Models for Automotive Embedded Systems Development

Henrik Lönn, Technology Specialist, Embedded Software Development, AB Volvo.
Our organization

Volvo Group CEO & Group Executive Board

Group Functions

BUSINESS AREAS

Group Trucks Technology

Renault Trucks

Mack Trucks

UD Trucks

Volvo Trucks

Volvo CE

Volvo Buses

Volvo Penta

Volvo Financial Services

Volvo Autonomous Solutions

Arquus

TRUCK DIVISIONS

Group Trucks Operations

Group Trucks Purchasing

Volvo Group Trucks Technology
Lönn: Models for Automotive Embedded Systems Development
Innovative technology offerings
FOR OUR CUSTOMERS AND THE SOCIETY

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>PeDESTrian and Cyclist Detection System</td>
<td>Automatic Traction Control</td>
<td>Predictive Cruise Control (I-See)</td>
<td>Collision Warning with Emergency Brake</td>
<td>MACK ® GuardDog ® Connect</td>
</tr>
</tbody>
</table>

Volvo Group Trucks Technology
Lönn: Models for Automotive Embedded Systems Development
Concepts and technologies for the future

**ACCELERATING THE TRANSITION TO A SUSTAINABLE FUTURE**

<table>
<thead>
<tr>
<th>AUTONOMOUS REFUSE TRUCK FOR URBAN ENVIRONMENT</th>
<th>AUTONOMOUS TRUCK FOR MINING</th>
<th>VOLVO CONCEPT TRUCK</th>
<th>VOLVO VERA</th>
<th>ELECTRIC ROAD SYSTEMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>FULLY ELECTRIC CONSTRUCTION EQUIPMENT WITH AUTOMATION</td>
<td>3D PRINTING FOR LIGHTER ENGINES</td>
<td>WIRELESS SENSORS TO REPLACE CABLES</td>
<td>PLATOONING</td>
<td>AND MANY MORE</td>
</tr>
</tbody>
</table>

Volvo Group Trucks Technology
Lönn: Models for Automotive Embedded Systems Development
Take Home Messages:
• Appropriate Software Architecture is Instrumental
• Software Architecture Competence is Critical for Swedish Industry
• Modeling is an Enabling Technology for Continuous Integration of Complex Products
Reduced Lead Times
Continuous Deployment
Continuous Integration
Quality
Reuse
Multiple cycle times
Multiple Aspects
Multiple Domains
Multiple Teams

Automation
Connectivity
Electrification
Control Strategies
Architecture (re)volution
Why
What
How
Get Things Right - Continuously

Product Cost Flexibility

Mechanical vs. Software Design Flexibility

Cumulative Resources

Cumulative Resources – Continuous Deployment

Hours

Conception Finalization

Conception Finalization

Volvo Group Trucks Technology
Lönn: Models for Automotive Embedded Systems Development
Example: Testing
Example: Road and Track Testing

- When Simulation is too difficult
- When Simulation is Impossible
- Because it is “Free”
Example: Analysis and Simulation Based “Testing”

- To exercise rare situations
- To exercise dangerous situations
- To accelerate time
- To accelerate coverage
- To be repeatable
- To allow debugging
- To cover large sets of configurations
- To cover fictuous configurations
In General: Multi-Method

Multiple Engineering Activities

- Input must be semantically and syntactically valid
- Input must be consistent across methods

\[ F = m \frac{dV}{dt} = m \frac{d^2x}{dt^2} \]
Consistency and Accuracy

- Models are a representation of the real truck
- Data from factory, track, field is required to secure validity
- Models allow engineering rigor and engineering automation
Engineering Information and Activities

V model as reference
…not to prescribe timing and sequence…
V model vs. Continuous Integration and Delivery

Hourly
Every i:th iteration

Daily/Weekly/...
Every j:th iteration

Weekly
Every k:th iteration
Collaborate Efficiently

- Product complexity call for advanced models and tools
- Modelling is a significant Effort

*Vehicle Manufacturer must integrate models*
Control over Engineering Information

- Engineering documentation is a core asset
- Increased use of models represents risk
  - Vendor Lock-in
  - Complex Information
  - Complex access to information

*Vehicle Manufacturer must have control of their models*
Why
What
How
Vehicle Content

Solution independent description of vehicle
"Features" to decompose content
Application Software Architecture - Layering

- Route Management
- Situation Management
- Motion Management
- Motion Device Management
- Software Platform
Application Software Architecture - Layering

- Route Management
- Situation Management
- Motion Management
- Motion Device Management
- Software Platform
Virtual Engineering

...to allow

- Instant Deliveries
- Maximize Verification Confidence
- Exercise Dangerous and Rare Events
Virtual Integration and Simulation

- Simulation requires models of all relevant functional aspects
  - (Software)
  - Software platform
  - Electronics
  - Mechatronics
  - Physics
  - Environment
An information model that captures engineering information in a standardized way
Modelling & Simulation Technology

- Models - MIL
- Software - SIL
- Hardware - HIL
- Prototyping - RCP

Controller (SW & Electronics) → Devices → Plant (Mechanics, etc.) → Environment
Modelling Pattern to Support Integration

Model-in-the-loop: **Controller model** interfaces to models of sensors and actuators
Software-in-the-loop: **Software** interfaces to models of electronics
Hardware-in-the-loop: **ECU** interfaces to models of sensors and actuators
Hardware+-in-the-loop: **Sensor & actuator** interfaces to models of plant
Model Structure
Feature Models
Simulation Content

Behavior/Timing
Function Mockup Units + Execution Definition
Source Code + Execution Definition
Why
What
How
AUTOSAR
AUTOSAR - Technical Goals

- Increased Flexibility
  - Modularity
  - Scalability
  - Transferability
  - Re-usability
- Standardized platform
  - Off-the-shelf purchase & integration of comm, OS, diagnosis, drivers, etc.
  - Off-the-shelf hardware
- Standardized Interfaces
  - Off-the-shelf purchase & integration of common vehicle functions
AUTOSAR ECU SW Architecture

AUTOSAR Runtime Environment (RTE)

Standardized Interface

Operating System

Services

Communication

ECU Abstraction

Complex Device Drivers

Basic Software

Microcontroller Abstraction

ECU-Hardware

Standardized Interface

AUTOSAR Interface

Standardized Interface

AUTOSAR Interface

Standardized Interface

AUTOSAR Interface

AUTOSAR Interface

AUTOSAR Interface

AUTOSAR Interface

Standardized Interface

Standardized Interface

Standardized Interface

Standardized Interface

Standardized Interface

AUTOSAR Software

AUTOSAR Software

AUTOSAR Software

AUTOSAR Software

Application Software Component

Actuator Software Component

Sensor Software Component

...
AUTOSAR - Elements

- **Modelling**
  - Capture SW Components SW Component Template
  - Capture ECU resources: ECU Resource Description
  - Capture allocation and communication: System Description

- **Methodology**
  - Autogenerate ECU configuration
  - Autogenerate platform SW configuration
  - Autogenerate glue code (RTE)

- **Application Interfaces**
  - Standard interface definitions for well-established functions

- **Architecture**
  - Standard platform SW
  - Standard interfaces
AUTOSAR Software: Software Components

- Hardware independent (CPU and peripherals of ECU)
- Location independent (Communication technology, platform services)
- Specified by
  - Interface
  - Resource Requirements
  - Implementation constraints
AUTOSAR Software: Sensors and Actuators

- SensorActuatorSoftwareComponent adapts device interface to application functions
- Software architecture to separate ECU and device specific parts
- Support for re-allocation of sensors and actuators
AUTOSAR Basic Software

- Software platform for application SW components
- Standardized Components
  - System services (diagnostic protocols; NVRAM, flash and memory management)
  - Communication (CAN/LIN/FlexRay... framework, I/O management, Network management)
  - Operating System (OSEK based)
  - Microcontroller Abstraction
- ECU Specific Components
  - ECU Abstraction
  - Complex Device Driver
AUTOSAR Methodology

Covers process steps for

– ECU platform software generation and configuration

– Application software integration
EAST-ADL Representation

- System Model
  - Vehicle Level: Technical Feature Model
  - Analysis Level: Functional Analysis Architecture
  - Design Level: Functional Design Architecture
  - Implementation Level: HW Function

Features of the vehicle

Abstract functions

Concrete functions, Hardware topology, allocation to nodes

Software Architecture as represented by AUTOSAR

<<Realize>>
EAST-ADL Model on Design Level

Functional Structure

Design Level
EAST-ADL Model on Design Level

Functional Structure onto HW Architecture

<<FunctionalDesignArchitecture>> DemonstratorFDA

Application Functionality

<<LocalDeviceManager>> BrakePedal
<<DesignFunction>> BrakeController
<<LocalDeviceManager>> WheelSensorFL

VehicleSpeed

<<BSWFunct>> ABSFrontLeft
<<BSWFunct>> BrakeActuatorFL

<<HWFunct>> PedalIO
<<HWFunct>> PedalSensor
<<HWFunct>> BrakeIO
<<HWFunct>> BrakeActuatorFrontLeft
<<HWFunct>> WheelSensorFrontLeft

<<BSWFunct>> WSensIO
<<HWFunct>> WheelSensorFrontLeft

<<AnalysisFunction>> BrakePlantModel

<<BSWFunct>> BrakeActuatorFL
<<BSWFunct>> BrakeIO
<<BSWFunct>> WSensIO
<<BSWFunct>> WheelSensorFL

<<BSWFunct>> WheelSensIO
<<BWSFunction>> BrakeActuatorFL
<<BWSFunction>> BrakeIO
<<BWSFunction>> WSensIO
<<BWSFunction>> WheelSensorFL

<<LocalDeviceManager>> BrakePedal
<<LocalDeviceManager>> ABSFrontLeft
<<LocalDeviceManager>> WheelSensorFL

Brake Pedal
Brake Controller
Wheel Sensor FL

<<BSWFunct>> BrakeFrontLeft
<<BSWFunct>> BrakeFrontRight
<<BSWFunct>> BrakeRearLeft
<<BSWFunct>> BrakeRearRight

ECU Node
ABS_FrontLeft
ABS_FrontRight
ABS_RearLeft
ABS_RearRight

<<AnalysisFunction>> BrakePlantModel
<<BSWFunct>> BrakeActuatorFL
<<BSWFunct>> BrakeIO
<<BSWFunct>> WSensIO
<<BSWFunct>> WheelSensorFL

<<BSWFunct>> WheelSensIO
<<BWSFunction>> BrakeActuatorFL
<<BWSFunction>> BrakeIO
<<BWSFunction>> WSensIO
<<BWSFunction>> WheelSensorFL

<<LocalDeviceManager>> BrakePedal
<<LocalDeviceManager>> ABSFrontLeft
<<LocalDeviceManager>> WheelSensorFL

Brake Pedal
Brake Controller
Wheel Sensor FL

<<BSWFunct>> BrakeFrontLeft
<<BSWFunct>> BrakeFrontRight
<<BSWFunct>> BrakeRearLeft
<<BSWFunct>> BrakeRearRight

ECU Node
ABS_FrontLeft
ABS_FrontRight
ABS_RearLeft
ABS_RearRight

<<AnalysisFunction>> BrakePlantModel
<<BSWFunct>> BrakeActuatorFL
<<BSWFunct>> BrakeIO
<<BSWFunct>> WSensIO
<<BSWFunct>> WheelSensorFL

<<BSWFunct>> WheelSensIO
<<BWSFunction>> BrakeActuatorFL
<<BWSFunction>> BrakeIO
<<BWSFunction>> WSensIO
<<BWSFunction>> WheelSensorFL

<<LocalDeviceManager>> BrakePedal
<<LocalDeviceManager>> ABSFrontLeft
<<LocalDeviceManager>> WheelSensorFL

Brake Pedal
Brake Controller
Wheel Sensor FL

<<BSWFunct>> BrakeFrontLeft
<<BSWFunct>> BrakeFrontRight
<<BSWFunct>> BrakeRearLeft
<<BSWFunct>> BrakeRearRight

ECU Node
ABS_FrontLeft
ABS_FrontRight
ABS_RearLeft
ABS_RearRight

<<AnalysisFunction>> BrakePlantModel
<<BSWFunct>> BrakeActuatorFL
<<BSWFunct>> BrakeIO
<<BSWFunct>> WSensIO
<<BSWFunct>> WheelSensorFL

<<BSWFunct>> WheelSensIO
<<BWSFunction>> BrakeActuatorFL
<<BWSFunction>> BrakeIO
<<BWSFunction>> WSensIO
<<BWSFunction>> WheelSensorFL

<<LocalDeviceManager>> BrakePedal
<<LocalDeviceManager>> ABSFrontLeft
<<LocalDeviceManager>> WheelSensorFL

Brake Pedal
Brake Controller
Wheel Sensor FL

<<BSWFunct>> BrakeFrontLeft
<<BSWFunct>> BrakeFrontRight
<<BSWFunct>> BrakeRearLeft
<<BSWFunct>> BrakeRearRight

ECU Node
ABS_FrontLeft
ABS_FrontRight
ABS_RearLeft
ABS_RearRight

<<AnalysisFunction>> BrakePlantModel
<<BSWFunct>> BrakeActuatorFL
<<BSWFunct>> BrakeIO
<<BSWFunct>> WSensIO
<<BSWFunct>> WheelSensorFL

<<BSWFunct>> WheelSensIO
<<BWSFunction>> BrakeActuatorFL
<<BWSFunction>> BrakeIO
<<BWSFunction>> WSensIO
<<BWSFunction>> WheelSensorFL

<<LocalDeviceManager>> BrakePedal
<<LocalDeviceManager>> ABSFrontLeft
<<LocalDeviceManager>> WheelSensorFL

Brake Pedal
Brake Controller
Wheel Sensor FL

<<BSWFunct>> BrakeFrontLeft
<<BSWFunct>> BrakeFrontRight
<<BSWFunct>> BrakeRearLeft
<<BSWFunct>> BrakeRearRight

ECU Node
ABS_FrontLeft
ABS_FrontRight
ABS_RearLeft
ABS_RearRight

<<AnalysisFunction>> BrakePlantModel
<<BSWFunct>> BrakeActuatorFL
<<BSWFunct>> BrakeIO
<<BSWFunct>> WSensIO
<<BSWFunct>> WheelSensorFL

<<BSWFunct>> WheelSensIO
<<BWSFunction>> BrakeActuatorFL
<<BWSFunction>> BrakeIO
<<BWSFunction>> WSensIO
<<BWSFunction>> WheelSensorFL

<<LocalDeviceManager>> BrakePedal
<<LocalDeviceManager>> ABSFrontLeft
<<LocalDeviceManager>> WheelSensorFL

Brake Pedal
Brake Controller
Wheel Sensor FL

<<BSWFunct>> BrakeFrontLeft
<<BSWFunct>> BrakeFrontRight
<<BSWFunct>> BrakeRearLeft
<<BSWFunct>> BrakeRearRight

ECU Node
ABS_FrontLeft
ABS_FrontRight
ABS_RearLeft
ABS_RearRight

<<AnalysisFunction>> BrakePlantModel
<<BSWFunct>> BrakeActuatorFL
<<BSWFunct>> BrakeIO
<<BSWFunct>> WSensIO
<<BSWFunct>> WheelSensorFL

<<BSWFunct>> WheelSensIO
<<BWSFunction>> BrakeActuatorFL
<<BWSFunction>> BrakeIO
<<BWSFunction>> WSensIO
<<BWSFunction>> WheelSensorFL

<<LocalDeviceManager>> BrakePedal
<<LocalDeviceManager>> ABSFrontLeft
<<LocalDeviceManager>> WheelSensorFL

Brake Pedal
Brake Controller
Wheel Sensor FL

<<BSWFunct>> BrakeFrontLeft
<<BSWFunct>> BrakeFrontRight
<<BSWFunct>> BrakeRearLeft
<<BSWFunct>> BrakeRearRight

ECU Node
ABS_FrontLeft
ABS_FrontRight
ABS_RearLeft
ABS_RearRight

<<AnalysisFunction>> BrakePlantModel
<<BSWFunct>> BrakeActuatorFL
<<BSWFunct>> BrakeIO
<<BSWFunct>> WSensIO
<<BSWFunct>> WheelSensorFL

<<BSWFunct>> WheelSensIO
<<BWSFunction>> BrakeActuatorFL
<<BWSFunction>> BrakeIO
<<BWSFunction>> WSensIO
<<BWSFunction>> WheelSensorFL

<<LocalDeviceManager>> BrakePedal
<<LocalDeviceManager>> ABSFrontLeft
<<LocalDeviceManager>> WheelSensorFL

Brake Pedal
Brake Controller
Wheel Sensor FL

<<BSWFunct>> BrakeFrontLeft
<<BSWFunct>> BrakeFrontRight
<<BSWFunct>> BrakeRearLeft
<<BSWFunct>> BrakeRearRight

ECU Node
ABS_FrontLeft
ABS_FrontRight
ABS_RearLeft
ABS_RearRight

<<AnalysisFunction>> BrakePlantModel
<<BSWFunct>> BrakeActuatorFL
<<BSWFunct>> BrakeIO
<<BSWFunct>> WSensIO
<<BSWFunct>> WheelSensorFL

<<BSWFunct>> WheelSensIO
<<BWSFunction>> BrakeActuatorFL
<<BWSFunction>> BrakeIO
<<BWSFunction>> WSensIO
<<BWSFunction>> WheelSensorFL
EAST-ADL Model on Design Level

Timing/Triggering

<<FunctionalDesignArchitecture>> DemonstratorFDA

Application Functionality

<<LocalDeviceManager>> BrakePedal

<<DesignFunction>> BrakeController

<<LocalDeviceManager>> WheelSensorFL

<<DesignFunction>> ABSFrontLeft

VehicleSpeed

BSW Functionality

<<BSWFunction>> PedalIO

<<BSWFunction>> BrakeActuatorFL

<<BSWFunction>> WSensIO

<<BSWFunction>> BrakeActuatorFrontLeft

<<BSWFunction>> WheelSensorFrontLeft

HW Functionality

<<HWFunction>> PedalSensor

<<HWFunction>> BrakeIO

<<HWFunction>> BrakeActuatorFL

<<HWFunction>> WheelSensorFL

<<LocalDeviceManager>> BrakePedal

<<DesignFunction>> ABSFrontLeft

<<LocalDeviceManager>> BrakeController

<<DesignFunction>> BrakeController

<<LocalDeviceManager>> WheelSensorFL

<<FunctionTrigger>> ABSTrigger

<<PeriodicEventConstraint>> ABSPeriodConstraint

<<EnvironmentModel>> DemonstratorEM

Lönn: Models for Automotive Embedded Systems Development
Transfer Function – "Black-box" behavior
EAST-ADL Model on Design Level

Transfer Function – "Black-box" behavior

Simulation model `My_ABS.mdl`
Typically in plant model:
• Non-causal: "Power Ports" – {Torque, Speed}, {Pressure, Flow}, etc.
• Continuous Time: No triggers
Why
What
How
Variability & Product Lines
Variability - Background

- On average, less than 2 vehicles of the same configuration is built: Huge configuration space
- All engineering information is governed by configuration and variant
  - Requirements
  - Components
  - Tests
  - Annotations
  - ...
- Co-variation SW and mechanics/plant
  - SW configuration must be consistent with mechatronics and mechanics of truck
- Customer variability vs technical variability
  - Some variability is invisible to the user: Internal technical solutions
- Detailed variability vs. Exposed variability
  - Some details are only relevant for the component developer
Variability Mechanism 1: Include/exclude

Feature model for configuration

Artefacts subject to configuration
Variability Mechanism 2: Parameterize

Feature model for configuration

Artefacts subject to configuration
- Include/exclude
- Variant-specific calibration
Compositional Variability

\[
\text{thresholdValue} := \begin{cases} 
  \text{front} & ? 140 \\
  \text{rear} & ? 80 \\
  \text{undef} & 
\end{cases}
\]

Front

Rear

WiperController

WiperIntegrator

driver Request

rainSensor Request

rainSensorSig

rsThr:Threshold

wiperBtnSig

Variant 1

Variant 2

Lönn: Models for Automotive Embedded Systems Development

Volvo Group Trucks Technology
Compositional Variability

Volvo Group Trucks Technology
Lönn: Models for Automotive Embedded Systems Development
Why
What
How
Workflow
Engineering Workflow

- **Represent**
  - Models and Code
- **Resolve**
  - Configure and Resolve variability
- **Generate**
  - Automatic preparation for each tool
- **Simulate/Analyze/Verify**
  - Software centric simulations
  - Physics centric simulations
  - Arithmetic analysis
  - Formal verification
  - ...
Configuration

Software:
- AUTOSAR SWC Selection
- AUTOSAR SWC Parameterization
- Configuration ports
- Initialization values

Non-Software
- Logical Component Selection
- Function Mockup Unit Parameterization
- Configuration Ports
- Initialization values

Environment:
- Traffic Scenario:
  - OpenScenario Selection & Parameterization
- Roads:
  - OpenRoad Selection and Parameterization

Lönn: Models for Automotive Embedded Systems Development
Generation

Application SW Integration
Platform SW Integration
ECU, I/O, Sensors & Actuators
Plant Integration
Environment Integration

Volvo Group Trucks Technology
Lönn: Models for Automotive Embedded Systems Development
Simulation

Application Software
Platform Software

Simulated Mechatronics

Simulated Physics

Environment

Adapt Simulation Platform

Test Framework

Volvo Group Trucks Technology
Lönn: Models for Automotive Embedded Systems Development
Engineering Automation

Make Changes → Automatic Resolution and Build → Automatic Invocation → Automatic Test

Local PC or Cloud

Volvo Group Trucks Technology
Lönn: Models for Automotive Embedded Systems Development
Conclusion

- AUTOSAR software platform is automotive de-facto standard
- EAST-ADL supports automotive embedded systems modelling
  “starting” with needs and requirements and
  “ending” with an AUTOSAR SW architecture
- An agreed modelling language makes it possible
  - to understand engineering information from other departments/disciplines/companies
  - to exchange engineering models between different organizations and tools
  - to progress jointly on tools and methodology for modelling, analysis and synthesis
Summary

New Vehicle Functionality Development Efficiency

Short Lead Time Iterative & Incremental Development

Engineering Automation

Automatic Configurability Automatic Evaluation

Variability Modelling System & Plant Modelling

Model

Configure

Simulate