모델기반설계를 이용한 요구사항 기반 검증의 단순화

Application Engineer
홍 혁 기 부장
Key takeaways

- Verify and validate requirements earlier

- Identify inconsistencies in requirements by using unambiguous assessments

- Traceability from requirements to design and test
Challenge: Errors introduced early but found late

- Most errors introduced
- Unit test finds some errors
- Errors found during integration or in field
Cost of finding errors increases over time
Challenges with requirements based verification

- **Are all requirements implemented?**
- **Is the implementation functioning correctly?**
- **Is requirement interpreted correctly?**
- **How to avoid modifying the design for test?**

**Requirements** → **Specification** → **C/C++**/Hand code
Simulink models for specification

Model-Based Design enables:

- Early testing to increase confidence in your design
- Delivery of higher quality software throughout the workflow
Multiple languages to describe complex systems
Ad-Hoc Testing: Explore behavior and design alternatives

Requirements → Design Model → C/C++ → Hand code
Validate behavior earlier with simulation
Validate Behavior Earlier with Simulation
Complete Model Based Design

**Requirements** → **Design Model** → **Model used for production code generation** → **C/C++** → **Generated code**

**Simulink Models**

**Code Generation**
Systematically verify requirements

- Are all requirements implemented?
- Is the implementation functioning correctly?
- Are designs and requirements consistent?

Requirements Based Testing

Simulink Models

Requirements

Design Model

Model used for production code generation

C/C++

Generated code
Integrate with requirements tools and author requirements

- Import from:
  - Word / Excel
  - IBM® Rational® DOORS®
  - ReqIF™ standard
- Update synchronizes changes from source
- Edit and add further details to import
- Author requirements
- Export ReqIF
  - Enables roundtrip with external tools

Import
Update
Export

 Requirements Management Tools

External Requirements

Simulink Requirements

External Requirements

Authored Requirements
Roundtrip workflow with external tools thru ReqIF

- Import from:
  - Word / Excel
  - IBM® Rational® DOORS®
  - ReqIF™ standard
- Update synchronizes changes from source
- Edit and add further details to import
- Author requirements
- Export ReqIF
  - Enables roundtrip with external tools
Requirements Verification with Simulink

Requirements

- Transmission Operating Modes
- Reverse cannot be entered from drive
- Engine only starts in Park

Implemented By

Simulink / Stateflow

Verified By

Test Case

Inputs

- MAT / Excel file (input)
- Signal Editor

Assessments

- MAT / Excel File (baseline)
- Test Assessments

Test Harness

Simulink Test

MATLAB Unit Test
Requirements Verification with Simulink

**Requirements**

- **crs_req_func_spec**
  - 1. Driver Switch Request Handling
  - 1.1 Switch precedence
  - 1.2 Avoid repeating commands

**Test Case**

- **Inputs**
  - MAT / Excel file (input)
  - Signal Editor

- **Test Sequence**
  - Simulink Test

- **Test Assessments**
  - MATLAB Unit Test
  - Simulink / Stateflow

**Implemented By**

**Verified By**
### 1 Requirements for the basic Heatpump Controller

Temperature difference is defined as the difference between the room and the set temperature. The controller shall turn the fan on when the temperature difference has reached a certain level, to circulate the air. The controller shall turn the heatpump on when the temperature difference has reached another level, to heat or cool the space.

#### 1.1 Idle when Temperature in Range
If the temperature difference is less than 1 degrees, the system shall be idle with all signals off.

#### 1.2 Activate Fan
The fan shall activate when the temperature difference is greater than or equal to 1 degrees.

#### 1.3 Activate Heat Pump
The pump shall activate when the temperature difference is greater than or equal to 2 degrees for more than 2 seconds and stay active for at least 2 seconds.

##### 1.3.1 Cool Mode
If the room temperature is greater than the set temperature, the system shall cool the space.

##### 1.3.2 Heat Mode
If the room temperature is less than the set temperature, the system shall heat the space.

#### 1.4 Max Temperature Difference
The difference between the room temperature and the set temperature should never exceed 6 degrees.

### Requirements in DOORS
Example: Heat Pump Controller Implementation
Link requirements to implementation in model
Work with Model and Requirements with Requirements Perspective

- Requirement Annotations
- Badges
- Implementation and Verification Status
- Property Inspector

MATLAB EXPO 2019
## Track Implementation and Verification

<table>
<thead>
<tr>
<th>Index</th>
<th>ID</th>
<th>Summary</th>
<th>Implemented</th>
<th>Verified</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>Driver Switch Request Handling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1</td>
<td>1</td>
<td>Switch precedence</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>2</td>
<td>Avoid repeating commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td>3</td>
<td>Long Switch recognition</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td>4</td>
<td>Cancel Switch Detection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5</td>
<td>5</td>
<td>Set Switch Detection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.6</td>
<td>6</td>
<td>Enable Switch Detection</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Implementation Status**
- Implemented
- Justified
- Missing

**Verification Status**
- Passed
- Failed
- Unexecuted
- Missing

---

**MathWorks**

MATLAB EXPO 2019
Isolate Component Under Test with Test Harness

House Heating System
1. Plot temperature of wall, window, and roof (see code)
2. Plot heat flow through wall, window, and roof (see code)
3. Explore simulation results using sscexplore
4. Learn more about this example
Test Sequence Block: Step-based and temporal test sequences
Activate Heat Pump

If the temperature difference exceeds 2 degrees for more than 2 seconds, then the pump shall activate for at least 2 seconds.

\[(|x_1 - x_2| \geq x_3)^\varepsilon \land \Box_{[0,t_1]}(|x_1 - x_2| \geq x_3) \rightarrow \Box_{[0,t_2]} x_4\]
Translate textual requirements into unambiguous Temporal Assessments

• Compose assessments using form based editor

• View assessments as English-like sentence

• Link to requirements

• Review and debug temporal assessment results
Translate textual requirements into unambiguous Temporal Assessments

- Compose assessments using form based editor
- View assessments as English-like sentence
- Link to requirements
- Review and debug temporal assessment results

MATLAB EXPO 2019
Translate textual requirements into unambiguous Temporal Assessments

- Compose assessments using form based editor
- View assessments as English-like sentence
- Link to requirements
- Review and debug temporal assessment results

**Temporal Assessment Editor**
Translate textual requirements into unambiguous Temporal Assessments

- Compose assessments using form based editor
- View assessments as English-like sentence
- Link to requirements
- Review and debug temporal assessment results

MATLAB EXPO 2019
Translate textual requirements into unambiguous Temporal Assessments

• Compose assessments using form based editor
• View assessments as English-like sentence
• Link to requirements
• Review and debug temporal assessment results
Translate textual requirements into unambiguous Temporal Assessments

- Compose assessments using form based editor
- View assessments as English-like sentence
- Link to requirements
- Review and debug temporal assessment results
Demo: Temporal Assessments
Execute assessments to verify requirements
Locate implementation of requirement using link
Observers: Separate test/verification logic from design

- Access nested signals without signal lines or changing dynamic response
- Avoid modifying interface for testing
- Simplify design and test by avoiding additional signal lines
Observers: Separate test/verification logic from design

- Access nested signals without signal lines or changing dynamic response
- Avoid modifying interface for testing
- Simplify design and test by avoiding additional signal lines
Demo: Observers
Summary

- Verify and validate requirements earlier
- Identify inconsistencies in requirements by using unambiguous assessments
- Traceability from requirements to design and test
Learn More

Key products covered in this presentation:

- Simulink Requirements
- Simulink Test
- Simulink Real-Time

Learn more at Verification, Validation and Test Solution Page:
mathworks.com/solutions/verification-validation-validation.html