Leveraging MBD, auto-code generation and AUTOSAR to architect and implement an Engine Control Application for series production

Santhosh Jogi, Luigi Milia, Sebastiano Tesio

May 29th, 2018
Index

- FCA Global Powertrain Controls
- Development process overview
- Definition of Software Architecture with AUTOSAR
- Model-based design and auto-code generation
- Challenges in R2016a
- Conclusions and next steps
FCA Global Powertrain Controls

- We’re a global team whose mission is to build propulsion Control Systems, leveraging the specific competences in the organization.

- We’re using common development processes and tools to design, implement and deliver high quality products.

- We’re building the foundation for next generation of FCA Powertrain Control Systems adopting a modular and scalable approach.
Index

- FCA Global Powertrain Controls
- **Development process overview**
- Definition of Software Architecture with AUTOSAR
- Model-based design and auto-code generation
- Challenges in R2016a
- Conclusions and next steps
Centers of Excellence (COEs) are geographically distributed teams to leverage the key competences in each Region.

More than 200 developers spread among Italy, USA, Brazil, India.

Global governance and a rigorous process have been established, to manage the project plan and deliverables across all the stages.

Software Tools have been adopted to implement process and traceability in the development (Application Lifecycle Management).
Development process overview (2/2)

Powertrain Requirements Collection

- COE A
- COE B
- COE C

AUTOSAR Architecture

Model Based Design

Automatic code generation

Software build on production ECU

In-vehicle calibration and final validation

Virtual Verification

Virtual calibration

Hardware-in-the-Loop verification
Index

- FCA Global Powertrain Controls
- Development process overview
- Definition of Software Architecture with AUTOSAR
- Model-based design and auto-code generation
- Challenges in R2016a
- Conclusions and next steps
AUTOSAR Overview (1 / 2)

AUTOSAR (AUTomotive Open System ARchitecture) is an open and standardized automotive software architecture, jointly developed by automobile manufacturers, suppliers and tool developers.

Key features

- **Exchangeability** and **Integration** of functions between car makers and ECU suppliers
- **Increased scope** of automatic code generation and configuration
- **Implementation** and **standardization** of SW architecture and basic functions
- Increased use of Commercial **Off-the-shelf** Hardware

**AUTOSAR** provides a common software infrastructure for automotive systems based on a
- **standardized architecture**
- **application interfaces**
- **defined methodology**
AUTOSAR requires a **complete, exact definition** of the entire SW Architecture, including:

- Hundreds of SW Components
- Thousands of ports and interfaces
- Hundreds of data types, ranges, scalings
- Complete interconnection of SW Components
- Configuration of Real-Time Operating System
- Configuration of the Basic SW

It’s a **tool-driven approach**:

- Architecture Authoring Tools
- Configuration Tools
- Code Generation Tools
AUTOSAR Software Architecture

Interfaces
Components and interfaces view (simplified)

AUTOSAR Software Component

Application Software Component

AUTOSAR Interface

Actuator Software Component

AUTOSAR Interface

Sensor Software Component

AUTOSAR Interface

AUTOSAR Software

Application Software Component

AUTOSAR Interface

Interfaces:

VFB & RTE relevant

RTE relevant

BSW relevant

Possible interfaces inside Basic Software (which are not specified within AUTOSAR)

Standardized Interface

Standardized AUTOSAR Interface

Services

Communication

AUTOSAR Interface

ECU Abstraction

Complex Device

Drivers

Basic Software

AUTOSAR Runtime Environment (RTE)

Standardized Interface

Operating System

Standardized Interface

Standardized Interface

Standardized Interface

Standardized Interface

Standardized Interface

Basic Software

ECU-Hardware
AUTOSAR is a (complex!) methodology to handle complexity.

Some advantages for FCA PWT:

- It provides a formal language to design complex SW systems that can be shared and understood across distributed teams (using the proper tools).

- It requires to define the SW architecture up-front (top-down), so that possible integration problems among components are discovered and fixed early.

- It facilitates reuse of SW components across different applications and different ECUs.

- It increases the scope and amount of automatic code generation, from Application components only to interconnection generation, operating system, communication and basic SW configuration.

- It increases the scope of simulation from Application components only to almost the entire ECU SW → Virtual Verification.
FCA Global Powertrain Controls
Development process overview
Definition of Software Architecture with AUTOSAR
Model-based design and auto-code generation
Challenges in R2016a
Conclusions and next steps
Model-based design and auto-code generation

Architecture

ARXML files

Custom Scripts
#scripts > 200

Custom Library
#blocks ~ 150

Data dictionary

Custom Storage classes

MIL/SIL Simulation

Configurations
MIL/SIL/PROD

Autocoding

Embedded Coder

Code: .c / .h .arxml

Software Integration
Application - RTE - BSW

Production code: s19/a2l

Vector DaVinci

Matlab Simulink

R2015b/R2016a

MATLAB SYSTEMDES

INCA

AUTOSAR
Model-based design and auto-code generation

- **AUTOSAR Software Component:**
  - Runnables
    - Event Triggers
  - Sender / Receiver Ports
    - Inputs/Outputs
  - Variant Management
  - Client / Server communication
Model-based design and auto-code generation

- Internal Behaviour (Control logic)
  - Application of modeling rules

- Usage of FCA Custom Block Library and Simulink Data Dictionary
  - Test Point (linked to Data Dictionary)
  - Calibration (linked to Data Dictionary)
Model-based design and auto-code generation

- Internal Behaviour (Control logic)
  - Access to AUTOSAR server function through «function caller» block

- Look Up Table and Calibration parameters
- An «Harness» Model is required to simulate the Control logics
  - The «harness» contains:
    - Link to referenced model (Autosar Sw-Component)
    - AUTOSAR Servers (Simulink functions)
    - Inputs and Events
    - Outputs
Model-based design and auto-code generation

- Model vs Code verification through simulation with Simulink Data Inspector
  - MIL vs SIL comparisons
  - Documentation report

- Model Coverage is computed to ensure the model has been extensively tested
Model-based design and auto-code generation

- Code Generation
  - A specific model configuration is used
  - Only for referenced model (AUTOSAR Sw-C)
  - Floating point, ANSI C code
    - Only counters and boolean variables in fixed point
  - Deliverables: Source code (.c/.h) + .arxml

- Software Integration
  - Application SW + BSW + RTE (automatically generated)

- SW Configuration and Version management
  - RTC (IBM) is used for workflow instantiation, change management and SW repository
  - 200+ Sw-C’s (Simulink models, generated code, test reports, documentation…)
  - Basic Software (BSW)
  - Custom tools and libraries
Index

- FCA Global Powertrain Controls
- Development process overview
- Definition of Software Architecture with AUTOSAR
- Model-based design and auto-code generation
- Challenges in R2016a
- Conclusions and next steps
Challenges in R2016a

Given to the **complexity** and **number** of models, a close **collaboration** between FCA and MathWorks is necessary to tailor the tool-chain to specific FCA **requirements** and to address tool-chain **improvements**.

Some examples of **expected functionalities** not present in 2016a:

- logging internal signals in SIL;
- support to AUTOSAR Data Type Record across referenced models;
- mature support of AUTOSAR Variants;

A **weekly meeting** occurs to communicate issues and new requests from FCA to MathWorks:

- issues are reproduced and analyzed, **workarounds** or **patches** are provided in the AUTOSAR Support Package;
- new **requirements** are collected, submitted to Mathworks developers and released in the new MATLAB releases (2018a provides the above functionalities).
FCA Global Powertrain Controls
Development process overview
Definition of Software Architecture with AUTOSAR
Model-based design and auto-code generation
Challenges in R2016a

Conclusions and next steps
Conclusions and next steps

- FCA Global Powertrain has embraced AUTOSAR and model-based design to build the foundation of next generation Control Systems.

- Adoption of AUTOSAR enables the design of complex and re-usable SW applications, developed by geographically distributed teams.

- Model-based design enables the development of high quality code, AUTOSAR compliant, through graphical design, simulation and code generation.

- FCA Global Powertrain is further leveraging the AUTOSAR tool-driven approach to building Virtual ECU’s for early verification and calibration of entire ECU production SW in a Desktop PC environment.

May 29th, 2018