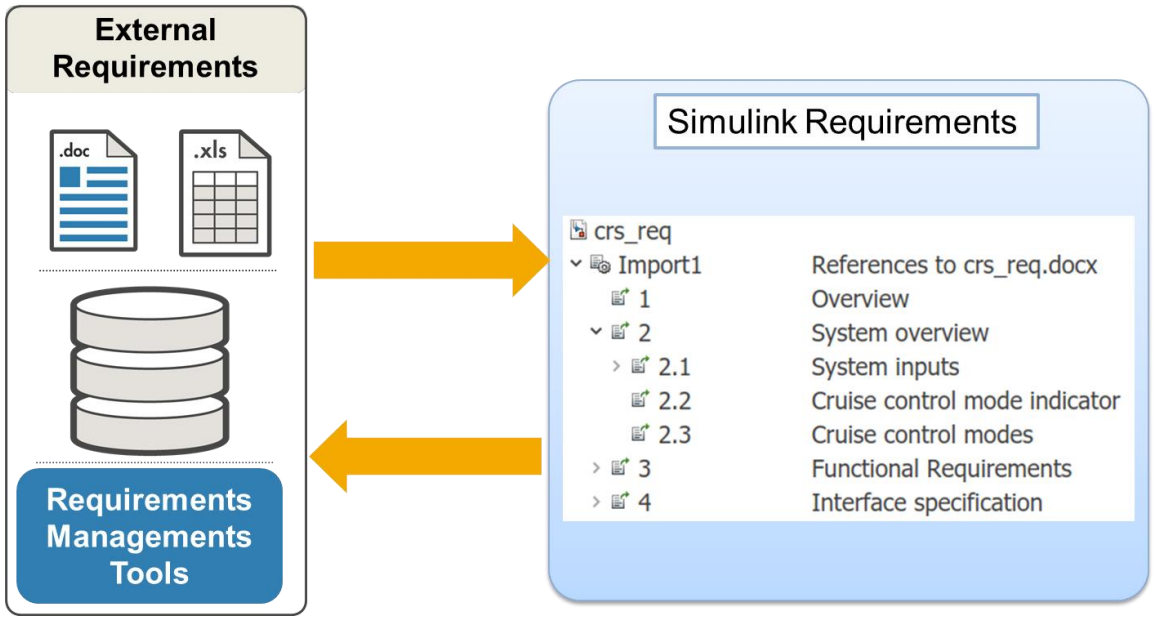


能够实施



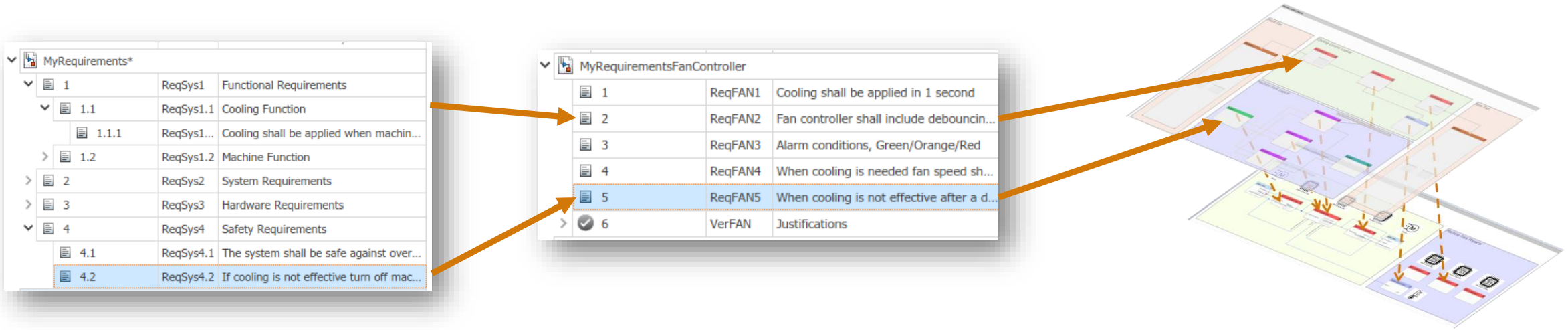
要点

- 可以直接在与架构和设计模型**相同的环境**中导入，编写和存储文本需求.



要点

- 可以直接在与架构和设计模型相同的环境中导入，编写和存储文本需求.
- 通过在多个需求和架构工件之间建立关系，可以了解**系统变更的影响**.



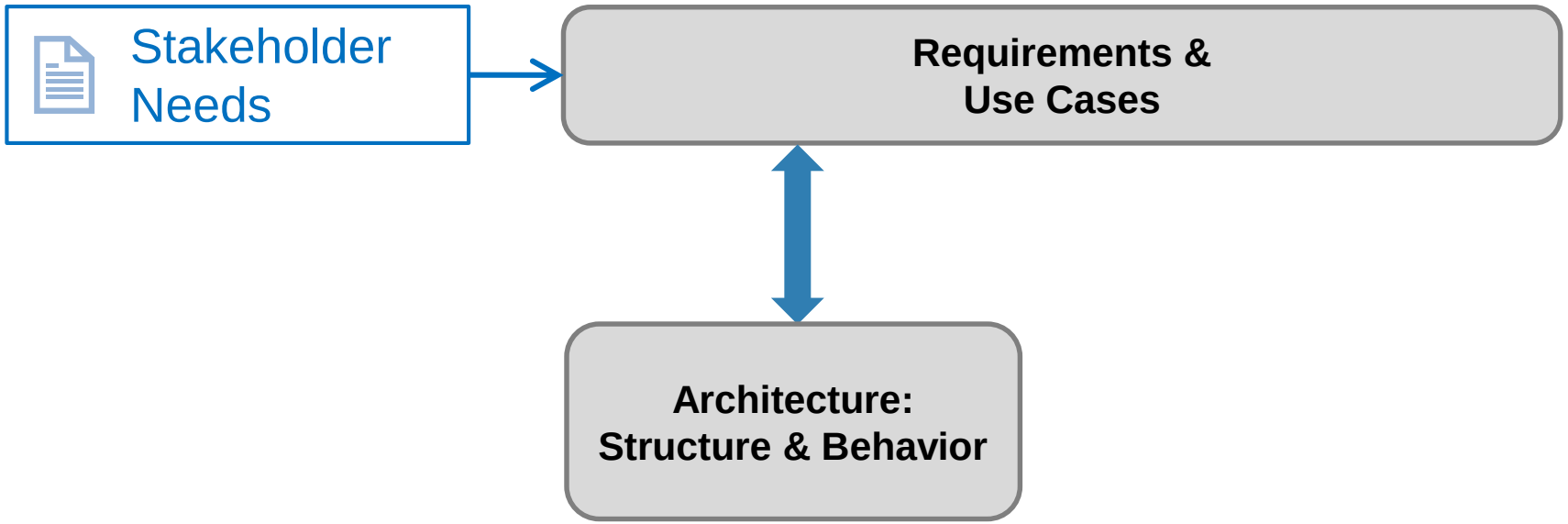
要点

- 可以直接在与架构和设计模型相同的环境中导入，编写和存储文本需求.
- 通过在多个需求和架构工件之间建立关系，您可以了解**系统变更的影响**.
- 可以通过可视化这些关系来**评估系统的完整性**.

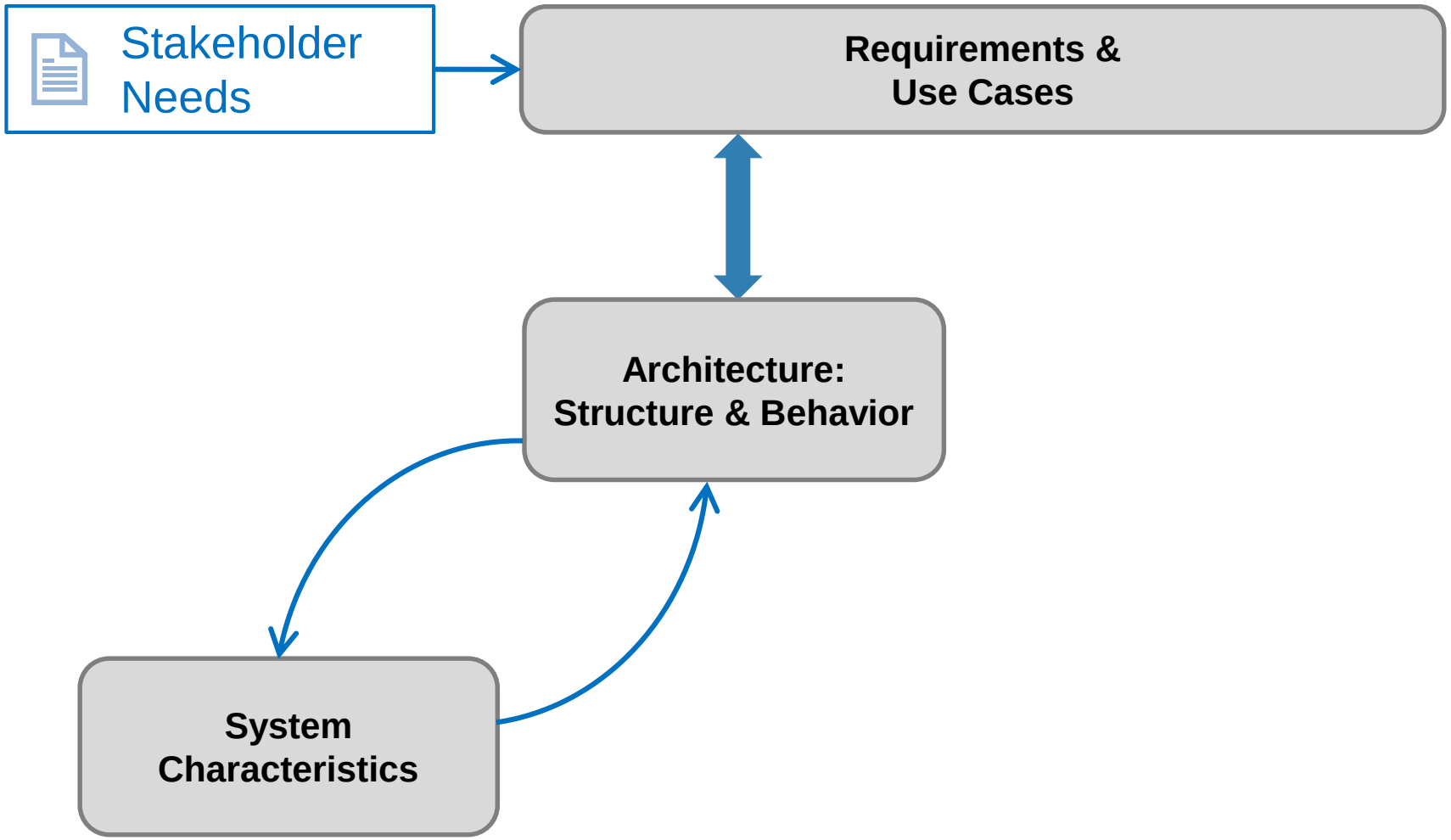
典型的系统工程任务

典型的系统工程任务

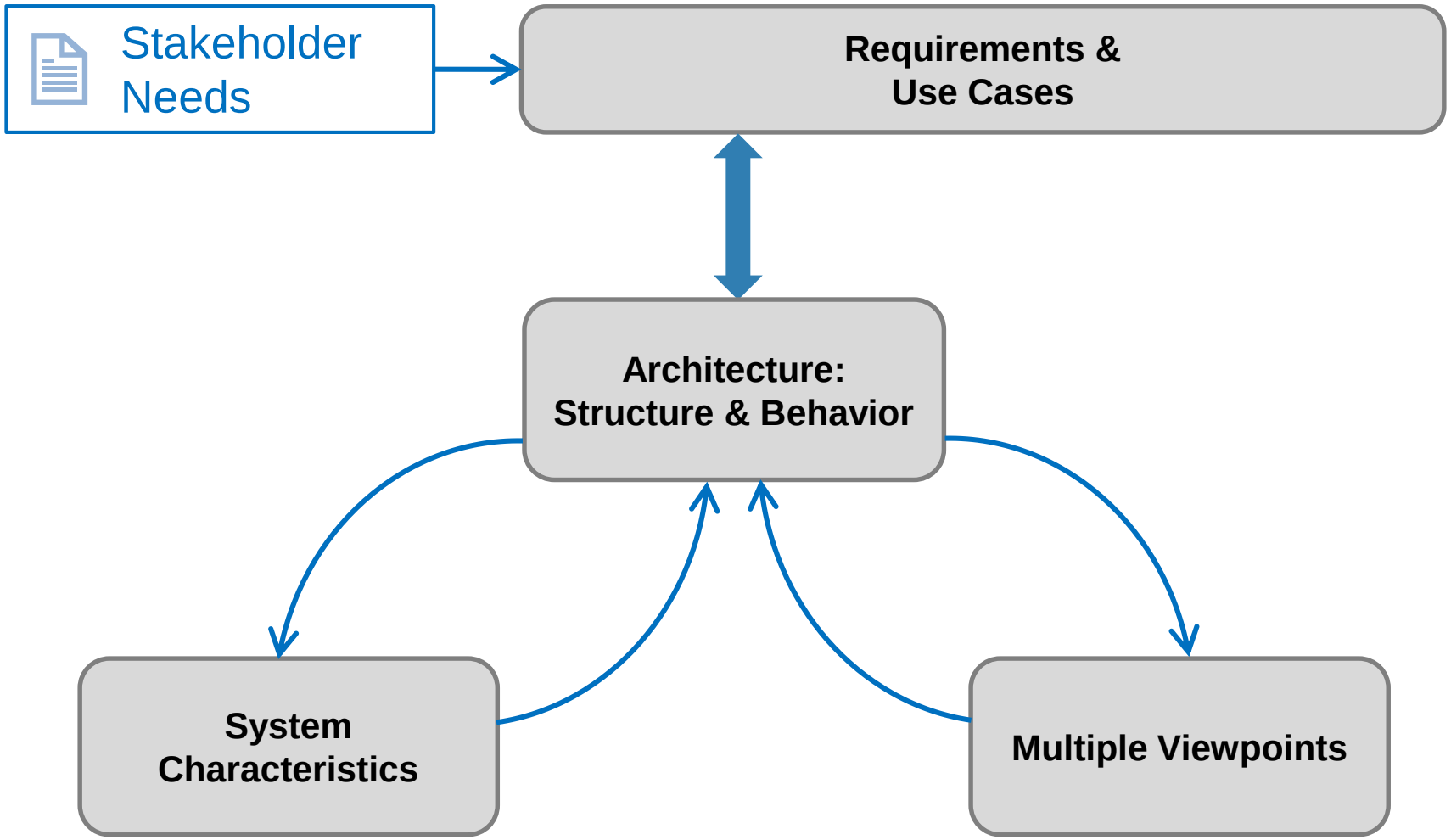
典型的系统工程任务


Stakeholder Needs
Requirements & Use Cases
Architecture: Structure & Behavior

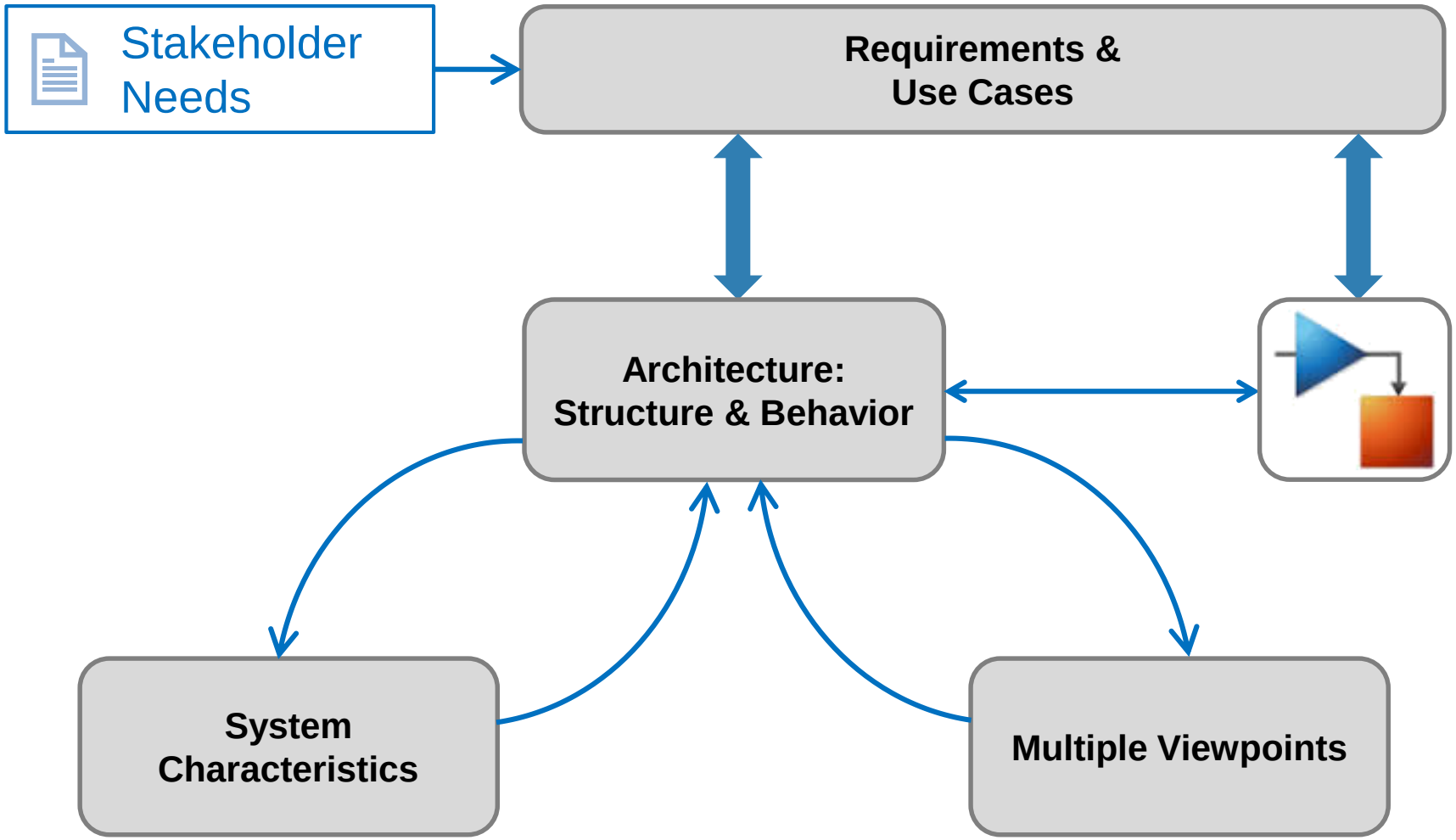
典型的系统工程任务


Stakeholder Needs
Requirements & Use Cases
Architecture: Structure & Behavior
System Characteristics

典型的系统工程任务

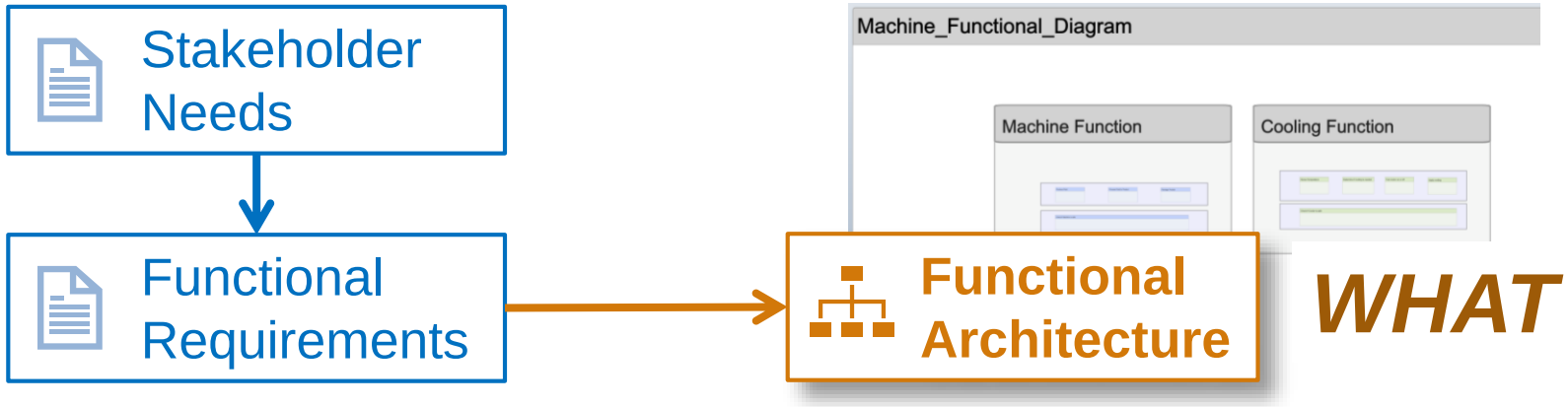

Stakeholder Needs
Requirements & Use Cases
Architecture: Structure & Behavior
System Characteristics
Multiple Viewpoints

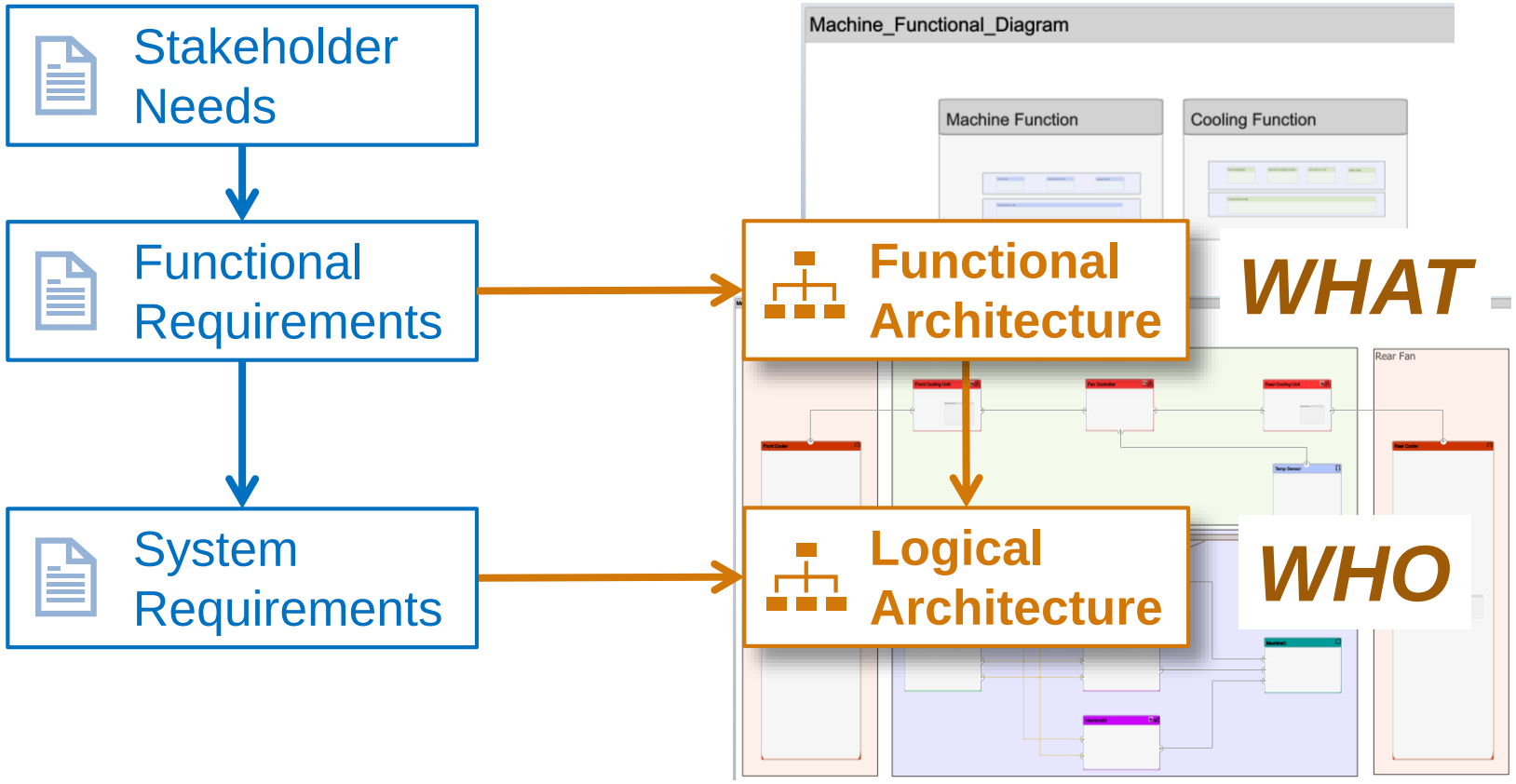
典型的系统工程任务

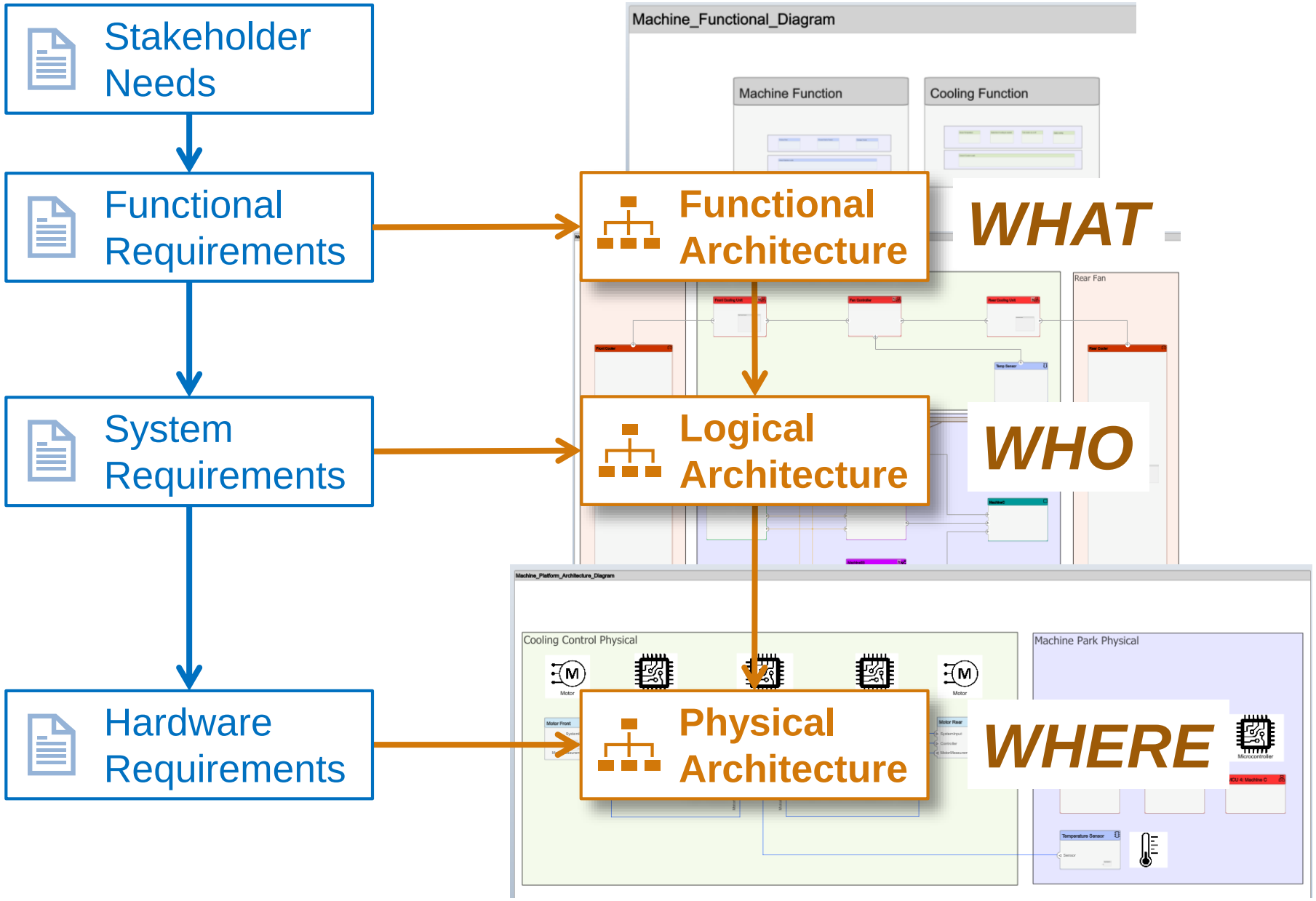

Stakeholder Needs
Requirements & Use Cases
Architecture: Structure & Behavior
System Characteristics
Multiple Viewpoints

**Requirements &
Use Cases**

**Architecture:
Structure & Behavior**


Stakeholder Needs
Functional Requirements
Machine_Functional_Diagram
Machine Function
Cooling Function
Functional Architecture
WHAT


Stakeholder Needs
Functional Requirements
System Requirements
Machine_Functional_Diagram
Machine Function
Cooling Function
Rear Fan
Functional Architecture
Logical Architecture
WHAT
WHO

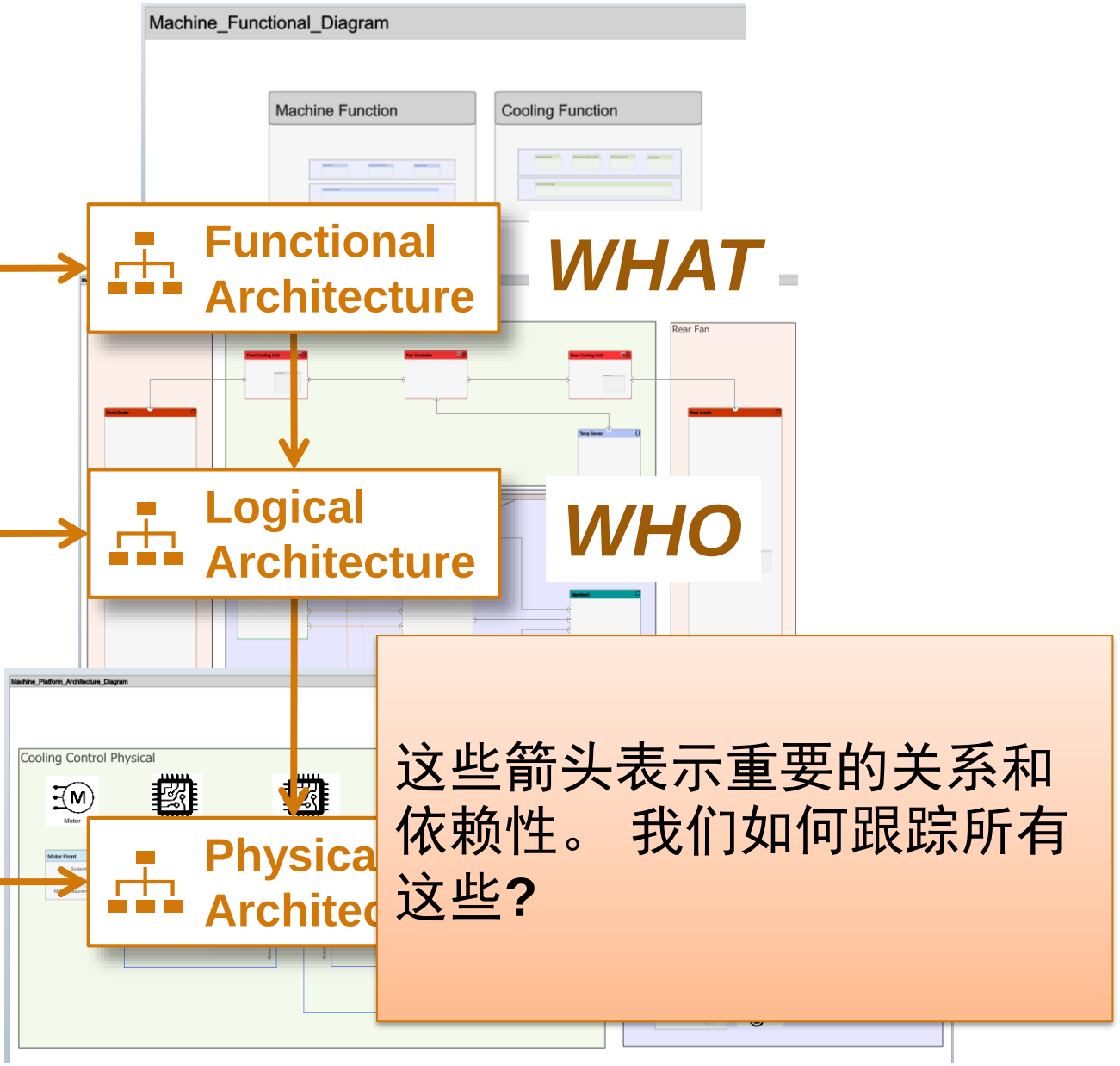

Stakeholder Needs
Functional Requirements
System Requirements
Hardware Requirements
Machine_Functional_Diagram
Machine Function
Cooling Function
Rear Fan
Functional Architecture
WHAT
Logical Architecture
WHO
Machine_Platform_Architecture_Diagram
Cooling Control Physical
Machine Park Physical
Physical Architecture
WHERE

Stakeholder Needs

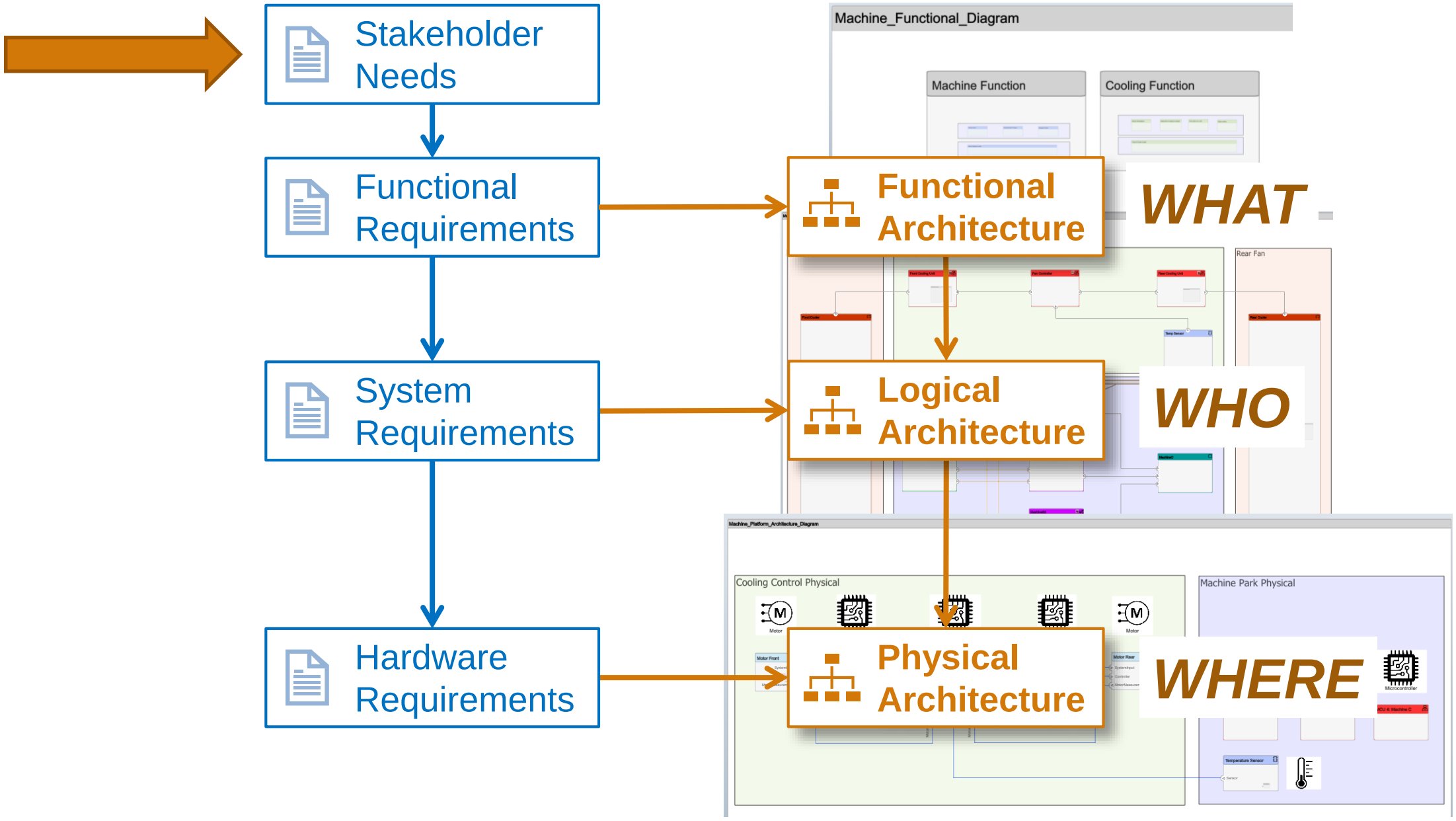
Functional Requirements

System Requirements

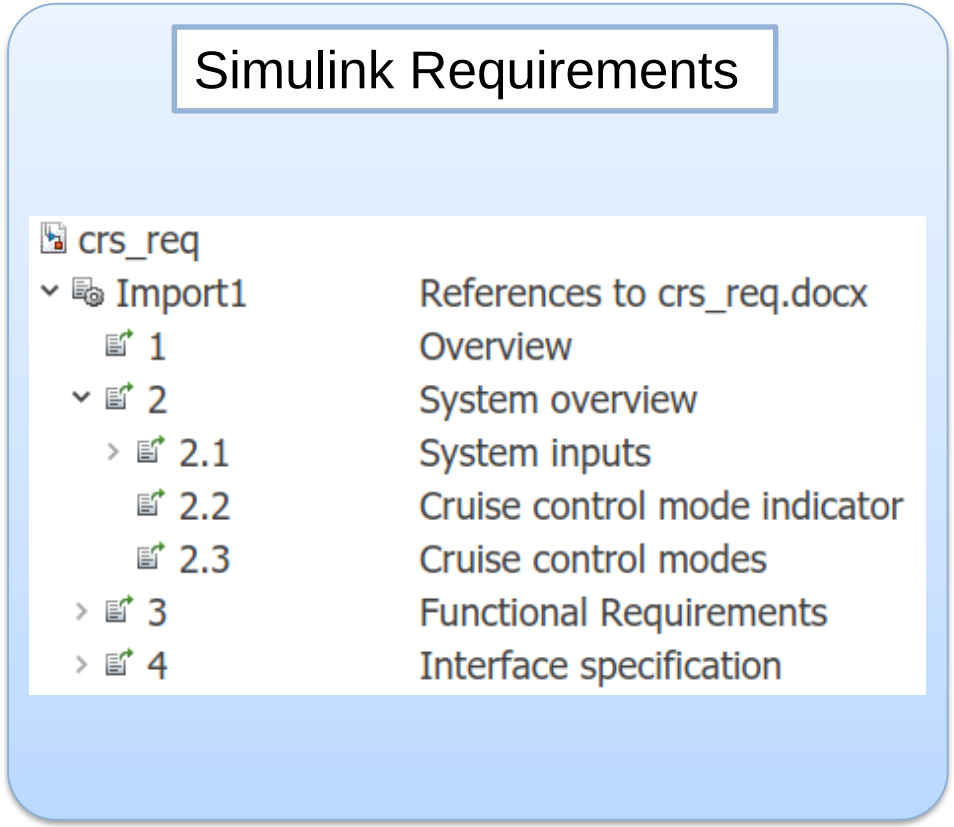
Hardware Requirements



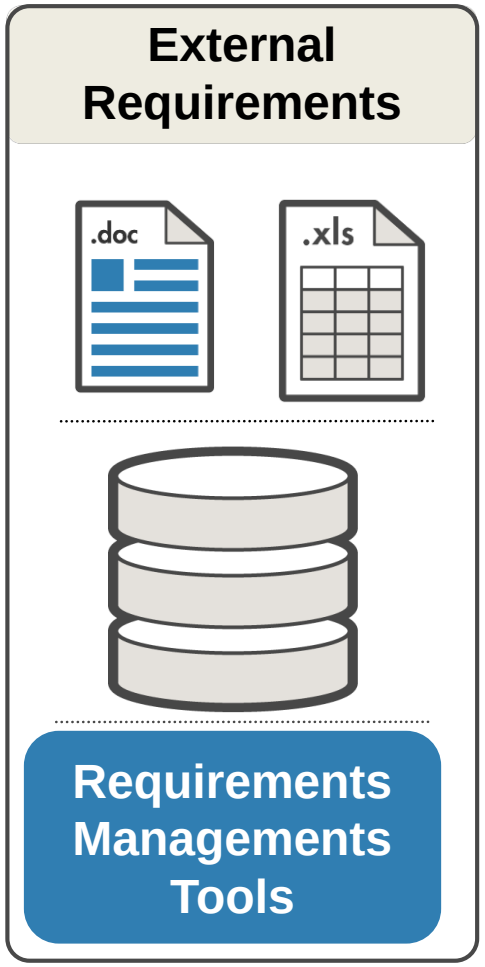
这些箭头表示重要的关系和依赖性。 我们如何跟踪所有这些?


Stakeholder Needs
Functional Requirements
System Requirements
Hardware Requirements
Functional Architecture
Logical Architecture
Physical Architecture
WHAT
WHO
WHERE
Machine_Functional_Diagram
Machine Function
Cooling Function
Rear Fan
Machine_Platform_Architecture_Diagram
Cooling Control Physical
Machine Park Physical

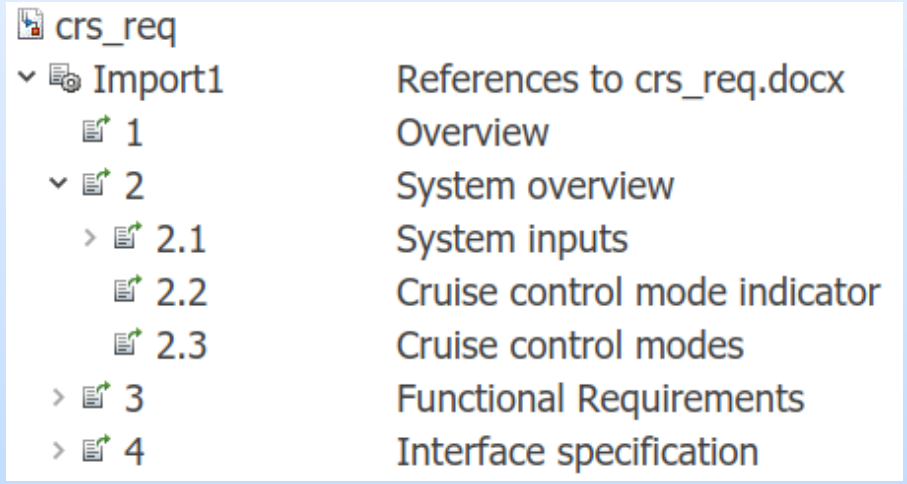
与第三方需求工具交换数据



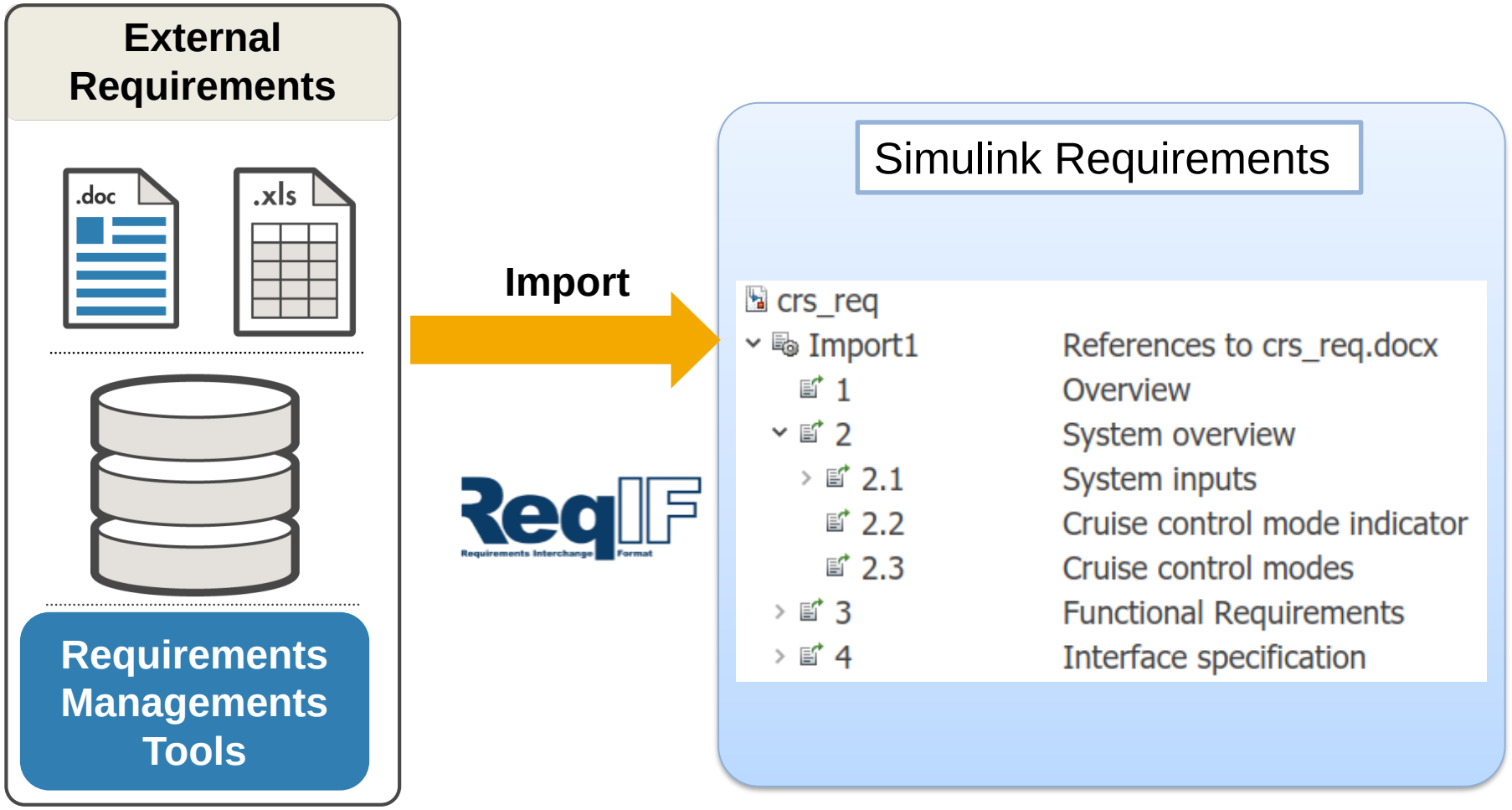
与第三方需求工具交换数据



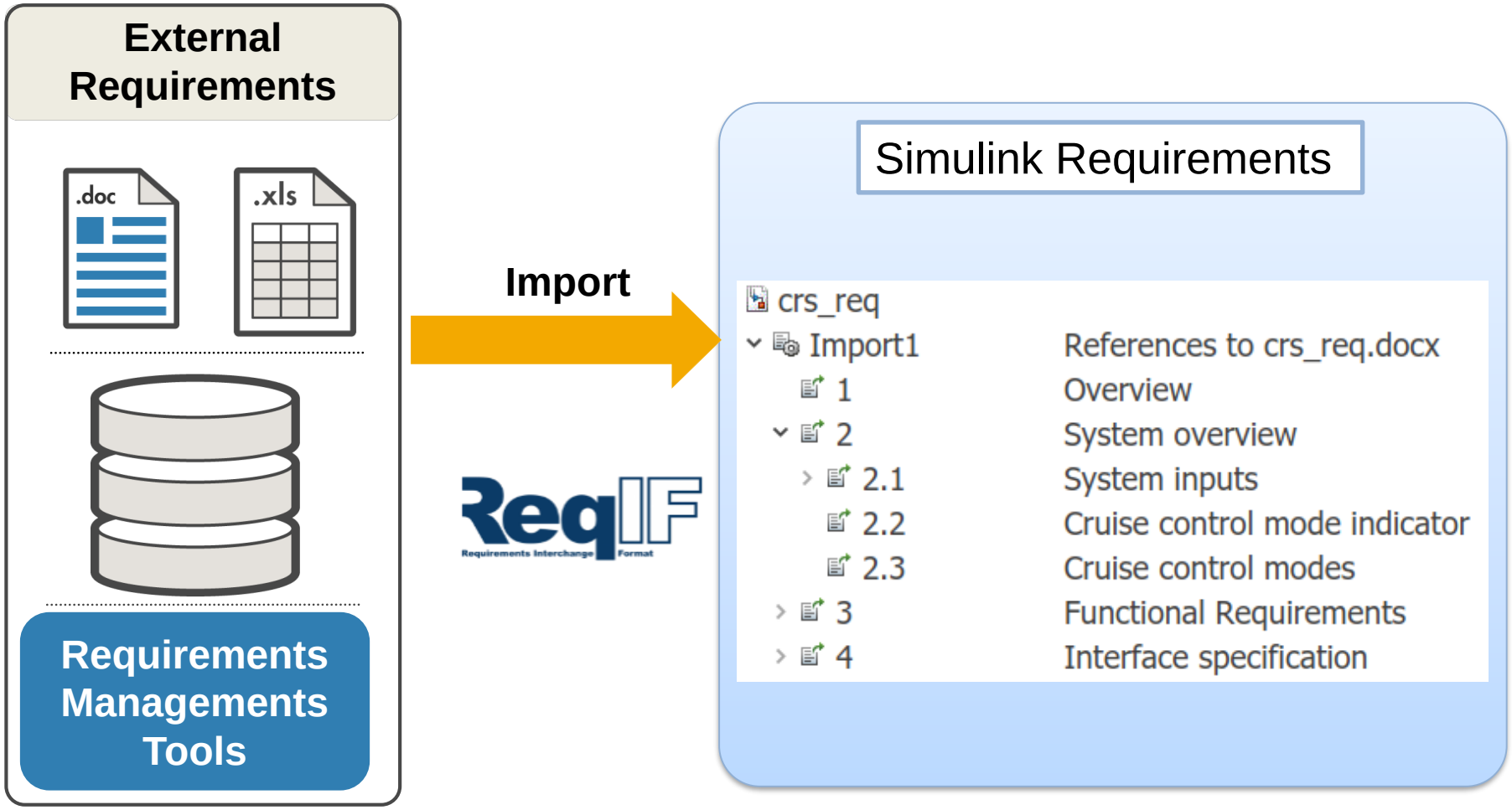
Simulink Requirements



与第三方需求工具交换数据



与第三方需求工具交换数据



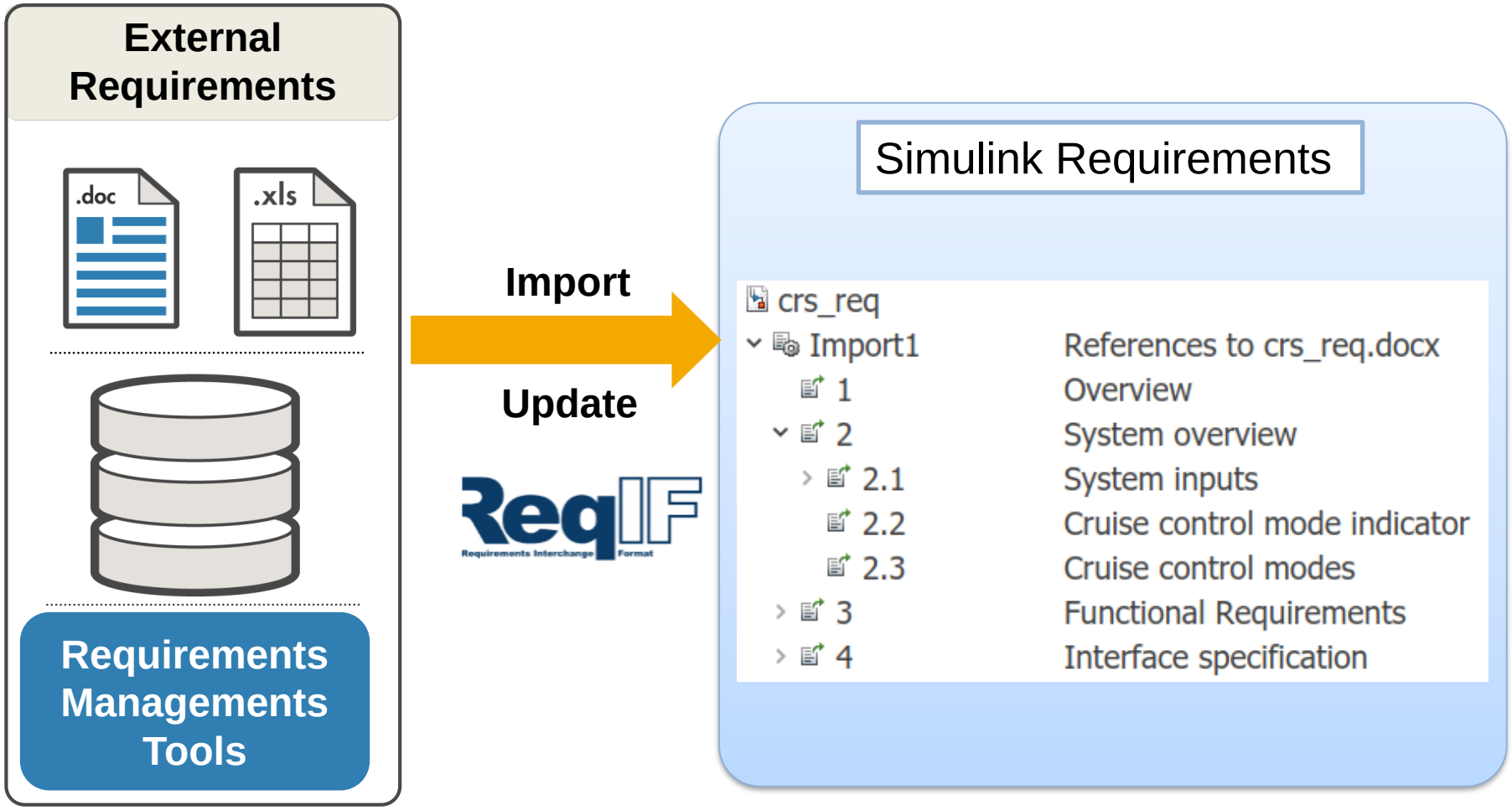
- 从哪里导入:
  - Word / Excel
  - IBM® Rational® DOORS®
  - DOORS Next
  - ReqIF™ standard
- 源变更后同步更改
- 导入后的需求，可以进行编辑并添加更多详细信息

R2019a

- 导出 ReqIF
  - 使用外部工具实现闭环

R2019a

与第三方需求工具交换数据



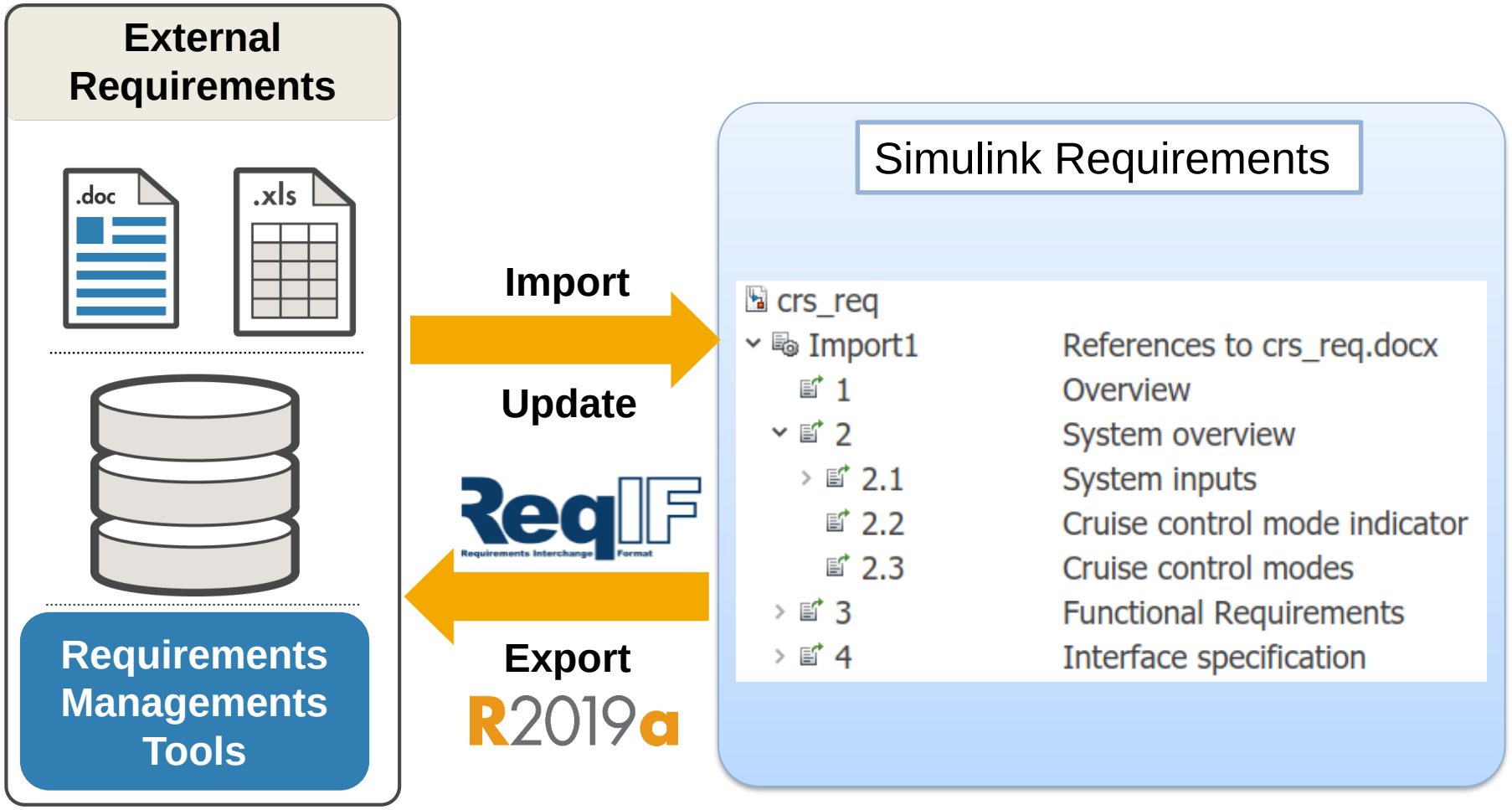
- 从哪里导入:
  - Word / Excel
  - IBM® Rational® DOORS®
  - DOORS Next
  - ReqIF™ standard
- 源变更后同步更改
- 导入后的需求，可以进行编辑并添加更多详细信息

R2019a

- 导出 ReqIF
  - 使用外部工具实现闭环

R2019a

与第三方需求工具交换数据



- 从哪里导入:
  - Word / Excel
  - IBM® Rational® DOORS®
  - DOORS Next
  - ReqIF™ standard
- 源变更后同步更改
- 导入后的需求，可以进行编辑并添加更多详细信息

R2019a

- 导出 ReqIF
  - 使用外部工具实现闭环

R2019a

Stakeholder Needs

Functional Requirements

System Requirements

Hardware Requirements

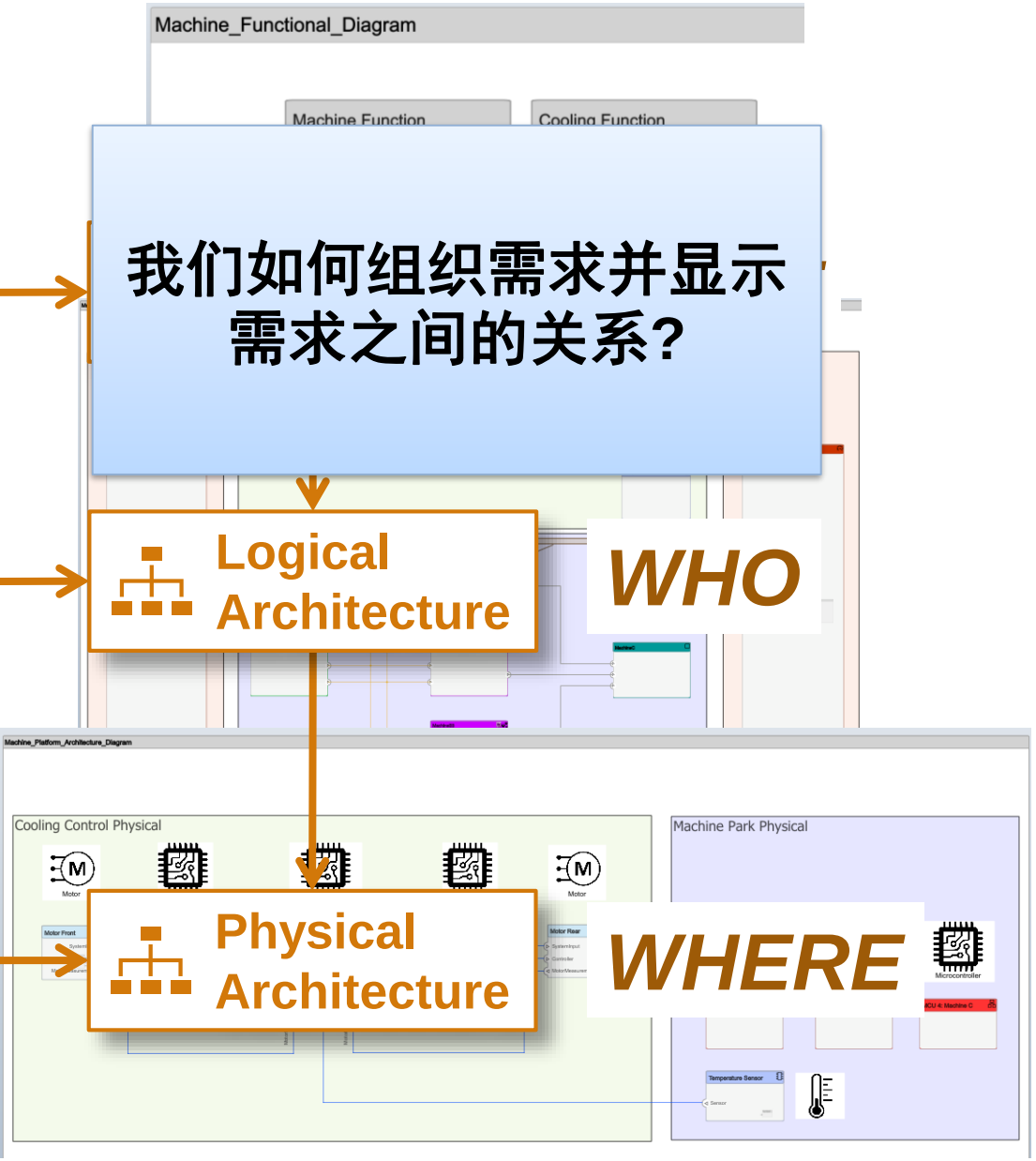
我们如何组织需求并显示需求之间的关系?

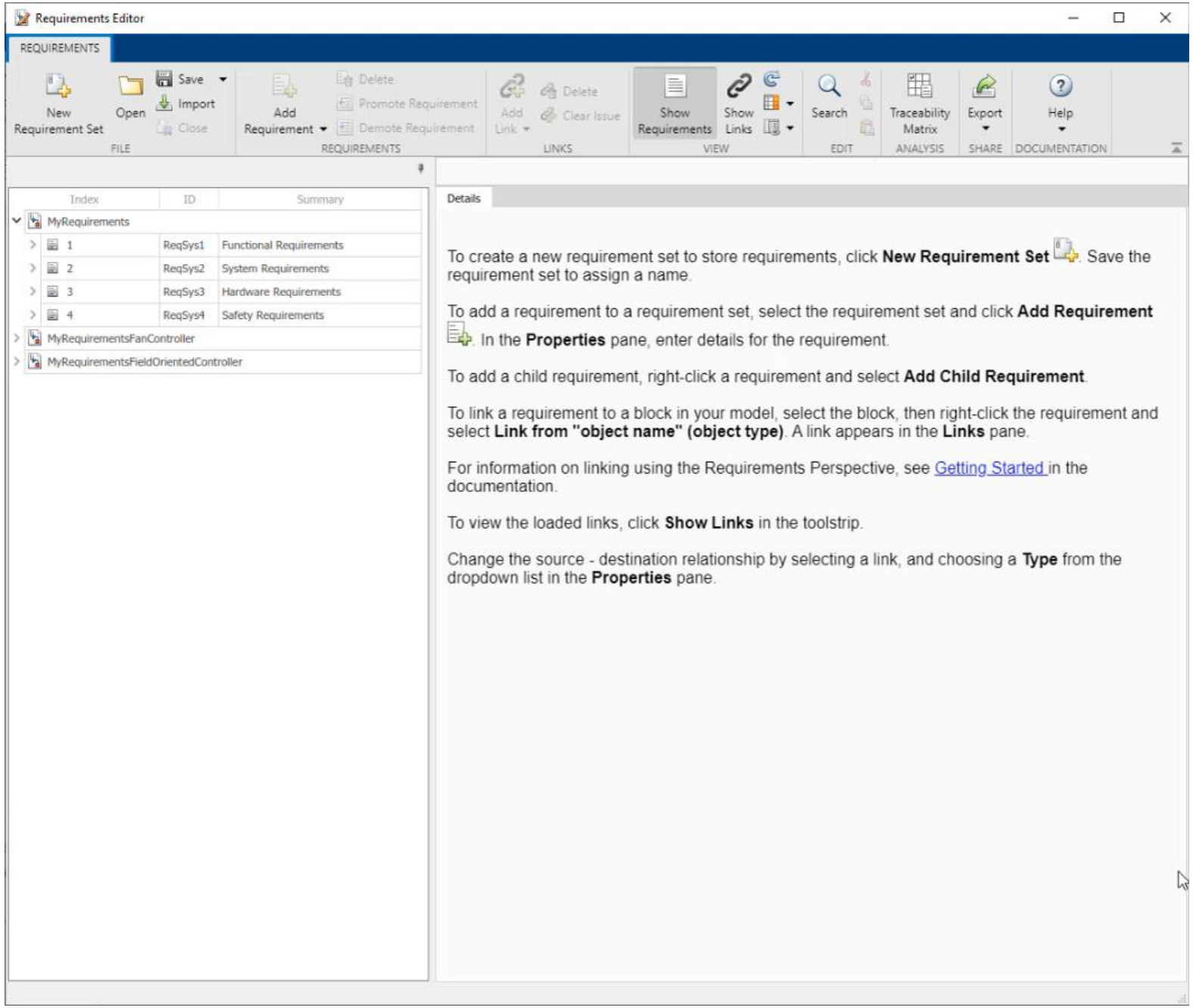
Logical Architecture

WHO

Physical Architecture

WHERE



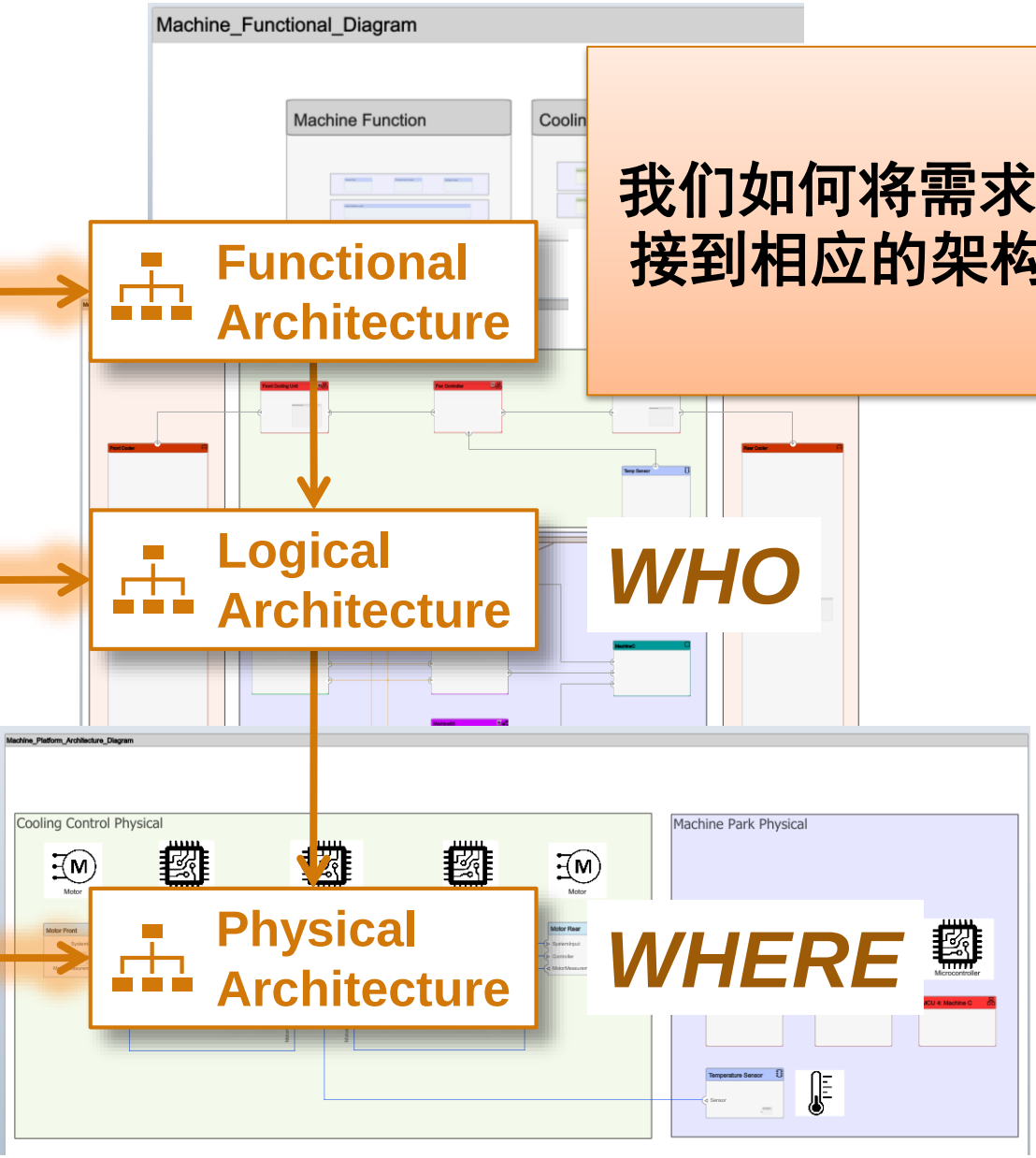

Requirements Editor
REQUIREMENTS
New Requirement Set
Open
Save
Import
Close
FILE
Add Requirement
Delete
Promote Requirement
Demote Requirement
REQUIREMENTS
Add Link
Delete
Clear Issue
LINKS
Show Requirements
Show Links
VIEW
Search
EDIT
Traceability Matrix
ANALYSIS
Export
SHARE
Help
DOCUMENTATION
Index
ID
Summary
MyRequirements
1 ReqSys1 Functional Requirements
2 ReqSys2 System Requirements
3 ReqSys3 Hardware Requirements
4 ReqSys4 Safety Requirements
MyRequirementsFanController
MyRequirementsFieldOrientedController
Details
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 the loaded links, click Show Links in the toolstrip.
Change the source - destination relationship by selecting a link, and choosing a Type from the dropdown list in the Properties pane.

Stakeholder Needs

Functional Requirements

System Requirements

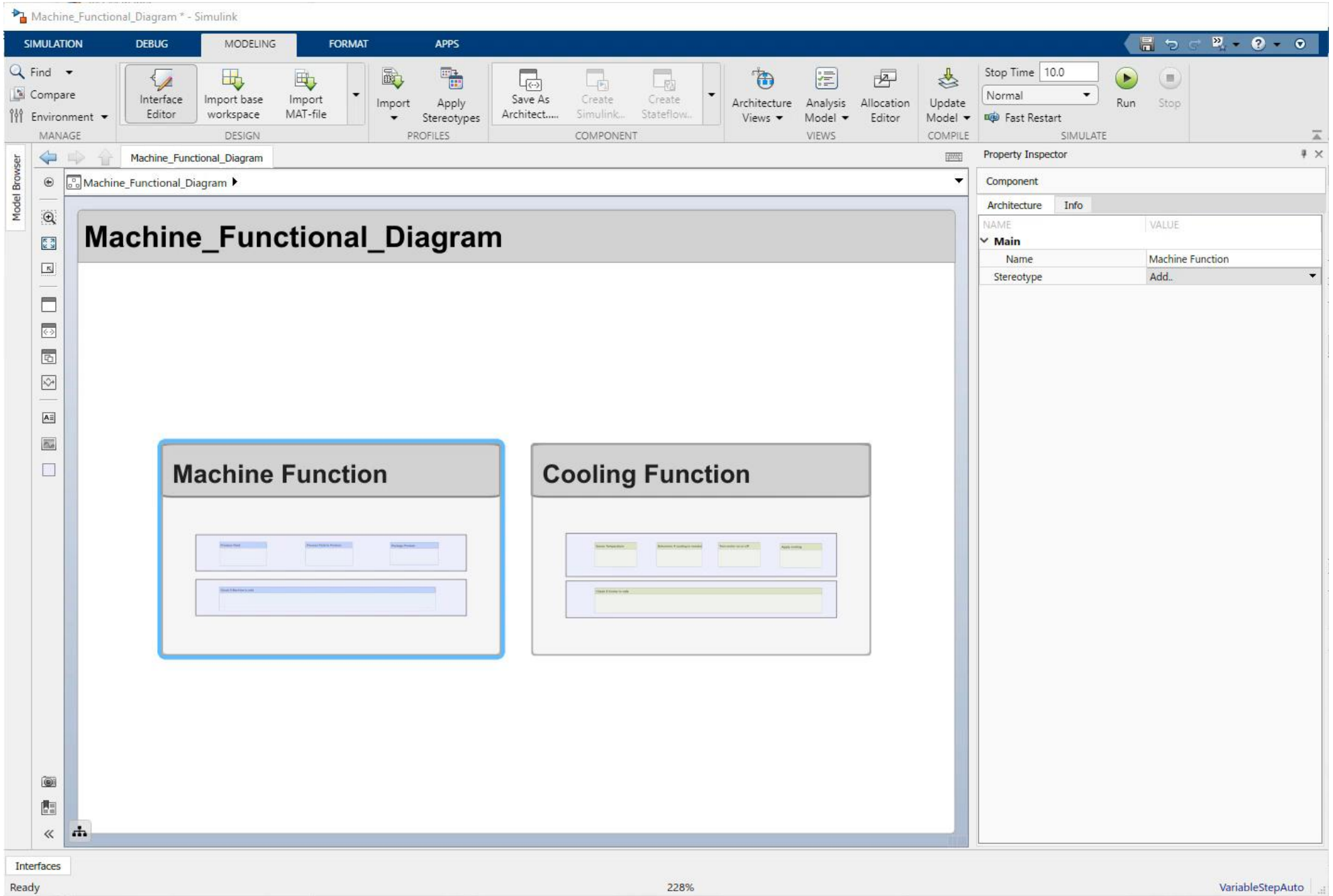
Hardware Requirements

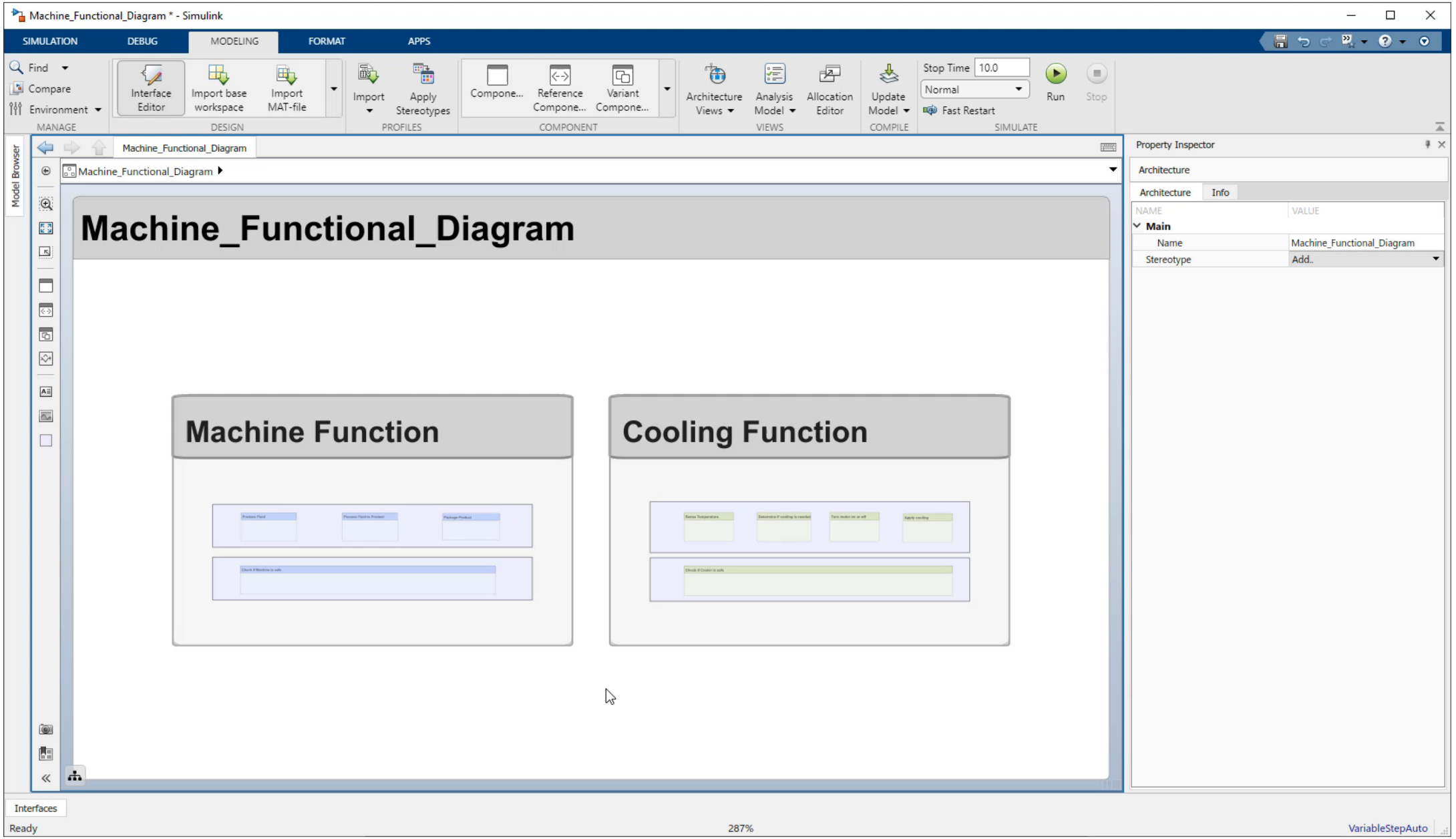


我们如何将需求链接到相应的架构?

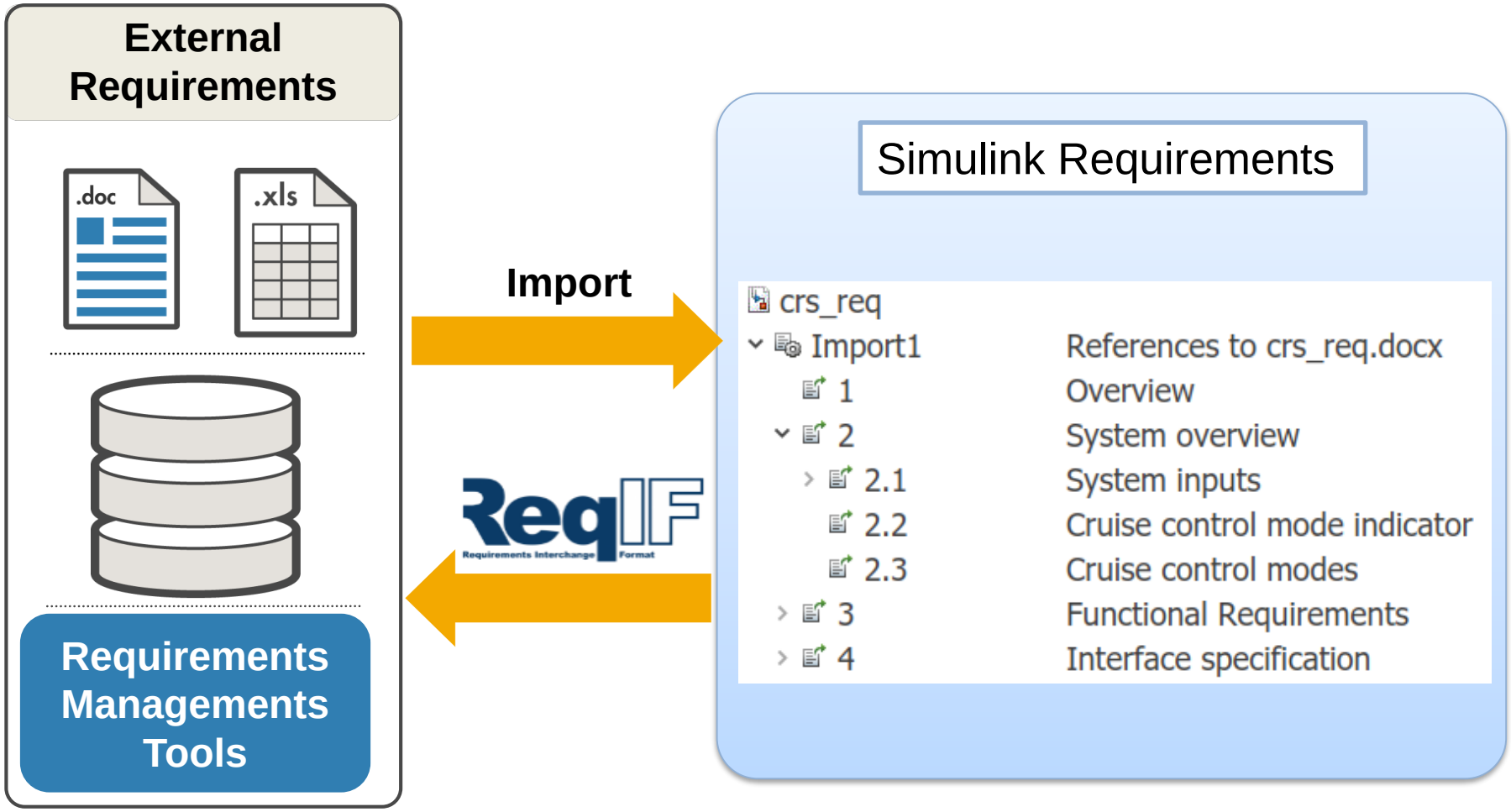
WHO

WHERE

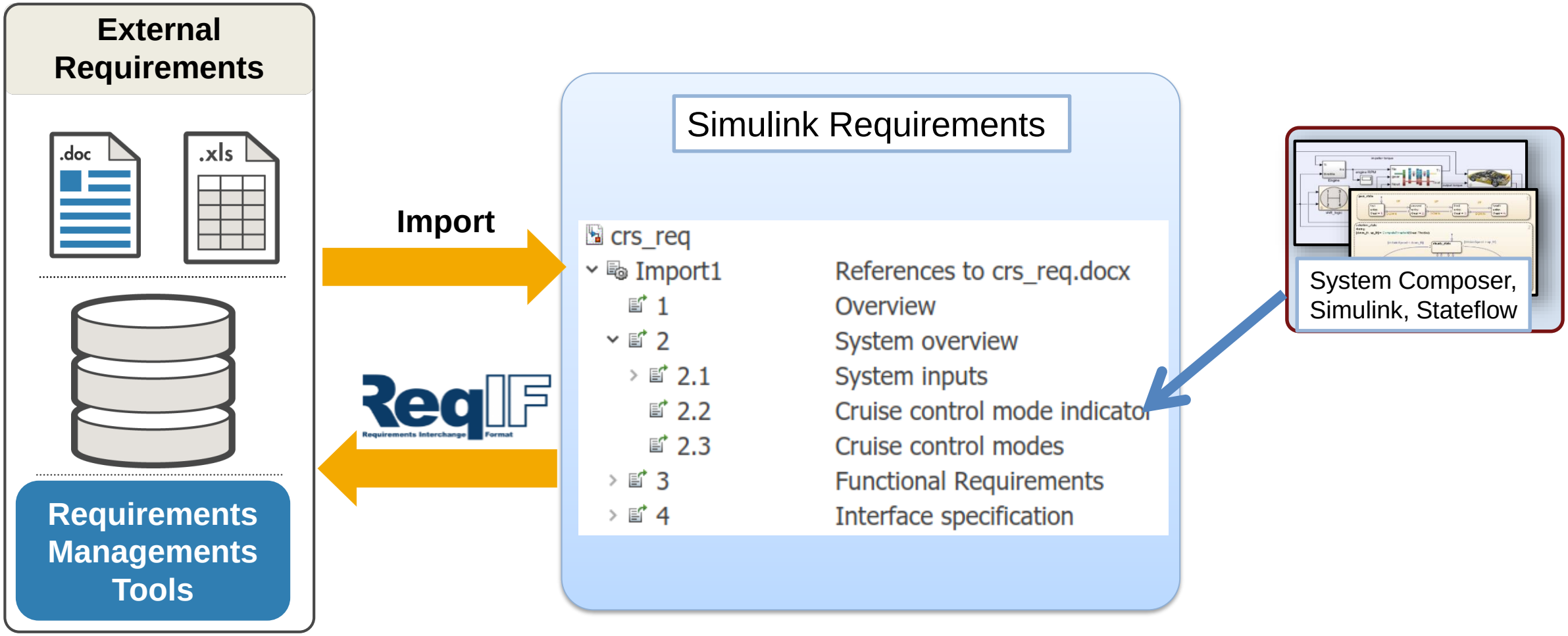

Machine_Functional_Diagram * - Simulink
SIMULATION
DEBUG
MODELING
FORMAT
APPS
Find
Compare
Environment
MANAGE
Interface Editor
Import base workspace
Import MAT-file
DESIGN
Import
Apply Stereotypes
PROFILES
Save As Architect.....
Create Simulink...
Create Stateflow...
COMPONENT
Architecture Views
Analysis Model
Allocation Editor
VIEWS
Update Model
COMPILE
Stop Time
10.0
Normal
Fast Restart
Run
Stop
SIMULATE
Model Browser
Machine_Functional_Diagram
Machine_Functional_Diagram
Machine_Functional_Diagram
Machine Function
Cooling Function
Property Inspector
Component
Architecture
Info
NAME
VALUE
Main
Name
Machine Function
Stereotype
Add..
Interfaces
Ready
228%
VariableStepAuto


Machine_Functional_Diagram * - Simulink
SIMULATION
DEBUG
MODELING
FORMAT
APPS
Find
Compare
Environment
MANAGE
Interface Editor
Import base workspace
Import MAT-file
DESIGN
Import
Apply Stereotypes
PROFILES
Compone...
Reference Compone...
Variant Compone...
COMPONENT
Architecture Views
Analysis Model
Allocation Editor
VIEWS
Update Model
COMPILE
Stop Time
10.0
Normal
Fast Restart
Run
Stop
SIMULATE
Model Browser
Machine_Functional_Diagram
Machine_Functional_Diagram
Machine_Functional_Diagram
Machine Function
Cooling Function
Property Inspector
Architecture
Architecture
Info
NAME
VALUE
Main
Name
Machine_Functional_Diagram
Stereotype
Add..
Interfaces
Ready
287%
VariableStepAuto

导出链接以追溯模型和测试



导出链接以追溯模型和测试



导出链接以追溯模型和测试

External Requirements

.doc .xls

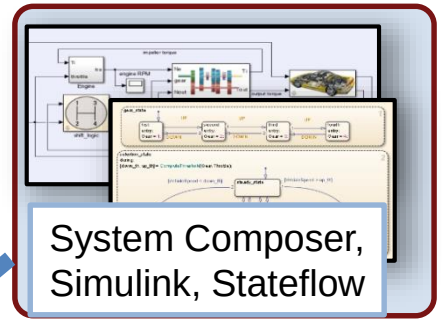
Requirements Managements Tools

Import

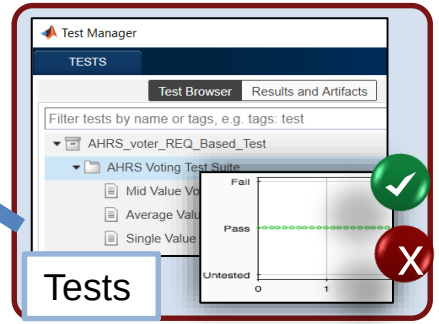
ReqIF Requirements Interchange Format

Simulink Requirements

crs_req
- Import1 — References to crs_req.docx
  - 1 — Overview
  - 2 — System overview
    - 2.1 — System inputs
    - 2.2 — Cruise control mode indicator
    - 2.3 — Cruise control modes
  - 3 — Functional Requirements
  - 4 — Interface specification

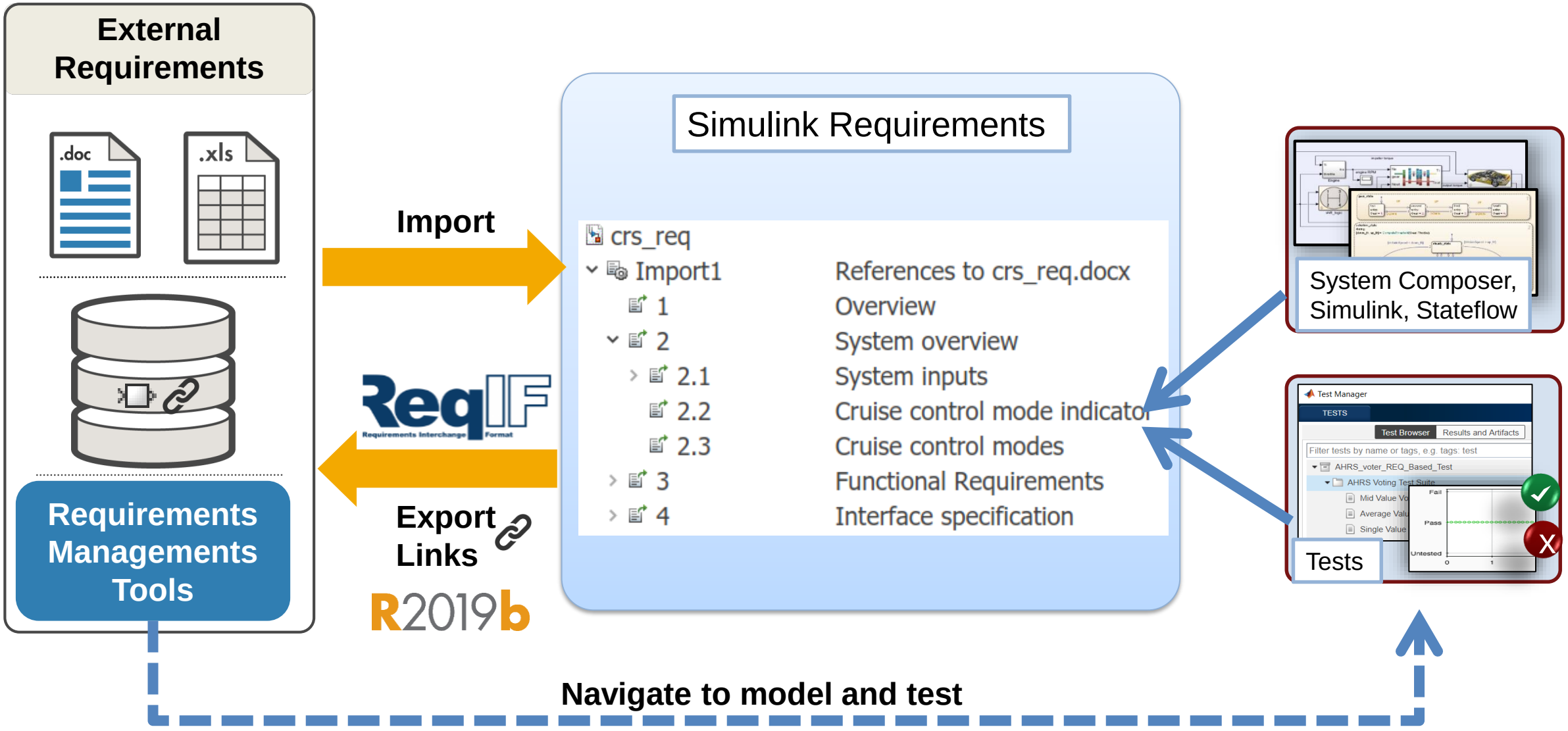


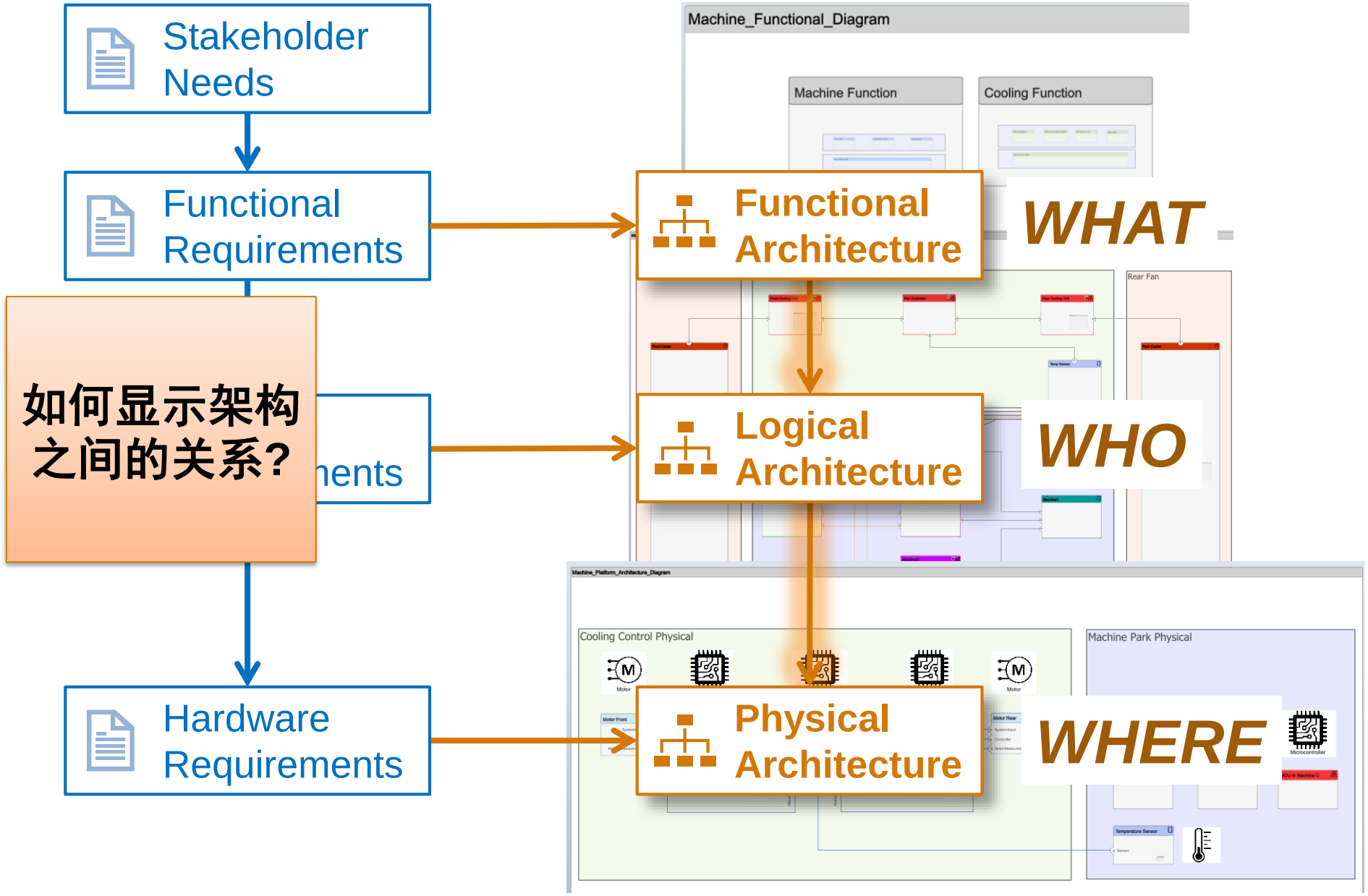
System Composer, Simulink, Stateflow



Tests

导出链接以追溯模型和测试

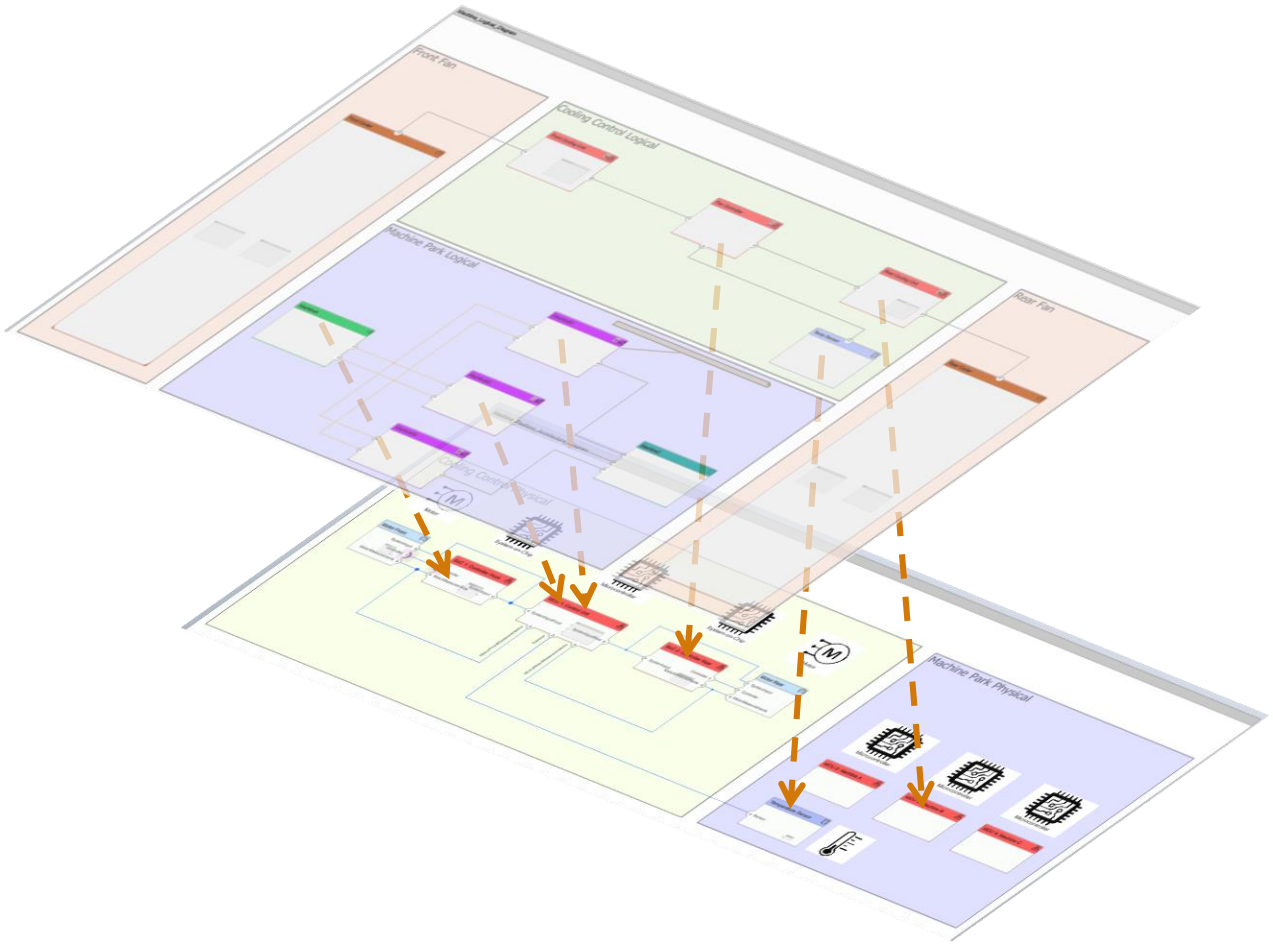



Stakeholder Needs
Functional Requirements
如何显示架构之间的关系?
ments
Hardware Requirements
Machine_Functional_Diagram
Machine Function
Cooling Function
Functional Architecture
WHAT
Rear Fan
Logical Architecture
WHO
Machine_Platform_Architecture_Diagram
Cooling Control Physical
Machine Park Physical
Physical Architecture
WHERE

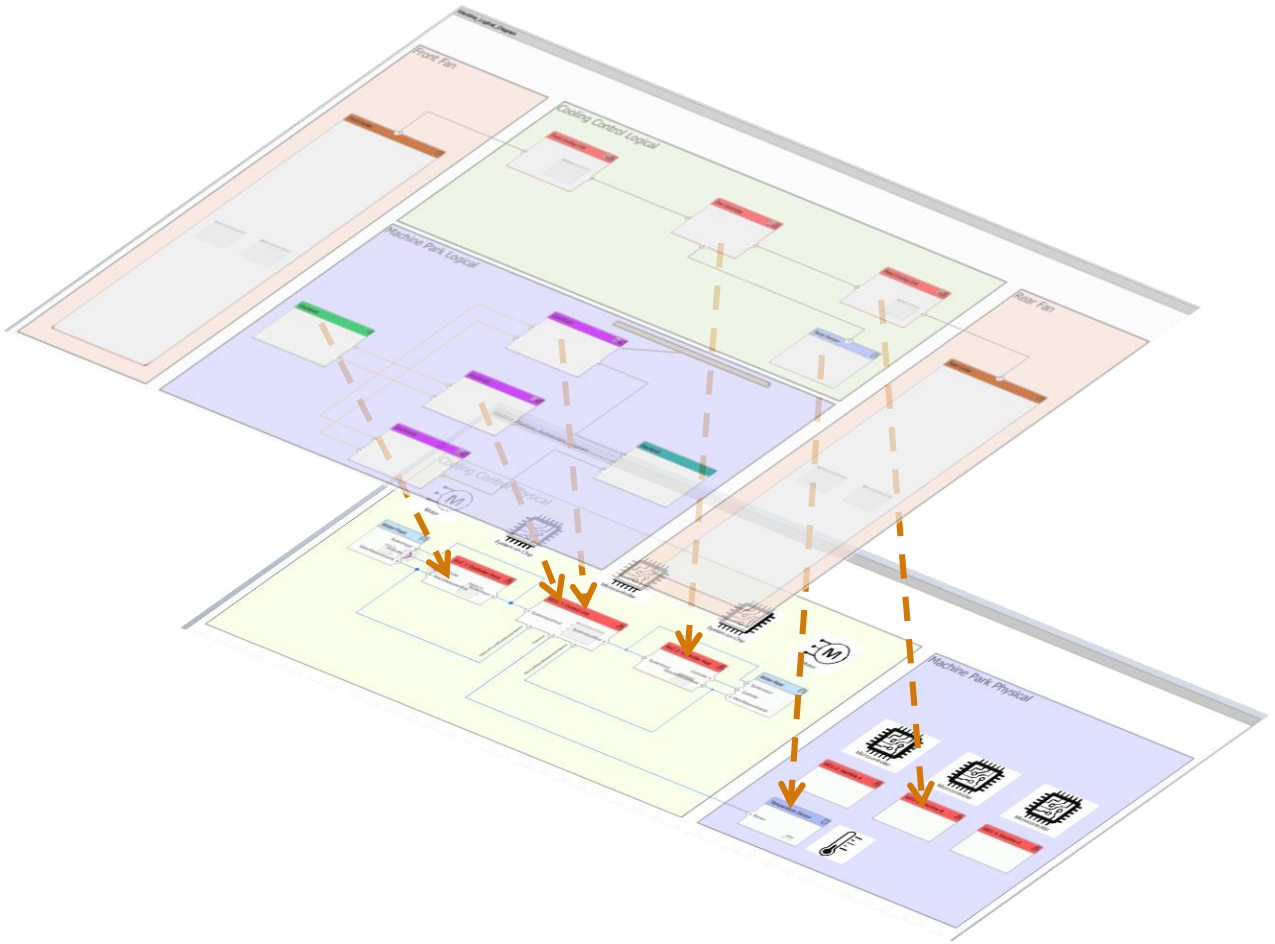
架构之间的分配


Front Fan
Cooling Control Logical
Machine Park Logical
Rear Fan
Cooling Control Physical
Machine Park Physical

架构之间的分配


Front Fan
Cooling Control Logical
Machine Park Logical
Rear Fan
Machine Park Physical

架构之间的分配



Allocation Editor

ALLOCATIONS

New Allocation Set | Open Allocation Set | Save Allocation Set — FILE
Add Scenario | Delete Scenario — SCENARIO
Synchronize — REFRESH
Port | Connector | Component | Allocated | Un-Allocated — ROW FILTER

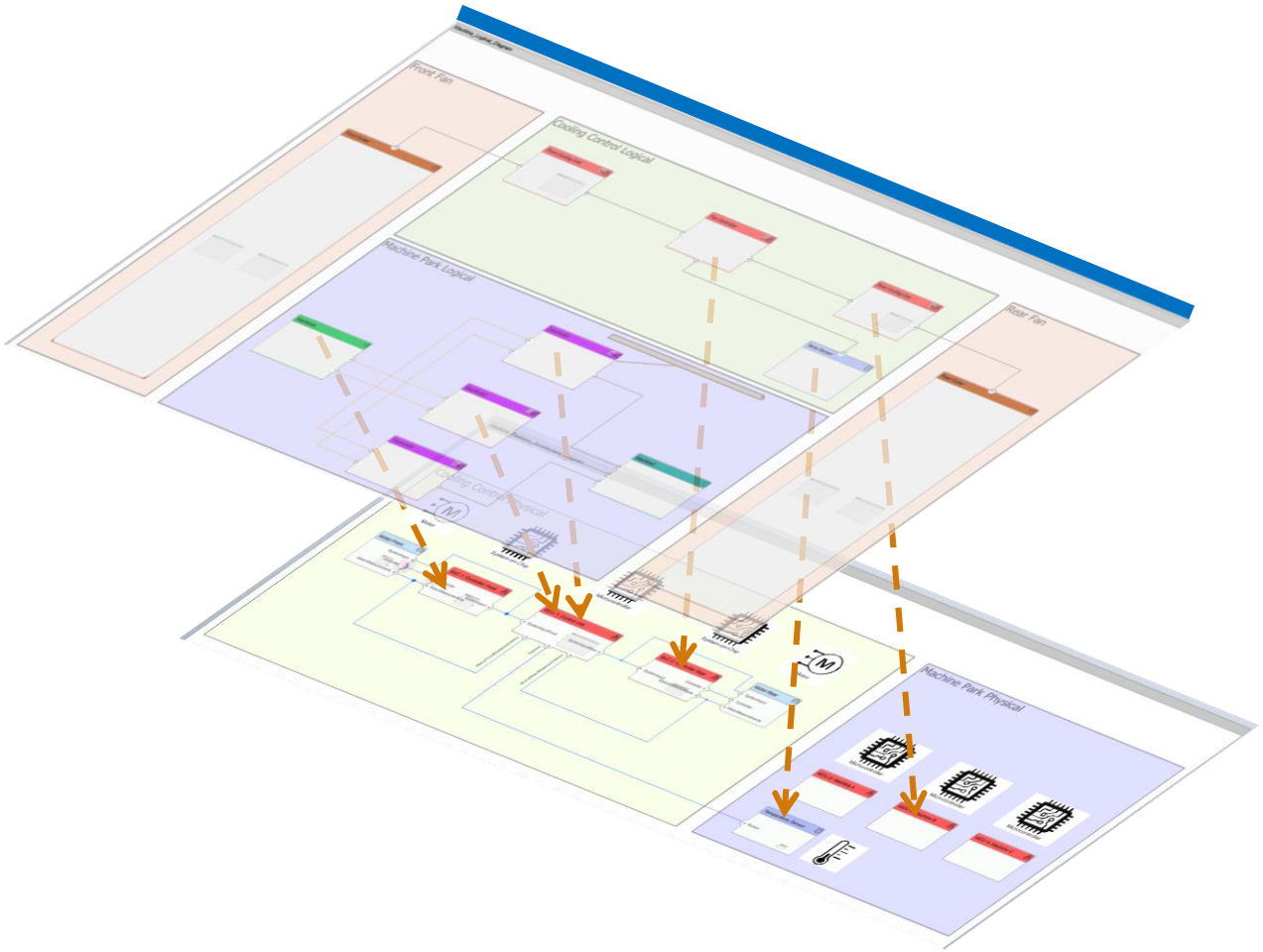
ALLOCATION SET BROWSER

- Functional_to_Logical
  - Scenario 1
- Logical_to_Physical
  - Scenario 1

Scenario 1

	Machine_Platform_Arc	MCU 1: Control Un	SoC 2: Controller F	Motor Front	Motor Rear	SoC 1: Controller F	Temperature Sensc	MCU 4: Machine C	MCU 2: Machine A	MCU 3: Machine B
Machine_Logical_Diagram										
Rear Cooling Unit			↑							
MachineB1										↑
Fan Controller		↑								
MachineB3										↑
MachineA									↑	
Front Cooling Unit						↑				
Front Cooler										
Motor				↑						
MachineB2										↑
MachineC								↑		
Rear Cooler										
Motor					↑					
Temp Sensor							↑			

架构之间的分配



Allocation Editor

ALLOCATIONS

New Allocation Set | Open Allocation Set | Save Allocation Set — FILE
Add Scenario | Delete Scenario — SCENARIO
Synchronize — REFRESH
Port | Connector | Component | Allocated | Un-Allocated — ROW FILTER

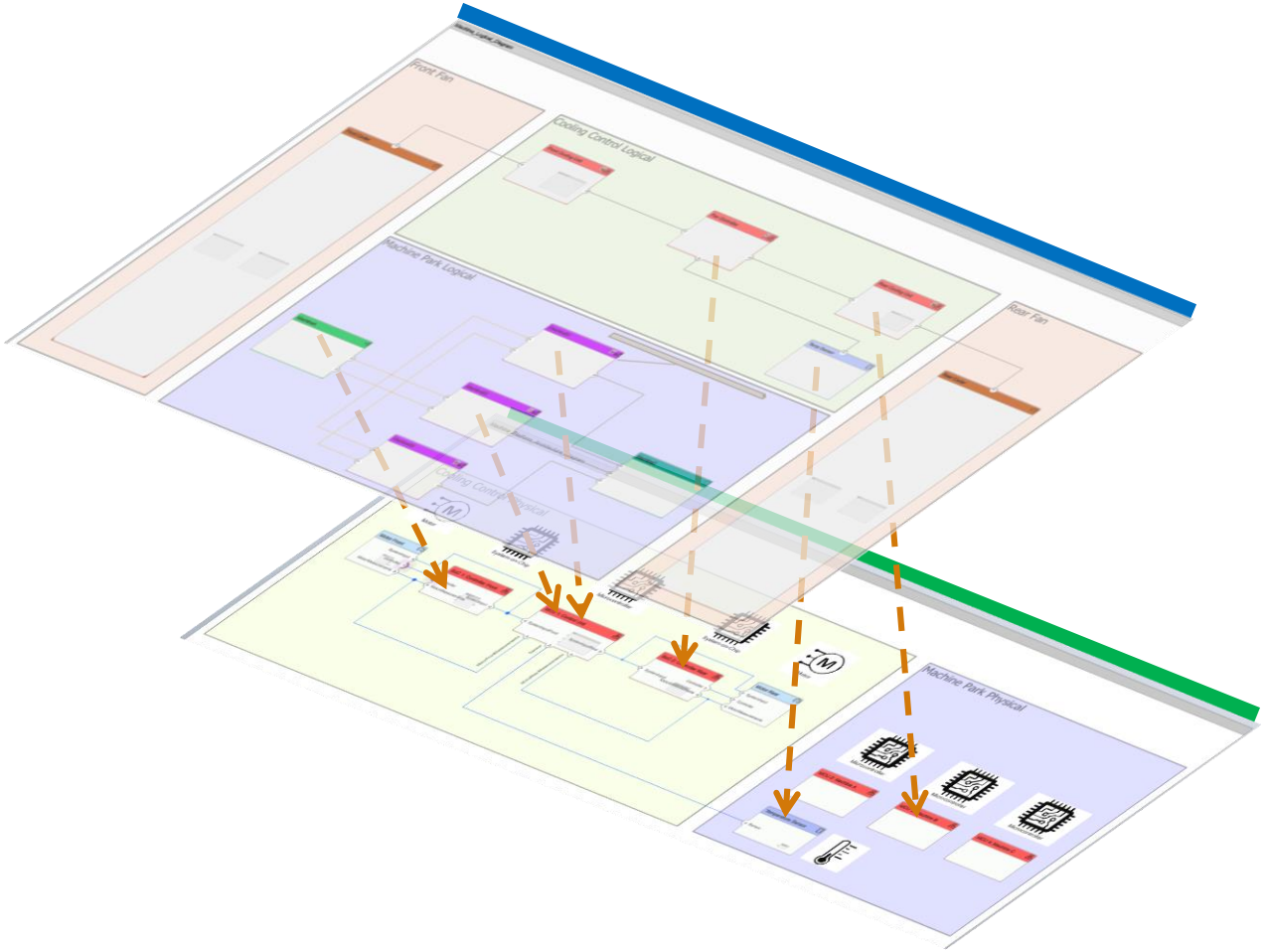
ALLOCATION SET BROWSER

- Functional_to_Logical
  - Scenario 1
- Logical_to_Physical
  - Scenario 1

Scenario 1

	Machine_Platform_Arc	MCU 1: Control Un	SoC 2: Controller F	Motor Front	Motor Rear	SoC 1: Controller F	Temperature Sensc	MCU 4: Machine C	MCU 2: Machine A	MCU 3: Machine B
Machine_Logical_Diagram										
Rear Cooling Unit			↑							
MachineB1										↑
Fan Controller		↑								
MachineB3										↑
MachineA									↑	
Front Cooling Unit						↑				
Front Cooler										
Motor				↑						
MachineB2										↑
MachineC								↑		
Rear Cooler										
Motor					↑					
Temp Sensor							↑			

架构之间的分配



Allocation Editor

ALLOCATIONS

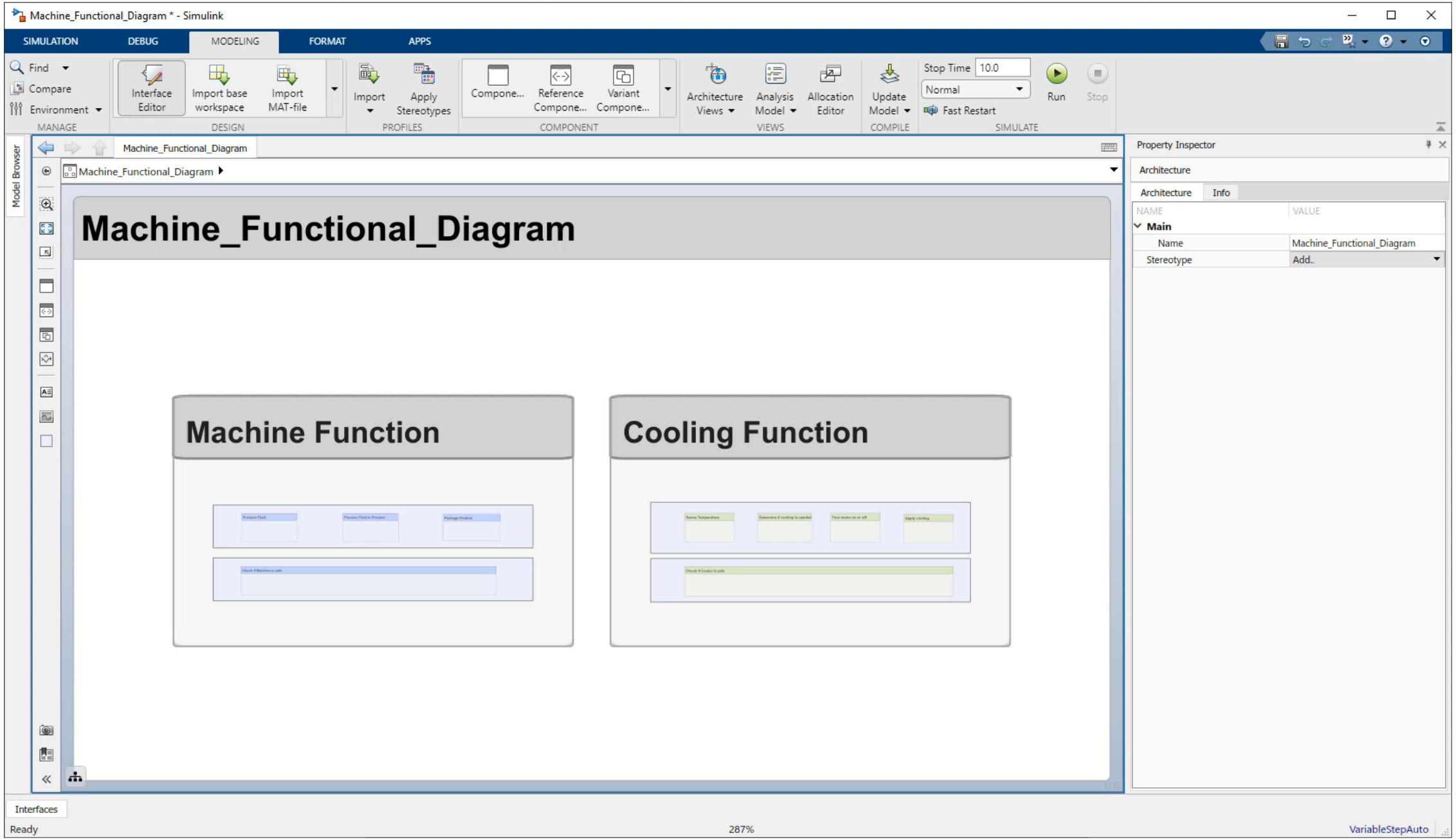
New Allocation Set | Open Allocation Set | Save Allocation Set — FILE
Add Scenario | Delete Scenario — SCENARIO
Synchronize — REFRESH
Port | Connector | Component | Allocated | Un-Allocated — ROW FILTER

ALLOCATION SET BROWSER

- Functional_to_Logical
  - Scenario 1
- Logical_to_Physical
  - Scenario 1

Scenario 1

	Machine_Platform_Arc	MCU 1: Control Un	SoC 2: Controller F	Motor Front	Motor Rear	SoC 1: Controller F	Temperature Sensc	MCU 4: Machine C	MCU 2: Machine A	MCU 3: Machine B
Machine_Logical_Diagram										
Rear Cooling Unit			↑							
MachineB1										↑
Fan Controller		↑								
MachineB3										↑
MachineA									↑	
Front Cooling Unit						↑				
Front Cooler										
Motor				↑						
MachineB2										↑
MachineC								↑		
Rear Cooler										
Motor					↑					
Temp Sensor							↑			


Machine_Functional_Diagram * - Simulink
SIMULATION
DEBUG
MODELING
FORMAT
APPS
Find
Compare
Environment
MANAGE
Interface Editor
Import base workspace
Import MAT-file
DESIGN
Import
Apply Stereotypes
PROFILES
Compone...
Reference Compone...
Variant Compone...
COMPONENT
Architecture Views
Analysis Model
Allocation Editor
VIEWS
Update Model
COMPILE
Stop Time
10.0
Normal
Fast Restart
Run
Stop
SIMULATE
Model Browser
Machine_Functional_Diagram
Machine_Functional_Diagram
Machine_Functional_Diagram
Machine Function
Cooling Function
Property Inspector
Architecture
Architecture
Info
NAME
VALUE
Main
Name
Machine_Functional_Diagram
Stereotype
Add..
Interfaces
Ready
287%
VariableStepAuto

定量评估不同的分配方案

Allocate Architectures in a Tire Pressure Monitoring System

R2021a

This example shows how to use allocations to analyze a tire pressure monitoring system.

View MATLAB Command

Overview

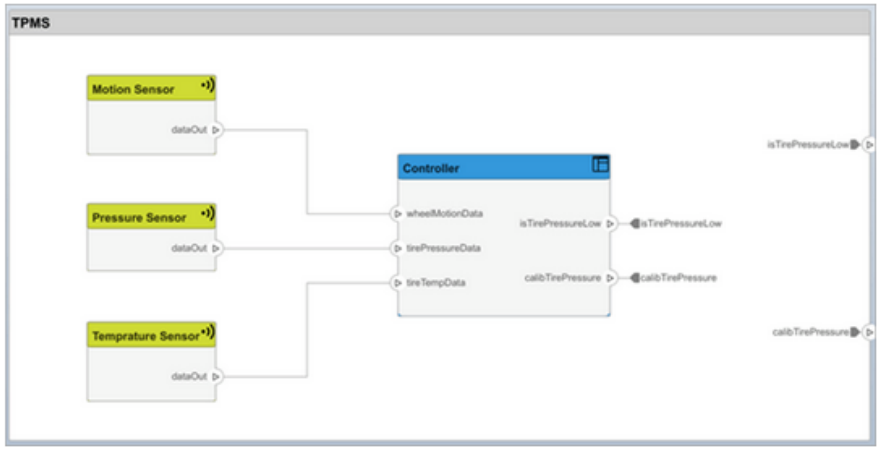
In systems engineering, it is common to describe a system at different levels of abstraction. For example, you can describe a systen not have any behavior associated with them but most likely trace back to some operating requirements the system must fulfill. We re *architecture*. In this example, an automobile tire pressure monitoring system is described in three different architectures:

1. Functional Architecture — Describes the system in terms of its high-level functions. The connections show dependencies betwe
2. Logical Architecture — Describes the system in terms of its logical components and how data is exchanged between them. Add simulation.
3. Platform Architecture — Describes the physical hardware needed for the system at a high level.

The allocation process is defined as linking these three architectures that fully describe the system. The linking captures the informa accessible to the others.

Use this command to open the project.

```
scExampleTirePressureMonitorSystem
```

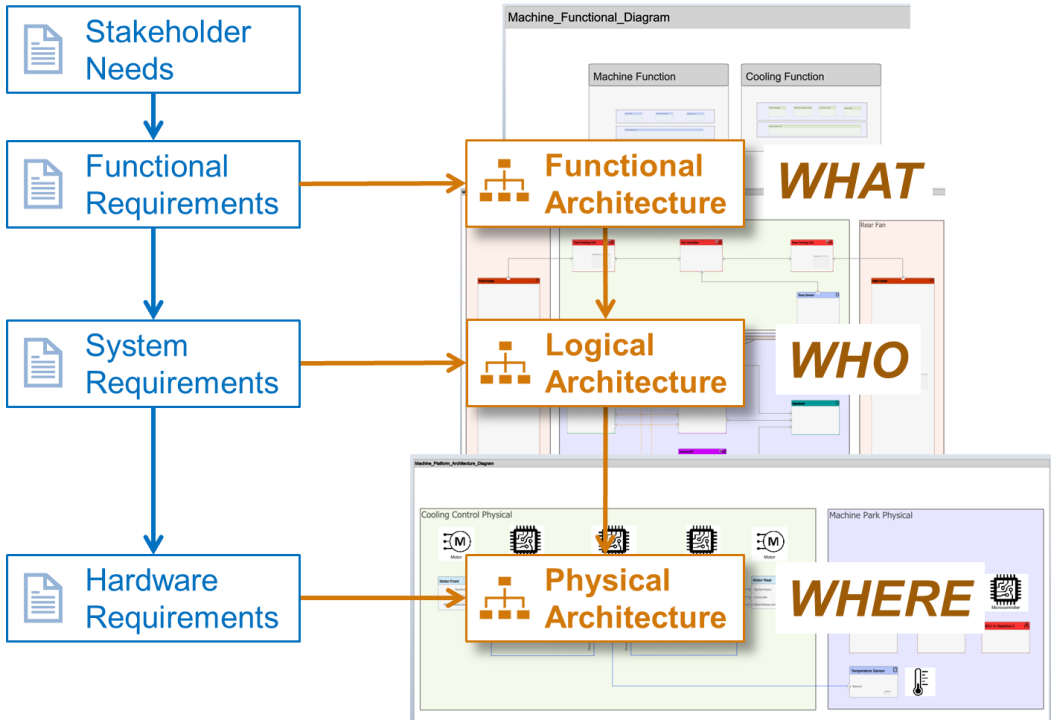


	Supplier A	Supplier B	Supplier C	Supplier D
Report Low Tire Pressure	1	0	0	0
Measure pressure on tire	0	0	1	0
Calculate Tire Pressure	0	1	0	0
Measure temprature of tire	0	0	0	1
Measure rotations	0	1	0	0
Calculate if pressure is low	1	0	0	0
Report Tire Pressure Levels	1	0	0	0
Measure Tire Pressure	0	0	0	0

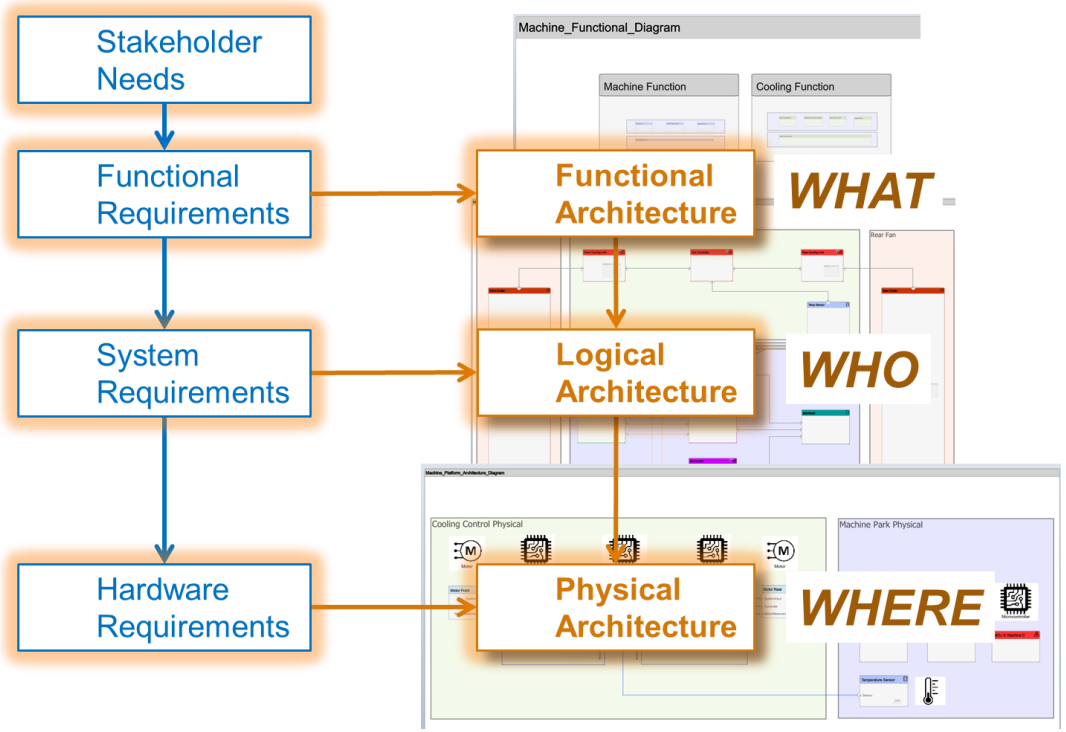
	Scenario 1	Scenario 2
Front ECUMemory Used (MB)	110	90
Front ECU Memory (MB)	100	100
Front ECU Overloaded	1	0
Rear ECU Memory Used (MB)	0	20
Rear ECU Memory (MB)	100	100
Rear ECU Overloaded	0	0

https://www.mathworks.com/help/systemcomposer/ug/allocate-architectures-of-the-tire-pressure-monitoring-system.html

要点

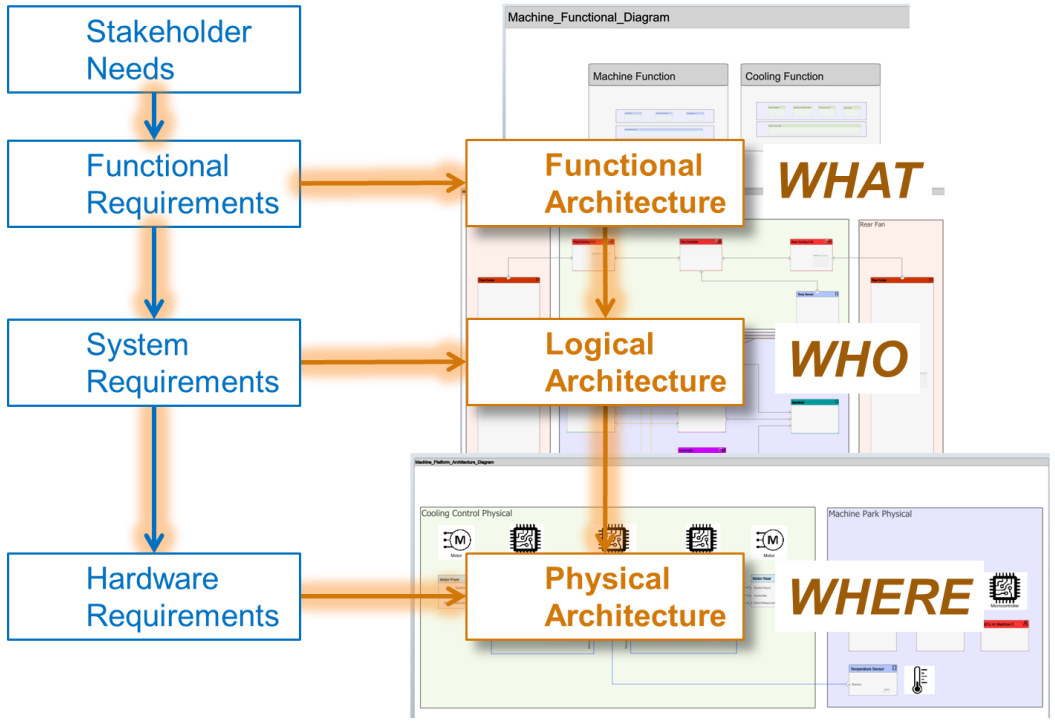

Stakeholder Needs
Functional Requirements
System Requirements
Hardware Requirements
Machine_Functional_Diagram
Machine Function
Cooling Function
Functional Architecture
WHAT
Logical Architecture
WHO
Physical Architecture
WHERE
Cooling Control Physical
Machine Park Physical

要点



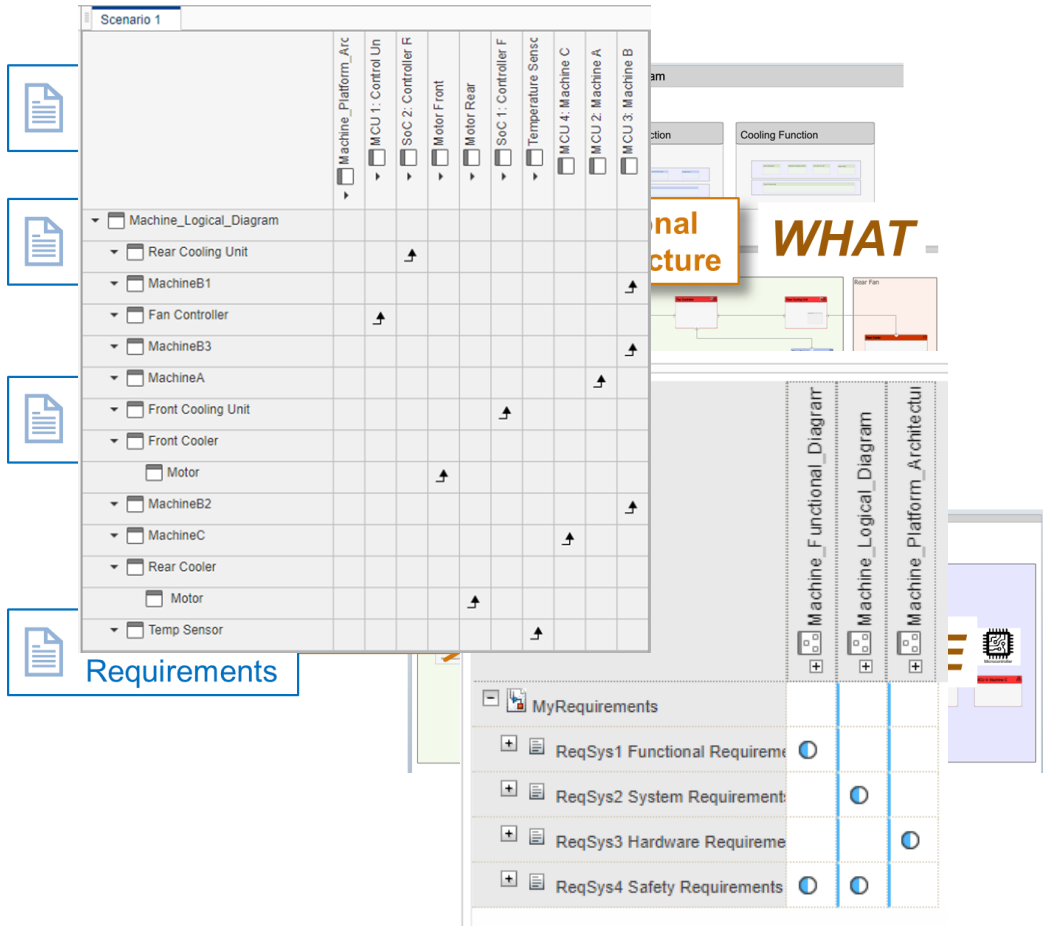
- 可以直接在与架构和设计模型**相同的环境**中导入，编写和存储文本需求.

要点



- 可以直接在与架构和设计模型**相同的环境**中导入，编写和存储文本需求.
- 通过在多个需求和架构工件之间建立关系，可以了解**系统变更的影响**.

Key Takeaways



- 可以直接在与架构和设计模型**相同的环境**中导入，编写和存储文本需求.
- 通过在多个需求和架构工件之间建立关系，可以了解**系统变更的影响**.
- 可以通过可视化这些关系来**评估系统的完整性**.

谁在使用MathWorks工具进行基于模型的系统工程?

谁在使用MathWorks工具进行基于模型的系统工程?

System Architecture Modeling for Electronic Systems Using MathWorks System Composer and Simulink

Christopher B. Watkins
Gulfstream Aerospace Corporation
Savannah, GA, U.S.
chris.watkins@gulfstream.com

Jerry Varghese
Gulfstream Aerospace Corporation
Savannah, GA, U.S.
jerry.varghese@gulfstream.com

Michael Knight
Gulfstream Aerospace Corporation
Savannah, GA, U.S.
michael.knight@gulfstream.com

Becky Petteys
The MathWorks, Inc.
Natick, Massachusetts, U.S.
bpetteys@mathworks.com

Jordan Ross
The MathWorks, Inc.
Natick, Massachusetts, U.S.
jordanr@mathworks.com

***Abstract*—Electronic system architectures have traditionally been documented as static block diagrams in tools such as Microsoft® Visio® or through a richer modeling approach such as Systems Modeling Language (SysML). These approaches did not fully meet the modeling needs for the Gulfstream authors, which led to an alternative approach.**

This paper introduces the Electronic System Architecture Modeling (eSAM) method, which leverages a new system architecture modeling tool called System Composer™. eSAM was created by the authors to define a standard method for applying the generic System Composer modeling constructs to build functional, physical, and logical architecture models of electronic systems. The eSAM methods are applied to an example avionics architecture to demonstrate capabilities needed for system modeling, collaborative OEM-supplier workflows, data management and ICD generation, systems integration activities, generation of system architecture deliverables for the avionics

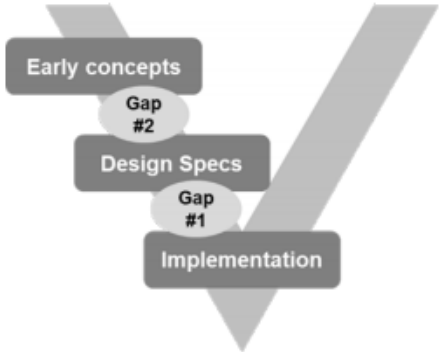
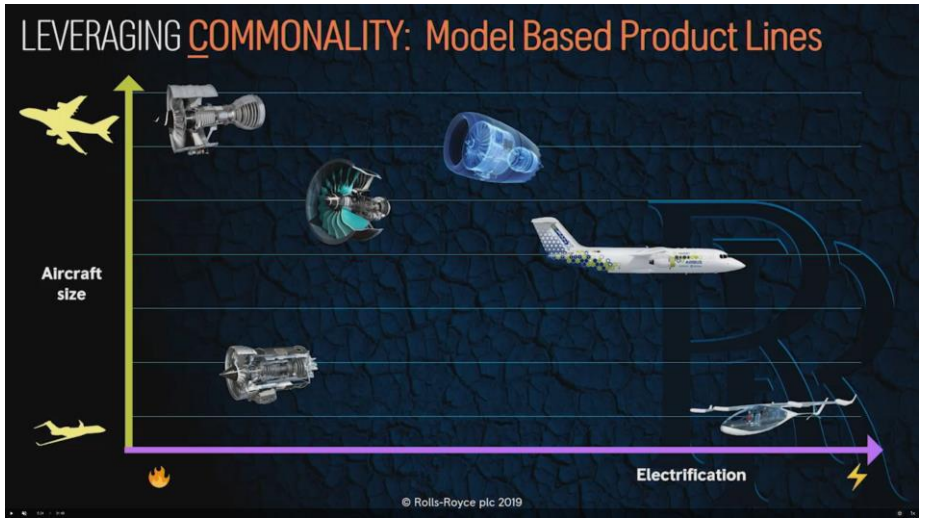


Figure 1: Simplified development process

https://ieeexplore.ieee.org/document/9256753

谁在使用MathWorks工具进行基于模型的系统工程?

Rolls Royce, UK Expo, Oct 2019



https://www.mathworks.com/videos/our-journey-towards-model-based-product-lines-1573233985120.html

System Architecture Modeling for Electronic Systems Using MathWorks System Composer and Simulink

Christopher B. Watkins
Gulfstream Aerospace Corporation
Savannah, GA, U.S.
chris.watkins@gulfstream.com

Jerry Varghese
Gulfstream Aerospace Corporation
Savannah, GA, U.S.
jerry.varghese@gulfstream.com

Michael Knight
Gulfstream Aerospace Corporation
Savannah, GA, U.S.
michael.knight@gulfstream.com

Becky Petteys
The MathWorks, Inc.
Natick, Massachusetts, U.S.
bpetteys@mathworks.com

Jordan Ross
The MathWorks, Inc.
Natick, Massachusetts, U.S.
jordanr@mathworks.com

***Abstract*—Electronic system architectures have traditionally been documented as static block diagrams in tools such as Microsoft® Visio® or through a richer modeling approach such as Systems Modeling Language (SysML). These approaches did not fully meet the modeling needs for the Gulfstream authors, which led to an alternative approach.**

This paper introduces the Electronic System Architecture Modeling (eSAM) method, which leverages a new system architecture modeling tool called System Composer™. eSAM was created by the authors to define a standard method for applying the generic System Composer modeling constructs to build functional, physical, and logical architecture models of electronic systems. The eSAM methods are applied to an example avionics architecture to demonstrate capabilities needed for system modeling, collaborative OEM-supplier workflows, data management and ICD generation, systems integration activities, generation of system architecture deliverables for the avionics

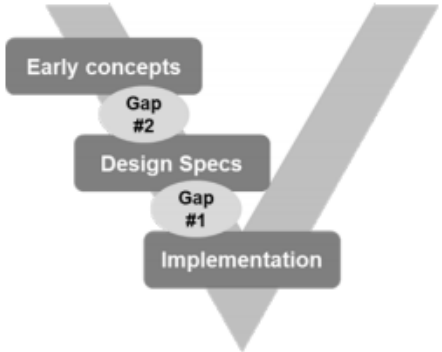


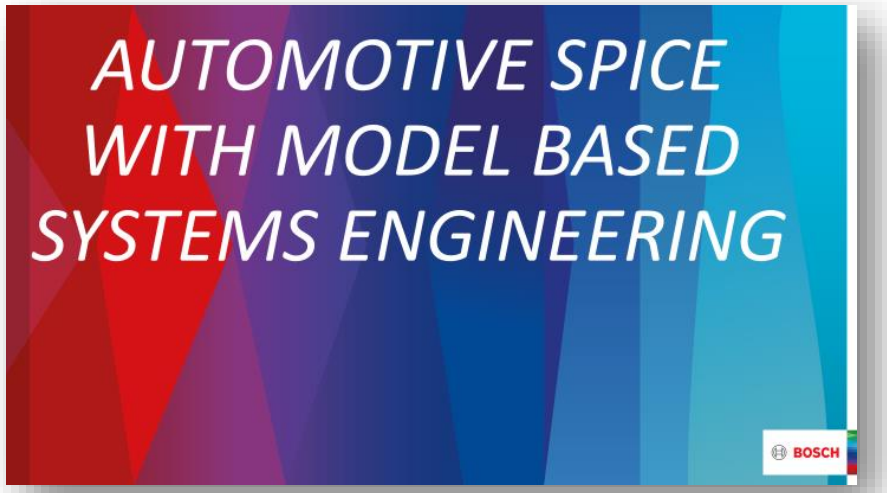
Figure 1: Simplified development process

https://ieeexplore.ieee.org/document/9256753

谁在使用MathWorks工具进行基于模型的系统工程?

MathWorks Automotive Conference 2021

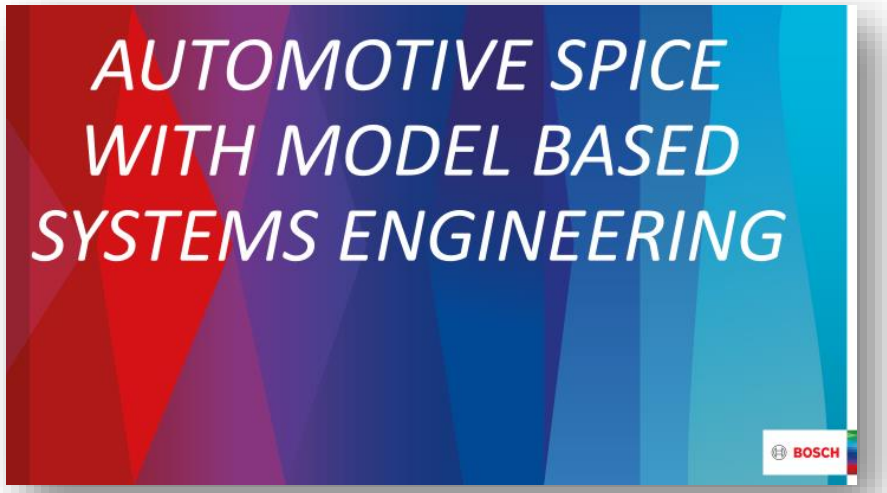
Felix Raab, Bosch

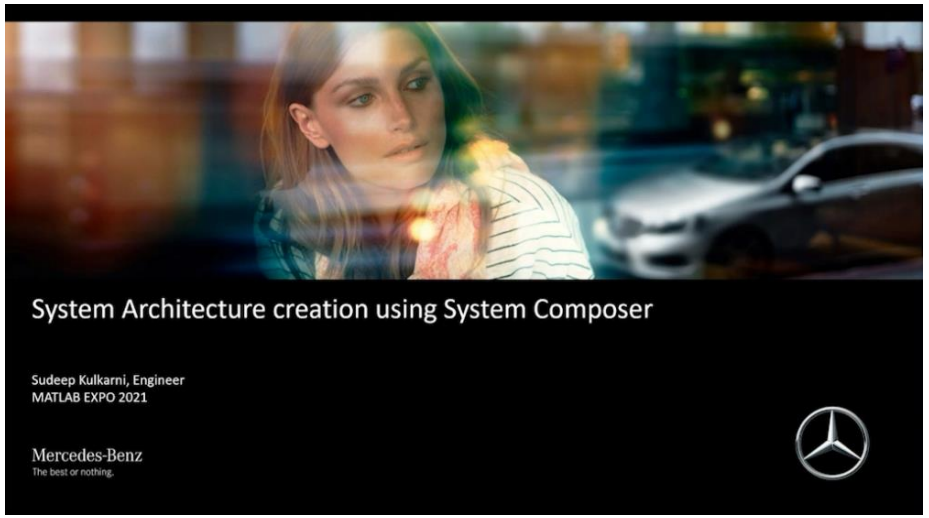


谁在使用MathWorks工具进行基于模型的系统工程?

MathWorks Automotive Conference 2021

Felix Raab, Bosch





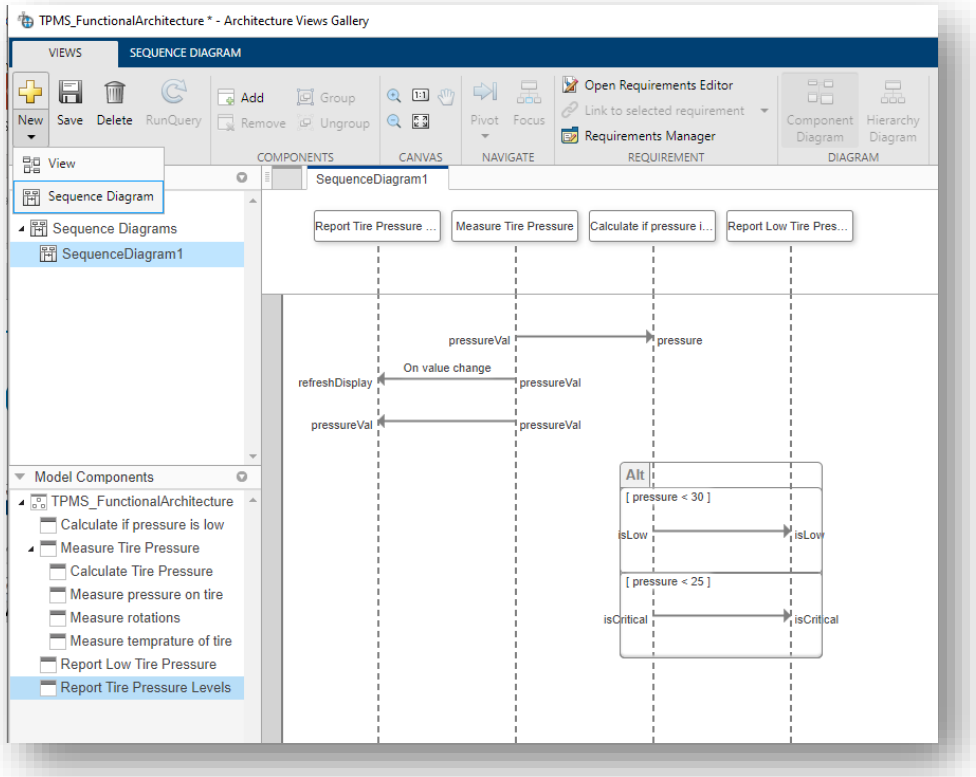
Sudeep Kulkarni,
Mercedes-Benz

新特性 R2021a

- System Composer

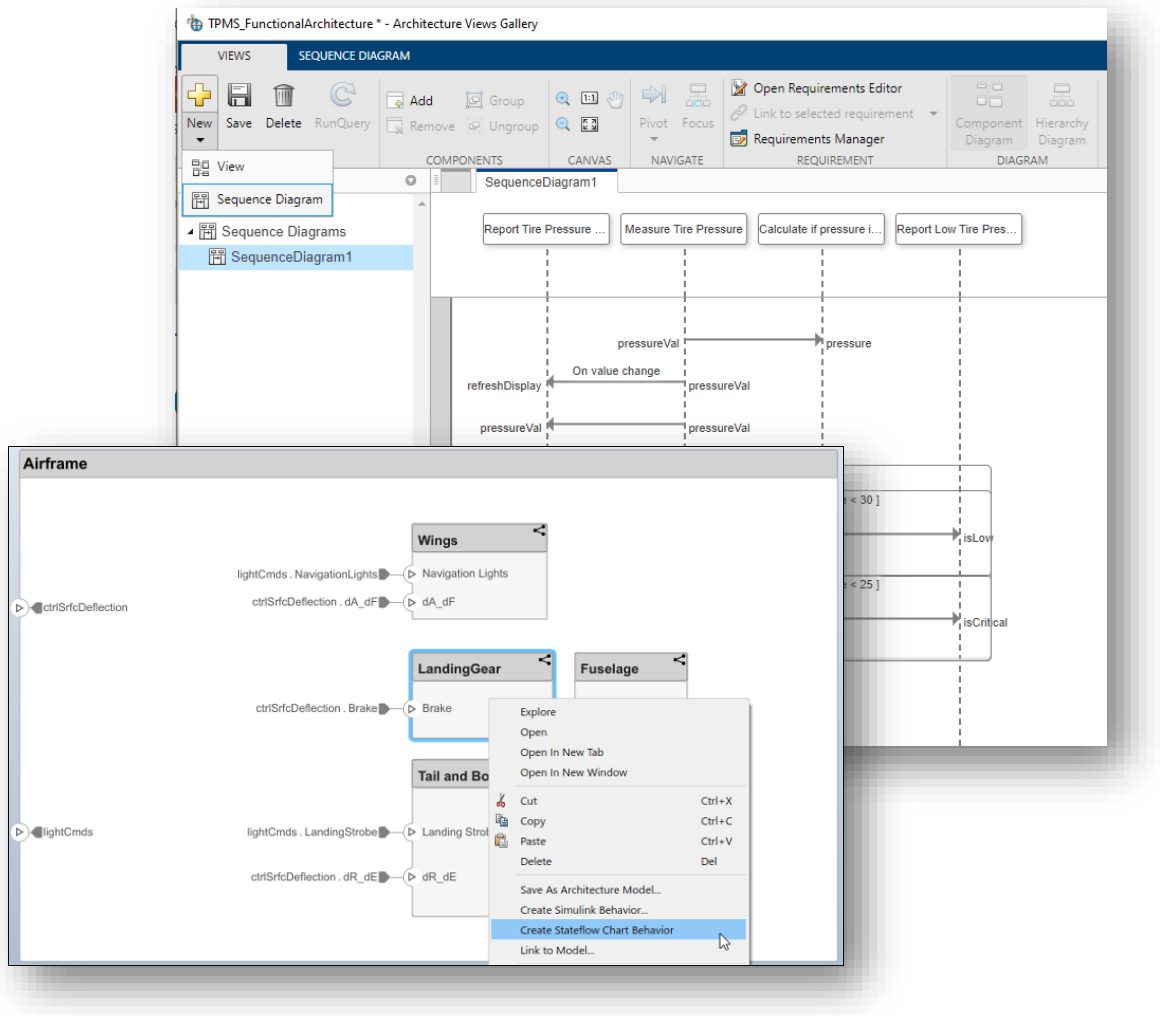
新特性 R2021a

- System Composer
  - Sequence diagrams



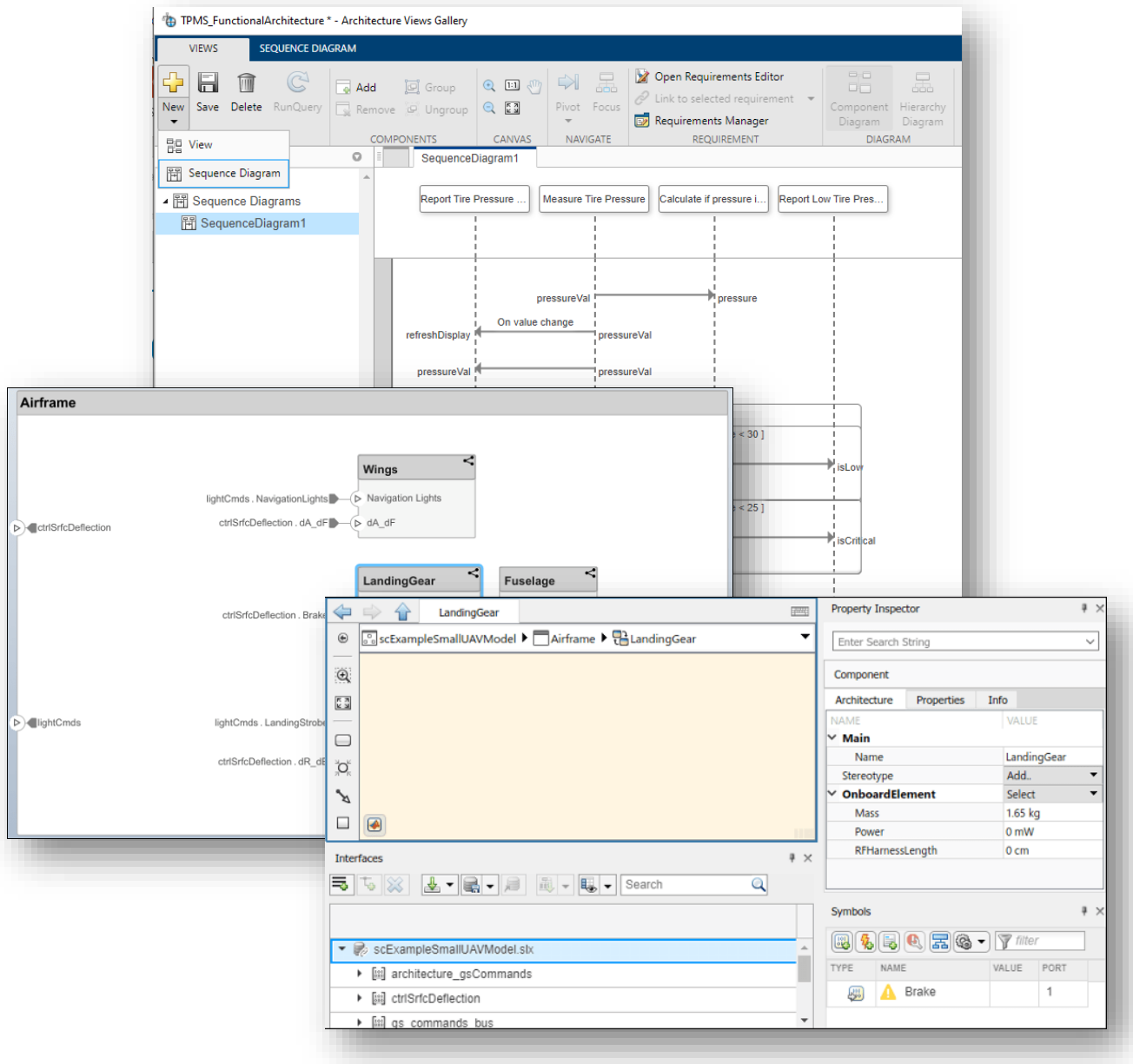
新特性 R2021a

- System Composer
  - Sequence diagrams
  - Stateflow charts in components



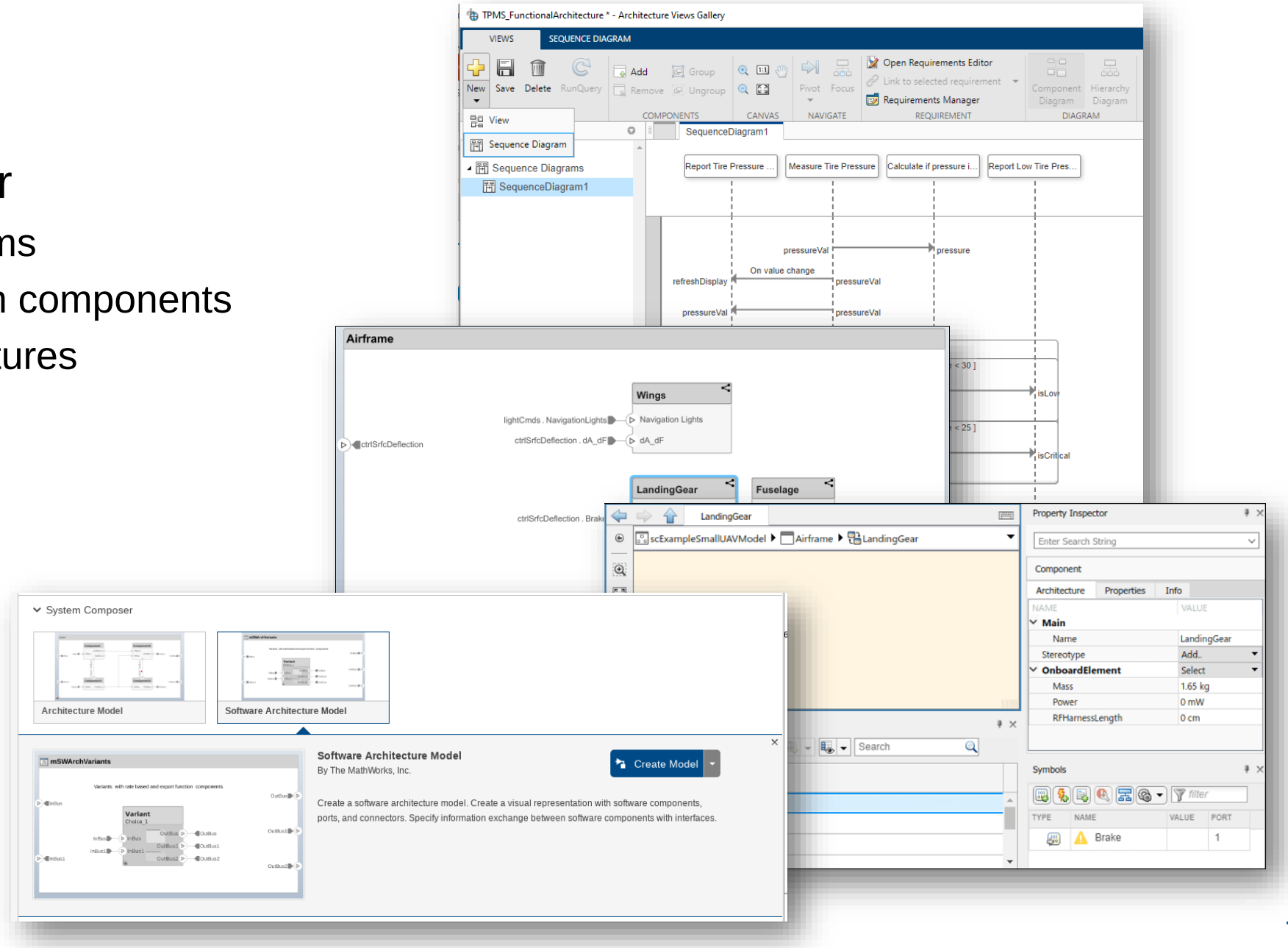
新特性 R2021a

- System Composer
  - Sequence diagrams
  - Stateflow charts in components



新特性 R2021a

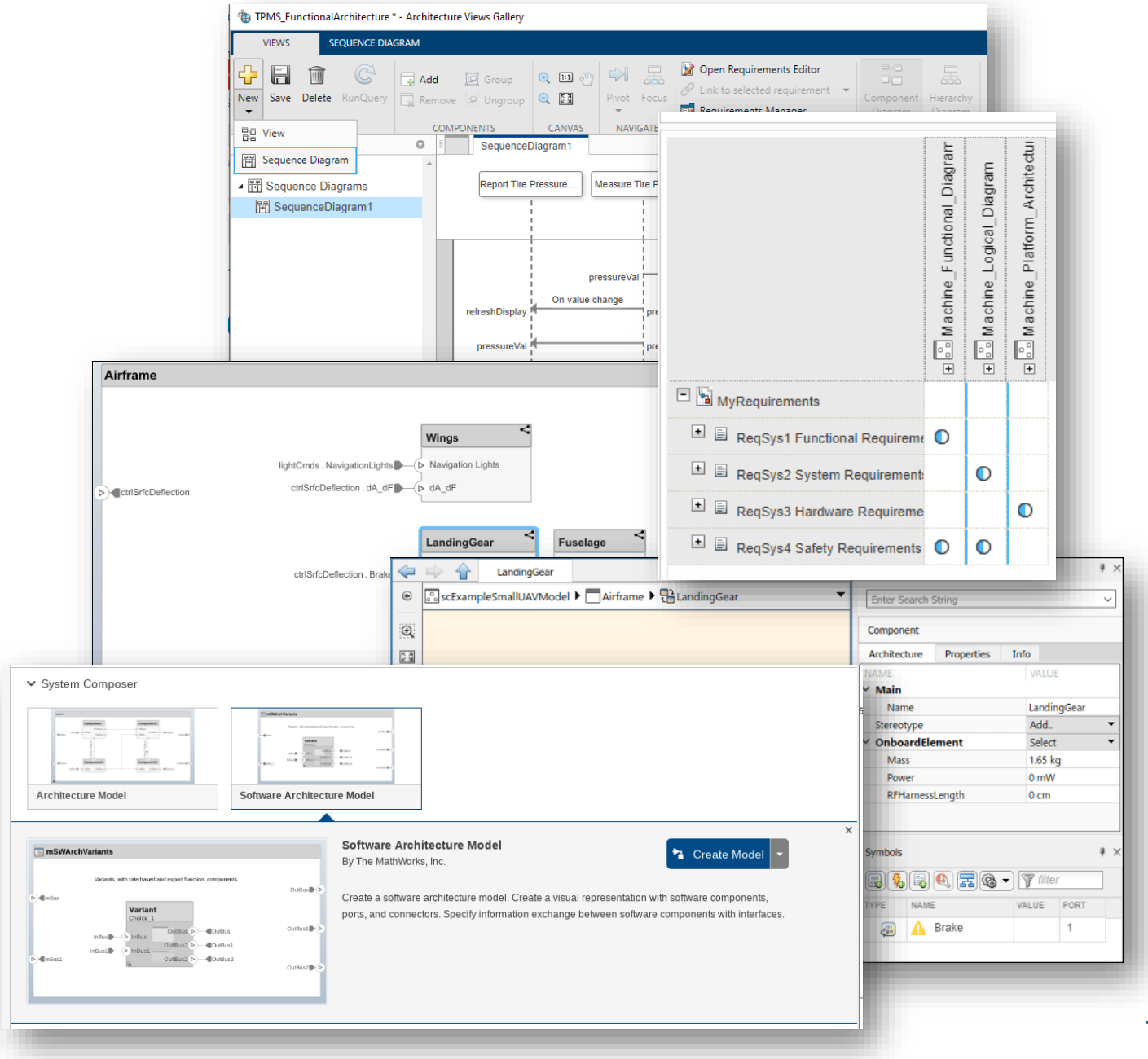
- System Composer
  - Sequence diagrams
  - Stateflow charts in components
  - Software architectures



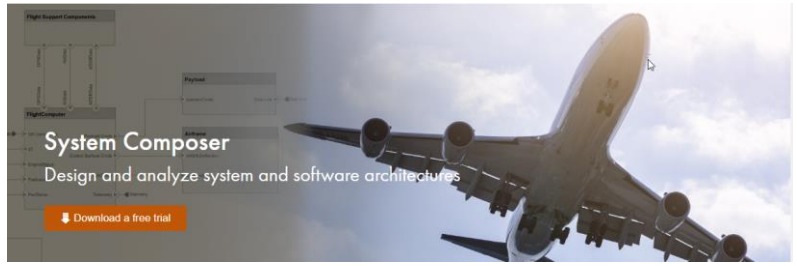
新特性 R2021a

- System Composer
  - Sequence diagrams
  - Stateflow charts in components
  - Software architectures

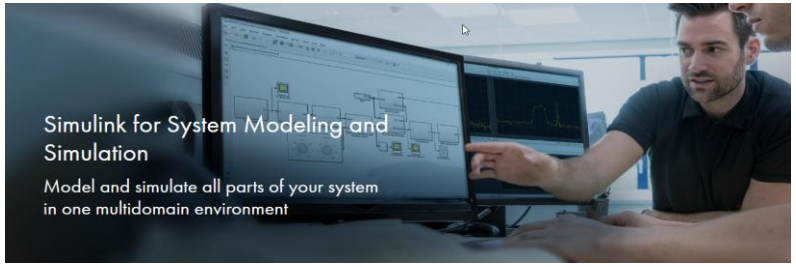
- Simulink Requirements
  - Editor improvements
  - Multi-artifact traceability matrix



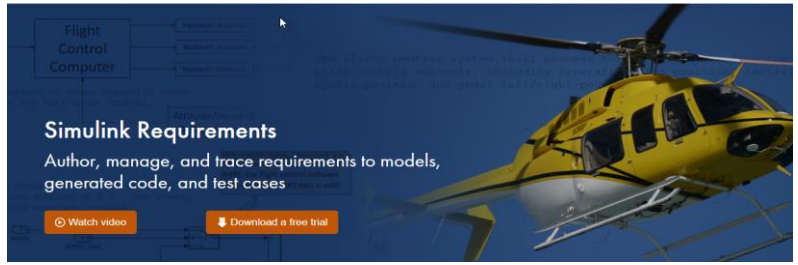
更多信息



[System Composer]

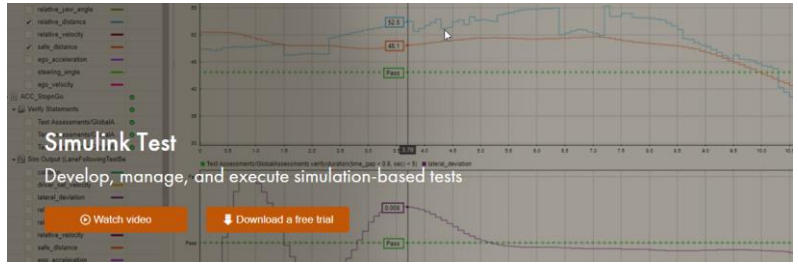


[Model-Based Systems Engineering]



[Simulink Requirements]

[System Modeling and Simulation]



[Simulink Test]

[AUTOSAR]

MATLAB EXPO 2021

Thank you