Practical Implementation of an AUTOSAR toolchain at Volvo Cars

Niklas Amberntsson
Technical Marketing Manager
Automotive Network Design

3rd AUTOSAR Open Conference
Frankfurt, May 11 2011
Overview

Topics covered:
1. Overview of tool-landscape
2. Objectives
3. Main technical challenges and solutions
4. Tool interoperability topics
5. Conclusions
Volvo Cars AUTOSAR tool chain

Function Design

Function Requirements

SW architecture

Function allocation

Logical Topology

ECU configuration

and integration

ECU platform SW

ECU extract generation

ECU Extract

Communication Design and analysis

Post-build config mgmt

© 2011 Mentor Graphics Corp. Company Confidential
www.mentor.com
VCC AUTOSAR tool chain

1. Function Design
   - ELEKTRA Vector Informatik
   - SW architecture
   - Function allocation
   - Logical Topology

2. SW Development
   - Matlab/Simulink Mathworks
   - Any products that fulfil VCC requirements

3. ECU configuration and integration
   - ECU Extract generation
   - Post-build config mgmt

4. Communication Design and analysis
   - VSA COM Designer Mentor Graphics
   - VSB Mentor Graphics

© 2011 Mentor Graphics Corp. Company Confidential
www.mentor.com
Data storage overview

ELEKTRA
- Requirements
- Abstract Functions
- SWCs
- ECU Topology
- SWC allocation
- Timing requirement

VSA COM Designer
- SWCs
- ECU Topology
- SWC Allocation
- Timing requirement
- Com strategy definition
- Signal to Frame packing
- Frame parameter definition
- Gateway definition
- Controller configuration

Multiple SWC and SYS description files + EADL

AR ECU extracts and other files

To ECU config tools etc
Objectives

Apart from the obvious objective of an integrated toolchain from Systems design, through network design to ECU configuration and implementation a set of other objectives were defined:

1. Support for industrial development process with iterative development
2. Capability to design communication matrix for all possible variants of the ECU topology
3. Ability to capture timing requirements at system level and to validate the network design against these requirements
4. Support for post-build configuration changes of communication configuration for ECUs
5. Support for creating mixed systems consisting of AUTOSAR and legacy ECUs
INDUSTRIAL DEVELOPMENT PROCESS WITH ITERATIVE DEVELOPMENT AND CONFIGURATION MANAGEMENT
Industrial development process with iterative development

- Iterative development between system design and Communication design must be possible

ELEKTRA

System and SWC data

VSA COM Designer

System and SWC data

Com. Data

New sys and SWC data

Com. Data

changed

New sys and SWC data

Com. Data

changed

changed

new

© 2011 Mentor Graphics Corp. Company Confidential
www.mentor.com
Industrial development process and iterative development

What was needed:

- An agreed package structure in exported files from ELEKTRA

- A way to describe the design-objects lifecycle
  - In Work, Frozen, Released

- Automatic rule-based merge
  - Identification of design objects through UUID
  - Taking life-cycle status of objects into account, such as
    - "If design data changed, and lifecycle status is "In Work", then overwrite previous design data"
Configuration management

- Version structures of design objects as defined in ELEKTRA are replicated in the VSA COM designer
- Enables user to compare versions of design objects etc

ECUs

ABS
  ABS_1

EMS
  EMS_1

AR SYS and SWC template
Configuration management

- Version structures of design objects as defined in ELEKTRA are replicated in the VSA COM designer.
- Enables user to compare versions of design objects etc.
Configuration management

- Version structures of design objects as defined in ELEKTRA are replicated in the VSA COM designer
- Enables user to compare versions of design objects etc

**Diagram:**
- ECUs
  - ABS
  - ABS_1
  - EMS
  - EMS_1
- ELEKTRA
- VSA
  - ABS
  - ABS_1
  - EMS
  - EMS_1
- AR SYS and SWC template
- Config 1
- Config 2

Com design data added in VSA COM Designer
Configuration management

- Version structures of design objects as defined in ELEKTRA are replicated in the VSA COM designer
- Enables user to compare versions of design objects etc

Diagram:
- ECUs
  - ABS
    - ABS_1
    - ABS_2
  - EMS
    - EMS_1

- ELEKTRA

- VSA
  - ABS
    - ABS_1
  - EMS
    - EMS_1

Com design data added in VSA COM Designer

Config 1

© 2011 Mentor Graphics Corp. Company Confidential
www.mentor.com
A COMMON COMMUNICATION DESIGN FOR ALL VARIANTS OF SYSTEM
Capability to design communication matrix for all possible variants of the vehicle

- Even though multiple ECUs may exist in variants with different communication interfaces, the objective is one communication design that supports all needs.
- But still each ECU variant must get exactly the communication configuration it needs.
Capability to design communication matrix for all possible variants of the vehicle

- **Selected strategy:**
  - A super-set design strategy, where the goal is to create one communication matrix that contains all possible communication between all possible ECUs including variants
  - Only variants that affect communication design are considered

- **Issues to solve:**
  - How to communicate variants of ECUs from System and SWC design to network design?
  - How to manage the communication design to make sure it fulfills all requirements from all possible variants of ECUs
Capability to design communication matrix for all possible variants of the vehicle

User decides for design strategy (which ECUs to merge to super ECUs). In this case “EMS”

Network design made for “super system”

Import – merge
Define “super ECUs”

ECU extracts for all ECU variants

Network design
MANAGING TIMING REQUIREMENTS AND ANALYSING THE COMMUNICATION DESIGN
Capturing timing requirements: prerequisites

- The input to the network design process is the signals to be exchanged between ECUs and their timing requirements.
- Timing requirements on signals originate from the functions using the signals.
- Functions are described in ELEKTRA, while the network design done in VSA COM Designer.
- The timing model must be able to express:
  - Vehicle Function timing requirements before function allocation.
  - Partitioning of available time after function has been allocated to ECUs.
  - Splitting of time on different network segments in case of gateway.
Capturing timing requirements

Function design with timing requirements

ECU topology

Delay 100ms

Internal signal

Network signal

Delay 200ms

© 2011 Mentor Graphics Corp. Company Confidential
www.mentor.com
Capturing timing requirements

Function design with timing requirements

- Delay 100ms
- Delay 200ms

ECU topology

Export to VSA COM designer

- Communication related info passed to network design in VSA COM Designer
- VSA COM Designer user can distribute time in case of gateways or chains of signals
Capturing timing requirements

Function design with timing requirements

ECU topology

Export to VSA COM designer

- Communication related info passed to network design in VSA COM Designer
- VSA COM Designer user can distribute time in case of gateways or chains of signals
Benefits of timing model

- Allow capturing functional timing requirements before function allocation to ECUs
- Allow partitioning of overall functional timing requirement onto ECUs and networks after function allocation
  —Used as "contracts" for later design steps
- Allow network design validation vs requirements already during system design
Timing model practical implementation

- The AUTOSAR timing model provides a generic language to communicate timing requirements
  - Various sorts of Events and Delays
- To apply this to network design, a subset of the AUTOSAR timing model has been used in order to reduce complexity for the user
  - Events focused on signals and sensor/actuators
Timing model practical implementation

- Sample screenshot showing frame timing analysis in VSA/VSA COM Designer
SUPPORTING POST-BUILD COMMUNICATION
CONFIGURATION CHANGES
Supporting post-build configuration changes to communication configuration

- An important requirement was to support post-build configuration changes of the communication configuration of an ECU

- Such as:
  - Changing signal to frame packing
  - Adding/removing gatewayed signals
  - Changing frame IDs and timing

- This must be well supported in the communication design tool
  - in order to ensure user knows exactly which changes he can do with post-build configuration
  - To keep track of existing configurations and where they are used
Benefits of post-build configuration

- **Save development effort and cost**
  - Enable reuse of ECUs between car-models with different communication matrix without changing the application SW

- **Save development time**
  - Enable OEM to do “quick fixes” for verification purposes
  - Such as adding gatewayed signals to support a new function
  - Enable verification of new Engine Controller in old vehicle
Supporting post-build configuration changes to communication configuration

- Separation of editors
  - for application SW properties like published/subscribed signals
  - for ECU post-build configuration properties like signal packing

- One ECU may have several configurations

- Still supporting ECUs that do not have post-build configuration support

Fixed SW properties like:
- Scheduling of COM functions
- Generation of signal values
- Publisher/subscriber latency

Configurable BSW properties like:
- Signal to frame packing
- Frame periods
- Frame IDs
- Gateway definitions
Supporting both AUTOSAR and pre-AUTOSAR ECUs

Since there is possibility for carry over of pre-AUTOSAR ECUs in vehicles, a requirement to support design of mixed systems was introduced.

Pre-AUTOSAR ECUs use Volcano VTP or equivalent BSW.

Considerations:
- Features of AUTOSAR and pre-AUTOSAR ECUs differ
- Details like signal packing rules, signal endianess differ
- File format for configuring ECUs differ

Solutions:
- Different consistency check rules are applied for AUTOSAR and pre-AUTOSAR ECUs
  - To avoid using unsupported features of any ECU
  - To ensure signal packing etc is compliant with each ECUs capabilities
- VSA COM designer can generate configuration files for Volcano VTP as well as Fibex, DBC and LDF files for test tools
Tool interoperability topics

■ What worked well:
  — Exchange of AUTOSAR objects that exist in both tools
  — Distribution of ownership of data between tools

■ Most problematic areas
  — When meta-model of tools don’t match perfectly, such as in case of port-interfaces that don’t exist separately from port in ELEKTRA
    – Required fine-tuning to create the port-interface during export, according to agreed principles
Tool interoperability topics

- Extensions for Version management and Import/merge
  - Since both tools should reflect the structures of versions of ECUs, it is not enough to transfer only the design data
  - Additional information to describe life-cycle status and design version has been extended as "Special Data Groups"

- Debugging interfaces between tools
  - Quite detailed analysis required, sometimes at XML-level
  - Time consuming with iterations to export/import data between companies/tools, provide feedback, wait for new export etc...
  - The better approach is probably to exchange tools between companies so each company can do “short-loops” to verify integration
Current status and remaining work

- **Current status**
  - The basic data exchange between tools work and covers a complete development process
  - Proven in "proof of concept" type projects

- **Remaining work short term**
  - Additional quality assurance, prove tool chain in design of large systems
  - Add additional export formats such as LDF, DBC, VTP for connectivity to various activities and tools
  - Add database support in VSA COM designer

- **Mid/Long term plan**
  - Move to AUTOSAR 4.0 data exchange and utilize variability concept and timing model
Conclusions (1)

- AUTOSAR certainly helps when integrating tools from different vendors
- An effort like this would almost be unthinkable without the common ”language” from AUTOSAR
- Initial division of responsibilities between tools actually worked out well
- Attention must be paid to the meta-models of tools to be integrated
  - Easiest if all involved tools support ”native AUTOSAR” meta-model
  - Requires careful consideration of how to map tool-specific meta-model to AUTOSAR
Conclusions (2)

- Even though the project is not concluded it is clear that the toolchain supports digital continuity from requirements to function design and allocation, SW architecture, communication design, and ECU extract generation.

- Compared with previous document based development processes it is a great step forward.

- The project shows that AUTOSAR helps good tool integration even between tool vendors.
Contacts

Ove Berndtsson
oberndts@volvocars.com

Niklas Amberntsson
niklas_amberntsson@mentor.com