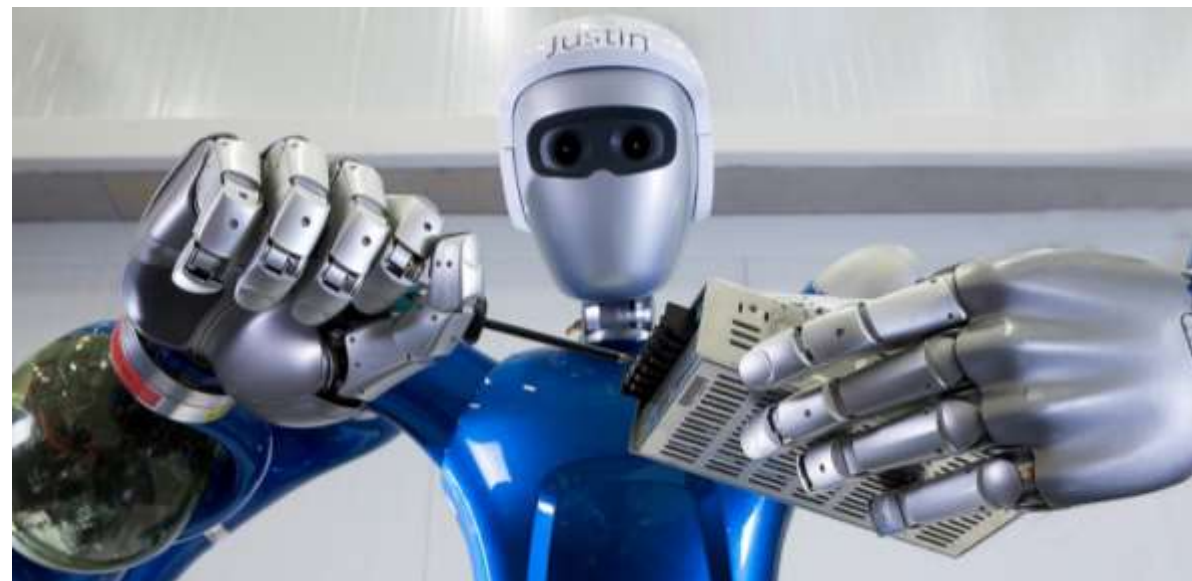


# MATLAB EXPO

在基于模型的敏捷设计中引入测试驱动开发  
杨超, MathWorks 中国应用工程师



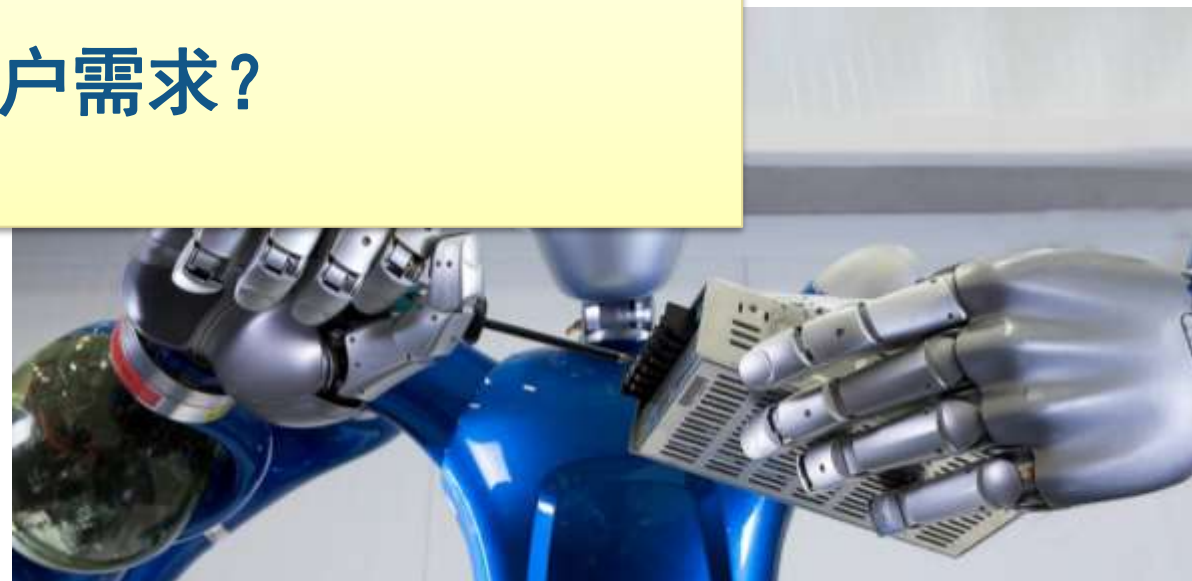
# 万物皆算法...



# 万物皆算法...

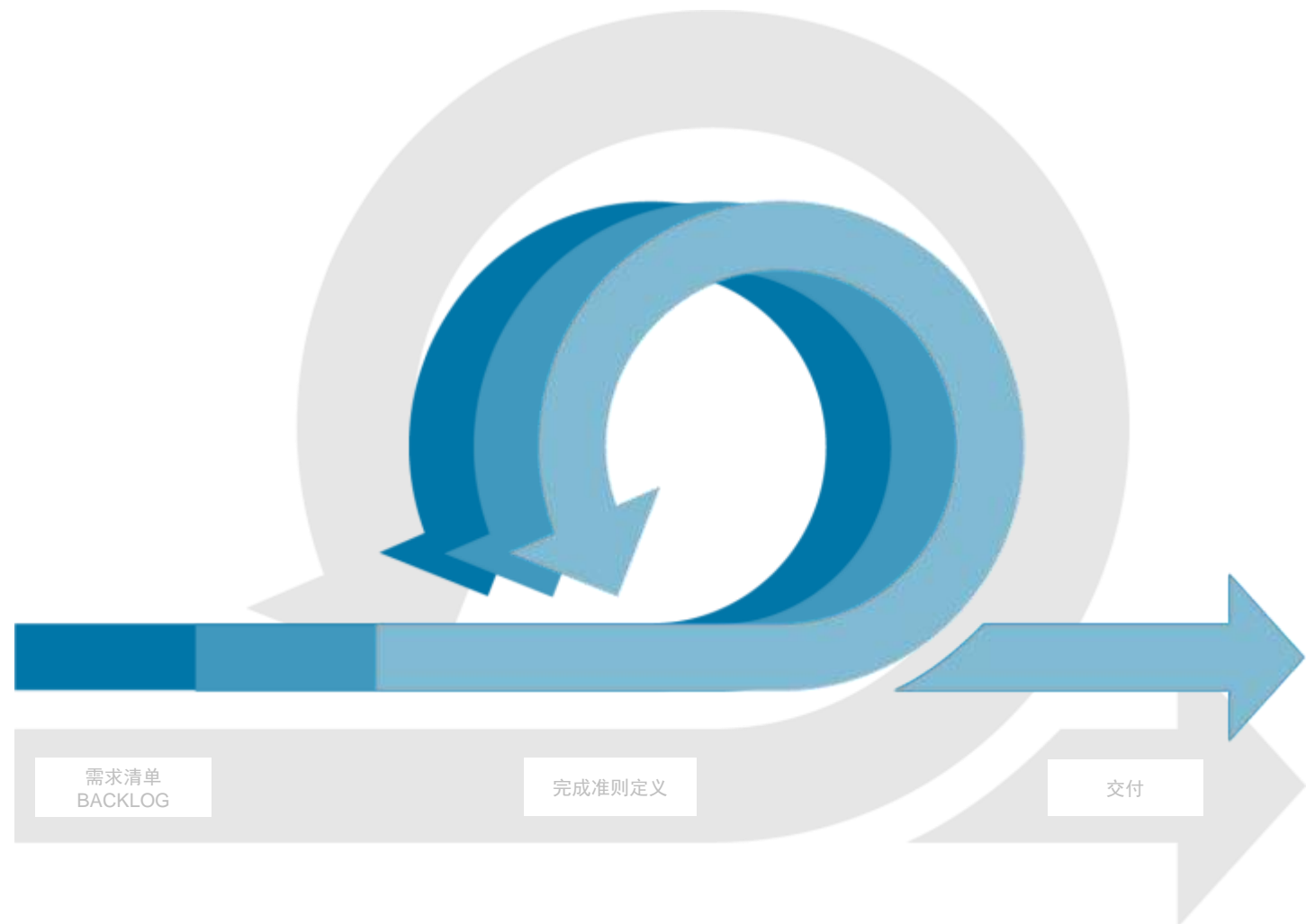


但是如何...  
更快交付?  
满足极具挑战的客户需求?  
保证质量?



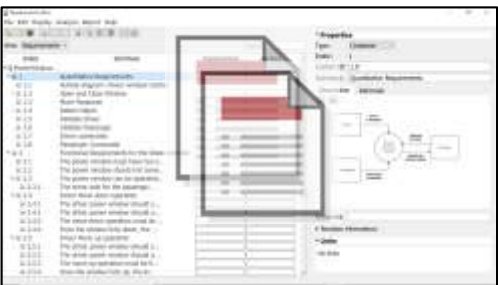
# 测试驱动的开发流程

1. 创建测试
2. 实施最少的设计以通过测试
3. 重构

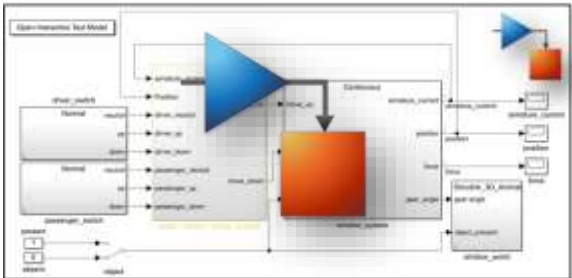


# Simulink为测试驱动开发（TDD）提供集成框架

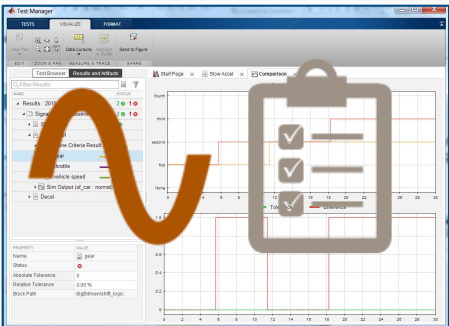
需求



实现



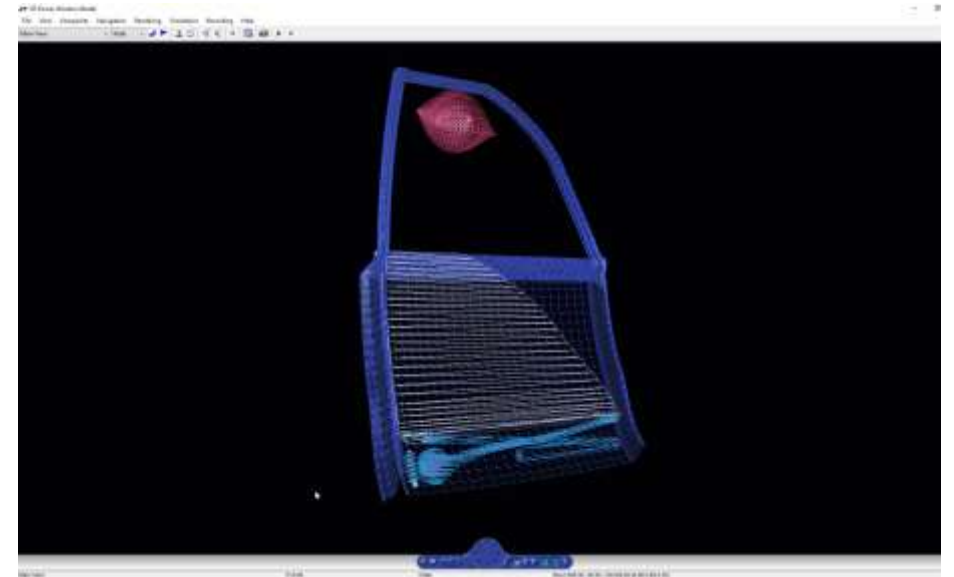
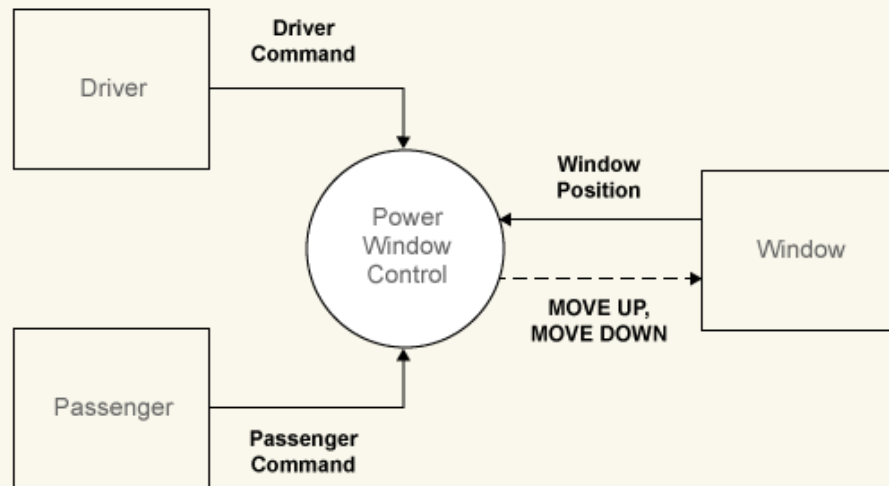
测试



# 从客户的顶层需求出发

## 使用者需求:

- 驾驶员和乘客应都能够控制车窗
- 如果检测到阻碍，车窗应停止关闭
- 车窗应能够全开和全关



# 需求捕获

The screenshot displays the Requirements Editor application. The main window is divided into a tree view on the left and a detailed view on the right.

**Tree View:**

- PowerWindo...
  - 1 Quantitative Requirements
    - 1.1 Activity diagram: Power window control
    - 1.2 Open and Close Window
      - 1.2.1 Fully Open
      - 1.2.2 Fully Close
    - 1.3 Move Response
    - 1.4 Detect Object
    - 1.5 Validate Driver
    - 1.6 Validate Passenger
    - 1.7 Driver commands
    - 1.8 Passenger Commands
  - 2 Controller Functional Requirements
    - 2.1 The power window must have two o...
    - 2.2 The power window should not move...
    - 2.3 The power window can be operated...
      - 2.3.1 The driver side for the passenge...
    - 2.4 Driver Move down operation
      - 2.4.1 Driver down button press
      - 2.4.2 Move down to end stop
      - 2.4.3 Move down automatically performance
      - 2.4.4 Enter neutral when fully down
    - 2.5 Driver Move up operation
      - 2.5.1 The driver power window should o...
      - 2.5.2 The driver power window should o...
      - 2.5.3 The move up operation must be fl...
      - 2.5.4 Once the window fully up, the dr...
    - 2.6 Passenger Move down operation
      - 2.6.1 The Passenger power window shoul...
      - 2.6.2 The Passenger power window shoul...
      - 2.6.3 The move down operation must be ...
      - 2.6.4 Once the window fully down, the ...

**Properties Panel (Right):**

- Type: Container
- Index: 2
- Custom ID: REQ 2
- Summary: Controller Functional Requirements
- Description: Both the driver and passenger can send commands to the window to move it up and down. The controller infers the correct command to send to the window actuator (e.g., the driver command has priority over the passenger command). In addition, diagram monitors the state of the window system to establish when the window is fully opened and closed and to detect if there is an object between the window and frame.
- Rationale: (Empty)
- Keywords: (Empty)
- Revision information: (Empty)
- Links: Implemented by: power\_window\_control\_system

**Diagram:** A diagram of a car window frame with labels: top, obstacle position, and bottom. A red diamond shape is positioned near the top of the window frame.

# 详情查看

需求  
详情

The screenshot shows the Requirements Editor interface. On the left is a tree view of requirements, and on the right is a table with columns for Index, Summary, Implemented, and Verified. The 'Implemented' and 'Verified' columns contain progress bars. A blue arrow points from the text '需求详情' to the tree view.

Index	Summary	Implemented	Verified
PowerWindo...		<div><div style="width: 20%;"></div></div>	<div><div style="width: 10%;"></div></div>
1	Quantitative Requirements	<div><div style="width: 100%;"></div></div>	<div><div style="width: 50%;"></div></div>
1.1	Activity diagram: Power window control	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
1.2	Open and Close Window	<div><div style="width: 100%;"></div></div>	<div><div style="width: 70%;"></div></div>
1.2.1	Fully Open	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
1.2.2	Fully Close	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
1.3	Move Response	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
1.4	Detect Object	<div><div style="width: 100%;"></div></div>	<div><div style="width: 50%;"></div></div>
1.5	Validate Driver	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
1.6	Validate Passenger	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
1.7	Driver commands	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
1.8	Passenger Commands	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2	Controller Functional Requirements	<div><div style="width: 10%;"></div></div>	<div><div style="width: 10%;"></div></div>
2.1	The power window must have two o...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.2	The power window should not move...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.3	The power window can be operated...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.3.1	The driver side for the passenge...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.4	Driver Move down operation	<div><div style="width: 100%;"></div></div>	<div><div style="width: 10%;"></div></div>
2.4.1	Driver down button press	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.4.2	Move down to end stop	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.4.3	Move down automatically performance	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.4.4	Enter neutral when fully down	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.5	Driver Move up operation	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.5.1	The driver power window should o...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.5.2	The driver power window should o...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.5.3	The move up operation must be fl...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.5.4	Once the window fully up, the dr...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.6	Passenger Move down operation	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.6.1	The Passenger power window shoul...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.6.2	The Passenger power window shoul...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.6.3	The move down operation must be ...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>
2.6.4	Once the window fully down, the ...	<div><div style="width: 100%;"></div></div>	<div><div style="width: 100%;"></div></div>



# 组织并创建需求层级

需求  
层级

Index	Summary	Implemented	Verified
PowerWindo...		<div style="width: 20%; background-color: blue;"></div>	<div style="width: 10%; background-color: green;"></div>
1	Quantitative Requirements	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 50%; background-color: green;"></div>
1.1	Activity diagram: Power window control	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
1.2	Open and Close Window	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 70%; background-color: green;"></div>
1.2.1	Fully Open	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
1.2.2	Fully Close	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
1.3	Move Response	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
1.4	Detect Object	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 50%; background-color: green;"></div>
1.5	Validate Driver	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
1.6	Validate Passenger	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
1.7	Driver commands	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
1.8	Passenger Commands	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2	Controller Functional Requirements	<div style="width: 10%; background-color: blue;"></div>	<div style="width: 5%; background-color: green;"></div>
2.1	The power window must have two o...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.2	The power window should not move...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.3	The power window can be operated...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.3.1	The driver side for the passenge...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.4	Driver Move down operation	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 10%; background-color: red;"></div>
2.4.1	Driver down button press	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.4.2	Move down to end stop	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.4.3	Move down automatically performance	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.4.4	Enter neutral when fully down	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.5	Driver Move up operation	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.5.1	The driver power window should o...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.5.2	The driver power window should o...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.5.3	The move up operation must be fl...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.5.4	Once the window fully up, the dr...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.6	Passenger Move down operation	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.6.1	The Passenger power window shoul...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.6.2	The Passenger power window shoul...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.6.3	The move down operation must be ...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>
2.6.4	Once the window fully down, the ...	<div style="width: 100%; background-color: blue;"></div>	<div style="width: 100%; background-color: green;"></div>

# 详细需求定义

The screenshot displays a requirement definition window with the following sections:

- Properties:** Type: Container, Index: 2, Custom ID: REQ 2, Summary: Controller Functional Requirements.
- Description:** Both the driver and passenger can send commands to the window to move it up and down. The controller infers the correct command to send to the window actuator (e.g., the driver command has priority over the passenger command). In addition, diagram monitors the state of the window system to establish when the window is fully opened and closed and to detect if there is an object between the window and frame.
- Rationale:** (Empty)
- Diagram:** A schematic of a window frame with labels: top, obstacle position, and bottom. A red obstacle is shown between the window and the top frame.
- Keywords:** (Empty)
- Revision information:** (Empty)
- Links:** Implemented by: [power\\_window\\_control\\_system](#)

描述 / 解释

链接  
看板

# 1. 创建测试

2. 实施最少的设计以通过测试

3. 重构



# 创建, 管理, 并执行基于仿真的测试

## Simulink Test

### Test Manager

- 创建, 管理, 组织测试

The Test Manager interface includes several key components:

- Test Browser:** A tree view showing test cases and their organization.
- Test Results:** A graphical view showing test execution results, including pass/fail status and timing.
- Reports:** A generated report titled "Report Generated by Test Manager" with fields for Title, Author, Date, and Test Environment.

### Test Harnesses

- 识别测试组件

The Test Harness interface shows a Simulink model with a component under test highlighted in a red dashed box. A red arrow points from this component to a detailed view of the component in the Test Harness window, which shows its inputs and outputs.

### Test Authoring

- 定义测试的输入、期望输出和偏差容限

The Test Authoring interface includes several key components:

- Test Sequence Editor:** A table defining test steps, including inputs (gear, speed, throttle) and expected outputs (speed, throttle).
- Signal Editor:** A tool for defining test signals, such as "Signal 1".
- Time-Series Data:** A tool for defining time-series data, such as "Time-Series Data".
- Temporal Assessments:** A tool for defining temporal assessments, such as "Temporal Assessments".

# 创建测试框架（Test Harness）以识别待测试的组件

The screenshot shows the Simulink interface for a model named 'slexPowerWindowExample'. The 'SUBSYSTEM BLOCK' tab is active, and the 'Add Test Harness' button is highlighted with a red box. The main workspace contains a block diagram of a power window control system. The diagram includes several subsystems: 'driver\_switch' and 'passenger\_switch' (each with 'Normal' and 'Neutral' states), 'power\_window\_control\_system' (highlighted in yellow), 'window\_system' (containing a 'Continuous' block), and 'window\_world' (containing a 'Simulink\_3D\_Animat' block). Signal lines connect these blocks, with outputs for 'armature\_current', 'position', and 'force' being monitored by scope blocks. A 'present' signal (1) and 'absent' signal (0) are also shown. The status bar at the bottom indicates 'Ready', '111%', and 'ode23'.

# 定义测试框架属性

Create Test Harness

Specify the properties of the test harness. The component under test is the system for which the harness is being created. After creation, use the block badge to find and open harnesses.

Component under Test: [slexPowerWindowExample/power\\_window\\_control\\_system](#)

Basic Properties | Advanced Properties | Description

Name:

*Harnesses saved internally.* [More information](#)

Sources and Sinks

⇒  ⇒

Create scalar inputs

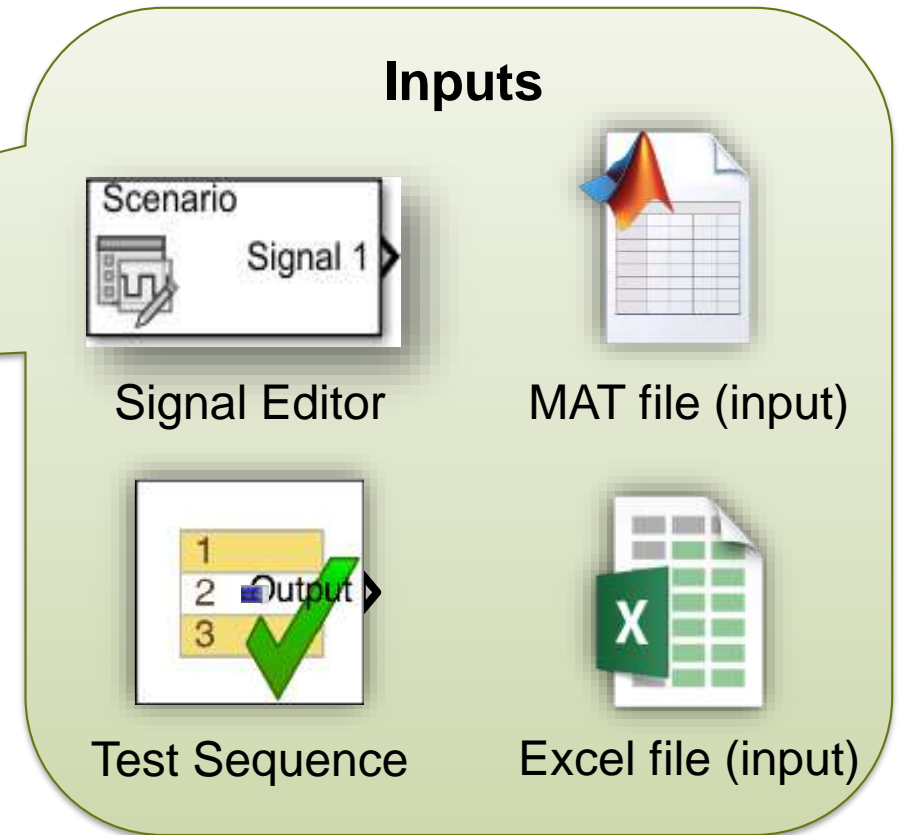
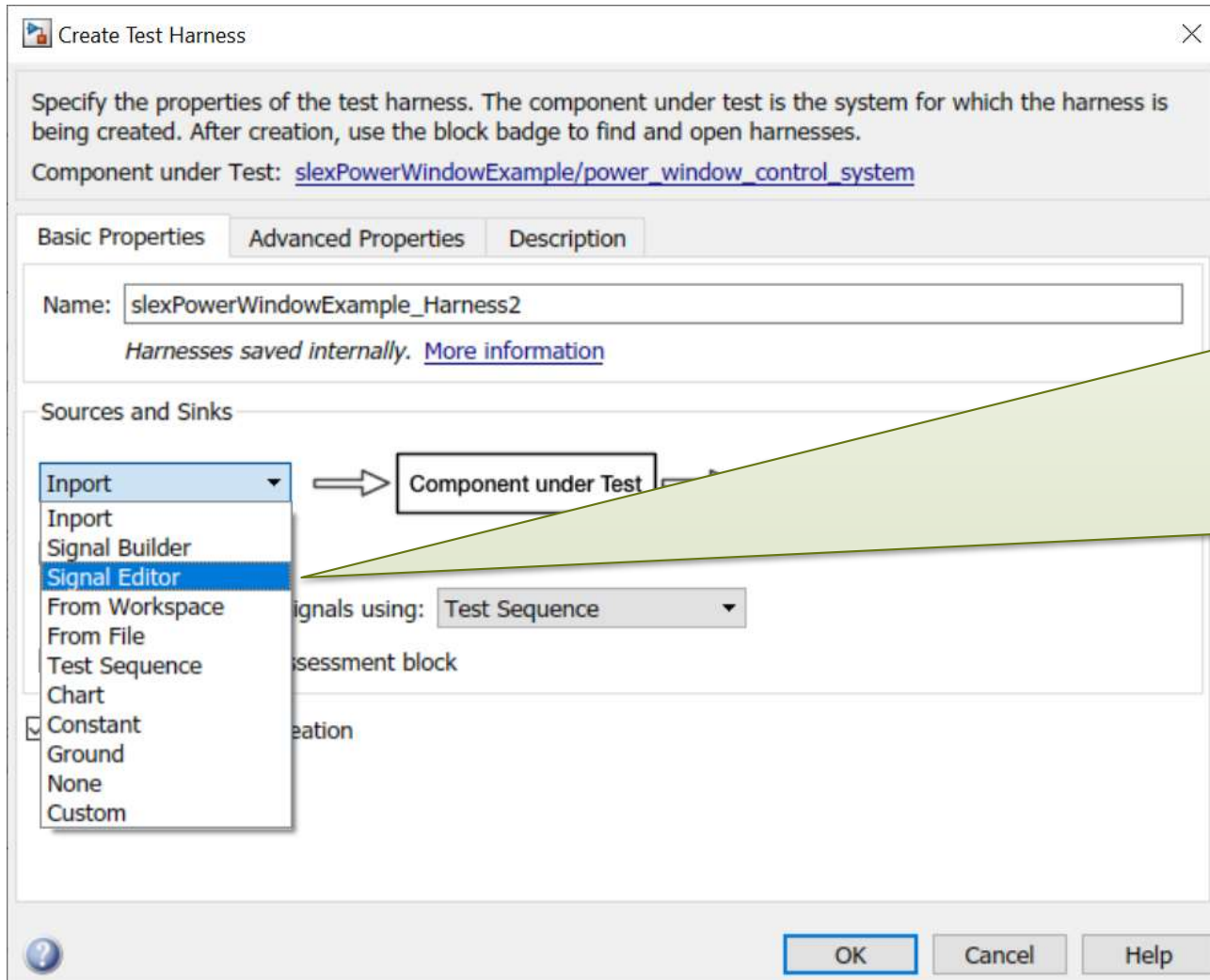
Generate function-call signals using:

Add separate Test Assessment block

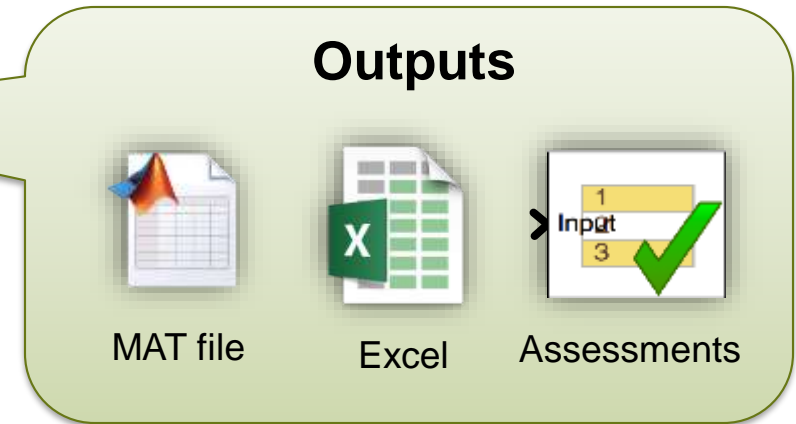
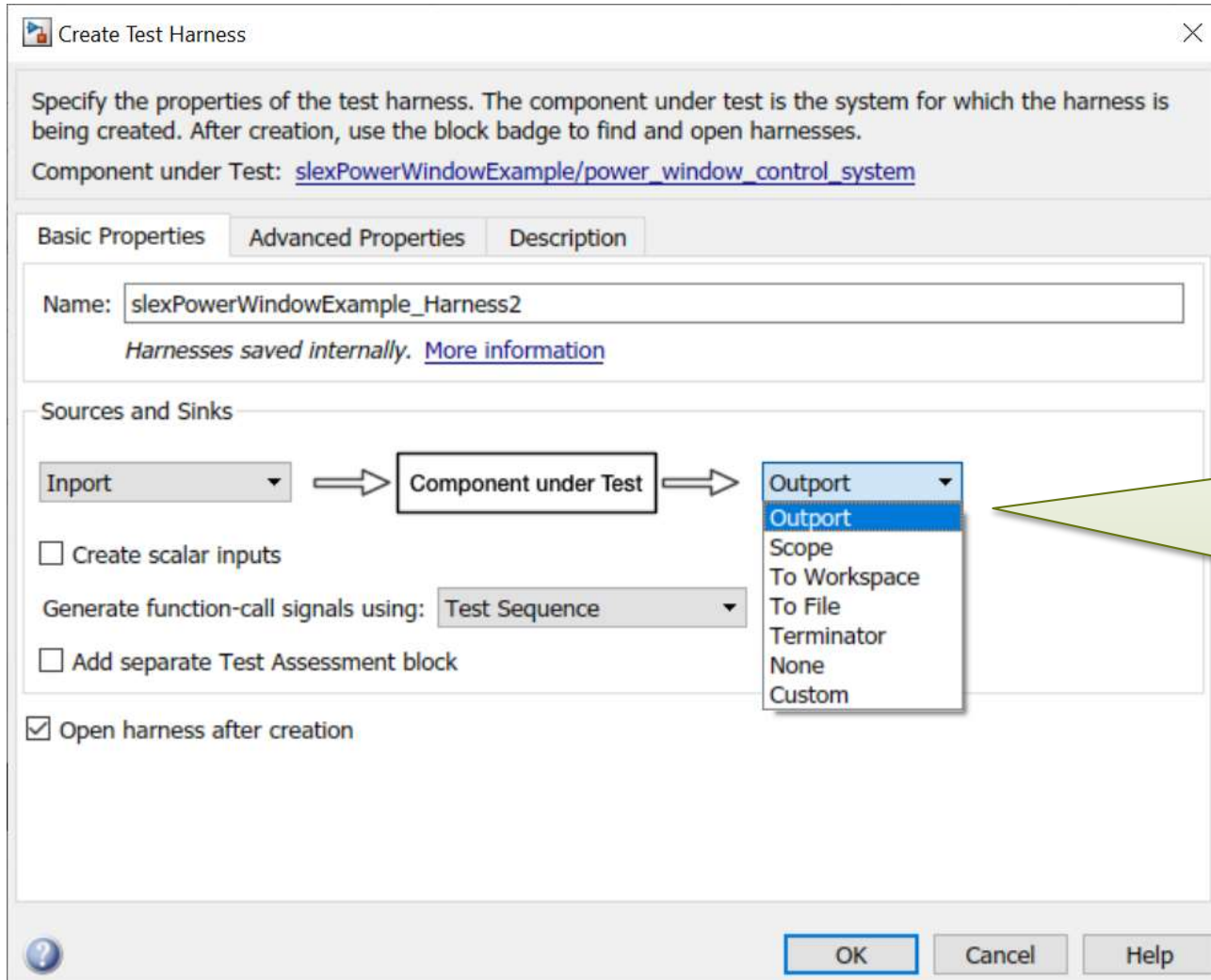
Open harness after creation

? OK Cancel Help

# 定义输入

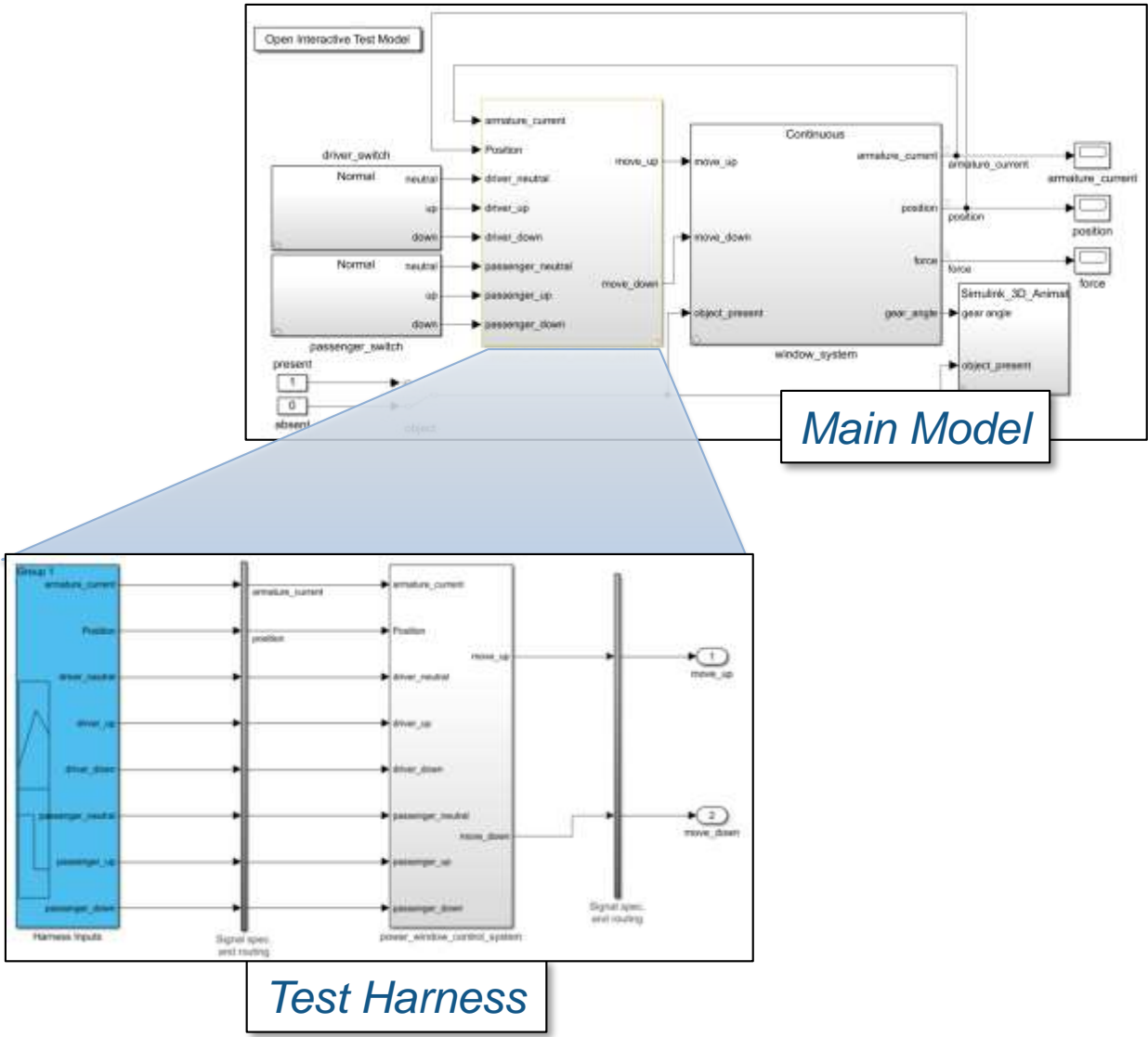


# 定义输出

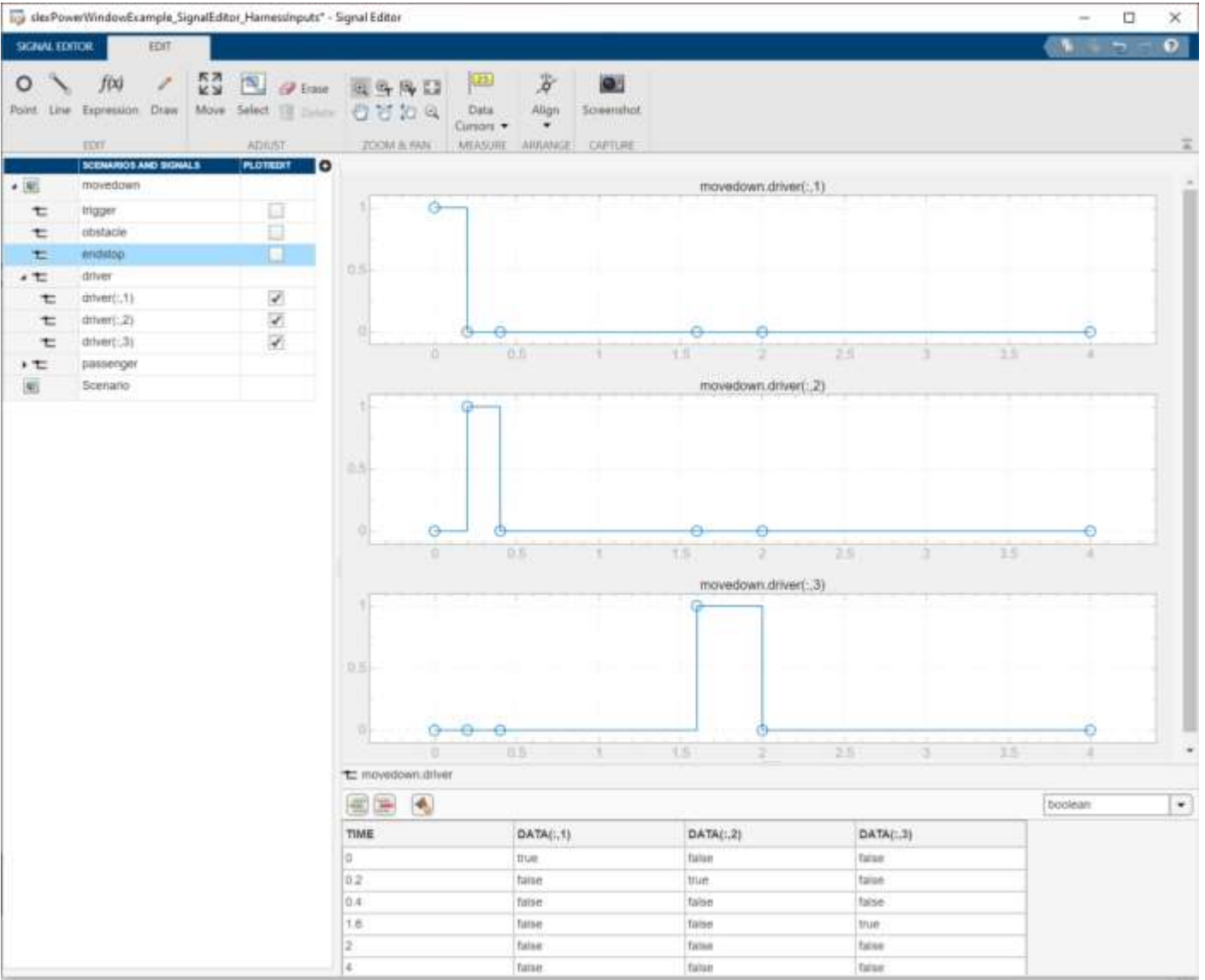




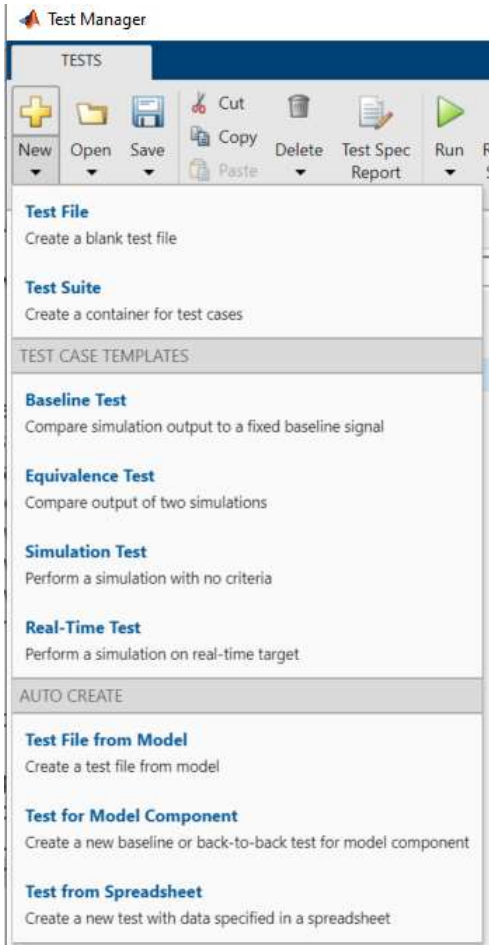
# 创建测试框架以识别待测试组件



# 利用Signal Editor编写测试输入

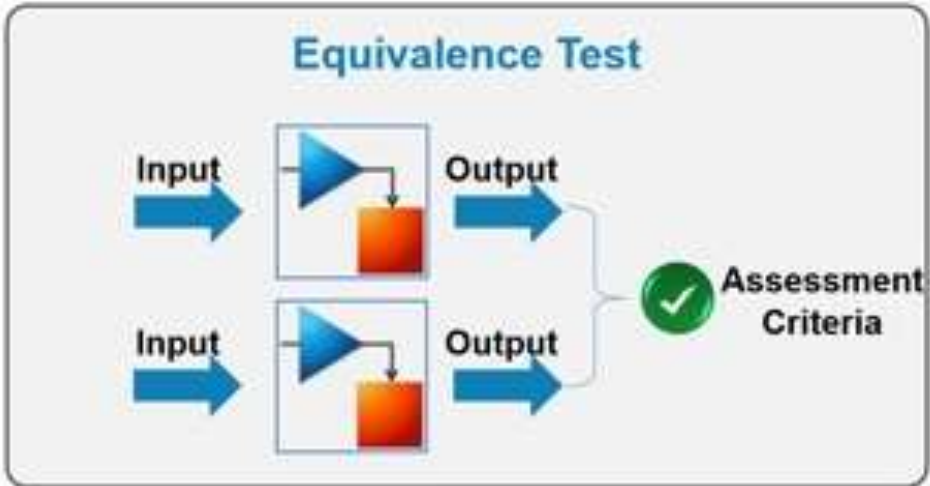
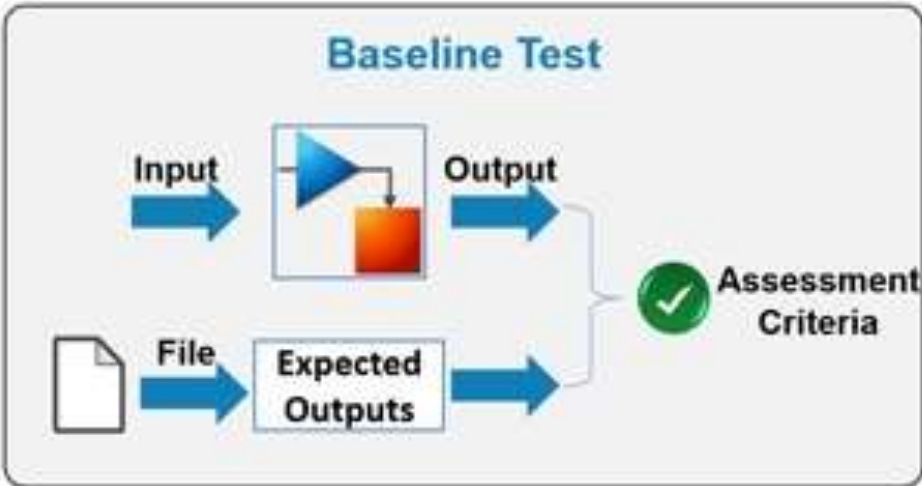


# 使用模板和向导（wizards）自动创建测试用例

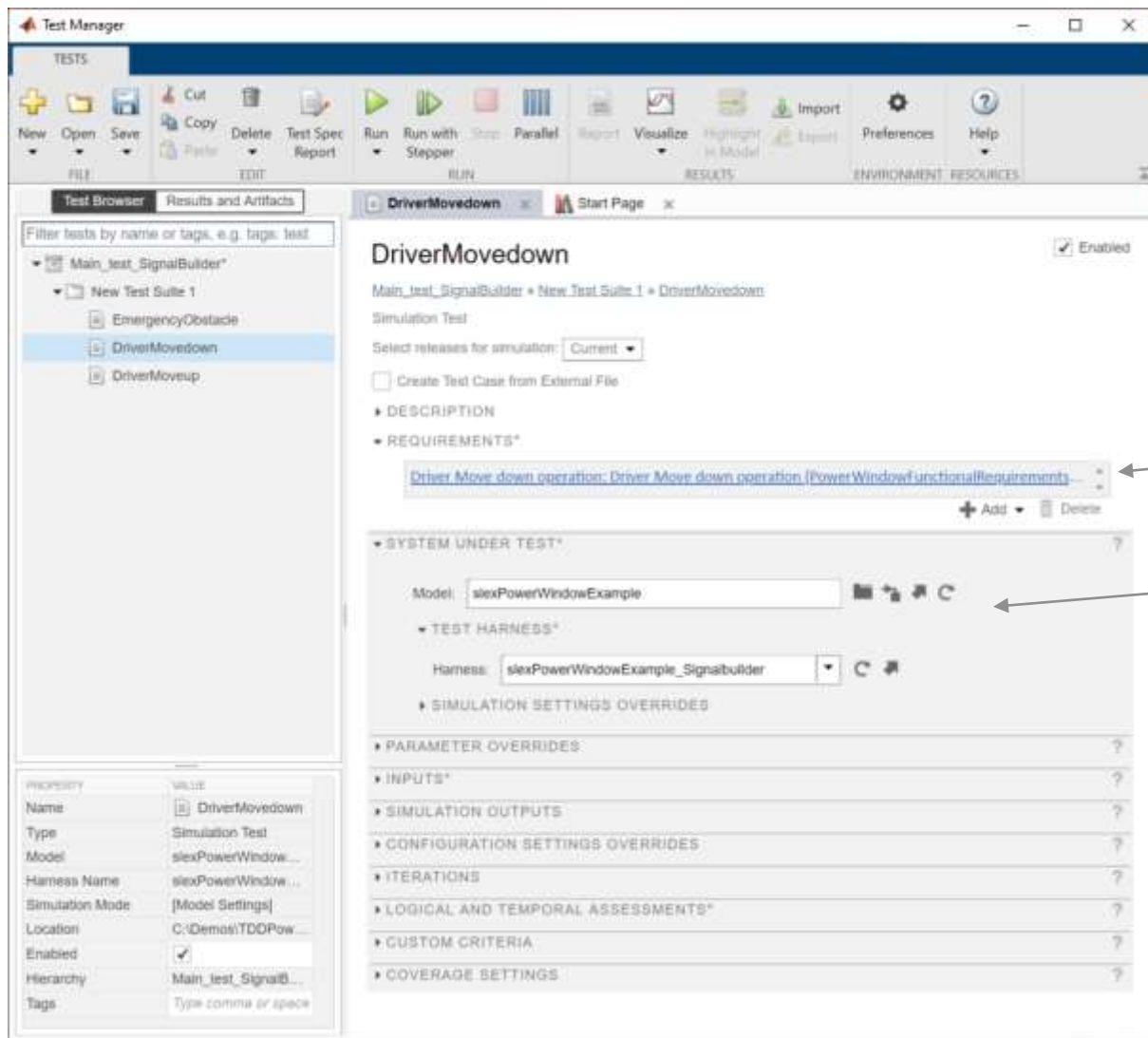


# 使用模板和向导 (wizards) 自动创建测试用例

## Test Case Templates



# 创建测试用例并关联到需求



关联到需求

定义要测试的模型

# 编译错误导致的测试失败

The screenshot shows the MATLAB Test Manager interface. The 'TESTS' tab is active, displaying a list of test results. The test case 'DriverMoveDown' is selected, showing a status of 'Failed' (red circle with a white exclamation mark). The 'Cause of Failure' is 'Errors running test case'. The 'ERRORS' section shows a red message: 'Signal Editor scenario 'movedown' not found in model slxPowerWindowExample\_SignalEditor.'

NAME	STATUS
Results: 2020-Mar-22 21:24:44	1
DriverMoveDown	
Logical and Temporal Assessm	

PROPERTY	VALUE
Name	DriverMoveDown
Status	1
Start Time	03/22/2020 21:24:48
End Time	03/22/2020 21:24:48
Type	Simulation Test
Test File Location	C:\Demos\TDDPowerWind...
Test Case Definition	

PROPERTY	VALUE
Name	DriverMoveDown
Outcome	1
Start Time	03/22/2020 21:24:48
End Time	03/22/2020 21:24:48
Type	Simulation Test
Test File Location	C:\Demos\TDDPowerWindow311\test\Mal...
Test Case Definition	
Rerun Test Case	
Tags	
Cause of Failure	Errors running test case

ERRORS

- Signal Editor scenario 'movedown' not found in model slxPowerWindowExample\_SignalEditor.

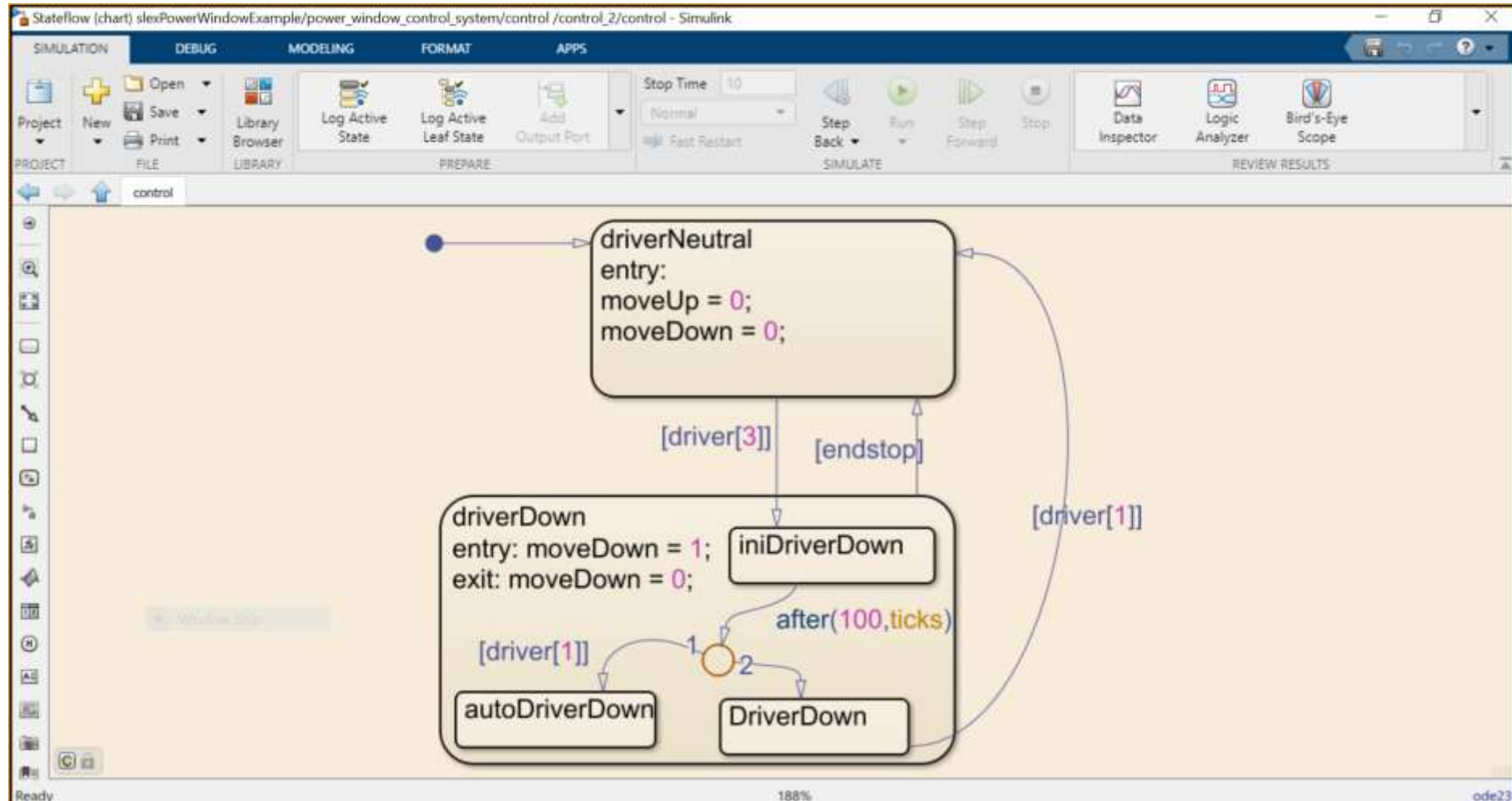
1. 创建测试

2. **实施最少的设计以通过测试**

3. 重构



# 给出仅足以通过测试的实现



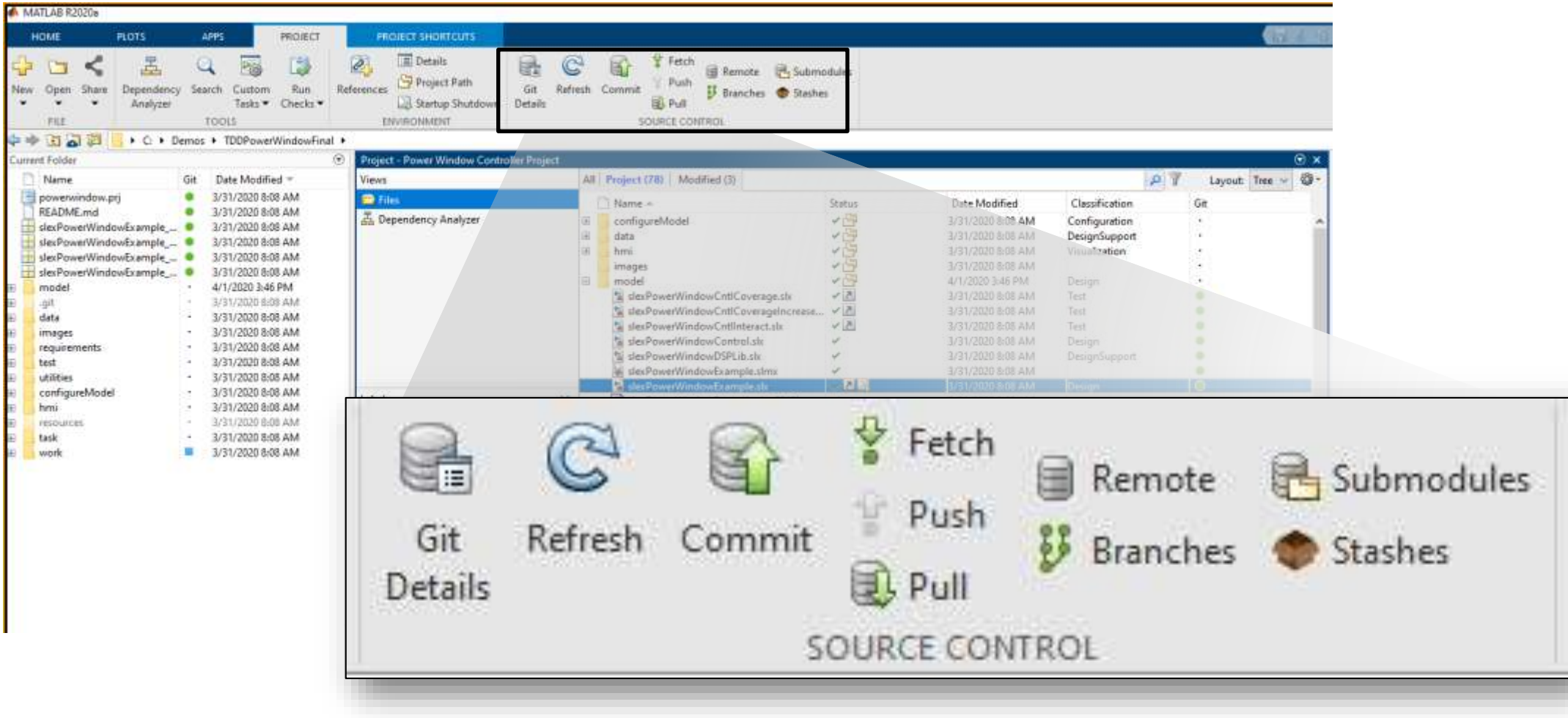


# 将实现关联到需求

The screenshot shows the Simulink Requirements Editor interface. The main workspace displays a stateflow chart with states like 'DriverUp' and 'autoDriverUp'. A callout box labeled '1.1.1: Fully Open' with the text 'If the up command is issued for between 200 ms and 1 s, the window must fully open.' has an arrow pointing to the 'autoDriverUp' state, labeled 'IMPLEMENTS'. The right-hand side shows the 'Property Inspector' for requirement 1.1.1, with fields for Index (1.2.1), Custom ID (1.1.1), and Summary (Fully Open). The bottom panel shows a 'Requirements' table with columns for Index, Summary, Implemented, and Verified.

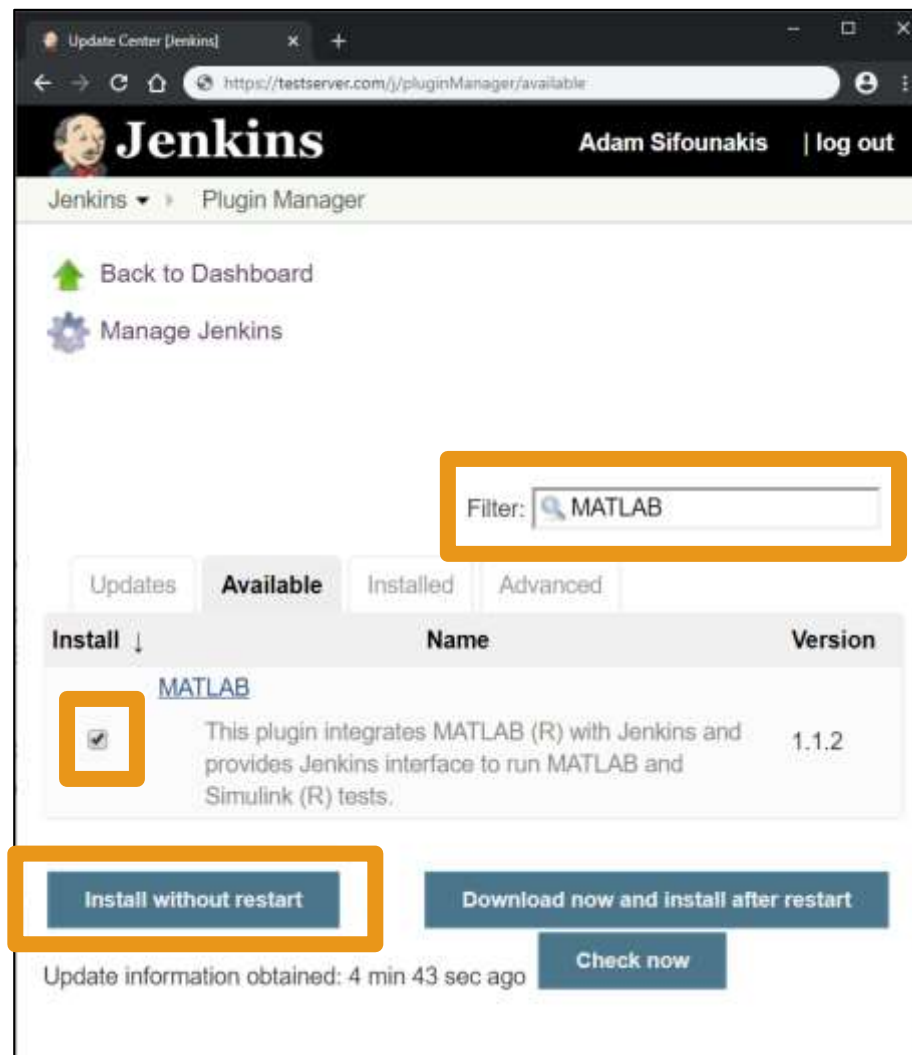
Index	Summary	Implemented	Verified
1.2	Open and Close Window	<input checked="" type="checkbox"/>	<input type="checkbox"/>
1.2.1	Fully Open	<input checked="" type="checkbox"/>	<input type="checkbox"/>
1.2.2	Fully Close	<input checked="" type="checkbox"/>	<input type="checkbox"/>

# 使用源代码管理直接从项目（Project）中管理工作件



# 利用持续集成扩展测试并自动化测试

- 规划自动化的代码和模型测试
- 在Jenkins中安装MATLAB插件



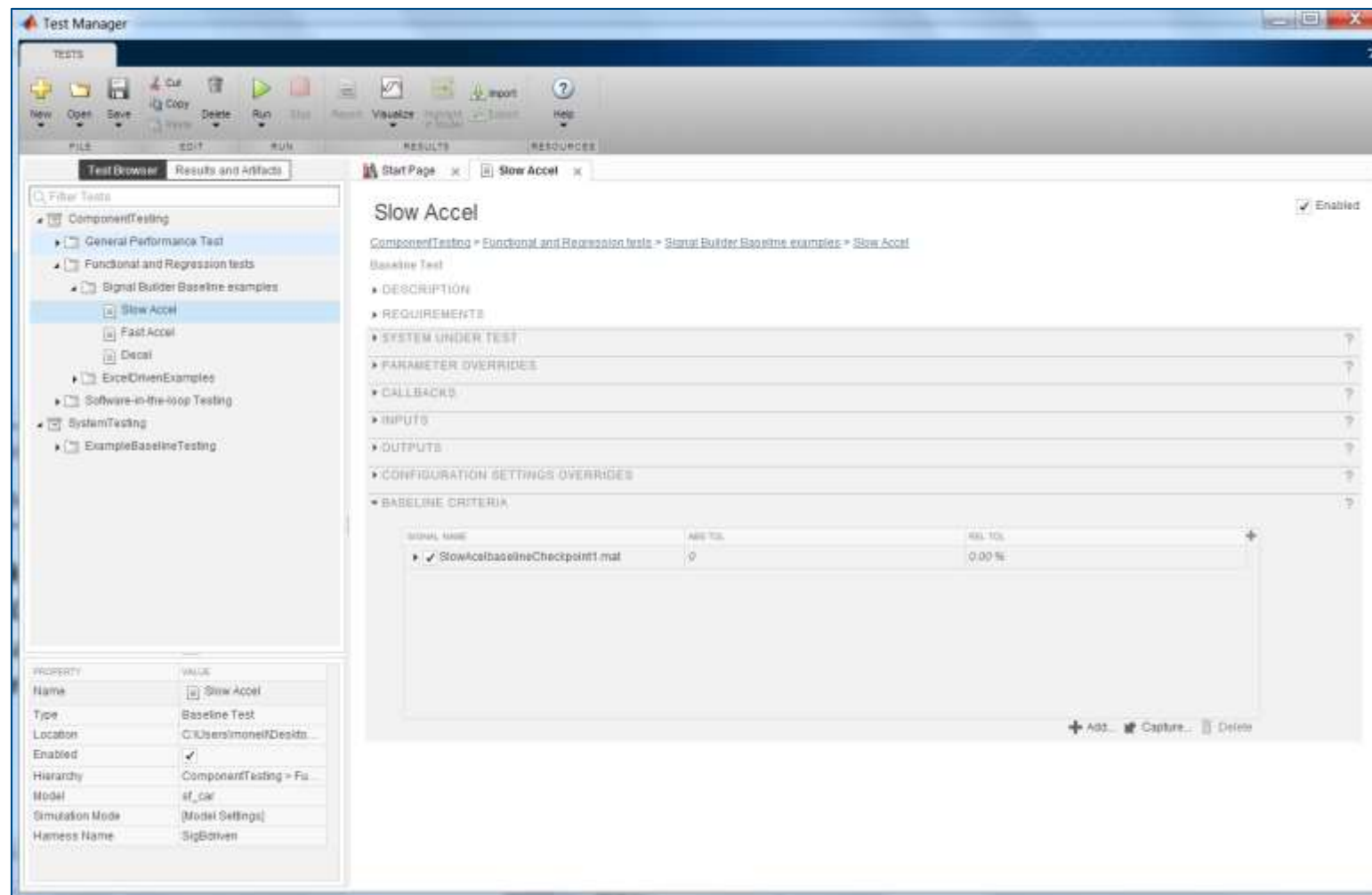
The screenshot shows the Jenkins Plugin Manager interface. The browser address bar indicates the URL is `https://testserver.com/j/pluginManager/available`. The Jenkins logo and user name "Adam Sifounakis" are visible at the top. The "Plugin Manager" section includes a search filter set to "MATLAB" and a table of available plugins. The "MATLAB" plugin is highlighted, and the "Install without restart" button is selected.

Install	Name	Version
<input checked="" type="checkbox"/>	MATLAB This plugin integrates MATLAB (R) with Jenkins and provides Jenkins interface to run MATLAB and Simulink (R) tests.	1.1.2



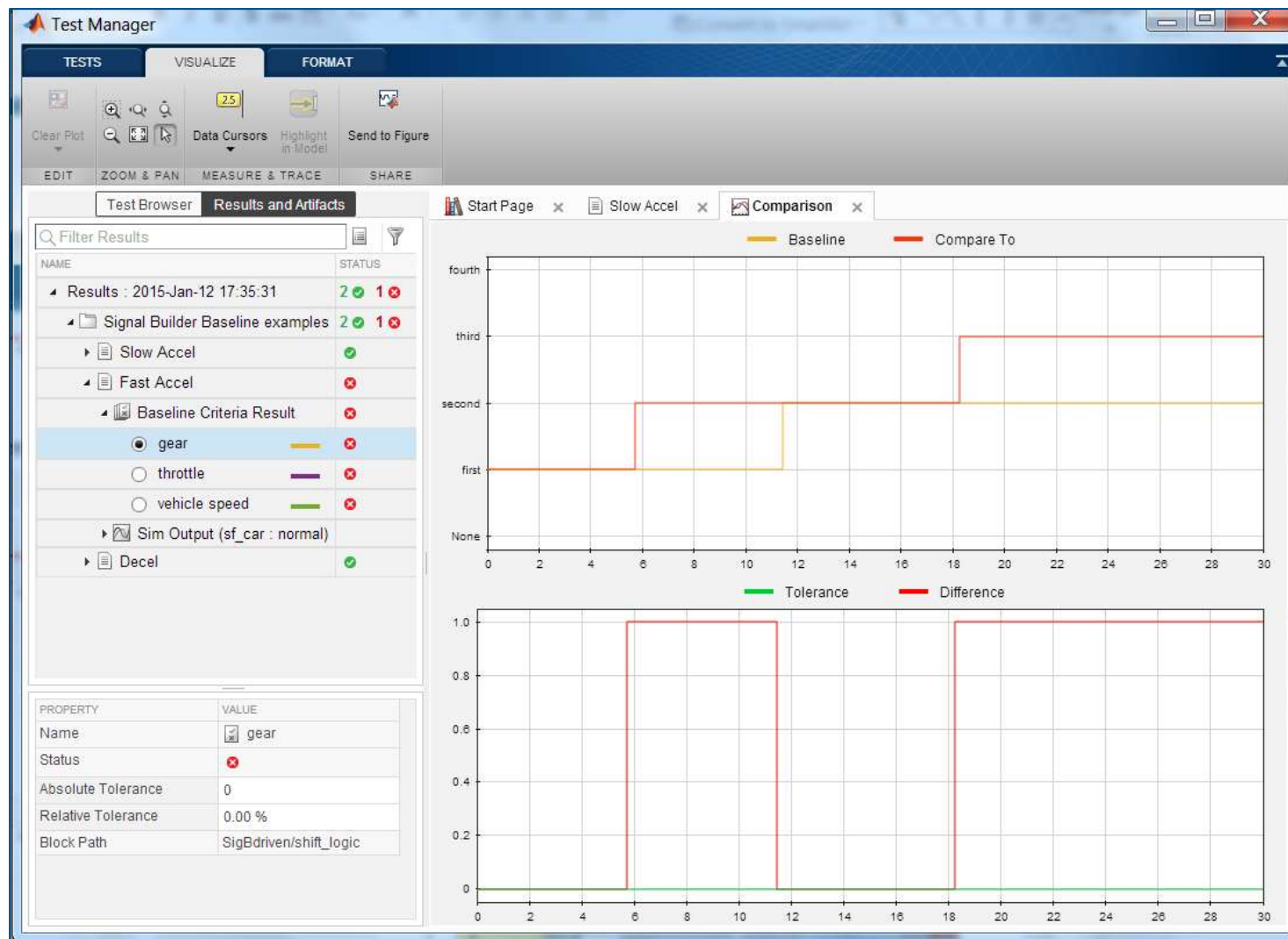
# 利用Test Manager执行测试

- 测试用例与文件分组
- 单独或批量执行

















# 利用Test Manager分析和调试测试结果

- 结果摘要查看
- 使用Simulation Data Inspector排查调试
- 将结果归档，导出，或生成报告



# 执行所有测试直至全部通过

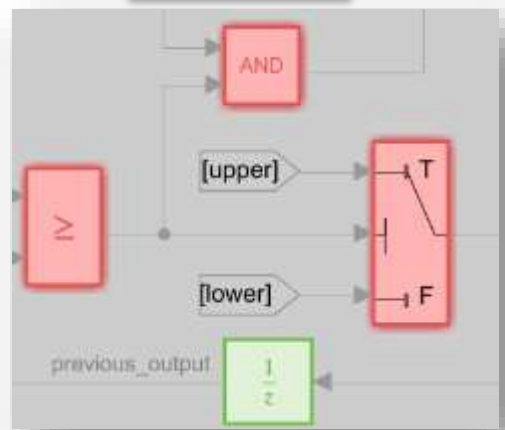
▼  PowerWindowControlUnitTest_sig 6 	
▶  DriverMoveDown	
▶  DriverMoveUp	
▶  EmergencyObstacle	
▶  Detect Object	
▶  Fully Close Window	
▶  PassengerMovedown	



# 使用覆盖度衡量测试完备性

- 测试缺陷
- 需求缺失
- 非预期功能
- 设计错误

Simulink



Stateflow



Code

A screenshot of a code editor showing a C++ function. The code is annotated with coverage information. A red arrow points to a line of code with the annotation "Coverage annotation". Another red arrow points to a line of code with the annotation "Links to model element". A tooltip is visible over a line of code, displaying "Tooltip with code coverage results". The code includes comments and function definitions.

# 生成测试报告以用于评审

**Create Test Result Report**

**Title Page Information**

Title:

Author:

Include MATLAB version

**Include in Report**

Results for:

Test requirements  
 MATLAB figures  
 Error and log messages  
 Simulation metadata  
 Coverage results  
 Plots of criteria and assessments  
 Plots for simulation output and baseline

2 rows x 1 columns of plots per page

**Output Options**

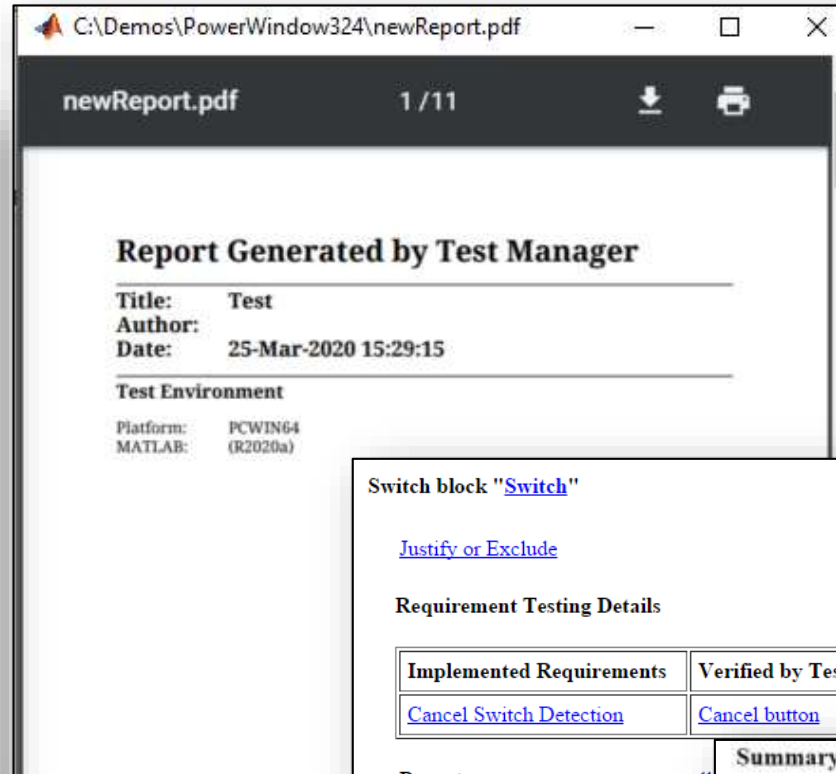
File Format:

File Name:

**Customization**

Template File:

Report Class:



Switch block "[Switch](#)"

[Justify or Exclude](#)

**Requirement Testing Details**

Implemented Requirements	Verified by Tests	Runs
<a href="#">Cancel Switch Detection</a>	<a href="#">Cancel button</a>	<a href="#">U1.2</a>

Parent: [crs\\_controll](#)

**Metric**

Cyclomatic Complexity	1	100%
Decision		100%
Execution		100%

**Summary**

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



1. 创建测试

2. 实施最少的设计以通过测试

3. **重构**

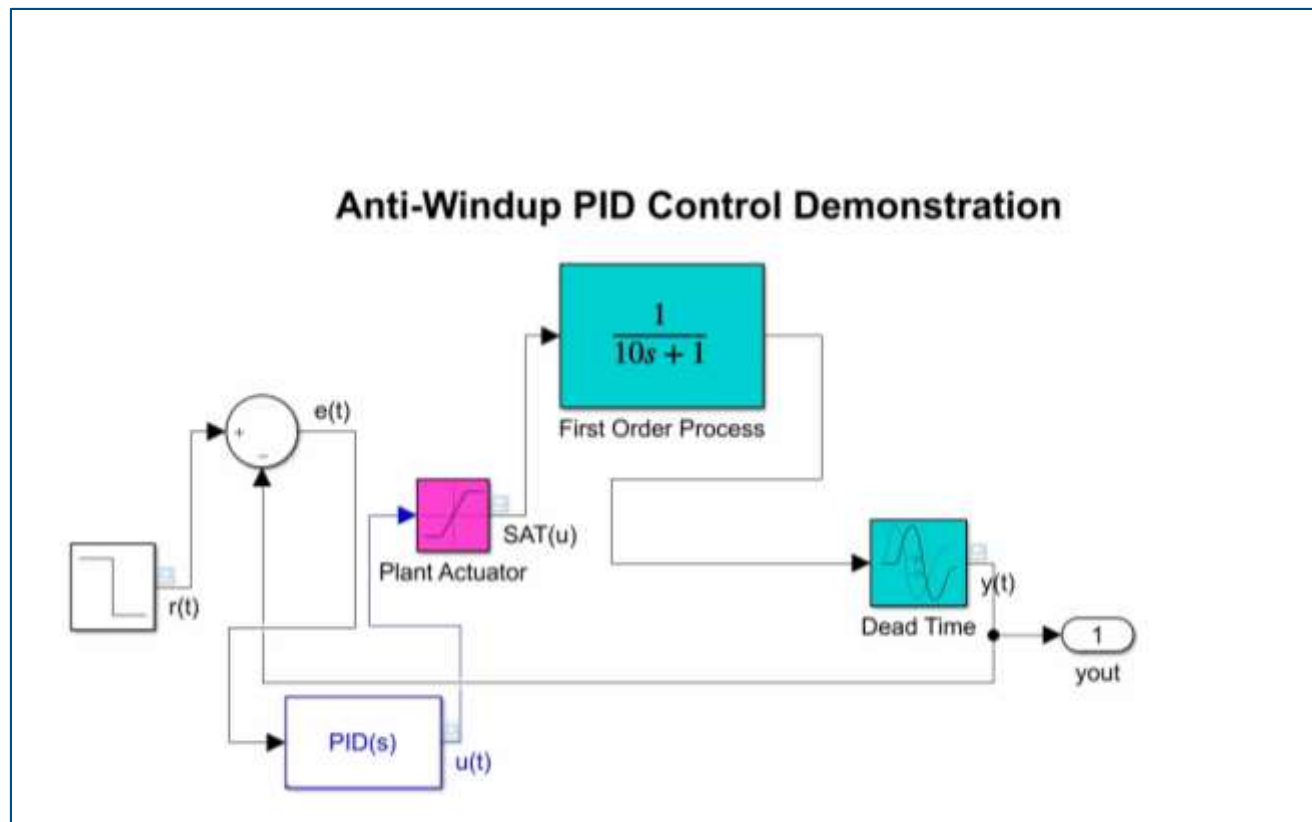


# 重构

- 重构是一种仅改善代码内部结构而不改变其外部行为的软件更改过程

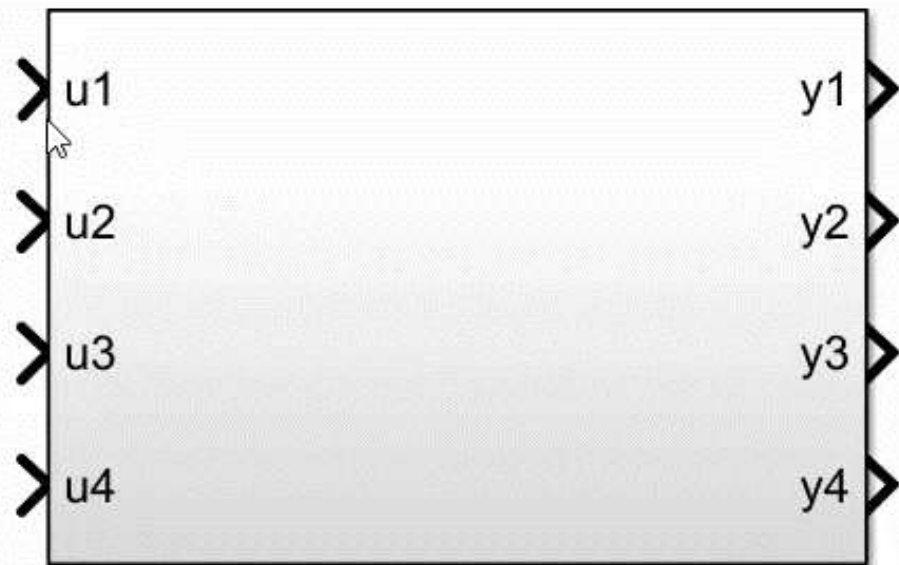
# 多种方式方法实现重构

- 重排布局



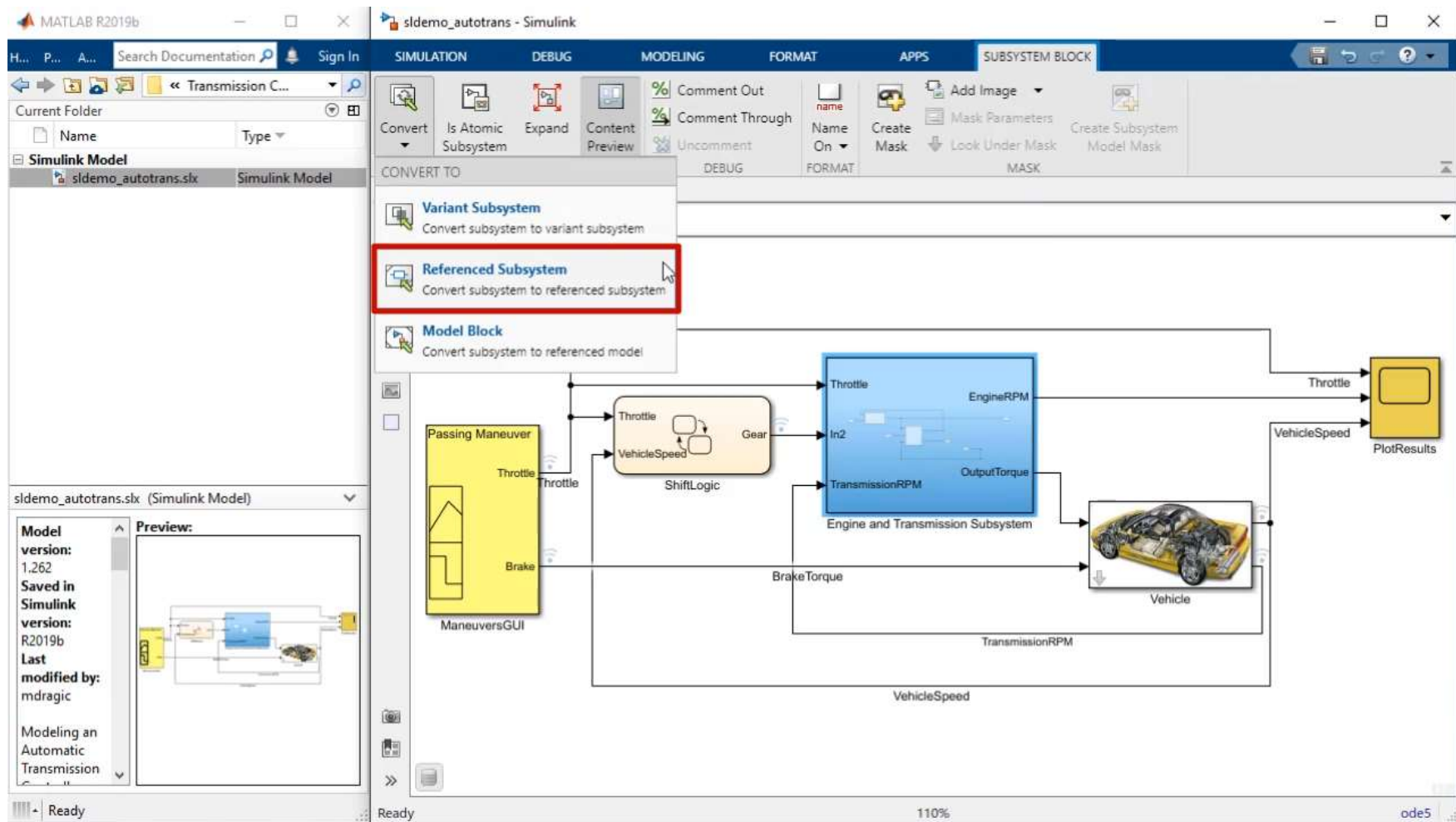
# 多种方式方法实现重构

- 重排布局



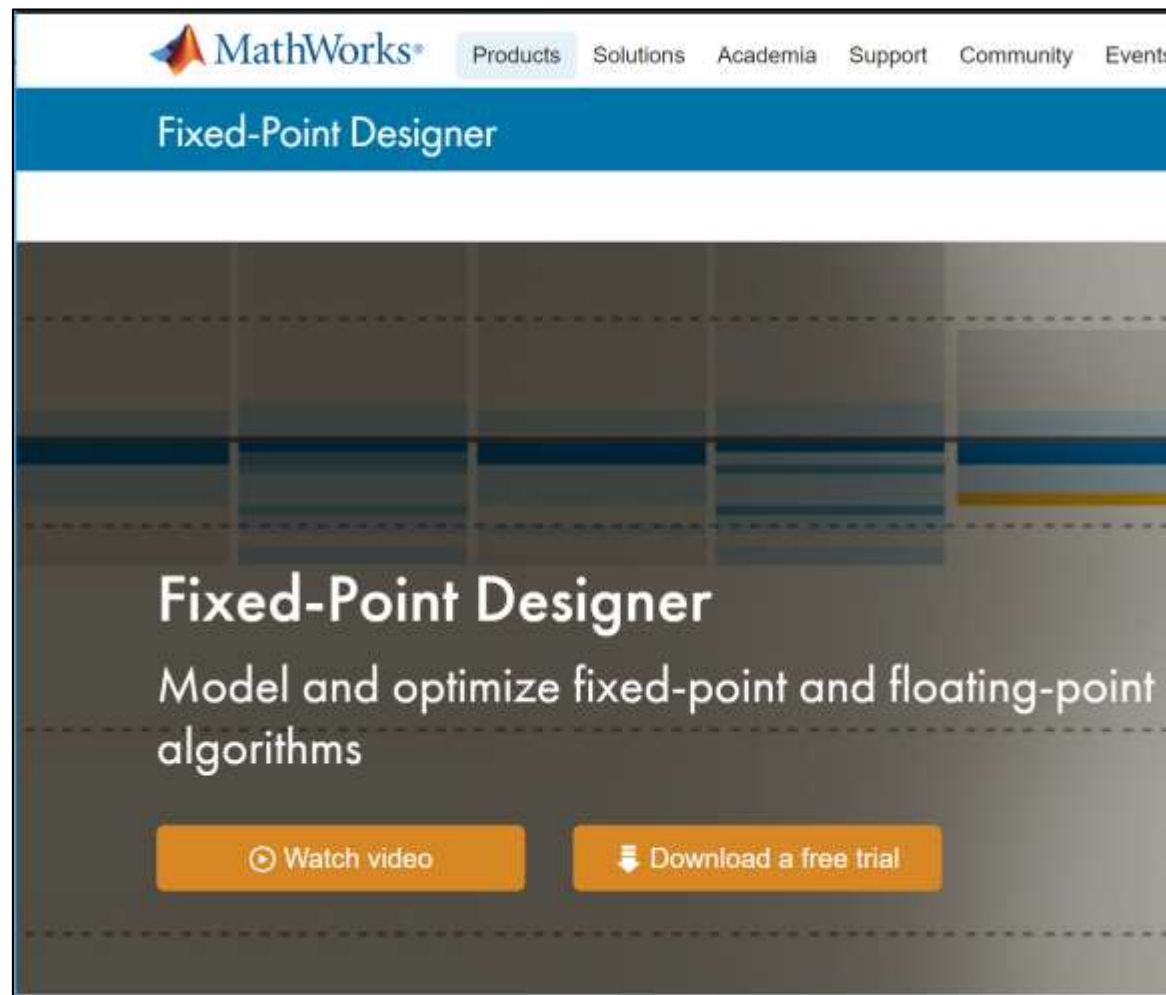
# 多种方式方法实现重构

- 重排布局
- 重构层次



# 多种方式方法实现重构

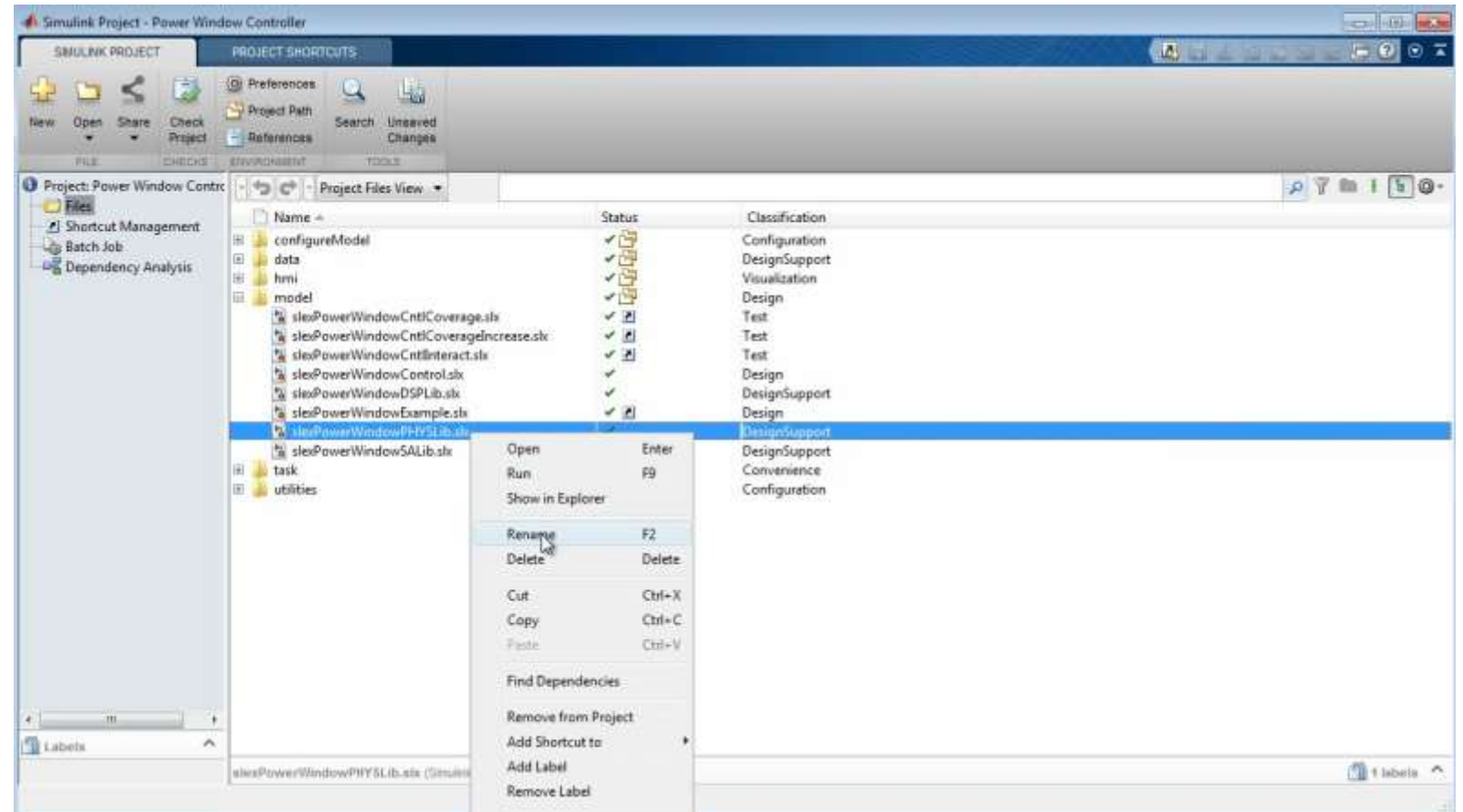
- 重排布局
- 重构层次
- 优化实现



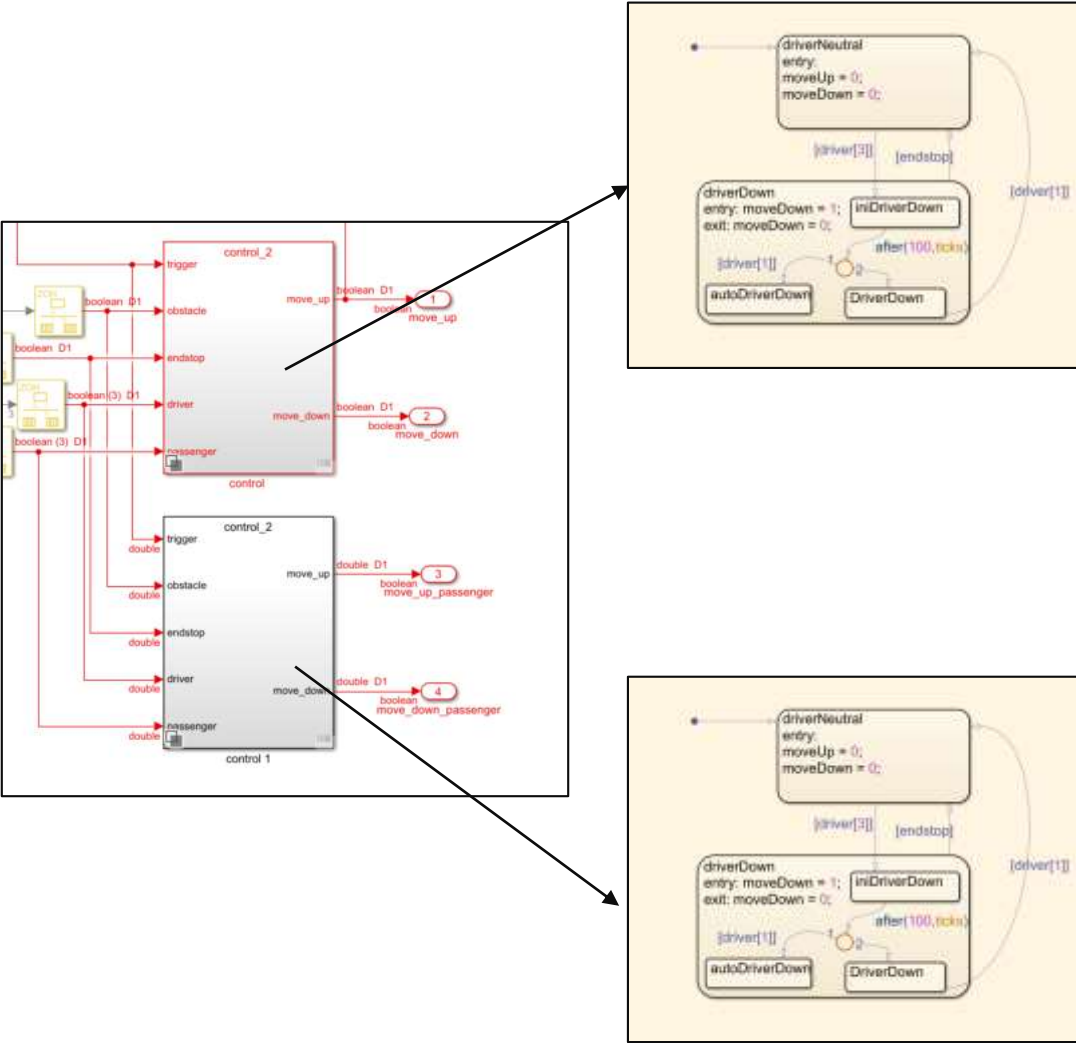
# 多种方式方法实现重构

- 重排布局
- 重构层次
- 优化实现
- 工程项目范围内的重命名

.... 更多!



# 通过整合冗余的Stateflow Chart实现重构



驾驶员和乘客的控制逻辑一致



# 使用Clone Detector App检查模型克隆（冗余）

The screenshot displays the Simulink Clone Detector application. The main workspace shows a Simulink model with several blocks highlighted in red, indicating detected clones. A 'CLONE DETECTOR' dialog box is open, showing options to 'Replace Clones' and 'Check Equivalency'. A tooltip for 'Replace Clones' explains: 'Create library blocks from clones. Copy model and replace its clones with library links'. Below the workspace, the 'Clone Detection Actions and Results' window is visible, showing a table of detected clone groups.

Group	Clones	Number of clones	Blocks Per Clone	Block Differences	Parameter Differences	Library Path to Replace Cl.	Similarity Plot
Exact Clone Group 1	2	24	0	0	0	newLibraryFile	[Plot]
Exact Clone Group 2	2	34	0	0	0	newLibraryFile	[Plot]
Exact Clone Group 3	2	1	0	0	0	newLibraryFile	[Plot]
Exact Clone Group 4	2	1	0	0	0	newLibraryFile	[Plot]
Similar Clone Group 1	4	17	1	1	1	newLibraryFile	[Plot]

# 测试驱动的开发流程

1. 创建测试
2. 实施最少的设计以通过测试
3. 重构



# 结论&要点



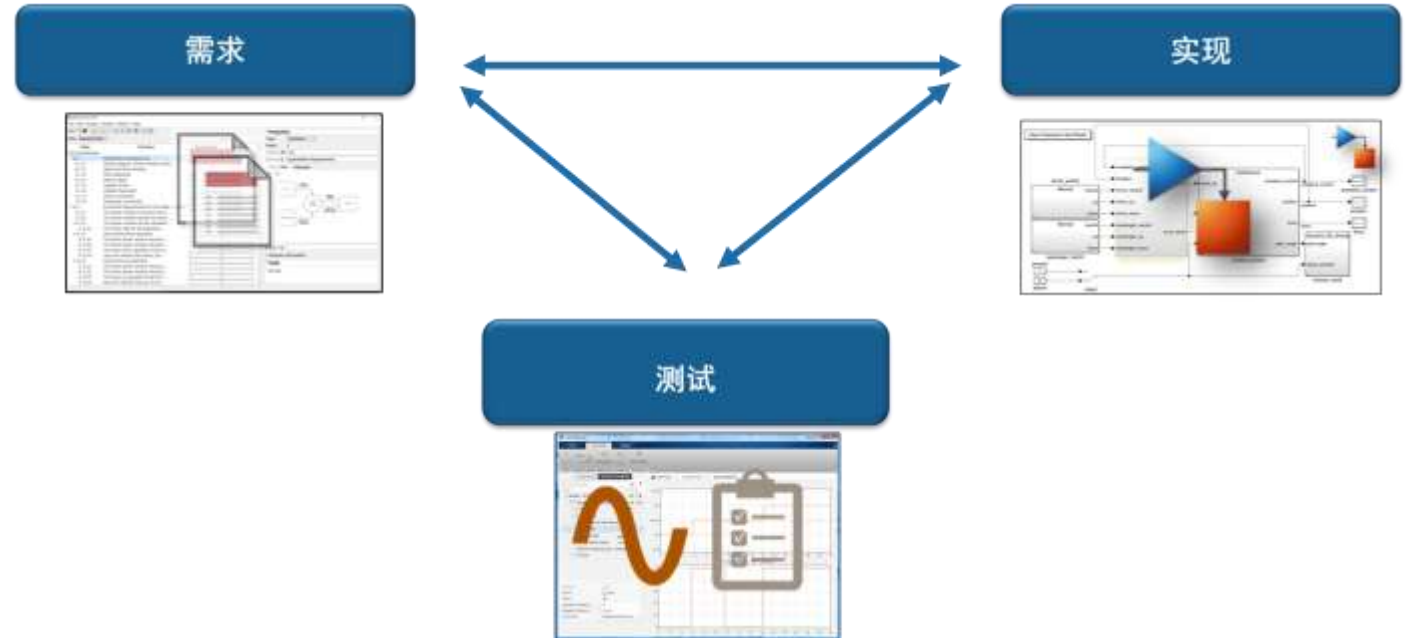
Simulink为TDD提供集成框架



系统性地验证需求



自动化测试以更快交付可用系统



# 测试驱动开发

## MATLAB和Simulink提供助力

- 基于模型的设计
- 需求管理
- 创建并执行测试
- 测试完备度评估
- 重构和合规性验证
- 持续集成
- 组织，管理和共享

- *Simulink and Stateflow*
- *Simulink Requirements*
- *Simulink Test*
- *Simulink Coverage*
- *Simulink Check*
- *MATLAB Plug in for Jenkins*
- *Projects*



# 了解更多

- [Agile System Development with Model-Based Design](#)
- [Agile Model-Based Design: Accelerating Simulink Simulations in Continuous Integration Workflows](#)
- [Verification, Validation, and Test Solution Page](#)
- [Continuous Integration Solution Page](#)