Model Based ADAS System Design

Vector Live Webinar
Agenda

- **Background**
  - PREEvision – Model Based System Design
  - AUTOSAR Workflow
  - ADAS System Design with Traceability and Consistency
  - Legacy Data Frontloading Methodology
  - Toolchain for Software Design
  - Toolchain for Testing
  - System Requirement Generation
From Traditional to a Library-based Development Approach

Background

Traditional development approach
- Development per project
- Low reuse = High variance
- High effort in project

Library-based development approach
- Define & manage reusable elements in global library
- Use of global library in projects = reuse & standardization
- Reduced effort in project

Transition
Background

From Document-based to Model-based Development

**Document-based development**
- Developing in numerous documents
- High effort in keeping documents in sync
- Difficult traceability and consistency

**Model-based development**
- Design systems once → reuse in end-to-end design
- Interconnected model from REQ-System-SW-HW-COM
- Increased efficiency, traceability and consistency

Transition
Agenda

Background

- **PREvision – Model Based System Design**
  - AUTOSAR Workflow
  - ADAS System Design with Traceability and Consistency
  - Legacy Data Frontloading Methodology
  - Toolchain for Software Design
  - Toolchain for Testing
  - System Requirement Generation
PREEvision – Model Based System Design

E/E Systems Engineering Overview
Agenda

Background
PREEvision – Model Based System Design

AUTOSAR Workflow
ADAS System Design with Traceability and Consistency
Legacy Data Frontloading Methodology
Toolchain for Software Design
Toolchain for Testing
System Requirement Generation
Software Architecture design
Hardware Architecture design
Software/Hardware mapping
Data mapping and signal routing
CAN, CAN FD, LIN, FlexRay design
Agenda

Background
PREEvision – Model Based System Design
AUTOSAR Workflow

- **ADAS System Design with Traceability and Consistency**
  - Legacy Data Frontloading Methodology
  - Toolchain for Software Design
  - Toolchain for Testing
  - System Requirement Generation
ADAS System Design with Traceability and Consistency

ADAS System Development Overview

- ADAS system scope: Sensor layer / Application & Arbitration layer / Actuator layer.
- Sensor, Application & Arbitration and Actuator layers are developed inhouse or by suppliers.
- Application & Arbitration development is a highly collaborative task.

Sensor
- “Radar” Supplier
- “Camera” Supplier
- “SENSOR” Team

Application & Arbitration
- AEB (“ADAS” Team)
- SCC (“ADAS” Team)
- LKAS (“A” Supplier)
- AEB + SCC
- SCC+LKAS
- AEB+SCC+LKAS
- Arbitration (“ADAS” Team)

Actuator
- “Powertrain” Team
- “Chassis” Team
- “C” Supplier

Development Complexity
- Sensor fusion: Radar/Camera/Lidar Sensor
- Interfaces: Sensor suppliers use different I/O Signals
- Highly collaborative development:
  - between ADAS systems
  - between ADAS and Other domain systems
  - coordinated by system architects / system owners
- Arbitration (Integration): managed by Integration engineer

Developed by Supplier
Developed by ADAS Team
Developed by Other Domain Team
CHALLENGES IN ADAS SYSTEM DEVELOPMENT

- **Duplicate Development between Systems**
  - e.g. object Detection, Environment Analysis
  - "AEB" System
  - "SCC" System

- **Duplicate Development between Teams**
  - "A Team" Sensor Fusion
  - "B Team" Sensor Fusion

- **Duplicate Development between Version**
  - "V1.0" AEB System
  - "V2.0" AEB System

**Challenges:**
- Duplicate Development → Efficiency
- Collaborative Systems → Consistency
- Vehicle Projects → Complexity

- "A" Vehicle
  - AEB

- "B" Vehicle
  - AEB + SCC

- "C" Vehicle
  - Standard AEB + SCC
  - Premium AEB + SCC + LKAS

**Example:**
- "V1.0" AEB System
- "V2.0" AEB System
Systems Engineering Approach to ADAS System Development

- Standardization of interfaces: Sensors and Actuators by supplier or collaborative systems use standard interfaces
- Reuse of modules: SW of the Perception and Control Library is reused on a module basis
- Separation of Application & Arbitration development from Perception and Control development
ADAS System Design with Traceability and Consistency

Product Line Engineering

- Global Library manages ADAS system Components and Interfaces
- System Library manages selectable system variants for application in specific vehicle projects.
- Vehicle Product Line consists of specific system variant with its sensors, applications and actuators for a specific vehicle.

**PREEvision Database**

- **Global Library**
  - Full ADAS System Model (150%)
    - Perception/Control Component
    - Specific Application/Arbitration
    - Blocks/Data types/Interfaces
  - Feature / Requirements
    - Logical Architecture
    - Software Architecture
    - Hardware Architecture
  - Full Model for *PL Reuse

- **System Library**
  - Partial ADAS System Model (120%)
    - Variants of Specific System
    - Variants of Specific Vehicle
    - Partial Communication Artifact
  - Feature / Requirements
    - Logical Architecture
    - Software Architecture
    - Hardware Architecture
  - Model for Deployment

- **Vehicle Product Line**
  - Specific ADAS System Model (100%)
    - Specific Component
    - Specific Vehicle System
    - Specific Network communication
  - Feature / Requirements
    - Logical Architecture
    - Software Architecture
    - Hardware Architecture
  - Specific Model for Production

* PL: Product Line
* M: Mapping between artifacts (Data based)
ADAS System Design with Traceability and Consistency

Software Layer Design

Software Composition Prototype

Driver Intention Analysis

Warning Control

AEB Longitudinal Application

SCC Longitudinal Application

AEB + SCC Arbitration

Software Component Types

ADAS Mode Selection

Driver Status Perception

Longitudinal Control

Warning Control

AEB + SCC Arbitration

System SW Library

AEB SW Architecture Design (120%)

AEB Root Composition

Driver Status Perception

AEB Longitudinal Application

Warning Control

AEB + SCC SW Architecture Design (120%)

AEB+SCC Root Composition

Driver Status Perception

AEB Longitudinal Application

SCC Longitudinal Application

AEB+SCC Arbitration

“A” Vehicle Product Line

“A” SW Architecture Design (120%)

AEB Root Composition

AEB

SCC Longitudinal Application

AEB + SCC Arbitration

“B” Vehicle Product Line

“AEB’ SW Architecture Design (100%)

AEB Root Composition

“AEB” Vehicle Product Line

“AEB” SW Architecture Design (100%)

AEB Root Composition

AEB

SCC Longitudinal Application

AEB + SCC Arbitration
ADAS System Design with Traceability and Consistency

Other Layers Design

- **Customer Feature / Requirements Design**
  - Feature and variant definition for all components of ADAS systems
  - Requirement definition for each function

- **Logical Architecture Design**
  - Describes functionality of Customer Features and Requirements for implementing software and abstracts them in the form of input-function-output to clarify

- **Software Architecture Design**
  - SWC creation and Interface (including data type) connection for function implementation
  - Definition of hierarchical structure and internal behavior of SWC

- **Network / Hardware Architecture Design**
  - Define all HWC of the ADAS system
  - Create connections to define the Network Topology
  - Distribute functionality by mapping SWC to HWC

- **Communication Design**
  - Create signals based on SWC port information, allocated to HWC in Network Topology
  - Create transmissions by synthesizing PDU and Frame
Variant Management

- Create Customer Feature relations using Variant Diagram
- Configure Alternatives by selecting Customer Features using variant templates
- Derive a new product line to create a vehicle-specific system using the propagated artifacts
- Propagate artifacts that are mapped to the Customer Features into a Architecture Variant container using Propagation Rule
ADAS System Design with Traceability and Consistency

PREEvision Demo
Agenda

Background
PREEvision – Model Based System Design
AUTOSAR Workflow
ADAS System Design with Traceability and Consistency

- **Legacy Data Frontloading Methodology**
  - Toolchain for Software Design
  - Toolchain for Testing
  - System Requirement Generation
## Legacy Data Frontloading Methodology

### AUTOSAR Based Model Frontloading

#### Simulink

<table>
<thead>
<tr>
<th>Items</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulink</td>
<td></td>
</tr>
<tr>
<td>System Model</td>
<td></td>
</tr>
<tr>
<td>Subsystem</td>
<td></td>
</tr>
<tr>
<td>In/Out Port</td>
<td></td>
</tr>
<tr>
<td>Connector</td>
<td></td>
</tr>
<tr>
<td>Interface</td>
<td></td>
</tr>
<tr>
<td>Data Type</td>
<td></td>
</tr>
<tr>
<td>Fixed-Point Data Type</td>
<td></td>
</tr>
</tbody>
</table>

#### PREEvision

<table>
<thead>
<tr>
<th>Items</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW Architecture</td>
<td></td>
</tr>
<tr>
<td>SW Component</td>
<td></td>
</tr>
<tr>
<td>SW Port</td>
<td></td>
</tr>
<tr>
<td>Connection</td>
<td></td>
</tr>
<tr>
<td>Interface</td>
<td></td>
</tr>
<tr>
<td>Data Element</td>
<td></td>
</tr>
<tr>
<td>Data Type</td>
<td></td>
</tr>
<tr>
<td>Computation Method</td>
<td></td>
</tr>
<tr>
<td>Internal Behavior</td>
<td></td>
</tr>
<tr>
<td>Event</td>
<td></td>
</tr>
<tr>
<td>Runnable Entity</td>
<td></td>
</tr>
<tr>
<td>Inter-Runnable Variable</td>
<td></td>
</tr>
<tr>
<td>Data Constraint</td>
<td></td>
</tr>
<tr>
<td>Variable Access</td>
<td></td>
</tr>
<tr>
<td>Data Constraint</td>
<td></td>
</tr>
</tbody>
</table>
Legacy Data Frontloading Methodology

DBC Import Process

- Start Design with Legacy DBC or ARXML files

**DBC Import Process**

**DBC Importer**

- Select import type:
  - Complete Import
  - Automatic Import with Mapping

**Text vs Data**

**SW – HW – COM Relation**
Agenda

Background
PREEvision – Model Based System Design
AUTOSAR Workflow
ADAS System Design with Traceability and Consistency
Legacy Data Frontloading Methodology

➢ Toolchain for Software Design
  Toolchain for Testing
  System Requirement Generation
Toolchain for Software Design

AUTOSAR SW Design Workflow and Tool Chain

**System Design / PREEvision**
- Architecture Design
- SWC Internal Behavior Design

**Export** > **Import**
- System Desc.

**Implementation / e.g. Matlab**
- System Model Design
- Algorithm Design
- ASW Code(.c/.h)

**Configuration / e.g. DaVinci**
- RTE Configuration
- BSW Configuration

**ASW Generation**
- SWC Desc.
- RTE / BSW Generation
- BSW Code(.c/.h)

**Compile & Build / e.g. Codewarrior**
- Compile & Debug
- Test & Build Target

---

Top-Down Approach

Bottom-Up Approach
Video Demo

PREEvision & Simulink Toolchain Tutorial Video
Agenda

Background
PREEvision – Model Based System Design
AUTOSAR Workflow
ADAS System Design with Traceability and Consistency
Legacy Data Frontloading Methodology
Toolchain for Software Design

▶ **Toolchain for Testing**
System Requirement Generation
Toolchain for Testing

Toolchain for SW design and testing

**System Design / PREEvision**
- System Design
- Test Spec Design / Test Analysis
- Traceability Matrix (.XML)
- System Desc. / ECU Extract

**Test Case Design / vTESTStudio**
- Test ITEM (.vti-tso)
- Test Case Design
- Trace Item Assign

**Test Execution / CANoe**
- Test Unit (.vtuexe)
- Test Execution
- NET Simulation
- Test Report (.XML)

**SW Design / Matlab & DaVinci**
- Algorithm Design
- BSW Configuration
- ASW Code (.c/.h)
- BSW Code (.c/.h)

**Compile / Chipset Tool**
- IMP Code (.c/.h)
- Compile & Build
- Debug
- Target Env.

**SW Test Execution / VectorCAST**
- Test Automation
- Coverage Test
- Statement Branch MC/DC
Toolchain for Testing

Video Demo

PREEvision
Test engineering Toolchain
Tutorial Video
Agenda

Background
PREEvision – Model Based System Design
AUTOSAR Workflow
ADAS System Design with Traceability and Consistency
Legacy Data Frontloading Methodology
Toolchain for Software Design
Toolchain for Testing

System Requirement Generation
System Requirement Generation

Automatic Report Generation

- Report creation using company's standard template
- Automatic configuration of diagram or table layout using mapping relation
System Requirement Generation

Video Tutorial

PREEvision Report Generation Tutorial Video
For more information about Vector and our products please visit

www.vector.com

Author:
Kim, Euiyeul
Vector Korea