ISO 26262 and Model-Based Design
A Done Deal?

MathWorks Automotive Conference 2012

Mirko Conrad
Development Manager Simulink Certification and Standards
ISO 26262 “Road Vehicles - Functional Safety”

- **Functional safety standard** for passenger cars
  - Concerned with avoidance of unreasonable risks due to hazards caused by malfunctioning E/E systems

- Facilitates modern software engineering concepts such as
  - Modeling and simulation
  - Early verification / validation
  - Code generation
ISO 26262 and Model-Based Design

- 2008: ISO/CD 26262
- 2009: ISO/DIS 26262
- 2010: Model Advisor checks for ISO 26262
- 2011: ISO 26262
- 2012: ISO 26262 Process Deployment Advisory Service
ISO 26262 and Model-Based Design A Done Deal?

MathWorks support

Typical challenges

Best practices

1. Understanding process and tool chain impact of meeting the standard
2. Gaining confidence in the software tools used
3. Demonstrating compliance and managing safety cases
**Typical Challenges**

**#1 Understanding Process and Tool Chain**

**Impact of Meeting the Standard**

<table>
<thead>
<tr>
<th>ISO 26262-1</th>
<th></th>
<th>• Vocabulary</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISO 26262-2</td>
<td></td>
<td>• Management of functional safety</td>
</tr>
<tr>
<td>ISO 26262-3</td>
<td></td>
<td>• Concept phase</td>
</tr>
<tr>
<td>ISO 26262-4</td>
<td></td>
<td>• Product development: system level</td>
</tr>
<tr>
<td>ISO 26262-5</td>
<td></td>
<td>• Product development: hardware level</td>
</tr>
<tr>
<td>ISO 26262-6</td>
<td></td>
<td>• Product development: software level</td>
</tr>
<tr>
<td>ISO 26262-7</td>
<td></td>
<td>• Production and operation</td>
</tr>
<tr>
<td>ISO 26262-8</td>
<td></td>
<td>• Supporting processes</td>
</tr>
<tr>
<td>ISO 26262-9</td>
<td></td>
<td>• ASIL-oriented and safety-oriented analyses</td>
</tr>
<tr>
<td>ISO 26262-10</td>
<td></td>
<td>• Guideline</td>
</tr>
</tbody>
</table>

- 10 parts
- 400+ pages
- 100+ work products

- 15+ software-related method tables
- 70+ methods

**Back-to-back comparison test**

**Qualified software tools**

**Estimation of required resources**

**Software architectural design specification**

**Design and coding guidelines**

**Simulation of dynamic parts of the design**

**Control flow analysis**

**Reference phase model for the software development**
Best Practices
ISO 26262 Gap Analysis

Actual software lifecycle

ISO 26262 compliant software lifecycle

Gap list; Roadmap to increase compliance
MathWorks Support

Process Deployment Advisory Service

Consulting services to quickly adopt Model-Based Design for ISO 26262

- Objectives
  - Identify gaps in current processes
  - Provide a roadmap to an optimized ISO 26262 process
  - Assist with deployment of that roadmap
  - Educate on the ISO 26262 standard

www.mathworks.com/services/consulting/areas/iso26262-process-deployment.html
Exemplary verification and validation processes for safety-related software created using Model-Based Design and production code generation.
Typical Challenges

#2 Gaining Confidence in the Software Tools Used

(1) Tool Classification

- **Tool use cases**
  - UC 1...n
- **Tool impact**
  - TI 1
  - TI 2
- **Tool error detection**
  - TD 1
  - TD 2
  - TD 3
- **Tool confidence level**
  - High
  - Medium

(2) Tool Qualification

- **ASIL**
- Qualification methods for TCL3
- Qualification methods for TCL2
- Qualification not required

Increasing qualification requirements

Software tool criteria evaluation report

Software tool qualification report

$10^2...10^3$ software tools
Best Practices

Start with Pre-qualified COTS Tools

Tool Vendor
Pre-qualification based on reference use cases / workflows

Certification Authority
Independent Assessment

Tool Qualification Package

Tool User
Project-specific adaptation

Templates and work products acc. to ISO 26262-8
Embedded Coder, Simulink Design Verifier, Simulink Verification and Validation, and Polyspace are pre-qualified to ISO 26262 for all ASILs.
Typical Challenges

#3 Demonstrating Compliance and Managing Safety Cases

“When claiming compliance with ISO 26262, each requirement shall be complied with”

“The safety case should progressively compile the work products that are generated during the safety lifecycle”
### Best Practices

#### Compliance Demonstration Templates / Tables

ISO 26262-6, Table 9 - Methods for the verification of software unit design and implementation

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL</th>
<th>Evidence</th>
<th>Notes</th>
<th>Findings</th>
<th>Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a Walk-through</td>
<td>++</td>
<td>o</td>
<td>o</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1b Inspection</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1c Semi-formal verification</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1d Formal verification</td>
<td>o</td>
<td>o</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1e Control flow analysis</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1f Data flow analysis</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1g Static code analysis</td>
<td>+</td>
<td>++</td>
<td>++</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1h Semantic code analysis</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

ISO 26262-6, 9 Software unit testing

ISO 2626-6, Table 10 - Methods for software unit testing

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL</th>
<th>Evidence</th>
<th>Notes</th>
<th>Findings</th>
<th>Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a Requirements-based test</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td></td>
</tr>
<tr>
<td>1b Interface test</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td></td>
</tr>
<tr>
<td>1c Fault injection test</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1d Resource usage test</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1e Back-to-back comparison test</td>
<td>+</td>
<td>+</td>
<td>++</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Management of functional safety**

Input document #12:

An overarching safety management is not present. All activities related to the management of functional safety were discussed in a project-specific way in the safety plan.

**Software tools and programming languages**

Software tools used were not qualified. Increased confidence from use for the target compiler was not established. Implementation was carried out in C. Programming guidelines and static analysis tools were used.
MathWorks Support

Process Compliance Demonstration

Annotated method tables with suggestions on how to use Model-Based Design processes and tools to apply the methods listed in ISO 26262-6

Table 9 – Methods for Verification of Software Unit Design and Implementation

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL</th>
<th>Applicable Model-Based Design Tools and Processes</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a Walkthrough</td>
<td>++</td>
<td>Simulink</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Simulink Report Generator – Web View, System Design Description (SDD) report</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Embedded Coder – Code generation report</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Code walkthroughs can be based on HTML code generation reports or code generation reports with an integrated Web View of the model.</td>
<td></td>
</tr>
<tr>
<td>1b Inspection</td>
<td>+</td>
<td>Simulink</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Simulink Report Generator – Web View, System Design Description (SDD) report</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Simulink Verification and Validation – Model Advisor checks</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unit design inspections can be supported by ISO 26262.</td>
<td></td>
</tr>
</tbody>
</table>

Table 10 – Methods for Software Unit Testing

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL</th>
<th>Applicable Model-Based Design Tools and Processes</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a Requirements-based test</td>
<td>++</td>
<td>Simulink Verification and Validation – Requirements Management Interface (RMI)</td>
<td>RMI can be used to establish bidirectional links between textual requirements and models.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IEC Certification Kit – Traceability matrix</td>
<td>Generated traceability matrices can be used to document and review existing links between textual requirements, models, and code.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Simulink – Signal Builder block</td>
<td>Signal Builder blocks can be used to create open-loop model tests. Dynamic test vector charts can be used to create closed-loop, reactive model tests.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Stateflow – Dynamic test vector charts</td>
<td>Component testing capabilities can be used to create model test harnesses. They also enable a requirements pane in the Signal Builder that can be used to link tests with textual requirements.</td>
</tr>
</tbody>
</table>

```
certkitiec;
certkitiec('selectItem','root');
certkitiec('selectItem','IEC Certification Kit (R2012a)','Certification Artifacts', ... 
'iec', 'certkitiec_mbd.pdf');
```
MathWorks Support
Artifact and Work Product Management

Artifacts explorer to access, modify, and manage work products; Integrated templates to demonstrate conformance with reference workflows

Checklist 1: Design Verification

<table>
<thead>
<tr>
<th>Technique / Measure</th>
<th>Associated Requirements</th>
<th>Used / Used to a limited degree / Not used</th>
<th>Interpretation in this application, Evidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Model review</td>
<td>• Inclusion of all model components</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(See &quot;Reviews and Static Analyses at the Model Level&quot;)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 Adherence to modeling standard</td>
<td>• Designation of a modeling standard</td>
<td>Used</td>
<td>Used Model Advisor to check adherence to the</td>
</tr>
<tr>
<td>(See &quot;Reviews and Static Analyses at the Model Level&quot;)</td>
<td>• Review the modeling standard as fit for purpose</td>
<td></td>
<td>ISO26262 Modeling Standards Checks</td>
</tr>
<tr>
<td></td>
<td>• Restriction to modeling constructs suited for production</td>
<td></td>
<td>dist_calc_report_264.html</td>
</tr>
<tr>
<td></td>
<td>• Evidence for using the modeling standard</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 Static analysis at the model level (if applicable)</td>
<td>• Evidence for using static analyzers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(See &quot;Reviews and Static Analyses at the Model Level&quot;)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 Supporting activities</td>
<td>• Documentation of the results of reviews and analyses</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(See &quot;Reviews and Static Analyses at the Model Level&quot;)</td>
<td>• Corrective action on failure of reviews and analyses</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

>> certkitiec;
ISO 26262 and Model-Based Design

Summary

- MathWorks is committed to supporting ISO 26262
  - Reference workflows for using Model-Based Design in the context of ISO 26262
  - Process deployment advisory service
  - ISO 26262 tool qualification kits
  - Process compliance demonstration
  - Artifact and work product management
  - ...

- We look forward to collaborating with you to accelerate the transition towards ISO 26262 compatible processes and tool chains for Model-Based Design