基于模型的设计在汽车功能安全项目中的应用

The application of MBD in automotive functional safety projects

Cheng Hui
KOSTAL ASIA
2015.06.18
1. MBD status in KOSTAL ASIA
   科世达亚洲MBD现状

2. KOSTAL ASIA software develop process (MBD)
   科世达亚洲软件开发流程

3. The application of MBD in functional safety projects
   MBD在功能安全项目中的应用
1. MBD status in KOSTAL ASIA
科世达亚洲MBD现状
Worldwide Footprint

LK Engineering Center

KONA - Engineering Center NAFTA

KOBRA

KOCR

KOSTAL ASIA Engineering Center ASIA

KOIN
Business Area: Automotive Electrical systems - Product Mix

**Mechatronics**
- Steering Column Modules
- DAC / RLS
- Roof Module
- E-Shifter

**Electronics**
- Body
- Door
- Seat
- Access Electronics
- On-Board Charger (OBC)
- Seat sw. / Module
- Door sw. / Module
- Faceplate
- Steering Wheel sw.

**Operating Elements & Switches**
Projects with MBD

BFM
Light and wiper controller
Sop: 2013.12

ESCL
ASIL-D required
Sop: 2016

PEPS
First Sop: 2014.05
Next Sop: 2015.08

SCM
Steering switch with LIN
Sop: 2016
MBD Cooperation

**KOSTAL ASIA**

- Training/ Technical Supporting

**MathWorks**

**KOSTAL ASIA**

- Workshop/ Alignment for Process, Tools, Methods

**KOSTAL Germany**
2. KOSTAL ASIA software develop process (MBD)
科世达亚洲软件开发流程
V-Model

system - Requirements analysis

System Architecture Design

Software Requirement Analysis

Software design

Software Construction

Software Integration

Software Testing

System Test

System Integration Test

Domain: Elec. Hardware

Domain: Mechanical

Domain: Software

Mechatronics level

Domain: Elec. Hardware

Domain: Mechanical

Domain: Software
V-Model

- System Architecture Design
- Software Requirement Analysis
- Software design
- Software Construction
- Software Integration

Domain: Elec. Hardware
Domain: Mechanical
Domain: Software

MBD in component level especially for application layer

Domain: Mechatronics level

Mathworks Annual / Cheng Hui / 2015.6.18  Slide 11
Flowchart of Model Based Design

**Step 1: Requirement Analysis**
- **Product:** Customer Requirement
  - System Requirement
- **Model Structure Interface definition:** arxml/xml
- **Tools:** EA/DaVinci/Kostal RTE
- **DOORS**
- **Software Requirement Analysis**

**Step 2: Architecture Design**
- **Tools:** Simulink/Stateflow
- **Product:** Data Dictionary
  - Model design docs
  - Model file
  - Execution order list

**Step 3: Model Design**
- **Tools:** Embedded Tester/Simulink Tester/V&V
- **Product:** Integ. Test Model
  - Integ. cases
  - Integ. Test Report

**Step 4: Unit Test**
- **Tools:** Embedded Tester
- **Product:** Test cases
  - Test cases
  - Test report

**Step 5: Integration Test**
- **Tools:** Embedded Tester
- **Product:** Integ. Test Report
  - Integ. cases
  - Integ. Test Report

**Step 6: Code Generation**
- **Tools:** Matlab Coder/Simulink Coder/Embedded Coder
- **Product:** Code Generation Report

**Step 7: Back to Back Test**
- **Tools:** Embedded Tester
- **Product:** Back to Back test Report

**Product:** Source Code
  - Code Generation Report

**Product:** Model Check Report
  - Unit Test Model
  - Test cases
  - Unit Test Report
  - Coverage Report

**MBD Process**
- **Design**
- **Validation & Verification**
The application of MBD in functional safety projects

MBD在功能安全项目中的应用
Structure of ISO26262

1. Vocabulary

2. Management of functional safety
   2-5 Overall safety management
   2-6 Safety management during the concept phase and the product development
   2-7 Safety management after the item’s release for production

3. Concept phase
   3-5 Item definition
   3-6 Initiation of the safety lifecycle
   3-7 Hazard analysis and risk assessment
   3-8 Functional safety concept

4. Product development at the system level
   4-5 Initiation of product development at the system level
   4-6 Specification of the technical safety requirements
   4-7 System design
   4-8 Item integration and testing
   4-9 Safety validation
   4-10 Functional safety assessment

5. Product development at the hardware level
   5-5 Initiation of product development at the hardware level
   5-6 Specification of hardware safety requirements
   5-7 Hardware design
   5-8 Evaluation of the hardware architectural metrics
   5-9 Evaluation of the safety goal violations due to random hardware failures
   5-10 Hardware integration and testing

6. Product development at the software level
   6-5 Initiation of product development at the software level
   6-6 Software architectural design
   6-7 Software unit design and implementation
   6-8 Software unit testing
   6-9 Software integration and testing
   6-10 Software integration and testing
   6-11 Verification of software safety requirements

7. Production and operation
   7-5 Production
   7-6 Operation, service (maintenance and repair), and decommissioning

8. Supporting processes
   8-5 Interfaces within distributed developments
   8-6 Specification and management of safety requirements
   8-7 Configuration management
   8-8 Change management
   8-9 Verification
   8-10 Documentation
   8-11 Confidence in the use of software tools
   8-12 Qualification of software components
   8-13 Qualification of hardware components
   8-14 Proven in use argument

9. ASIL-oriented and safety-oriented analyses
   9-5 Requirements decomposition with respect to ASIL tailoring
   9-6 Criteria for coexistence of elements
   9-7 Analysis of dependent failures
   9-8 Safety analyses

10. Guideline on ISO 26262
KOSTAL has many years of experience in safety management, starting with IEC61508 projects in 2003 (SIL3)

ISO26262 is managed by KOSTAL since 2010

There are 18 engineers have AFSP(Automotive Functional Safety Professional) certificate

ISO 26262 references (assessed projects)

ASIL-C/D:
- VW PQ25 Steering Angle Sensor (ASIL-D)
- SGM Electric Steering Column Lock (ASIL-D)
- Daimler BR222 Steering Angle Sensor (ASIL-D)

ASIL-B:

......
Safety relevant functionalities

Example

转向锁止作动器 (带转向锁止ECU)
Safety relevant functionalities

Example
Safety Path

[Diagram of a safety path system with various nodes and connections.]
What’s New for software development?

### Software unit testing

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1a</strong> Requirements-based test (a)</td>
<td>++ ++ ++ ++</td>
</tr>
<tr>
<td><strong>1b</strong> Interface test</td>
<td>++ ++ ++ ++</td>
</tr>
<tr>
<td><strong>1c</strong> Fault injection test (b)</td>
<td>+ + + ++</td>
</tr>
<tr>
<td><strong>1d</strong> Resource usage test (c)</td>
<td>+ + + ++</td>
</tr>
<tr>
<td><strong>1e</strong> Back-to-back comparison test between model and code, if applicable (d)</td>
<td>+ + ++ ++</td>
</tr>
</tbody>
</table>

### Methods for deriving test cases for Software unit testing

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1a</strong> Analysis of requirements</td>
<td>++ ++ ++ ++</td>
</tr>
<tr>
<td><strong>1b</strong> Generation and analysis of equivalence classes (a)</td>
<td>+ ++ ++ ++</td>
</tr>
<tr>
<td><strong>1c</strong> Analysis of boundary values (b)</td>
<td>+ ++ ++ ++</td>
</tr>
<tr>
<td><strong>1d</strong> Error guessing (c)</td>
<td>+ + + +</td>
</tr>
</tbody>
</table>

### Structural coverage metrics at the software unite level

<table>
<thead>
<tr>
<th>Methods</th>
<th>ASIL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1a</strong> Statement coverage</td>
<td>++ ++ + +</td>
</tr>
<tr>
<td><strong>1b</strong> Branch coverage</td>
<td>+ ++ ++ ++</td>
</tr>
<tr>
<td><strong>1c</strong> MC/DC (Modified Condition/Decision Coverage)</td>
<td>+ + + ++</td>
</tr>
</tbody>
</table>
Back to Back TEST

SIL TEST

PIL TEST

PIL API Components

Simulink
Host-Side Comms
Target-Side Comms

Build Process Launcher

PIL Application (harness to test algorithm object code)

Algorithm Object Code (generated from the model)

Real Target or Simulator

© 2008, Leopold Kostal GmbH & Co. KG. Contents and presentation are protected worldwide. Any kind of using, copying etc. is prohibited without prior permission. All rights incl. Industrial property rights are reserved.
This is test case 1.
The benefit of MBD at unit test phase

- Long test time
- Multi Test Tools
- More Test Cases

Manual code

- More effective when executive test
- Test case reuse saves time
- Test result visualization

Model
Conclusion

Advantages of Model-Based Design

- Executable models
  - unambiguous
  - only “one truth”

- Simulation
  - reduces “real” prototypes
  - systematic “what-if” analysis

- Design with Simulation
- Continuous Test and Verification

- Automatic Code Generation
  - minimizes coding errors

- Test with Design
  - detects errors earlier

Time

Efficiency
Thank You