Model Based Development and Code Generation for Automotive Embedded Systems

April 26, 2017 | Dr. Gergely Pintér, Dr. Máté Kovács
thyssenkrupp Steering

ingenieuring.tomorrow.together.
# Agenda

<table>
<thead>
<tr>
<th>Model Based Development and Code Generation for Automotive Embedded Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Automotive Environment</strong></td>
</tr>
<tr>
<td>Why do we need sophisticated modeling and code generation tools?</td>
</tr>
<tr>
<td><strong>AUTOSAR Standard</strong></td>
</tr>
<tr>
<td>Brief overview on the modeling language</td>
</tr>
<tr>
<td><strong>ThyssenKrupp’s AUTOSAR Authoring Tool</strong></td>
</tr>
<tr>
<td>Which are the main features and technologies used in our own modeling environment?</td>
</tr>
<tr>
<td><strong>Working with Models</strong></td>
</tr>
<tr>
<td>Editing, automatic model transformation and code generation</td>
</tr>
<tr>
<td><strong>Summary</strong></td>
</tr>
<tr>
<td>Brief wrap-up of the discussion</td>
</tr>
</tbody>
</table>
Agenda

Model Based Development and Code Generation for Automotive Embedded Systems

Automotive Environment
Why do we need sophisticated modeling and code generation tools?

AUTOSAR Standard
Brief overview on the modeling language

ThyssenKrupp’s AUTOSAR Authoring Tool
Which are the main features and technologies used in our own modeling environment?

Working with Models
Editing, automatic model transformation and code generation

Summary
Brief wrap-up of the discussion
thyssenkrupp produces **Electronic Power Steering systems** for premium car manufacturers, where efficient operation, outstanding steering experience and high performance are of paramount importance.
Automotive Environment
Embedded computing resources
Automotive Environment
Embedded computing resources

Steering Column
- Torque measurement
- Input of steering system

ECU with Electromotor
- Steering support
- Output of steering system

Steering Wheel
- User interface of the steering system
Automotive Environment
Embedded computing resources
Automotive Environment
Embedded computing resources
Automotive Environment
Embedded computing resources
32-bit dual core automotive grade, safety qualified microcontroller, executing a sophisticated steering application on a hard real time operating system.
Automotive Environment

Embedded software
Automotive Environment
Embedded software

- Software of a sophisticated steering system is huge…
  - Over 150 software components
  - 20% of them is automatically generated
  - Over 20 basic software modules
    - Nontrivial configuration code automatically generated from communication model
  - Need to integrate 3rd party code (e.g., components provided by the OEM)
  - …around 500k lines of highly optimized safety critical code (MISRA-compliant C and assembly)
- Process, safety and robustness requirements:
  - Automotive SPICE
  - ISO-26262 compliance (ASIL-D)

Strong **modeling foundations** and **code generation** are inevitable for reaching quality and **time to market** goals
## Agenda

### Model Based Development and Code Generation for Automotive Embedded Systems

#### Automotive Environment
*Why do we need sophisticated modeling and code generation tools?*

#### AUTOSAR Standard
*Brief overview on the modeling language*

#### ThyssenKrupp’s AUTOSAR Authoring Tool
*Which are the main features and technologies used in our own modeling environment?*

#### Working with Models
*Editing, automatic model transformation and code generation*

#### Summary
*Brief wrap-up of the discussion*
AUTOSAR Standard
Actually a family of standards

- AUTOSAR is a family of standards
  - Component oriented modeling language
    - Like UML mapped to specialties of the automotive domain
  - Rich basic software library
    - Like standard C library developed for automotive applications
  - Proposal for development method, standardized application interfaces, etc.
  - Way for organizing development and integration activities into a process
AUTOSAR Standard
A sophisticated modeling language for the automotive domain

- Modeling language
  - Components, ports, type-safe interfaces
  - Internal behavior, executable entities mapped to OS tasks
    - Distributed deployment to any number of ECUs in the vehicle
  - Well-defined metamodel and XML-based persistence rules

- Basic software stack
  - Communication, diagnostic functions, memory management and a real-time operating system
  - Well-defined configuration metamodel and XML-based persistence rules

AUTOSAR models in XML files are now the common language between OEMs and suppliers
AUTOSAR Standard
Layered organization of automotive software

Application Layer

Runtime Environment (RTE)

Base Software

Libraries

Microcontroller Hardware
Agenda

Model Based Development and Code Generation for Automotive Embedded Systems

Automotive Environment
Why do we need sophisticated modeling and code generation tools?

AUTOSAR Standard
Brief overview on the modeling language

ThyssenKrupp’s AUTOSAR Authoring Tool
Which are the main features and technologies used in our own modeling environment?

Working with Models
Editing, automatic model transformation and code generation

Summary
Brief wrap-up of the discussion
AUTOSAR Architect

Key features

AUTOSAR Architect

- Requirement editor
- Form editors (SWC)
- Test suite editor
- Model browser
- Rich editors (SWC)
- Configuration editor
- Wizards
- Diagram editors
- Importers and exporters

Graphical user interface

Automation

Command processor
Script interpreter

Code generation

- RTE
- Cpx. SWCs
- BSW₁, Conf.
- BSW₂, Conf.
- MCAL₁, Conf.
- MCAL₂, Conf.

Custom code generators

Variant handling

- Requirement metamodel
- AUTOSAR SWC metamodel
- AUTOSAR BSW metamodel
- Diagram metamodel
- Test suite metamodel
- Consistency checking

Model management

Persistence

Persistence

- Xtend support
- JET + Velocity support
- 3rd party template support
- Custom code generators
**AUTOSAR Architect**

**Key features**

**Persistence**
- Responsible for reading and writing XML files storing models and (ARXML)
- Key challenges: files of several 100MB, support for various schema versions
- Key technologies: XML, EMF
AUTOSAR Architect

Key features

Model management

- In-memory representation of requirements, models, configurations, diagrams, test specifications, etc.
- Provides instant feedback about modeling errors by continuous model consistency checking
- Metamodels of software components and basic software configuration are standard, others are defined by thyssenkrupp
- Key challenges: huge models, several millions of objects, rich interconnection structure, large number of consistency rules
- Key technologies: EMF, IncQuery

Persistence

Graphical user interface

Model management

- Requirement editor
- Form editors (SWC)
- Test suite editor
- Model browser
- Rich editors (SWC)
- Configuration editor
- Wizards
- Diagram editors
- Importers and exporters

Model management

- Requirement metamodel
- AUTOSAR SWC metamodel
- AUTOSAR BSW metamodel
- Diagram metamodel
- Test suite metamodel
- Consistency checking
AUTOSAR Architect

Key features

- **Graphical user interface**
  - Visual representation of requirements, models and configurations
    - Basic editing on forms
    - Rich editors for complex entities (e.g., internal behavior models of components)
    - Visualization of components’ externals and internal structure
    - Wizards and import-export facilities for straightforward interaction with non-AUTOSAR inputs (e.g., CANDB)
  - Key challenges: performance, maintaining consistency between model space and any number of editors
  - Key technologies: JFace, Eclipse Forms, Graphiti

- **Model browser**
- **Form editors (SWC)**
- **Test suite editor**
- **Configuration editor**
- **Importers and exporters**
- **Diagram editors**
- **Rich editors (SWC)**
- **Requirement editor**
- **Wizards**

- **Automation**
  - Command processor
  - Script interpreter

- **Code generation**
  - RTE
  - Cpx. SWCs
  - BSW₁ Conf.
  - BSW₂ Conf.
  - MCAL₁ Conf.
  - MCAL₂ Conf.
  - JET + Velocity support
  - 3rd party template support
  - Custom code generators

- **Key features**
  - Persistence
  - Xtend support
  - JET + Velocity support
  - Custom code generators
  - 3rd party template support
  - RTE
  - Cpx. SWCs
  - BSW₁ Conf.
  - BSW₂ Conf.
  - MCAL₁ Conf.
  - MCAL₂ Conf.

- **Importers and exporters**
- **Variant handling**
- **Graphical user interface**
AUTOSAR Architect

Key features

**Automation**
- Enables running code generators without opening the graphical user interface
- Provides a scripting interface for performing frequently occurring tasks
- Key technologies: JavaScript
AUTOSAR Architect

Key features

- **Requirement editor**
- **Form editors (SWC)**
- **Test suite editor**
- **Model browser**
- **Rich editors (SWC)**
- **Configuration editor**
- **Wizards**
- **Diagram editors**
- **Importers and exporters**

### Graphical user interface

- **Command processor**
- **Script interpreter**

### Automation

### Code generation

**Code generation**

- Significantly increases productivity by replacing error prone manual implementation by automatic code generation
  - RTE: safety critical glue code connecting components to each other and the basic software stack (data transfer, mutual exclusion, etc.)
  - Basic software configuration: CAN / FlexRay configuration, non-volatile memory layout, diagnostic services, network management, resources of the operating system
  - Custom software component generation
- Key challenges: performance, safety relevance
- Key technologies: Xtend, JET, Velocity
# Agenda

## Model Based Development and Code Generation for Automotive Embedded Systems

### Automotive Environment
Why do we need sophisticated modeling and code generation tools?

### AUTOSAR Standard
Brief overview on the modeling language

### ThyssenKrupp’s AUTOSAR Authoring Tool
Which are the main features and technologies used in our own modeling environment?

### Working with Models
Editing, automatic model transformation and code generation

### Summary
Brief wrap-up of the discussion
# Agenda

## Model Based Development and Code Generation for Automotive Embedded Systems

### Automotive Environment
Why do we need sophisticated modeling and code generation tools?

### AUTOSAR Standard
Brief overview on the modeling language

### ThyssenKrupp’s AUTOSAR Authoring Tool
Which are the main features and technologies used in our own modeling environment?

### Working with Models
Editing, automatic model transformation and code generation

- **Basic modeling features**
- **Code generation: synthesis of a basic software module’s configuration code**
- **Model to model transformation: synthesis of a software component**

### Summary
Brief wrap-up of the discussion
Agenda

Model Based Development and Code Generation for Automotive Embedded Systems

Automotive Environment
Why do we need sophisticated modeling and code generation tools?

AUTOSAR Standard
Brief overview on the modeling language

ThyssenKrupp’s AUTOSAR Authoring Tool
Which are the main features and technologies used in our own modeling environment?

Working with Models
Editing, automatic model transformation and code generation

Basic modeling features

Code generation: synthesis of a basic software module’s configuration code

Model to model transformation: synthesis of a software component

Summary
Brief wrap-up of the discussion
AUTOSAR Architect in Action

Layout of the tool
AUTOSAR Architect in Action
Eclipse Forms editors
AUTOSAR Architect in Action

A diagram implemented in Graphiti
AUTOSAR Architect in Action
The Model Navigator

Model structure visualized by EMF Containment Tree
AUTOSAR Architect in Action

Data binding

Seamless synchronization of two views over the same model element ensured by EMF data binding.
AUTOSAR Architect in Action
Composition diagram

Complex component structures visualized on Graphiti-based structure diagrams
AUTOSAR Architect in Action
Checking the consistency of models

Instantaneous feedback about modeling problems provided by the VIATRA engine
Message: sophisticated modeling tools are essential for ruling complexity of automotive software models
Agenda

Model Based Development and Code Generation for Automotive Embedded Systems

Automotive Environment
Why do we need sophisticated modeling and code generation tools?

AUTOSAR Standard
Brief overview on the modeling language

ThyssenKrupp’s AUTOSAR Authoring Tool
Which are the main features and technologies used in our own modeling environment?

Working with Models
Editing, automatic model transformation and code generation

Basic modeling features

Code generation: synthesis of a basic software module’s configuration code

Model to model transformation: synthesis of a software component

Summary
Brief wrap-up of the discussion
We would like to generate configuration code of the Communication (COM) module.
Configuration of the Communication module specifies list and internal structure of PDUs, PDU groups etc.
AUTOSAR Architect in Action

Code generation

Having prepared the configuration we need to set up a run configuration for the code generator.
All module configurations are listed here that are referred by the ECU Configuration Value Collection instance.
AUTOSAR Architect in Action

Code generation

Generated configuration code of the Communication module is around 15-20 kLoc of ANSI-C code, primarily with initialized constant structures…
Code generation

...and executable code which implements packing / unpacking user data to / from frames of the underlying automotive communication bus.
Message: code generation dramatically increases productivity and code quality
Agenda

Model Based Development and Code Generation for Automotive Embedded Systems

Automotive Environment
Why do we need sophisticated modeling and code generation tools?

AUTOSAR Standard
Brief overview on the modeling language

ThyssenKrupp’s AUTOSAR Authoring Tool
Which are the main features and technologies used in our own modeling environment?

Working with Models
Editing, automatic model transformation and code generation

Basic modeling features

Code generation: synthesis of a basic software module’s configuration code

Model to model transformation: synthesis of a software component

Summary
Brief wrap-up of the discussion
For safety relevant data we have to apply and check end-to-end data protection which is implemented in **software components** (called communication wrappers); most of the **model and code** of these components can be generated...

...thus we instruct the tool to automatically construct a software component

```
Configure NvBlockSwComponentType-s in NvM
Extract Model Elements
Create/Update Ports, Interfaces, DataTypes, CompuMethods and Functions
Create/Update Communication RxWrapper configuration
Create dummy GroupSingal-s in receive ComSignalGroup-s for...
```
AUTOSAR Architect in Action
Automatic Model Transformation

We can automatically synthesize a model of data types and interfaces based on the information available in the communication model…
AUTOSAR Architect in Action
Automatic Model Transformation

...AUTOSAR model of communication wrapper components (including interfaces, ports, data access etc.)...
AUTOSAR Architect in Action
Automatic Model Transformation

…and finally source code of these components which implement end-to-end protection (4-5kLoc in case of typical projects)
Message: transformation reduces time necessary for maintaining systematically constructible models
# Agenda

## Model Based Development and Code Generation for Automotive Embedded Systems

### Automotive Environment
Why do we need sophisticated modeling and code generation tools?

### AUTOSAR Standard
Brief overview on the modeling language

### ThyssenKrupp’s AUTOSAR Authoring Tool
Which are the main features and technologies used in our own modeling environment?

### Working with Models
Editing, automatic model transformation and code generation

### Summary
Brief wrap-up of the discussion
Summary

• Automotive application domain implies serious design quality requirements
  – Addressed by building steering software on AUTOSAR standard
• In-house developed authoring tool: AUTOSAR Architect
  – Supporting all activities of Model Driven Development
  – Built on various state of the art Eclipse technologies