Simulation of Complete Systems at ZF using Modelica Standards

Jochen Köhler[1] Julian King[2] Michael Kübler[3]

[1]R&D, ZF Friedrichshafen AG, Germany, jochen.koehler@zf.com
[2]R&D, ZF Friedrichshafen AG, Germany, julian.king@zf.com
[3]R&D, ZF Friedrichshafen AG, Germany, michael.kuebler@zf.com

Abstract

In this paper we describe how ZF is using existing and upcoming Modelica standards for simulating a variety of systems in automotive industry. In particular, Modelica is employed for driveline modeling. The FMI standard is used to transport models over tool boundaries. The novel SSP standard will contribute towards interconnecting FMUs and defining complete system architectures.

Keywords: Longitudinal vehicle dynamics, FMI, SSP, Parameterization, system architecture, autonomous driving

1 Introduction

ZF is a major automotive supplier, with an extensive product portfolio ranging from driveline and chassis technologies to active and passive safety systems for autonomous driving. In this context, detailed simulation models mimicking vehicle dynamics and the involved actuators/sensors are vital for developing reliable control software as well as for supplementing time- and cost-intensive test bed measurements. Simultaneously, in order to optimize the interplay between these components, the demand for holistic simulation approaches is steadily increasing. Complete system analysis is becoming recognized as an important factor for fast and robust system engineering activities as it allows for directly examining the global system behavior and for identifying relevant feedback loops. Modelica is especially suited to this task due to its inherent multi-physics capabilities. Furthermore, the associated Functional Mock-up Interface (FMI) standard facilitates an efficient integration and coupling of specialized sub-models (possibly developed in other tools), thereby going beyond an isolated analysis of single components.

2 Modeling drivelines with Modelica

Facing an enormous variety of driveline concepts and continuously decreasing innovation cycle times, simulation has become a backbone for developing gearbox control software at ZF. In particular, Modelica was introduced in our company over ten years ago and nowadays represents a standard approach for modeling a wide range of distinct transmission and related actuator concepts. Building on the freely available Modelica Standard library, more than ten context-specific ZF Modelica libraries have been created so far. These in-house libraries are made accessible throughout the company and preserve corporate modeling know-how in several highly relevant areas, such as

- Transmission and driveline components (Köhler, 2005)
- Extensions for hybrid and electrical powertrains (HEV/PEV/EV), including complete hybrid driving strategies (Köhler *et. al.*, 2006)
- Model parameterization tools and export templates to other platforms (Kellner, 2006)
- Combustion engine dynamics and exhaust after-treatment (Kuberczyk, Köhler, 2013)
- General mechatronic solutions, hydraulic and pneumatic actuators (based on Modelica Fluid) (King *et. al.*, 2014)

The above-mentioned central libraries are consequently re-used within distinct modeling projects, thereby generating thoroughly tested component models characterized by a high degree of reliability and robustness. Also, many component models are available in varying levels of detail and function, from highly resolved descriptions encompassing all relevant physical effects to simplified formulations optimized for real-time use (King *et. al.*, 2014).

A major advantage of the object-oriented modeling approach underlying Modelica is that it allows for an easy exchange of such component variants without having to modify other parts of the global model, thereby offering the possibility to quickly switch between different model configurations. Furthermore, Modelica models directly reflect the physical structure of the system under scrutiny. This fact enables a rapid transfer of system knowledge into model equations. On the other hand, also some shortcomings have to be mentioned. Due to the symbolic simplification of the system equations by the Modelica front-end tool, efficient debugging is difficult and the transparency of

DOI
10.3384/ecp1612424

the generated run-time C-Code is limited. This hampers the development of complex functions as compared to scripting languages. Moreover, to the authors' experience, Modelica is not yet widely known among young engineers. For all these reasons, further efforts are necessary to improve the interaction between Modelica models and models/methods originating from other tools. The Functional Mock-up Interface (FMI) offers great promise in this regard.

3 Using FMI to modularize system components

Modelica models can be exported to a wide range of simulation tools – usually via C-Code. At ZF this process has been extended to support even tools developed internally. To enhance the flexibility for the usage of simulation tools and to reduce the effort for maintaining this process there is a need to use established standards.

FMI satisfies this need since there is a broad list of tools supporting FMI. We also enriched internally developed tools to support FMI, which are used for system simulations, e.g. in case of CO2 analysis or software development and test. They are able to handle FMI 1.0 and 2.0 both in model-exchange and co-simulation mode. The verification of this FMI interface is done by applying the FMI cross-check rules.

In the past mostly monolithic models have been used as FMU, but recently the demand for a modular setup is increasing, internally and with customers of ZF. This is motivated by two requirements on virtual product development: efficiency and quality. Both requirements can be fulfilled if models, once built up by a modeling expert reflecting all needed physical effects, are reused and exchanged amongst different departments or even companies.
A crucial point here is a feasible definition of the simulation architecture for all relevant use cases. Therefore one needs to think about a proper definition of the interface signals, in order to

- enable an easy integration of existing models
- replace models of different levels of details
- regard existing solutions (e.g. within other simulation tools) inside the company and with customers.

The usability must not be neglected; therefore our aim is to decompose the overall system to smaller, but still reasonable sub-systems. These sub-systems mostly represent the components developed by the collaborating partners, e.g. in the case of analyzing longitudinal dynamics models of a combustion engine, a transmission system and vehicle dynamics, as shown in Figure 1.

When cutting tightly coupled systems to modules also numerical issues such as algebraic loops or stiff systems must be considered. FMI for model-exchange might be an option for such a simulation setup, but FMI for co-simulation is in focus because it enables the comparability of the simulation results and shows a feasible performance in most tools. Figure 1 shows an example of a modular setup used for software development and test. The focus of the modular setup currently is on physical or simplified logical models but will also be extended on virtual ECUs.

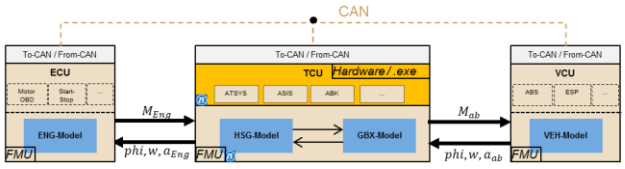

Figure 1. Structure of a driveline

This definition of interfaces needs discussions and it is good to take all parties into account. Therefore parts of those discussions are also handled in a cross-company approach, e.g. the "Smart Systems Engineering" project of the ProSTEP iViP Association (cf., http://www.prostep.org/en/projects/smart-systems-engineering.html).

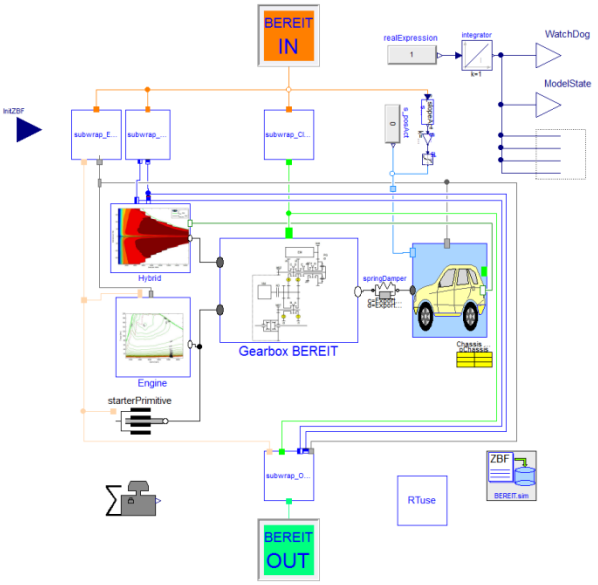

Figure 2. Model structure of the BEREIT range extender transmission concept in Dymola

The use of FMI to exchange behavioral simulation models has become an official standard in ZF. A prototypic example is the joint research project "BEREIT" (Bezahlbare Elektrische Reichweite durch Modularität und Skalierbarkeit – Affordable Electric Range by Modularity and Scalability), supported by the German Federal Ministry of Economics and Technology (cf., http://pt-em.de/de/1508.php). The goal here was to develop gearbox control software for a modular range extender concept for electric vehicles, featuring three possible operation modes: a purely

electric mode, a hybrid mode with load point shifting, and an "EDA" mode, which allows for charging the battery while the vehicle is in standstill or driving at low speeds (Roske *et. al.*, 2006). In order to minimize transmission losses, only dog clutches (which are synchronized by the electric motor) are used as switching elements for shifting gears.
Figure 2 shows the top-level model in Modelica/Dymola, including blocks for loading ZBF parameterization data as explained below (see chapter 5), as well as two connectors containing all input and output signals for communicating with the control software. This minimal interface remains the same for all model configurations, while the driveline modules may be freely exchanged. Typically, either the whole model or the gearbox part is then compiled as a co-simulation FMU to be imported into the in-house SiL-platform SOFTCAR (see below), which has been enabled to handle both FMI 1.0 and 2.0.

In Modelica an extensive use of expandable connectors is used to exchange measured signals e.g. between controller and plant model. Unfortunately this approach analog to a CAN-Bus in real vehicles currently cannot be obtained when switching to FMI. Therefore wrapper models and mapping functionalities are needed for the seamless integration of those FMUs, which will be addressed by the Modelica SSP project, as mentioned below.

Establishing FMI as the all-in-one solution for model exchange in ZF is still on-going since all the processes need to be mature enough and all issues which arise with the use of FMI need to be solved.

One major drawback so far is the lack of proper protection of the intellectual property within the exported FMU. To overcome this we established a post-procedural encryption with the following options:

- selection of signals and parameters to hide
- definition of a period of validity
- checksum usage to guard against manipulation

Another main issue is the consistent parameterization of a system simulation setup by a bunch of modular sub systems, which motivated the SSP ("system structure and parameterization") project.

4 Using "SSP" to specify system architectures

4.1 Motivation

Same as for the products itself that ZF offers, each simulation model has to be provided to several customers. The goal is to reuse the same simulation model of a certain product as a FMU. The challenge here is the adoptions that are needed for any customer, because there is no standard way to couple things especially in the area of personal cars. One could think of including the "kernel" FMU into a wrapper with all the signal modifications but this means quite an overhead, because usually these modifications are quite simple (linear manipulation or mapping tables for discrete values).

In Addition to that it has to be taken into account that also the receiver of the simulation model (e.g. the customer but also another department in ZF) has to integrate this FMU into his complete system. Assuming that the complete system is built from several FMUs as component representation the main additional information that defines the system architecture is the connection of all FMUs and a possible hierarchical arrangement. This requirement isn't focused by the FMI standard.

As a consequence a new Modelica Association project called "System Structure and Parameterization" (abbreviation SSP) was initiated after the Modelica Conference 2014 in Lund. One main goal of this project is to define a tool independent format to be able to specify the structure of a simulated system. So one has to define this structure only once in any (authoring) tool and can transfer it to any (integration) tool to include the system components and simulate the system. This is especially interesting when a system is simulated on different platforms like MIL, SIL or HIL.

First results of the project were presented 2015 at the Modelica Conference in Paris. There is a first draft for defining system structures. A XML schema was developed therefore. Three prototypes of tools were presented that are able to read and write such files.

ZF is developing its own software for integration of physical models of driveline with ECU software code called SOFTCAR in parallel to commercial ones. On top of that another tool is being developed that generates complete simulation models for the simulation platforms SOFTCAR and dSpace® out of selected FMUs, several (CAN-)bus specifications and parameterization definitions. For handling all these system architectures and parameterization data, the SSP approach will be used.

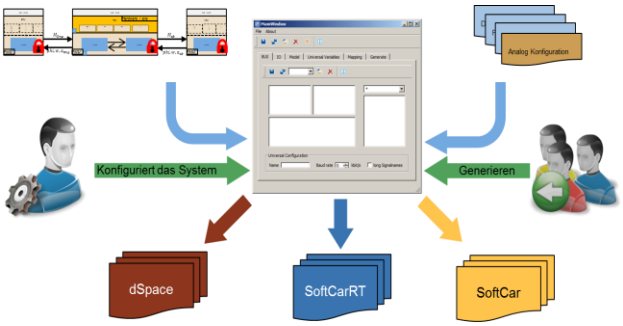

Figure 3 Sketch of mapping tool

The element in the SSP XML schema called "signal dictionary" is here very useful for connecting inputs and outputs of FMUs not directly but with a separate instance where you can define project independent

DOI
10.3384/ecp1612424

signal names that can be reused in many projects. Signal dictionaries can also be thought as "special" version of FMUs without any dynamics. Another useful feature is that the signal dictionaries can be used easily for visualization of these signals.

4.2 SSP for simulation of autonomous vehicles

Usage of SSP is also intended in ZF for developing reusable system architecture for ADAS (advanced driver assistance systems) and autonomous vehicles. In these systems there's a lot of feedback and interdependencies between all involved components. So the need of being able to simulate not only the isolated components but also the complete system is evident for a fast development process.

The system architecture is quite complex and the variety of combining components is very high. ZF offers a lot of components needed for autonomous driving:

- Sensors
  - Radar
  - Cameras
- Actuators
  - Steering
  - Braking
  - Transmission
  - Electric Machines for traction
- (Active) chassis components
- High performance ECUs
- Functions for autonomous driving

There's a little gap to end up with a complete virtual system by adding the missing parts.

- Engine
- Car body
- Driver
- Environment

The first three parts can be modeled quite simple for the purposes here. Having a model of the environment is very important and this is probably provided by a tool vendor.

4.3 Requirements

The system architecture to be defined has to fulfill the following requirements:

- It must fit to real systems.
- Defined modules have to correspond to real components.
- Variations of modules can be exchanged – also by real components on HiL-platforms.
- It must be possible to include also modules from OEM / other suppliers.
- The numeric coupling of modules must be robust for efficient and stable simulations.
- It must be possible to both simulate the complete systems and fragments of it.
- It must be usable on a variety of platforms (MiL, SiL, HiL).

Having this system architecture also allows extracting single components as empty template. You can give this template to somebody else for implementing. When finished, this new component model can be easily integrated in the complete system.

Of course this work just starts now. But it gives a good overview of the opportunities by using the SSP approach.

5 Using SSP for parameterization

The project "System Structure and Parameterization" also focuses to parameterization of components. In the FMI standard, the dynamics of a component and its parameterization is not separated. We experienced that this approach can be inconvenient sometimes. Another problem of this entity is our requirement to use the ZF internal standard for parameterization called "ZBF". This is a simple format of ASCII files to specify simulation parameters separately from simulation models and tools. Already before using FMI we used it to be able to parameterize models just during the initialization.

The advantages are:

- A Model can be provided with multiple parameter sets without the need to rebuild it.
- The parameter sets can be either readable or encrypted.
- Parameter definitions can be used for multiple components without duplication ("single source" pattern).
- Parameters can be hided to the model user if wanted.
- Parameters can be modified by the model user even though the model is a black box, if wanted (Intellectual property issues)

SSP can be used to have all these features without being forced to give up the internal standard because there will be a functionality to implement your own adapters to the SSP API. The part of parameterization of SSP is under development at the moment. As soon as there is a first version, ZF will implement its own adapter. So it's possible to take benefit of the other coming features of SSP:

- The possibility to handle parameters within any authoring tools that support SSP seamlessly.
- Use same parameterization approach also for entire system models with many components (either FMUs or proprietary models)
- Enrich parameterization data by meta data
- Handle IP issues for parameterization

- Efficient handling of complex parameters (lookup tables etc.)

SSP brings its own data format as a XML definition. But with the possibility to implement your own adapter it's also possible to transfer proprietary formats to the SSP standard format.

6 Outlook

For ZF it is important to contribute to Modelica projects like FMI and SSP to be sure to get powerful standards and reliable tools supporting them in order to do the jobs that have to be done. The number of simulation tasks will grow rapidly in future, so it's inevitable to be efficient in this context.

Also the cooperation of industrial users on these projects benefits everybody; industry gets tools that fit their needs and tool vendors can offer more attractive software.

Especially the work on SSP is not finished yet. Some effort has to be made to bring the upcoming standard to a mature status so it can be used in daily business. We try hard to make this happen quickly by evaluating the results very early to get experience with it. Another important point is the close cooperation with the FMI project group.

Acknowledgements

Thanks to the Modelica Design group for being open to suggestions and requirements of industry.
The great engagement of SSP project members helps a lot to create and establish the new standard.

References

M. Kellner and M. Neumann and A. Banerjee and P. Doshi. Parameterization of Modelica Models on PC and Real Time Platforms, *Proceedings of the 5th Modelica Conference 2006, pp. 267-274*, Vienna, Austria , 2006

J. King and J. Köhler and M. Kübler. Multi-platform simulation models for the development of transmission control software in commercial vehicles. *Simulation and testing for automotive electronics V, pp. 56-65*, 2014

J. Köhler and A. Banerjee. Usage of Modelica for transmission simulation in ZF. *Proceedings of the 4th Modelica Conference 2005, pp. 587-592*, Hamburg, Germany

J. Köhler and T. Mauz and J. Schnur. Systematische Entwicklung von Simulationsmodellen und Fahrstrategien für hybride Antriebe. *VDI Berichte, pp. 473-500*, 2006

J. Köhler and R. Kuberczyk. Simulationsverfahren zur Auslegung eines PHEV-Antriebsstrangs. *VDI-Tagung Plug-In-Hybride*, 2013

R. Kuberczyk and J. Köhler and S. Blattner. Benchmark of Saving Potentials of Diesel-Hybrid Vehicles. *22. Aachener Kolloquium*, 2013

M. Roske and F.-D. Speck and S. Kerschl. Das elektrodynamische Anfahrelement - ein Hybridantrieb mit erweiterter Anfahrfunktionalität. *VDI-Tagung Getriebe in Fahrzeugen*, 2006.

DOI 10.3384/ecp1612424