Process Pitfalls in ISO 26262 Compliance

MathWorks Consulting
Jason Moore
Functional safety industry trend

Driver assistance is a priority for major auto players

Mentions of “advanced driver assistance,” “ADAS,” and “safety,” on automaker and Tier-1 supplier earnings calls

- Internal IEC Certification Kit data (Normalized)
  - Support for ISO 26262

Growing interest in safety

ISO 26262

Is your organization ready for ISO?
Multifaceted support for ISO 26262
Multifaceted support for ISO 26262

- IEC Certification Kit
  - Model-Based Design Reference Workflow
  - Tool Qualification Package
    - Software Tool Criteria Evaluation Report
    - Software Tool Qualification
    - Tool Validation Suite
    - …etc.
Multifaceted support for ISO 26262

- **Targeted Features**
  - Model Metrics Dashboard
  - Model Testing Dashboard

Design Compliance

Verification Compliance

Tools

Reference Examples

Consulting Services
Multifaceted support for ISO 26262

- **Best Practice Paper**
  - (2018) *Model Quality Objectives*
    - Recommended model metric and threshold
    - How to achieve Freedom from Interference?
  - (2020) *An ISO 26262 Workflow for Automated Driving Applications Using MATLAB: Guidelines and Best Practices*
    - Use of MATLAB as part of ISO 26262 workflow
Multifaceted support for ISO 26262

- Reference Application

Architecture Design with System Composer

ISO 26262-6 Workflow Example

Component/Unit Design with Simulink
Multifaceted support for ISO 26262

Range of Consulting Services

- ISO Jumpstart
- Process Establishment
- Tool Qualification Support
- Process Gap Analysis
- Model Review

Consulting Services

Tools

Reference Examples

Create qualification artifacts
- Tool requirement
- User manual
- Test cases
- Expected results
- Traceability matrix
- TCL classification
- Reference workflow
- …etc.

Table 8 – Methods for deriving test cases for software and testing

<table>
<thead>
<tr>
<th>Methods</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection of requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specification and analysis of requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis of test cases</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design of test cases based on requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Observations based on our work with industry

Common pitfalls

- Unaware of ISO requirements
- Legacy components developed outside of ISO
- No clear mapping of ISO requirement to workflow
- Lack of tool implementation methods against ISO requirement
- No architecture consideration
- HIL-centric verification workflow
- No justification on method selections
- No clear definition of required work product
- Lack of consistency in work product
- No upfront consideration to tool qualification
- Lack of coordination between functional safety and software development
- Underestimate the effort (cost and timing) required for ISO project
- …etc.
Observations based on our work with industry

Common Themes

- Process not clearly defined or documented
- Lack top-down architectural design approach
- Poor tool qualification awareness
Process not clearly defined or documented

ISO 26262-6
• 15 Tables
• 90 Topics/Methods/Principles

- Which topics/method/principles were chosen?
- What justification were used?
- What evidence were captured?
- What are the implementation steps?
- …etc.?
Process not clearly defined or documented
Define process: from ISO Requirement down to Detail Work Instructions
Process not clearly defined or documented

Define process: from ISO Requirement down to Detail Work Instructions

ISO recommendations

Assessment Principle

Define ➔ Execute ➔ Archive

Mapping to Engineering Task

Decision to follow recommendation

Table 7 – Methods for software unit verification

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL D</th>
<th>Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a Walk-through</td>
<td>0</td>
<td>No*</td>
</tr>
<tr>
<td>1b Pre-programming</td>
<td>+</td>
<td>No*</td>
</tr>
<tr>
<td>1c Inspection</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1d Semi-formal verification</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1e Formal verification</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1f Control flow analysis</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1g Data flow analysis</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1h Static code analysis</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1i Static analysis based on abstract interpretation</td>
<td>=</td>
<td>No</td>
</tr>
<tr>
<td>1j Requirement-based test</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1k Interface test</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1l Fault injection test</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1m Resource usage evaluation</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1n Back-to-back comparison between model and code, if applicable</td>
<td>++</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 8 – Methods for deriving test cases for software unit testing

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL D</th>
<th>Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a Analysis of requirements</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1b Generation and analysis of equivalence classes</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1c Analysis of boundary values</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1d Error guessing based on knowledge or experience</td>
<td>++</td>
<td>No*</td>
</tr>
</tbody>
</table>

Table 9 – Structural coverage metrics at the software unit level

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL D</th>
<th>Applicable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a Statement coverage</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1b Branch coverage</td>
<td>++</td>
<td>Yes</td>
</tr>
<tr>
<td>1c MC/DC (Condition / Decision Coverage)</td>
<td>++</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Derive Consistent Work Instructions
Process not clearly defined or documented
Define process: from ISO Requirement down to Detail Work Instructions

Assessment
Principle

Define → Execute → Archive

Model Metric Dashboard
Process not clearly defined or documented
Define process: from ISO Requirement down to Detail Work Instructions
Process not clearly defined or documented

Define process: from ISO Requirement down to Detail Work Instructions

Assessment Principle

Define → Execute → Archive

Coverage Results

Test Results

Verification Report

Verification Plots

Simulink Test
(Test Manager)
Lack top-down architectural design approach

- “Bottom up” (legacy) vs “top down” (functional safety) approach

- Concepts:
  - Static and Dynamic architecture description (ISO 26262-6:2018 Clause 7.4.5)
  - Criteria for coexistence of elements (ISO 26262-9: 2018 Clause 6)
  - Safety-Oriented analysis (ISO 26262-9:2018 Clause 8)
  - Analysis of dependent failures (ISO 26262-9:2018 Clause 7)
  - Software partitioning using Freedom From Interference (ISO 26262-6:2018 Annex D)
Lack top-down architectural design approach
Perform architectural review – Freedom From Interference

- ISO 26262-6 (Annex D)
  - Timing and execution
  - Memory
  - Exchange of information

<table>
<thead>
<tr>
<th>Model architecture</th>
<th>Signal routing and definition</th>
<th>Code generation configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Use model reference for unit-level models</td>
<td>• Group bus signals by ASIL, feature, and rate</td>
<td>• Determine a code placement strategy</td>
</tr>
<tr>
<td>• Pick a strategy for grouping units into features</td>
<td>• Pass only necessary signals to units</td>
<td>• Use different name tokens for shared utilities</td>
</tr>
<tr>
<td>• Split ASIL and QM levels at the top level of the model</td>
<td>• Eliminate algorithm content at the integration level</td>
<td></td>
</tr>
</tbody>
</table>
**Poor tool qualification awareness**

- How do you qualify a tool?

<table>
<thead>
<tr>
<th>Method</th>
<th>TCL 2</th>
<th>TCL 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ASIL A</td>
<td>ASIL B</td>
</tr>
<tr>
<td>1a   Increased confidence from use</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>1b   Evaluation of the tool development process</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>1c   Validation of the software tool</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>1d   Development in compliance with a safety standard</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Source: ISO 26262:2018 Clause 11.4.6.1, Tables 4&5

+ … Recommended  ++ … Highly recommended
Poor of tool qualification awareness
Leverage tool qualification kit from supplier

- Utilize vendor provided tool qualification content as much as possible
- Have a plan to qualify any custom tools or use cases not covered by the tool vendor
Summary

- Process not clearly defined or documented
  - Document process: from ISO Requirement down to Detail Work Instructions

- Lack top-down architectural design approach
  - Review architecture – Implementation of Freedom From Interference

- Poor tool qualification awareness
  - Leverage tool qualification content from tool vendor
Presenter contact info and poll questions

Please contact me at jasonm@mathworks.com with questions

- Poll question : How would you rate your organizations activity on ISO 26262
  a. No interest
  b. Some interest but no activity
  c. Currently implementing an ISO 26262 compliant process
  d. Struggling to implement an ISO 26262 compliant process
  e. Already fully ISO 26262 compliant

- If you would like to an individual follow-up, please let us know in the WebEx poll area.