Co-Simulation of Components, Controls and Power Systems based on Open Source Software

Matthias Stifter, *Member, IEEE*, Edmund Widl, *Member, IEEE*, Filip Andrén, *Member, IEEE*, Atiyah Elsheikh, *Member, IEEE*, Thomas Strasser, *Senior Member, IEEE* and Peter Palensky, *Senior Member, IEEE*
AIT Austrian Institute of Technology, Energy Department, Vienna, Austria
{givenname.surname}@ait.ac.at

***Abstract*—There exists no universal tool to analyze the increasing complexity in smart grids. Domain specific simulation and engineering tools partly address the challenges of complex system behavior. Different component technologies, customer behavior and controls in the power networks are interacting in a highly dynamic manner. Results of isolated simulations may be not accurate enough on the system level. Free and open available tools like GridLAB-D, PSAT, OpenModelica and 4DIAC are well known and widely used because of their excellent domain specific expertise. With co-simulation approaches the individual strengths of each tool can be exploited to model and simulate the various aspects of complex smart grids. The achieved level of detail and realism potentially surpasses the results that the individual analyses would gain.**

This paper demonstrates a local smart charging control strategy implemented with the IEC 61499-based standard for distributed control systems. It is simulated with different electric vehicle driving patterns, modeled with the multi-agent environment GridLAB-D. Battery models are defined in OpenModelica and embedded as individual dynamic loads. The power system is simulated using PSAT. This work shows that boundaries and restriction in terms of modeling cross-domain specific problems can be overcome by coupling these open source applications.

***Index Terms*—Open source software, Power system analysis computing, Power system simulation, Smart grids.**

I. Introduction

Over the last decades excellent domain specific simulation tools have been developed to simulate dynamic system behavior for power and energy systems. Each one has its own right to exist, since they all focus on different phenomena, e.g., electrical systems, cyber-physical systems, controls or stochastic environmental behavior. These tools make it possible to model, simulate and understand the related problems. However, by coupling these existing simulation tools, it is possible to extend the systems and model more complex dynamic behavior.

The main contribution of this work is to show that a complex problem can be modeled and split up in separate subsystems and simulated using open source solutions. The proposed coupling of these simulation environments enables the investigation of interacting dynamic systems. Understanding and simulating these systems is essential for the development of intelligent energy systems.

This work is partly funded by the Austrian Climate and Energy Fund - Project DG-EV-HIL.
Invited Paper for the TF on OSS: "Open Source Tools for Smart Grid Applications", IEEE PES General Meeting 2013.

This paper is organized as follows: Section II provides an overview of the related work regarding co- & hybrid simulation with a focus on Open Source Software (OSS) in the domain of intelligent energy systems. In the following Section III a co-simulation framework based on OSS is introduced. The usage of this simulation environment is demonstrated on a smart charging case study in Section IV. Finally, in Section V the conclusions are provided.

II. Related Work

A detailed simulation of energy systems, including controls and active components, normally requires hybrid models, i.e. models combining aspects of both continuous time-based (physics-related) and discrete event-based (controls-related) simulation approaches. Unfortunately, such simulations are a challenging subject [1], and only a few tools are available to accomplish this task.

Today, the most popular tool in this direction is the proprietary simulation environment Simulink (developed by MathWorks), which works atop of MATLAB and allows to study hybrid models in a quite convenient way. However, Simulink is not able to cover the full complexity of typical energy systems all by it self, but needs at least to be extended with other proprietary toolboxes [2].

An alternative to all-in-one tools like Simulink is the development of customized co-simulation frameworks, that meet the specific needs of a certain scenario or application. Such an approach has the advantage that it allows to use the most appropriate expert tools for each domain. The disadvantage is simply the difficulty to make different tools work together properly and efficiently. Concerning the simulation of power systems, today only a few approaches and tools in this direction exist [3]–[5].

The co-simulation framework presented here is based on the approach proposed by [4], with the proprietary components replaced by standard open source solutions that have already been proven useful for power system applications [6].

III. Co-Simulation Tool Chain based on OSS

An overview of the different problem-oriented simulation tools and their corresponding interfaces that are integrated in the proposed co-simulation approach is shown in Figure 1. This environment enables a dynamic simulation of otherwise statically modeled behavior, like smart charge control, ancillary services, traffic patterns of EV or battery technologies.

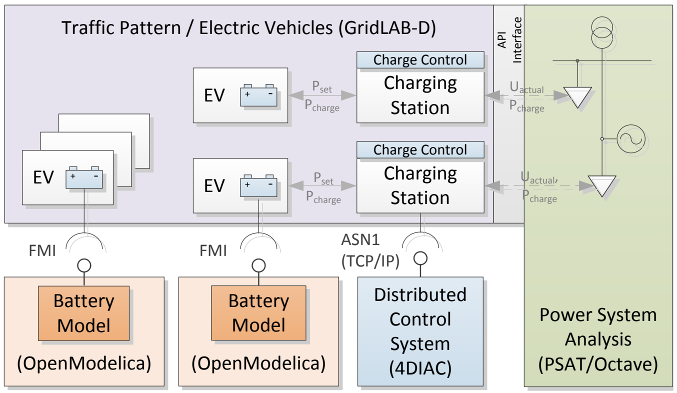

Fig. 1. Overview of the co-simulation approach based on OSS and interfaces.

A. *Discrete event-based micro-simulation*

The core of the co-simulation framework is GridLAB-D [7], which is responsible for the simulation control, i.e. the correct and deterministic execution of all components. GridLAB-D is an open source tool for discrete event-based micro-simulation that aims primarily at the analysis of power distribution systems. It provides several plug-in modules for simulating energy generation, distribution and consumption as well as related topics such as controls, network communication or markets. Its design also aspires to be flexible, in order to facilitate the coupling with other tools and simulation environments.

For the use-case presented below, a new plug-in has been developed that allows to represent all the components of the simulation by a dedicated object in GridLAB-D. These objects are responsible to update their own internal states according to the global simulation time. GridLAB-D's simulation core looks for changes in the states of these objects during runtime (discrete events) and enables the synchronized interaction between them. Each object has to inform the simulation core when the next change of its internal state is supposed to happen, assuming all other objects remain in their current state. Therefore, every time an event occurs, all objects have to be synchronized.

B. *Continuous time-based physics simulation*

Physical components typically have to be described by systems of continuous time-based differential algebraic equations. For the co-simulation framework presented here, the corresponding models are implemented with the help of OpenModelica [8], an open source general-purpose multi-domain (physics) simulation environment. It comes with a large standard library of components from various domains, e.g. electrical, thermal, mechanical or control-oriented elements, facilitating the rapid prototyping and development of complex models.

OpenModelica is based on the open standardized Modelica language, which is an object-oriented, equation-based language dedicated to multi-domain modeling. In contrast to a block-diagram approach (e.g. Simulink), it relies on universal acausal modeling concepts (e.g. energy conservation laws in physical domains). This approach has led to a considerable progress in the simulation of complex heterogeneous systems, i.e. systems comprising of elements from different simulation domains [9]. Hence OpenModelica is a particularly useful asset within the co-simulation framework in order to model real-world components conveniently and efficiently.

C. *Power system simulation*

For the analysis of the power system the open source tool PSAT [10] is used, a toolbox available for MATLAB/Simulink and for GNU Octave. It provides a variety of static and dynamic component models to support accurate power system modeling, including loads, machines, controls and regulating transformers. Among other things, PSAT is capable of steady-state analysis, like power flow, optimal power flow, small signal stability and time domain simulation.

When used as a stand-alone application all operations can be assessed by means of graphical user interfaces. However, for the purpose of the co-simulation framework presented here, PSAT is used in command line mode in GNU Octave.

To make a coupled simulation possible, the the power flow calculation must be controlled and invoked from the simulation control to synchronize the simulation. Changes to the power system state are possible by accessing the appropriate model parameters (e.g. transformer tap position, active and reactive power of the loads). By accessing the resulting matrices (e.g. voltages, power) it is possible to communicate actual model states to the interfaced applications. Simulation control is responsible for advancing to the next time step, where the power flow calculation is repeated with the respective changes of the model.

D. *Distributed automation and control system*

The aim of the chosen approach is to specify control algorithms in a standard-based way and to enable co-simulation to emulate the smart grid behavior. This emulation is necessary at different development steps, starting from control algorithm development and optimization to the validation of the implementation on an embedded control device. Due to the distributed nature of smart grids and their components, this typically applies also to the control system [11].

The control approach applied in the context of this paper is based on the IEC 61499 reference model for distributed automation [12]. This international standard covers the definition as well as the implementation phase of embedded controllers in general. In the German smart grid standardization roadmap the IEC 61499 refence model is suggested as a promising candidate for implementing smart grid controls [13]. Especially as implementation language for logical nodes according to the interoperability and communication standard IEC 61850 [14] the IEC 61499 reference model has big potential.

One of the major benefits using IEC 61499 lies in the fact, that the corresponding control models can be considered as executable models. This means they can be executed within a simulation environment the same way as they are executed on an embedded device, apart from minor adaptions [6]. In the OSS-based co-simulation environment proposed in this paper the IEC 61499 compliant OSS 4DIAC is used, since it provides an open and extensible control environment [15].

E. Simulator coupling using standardized interfaces

One of the main challenges was the coupling of GridLAB-D's discrete event-based simulation with the continuous time-based Modelica models. The prerequisite to do so is OpenModelica's ability to export self-contained stand-alone components according to the open standardized interface specification Functional Mockup Interface (FMI) for Model Exchange [16], the foundation for a new trend of co-simulation-based applications [17]–[19]. The FMI standard defines the API of the stand-alone components (so-called Functional Mockup Units) and provides the functionality to create and run instances of a model. These self-contained components are used to accomplish the coupling to GridLAB-D. This is done by deploying a dedicated FMI wrapper developed for the synchronous interaction between a discrete event-based environment and a continuous equation-based component.

The coupling between GridLAB-D and PSAT is technically rather simple, since GNU Octave's code base is completely open source. This allows to incorporate all functionality concerning matrix creation and manipulation as well as higher-level command parsing (e.g. evaluation of function calls) directly into other applications. The co-simulation framework uses a thin wrapper layer in order to enable GridLAB-D to use C arrays instead of GNU Octave's built-in data types. In addition it provides some other convenience functionalities (e.g. changing of working directory or execution of scripts).

For the coupling of the distributed control environment 4DIAC with the co-simulation framework the abstract Client/Server communication pattern is used. This pattern is represented by a pair of two IEC 61499 communication function blocks using the Abstract Syntax Notation 1 (ASN.1) specification over TCP/IP sockets (as defined in the IEC 61499 standard) for the data transfer. In the present case the IEC 61499 compliant controller acts as data server and dedicated GridLAB-D objects as the clients. The main reason for using such a simple communication protocol for the exchange of data instead of the FMI specification lies in the fact, that the same controller model can also be used in an embedded control environment with some small modifications. Only the communication function block has to be exchanged with a function block interacting with the real world (e.g. via analogue or digital I/Os).

F. Compiler, libraries and platform issues

The co-simulation framework has been implemented on a commercially available server running Windows 7 Enterprise in a virtual machine. Since both OpenModelica and GNU Octave use the open available MinGW development environment, GridLAB-D has been accordingly modified to compile under MinGW (GCC 4.6.2) in order to avoid compatibility issues. For the communication with 4DIAC the open Boost.Asio library has been used (also compiled with MinGW GCC 4.6.2).

The 4DIAC environment is available for different PC (Linux, Win32, Windows Cygwin) and embedded platforms/environments (VxWorks, ThreadX, eCos). For its usage in the co-simulation environment the native Win32 porting has been employed.

Since there are open versions of all the utilized tools and libraries available for Unices, it is definitely possible to port this co-simulation framework e.g. to Linux. However, this has not been done so far.

IV. Use-case: Smart Charging

The time base of the investigated phenomena in this work is in the scale of seconds up to minutes and fractions of an hour. For this reason the power system model is simulated in steady-state for every time step. It is sufficient to model non-dynamic loads with static profiles, as is usually the case in combination with standardized statistic profiles. When it comes to dynamic behavior like in the case of vehicle batteries, where the actual state of charge (SOC) depends on the amount of power and time spent charging, the model has to reflect the according changes accurately. Examples of such dynamic behavior are: effects of the limited charging power on the electric vehicle's range, available energy stored in an electric energy storage, changes in demand due to environmental temperature.

A. Description of the use-case

As a simple use-case, a local voltage control algorithm prevents voltage violations due to charging of electric vehicle batteries. The controller can influence the amount of active power charged into the battery.

1) EV driving pattern: The driving patterns of the individual EVs, i.e. their sequences of parking, driving and charging, are based on real-world data of 6 locations for the year 2011 of a commercial car rental provider, including approximately 7700 trips. The data has been processed to extract statistical information regarding e.g. departure and arrival times or journey distances. This allows to deduce (simplified) distributions of the relevant quantities which can be used to generate individual driving patterns for each EV dynamically during run-time.

2) Battery model: The battery of each EV is simulated independently, using a Modelica model of a Li-Ion battery, accurately implementing the constant current / constant voltage charging phases, created with a professional library dedicated to the description of the dynamic behaviour of batteries (Electric Energy Storage library) [20]. When connected to a charging point, a charging profile depending on the battery's SOC is applied, that reduces the charging power towards zero as soon as the SOC exceeds a certain threshold.

3) Smart charging controller: With the help of 4DIAC a smart charging controller has been implemented for each charging station. The smart charging controller acts as an addition to a normal charging station by providing local ancillary services that can be used by the grid operator. In this case the ancillary service is a local voltage controller, which can prevent voltage violations by limiting the amount of active power charged by the EV battery. The set points used by the controller can be configured by the grid operator.

The IEC 61499 application used for one charging station, in this case charging station 1 (`CP1`), is shown in Figure 2. A `SERVER` function block is used for the communication with GridLAB-D.

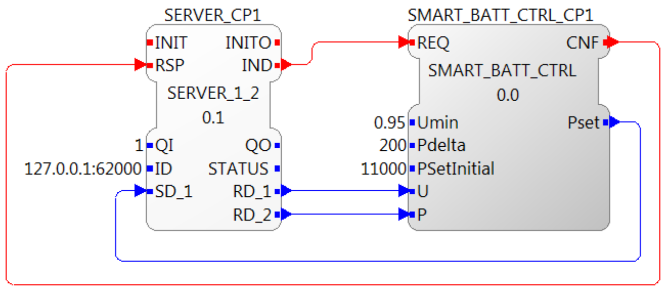

Fig. 2. Function block network for smart charge controller

Each charging controller uses a simple charging algorithm, which is implemented in the basic function block `SMART_BATT_CTRL`, shown in Figure 2. The purpose of this function block is to decreases the active power charged by the battery in case of under voltage in the grid, as depicted in Figure 3.

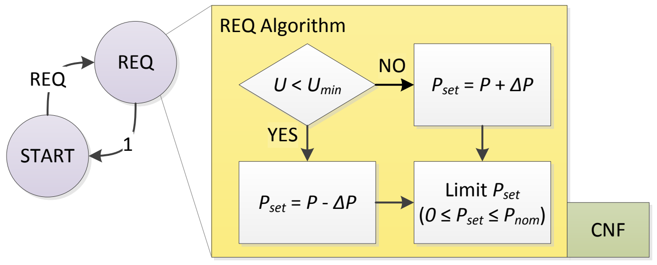

Fig. 3. Simple charging algorithm

The execution of the function block is controlled by two states, i.e. `START` and `REQ`. An `REQ` event triggers a transition to the `REQ` state where the algorithm is executed. In case an under-voltage is detected the set point for the active power of the battery P_{set} is decreased with ΔP. This is done each time the function block receives a `REQ` event and as long as the voltage is below the limit U_{min}. Once the voltage is back in an acceptable range P_{set} is increased. After the algorithm is executed a `CNF` event is triggered by the function block and the internal state machine returns to the `START` state.

4) Electrical network: For the demonstration of the co-simulation concept a simple low voltage network model has been selected. Loads represent households and charging points, modeled and simulated with PSAT in GNU Octave. The loads of the household are represented by static profiles for active and reactive power and the power needed for charging the electric vehicle batteries is determined by the battery model, depending on the actual battery SOC and controller state.

Figure 4 shows three low voltage (LV) networks connected via a transformer to the medium voltage (MV) networks. The charging points (CP) are also modeled as loads and are dynamically coupled to the battery via the smart charging controller.

B. Results from simulation

Figure 5 shows the results of a simulation run, where the uncontrolled charging (blue circles) of an EV causes the voltage level to drop below the threshold. The comparison with the controlled case (green crosses) shows that a reduction of the charging power, according to the algorithm described above, indeed mitigates the effect. Note the increased number of updates in the controlled case. During this phase the controller notifies the simulation core that the internal state needs to be updated more frequently, thus increasing the simulation step resolution. This allows to investigate interesting dynamic effects more precisely.

In addition, Figure 6 shows the according impact on the charging point's power consumption (above) and the battery's SOC (below) over time. The consequences of the variation in the charging power can be clearly seen as a dynamic response in the battery's SOC evolution. Note also the adaption of the charging power in correlation with the voltage levels caused by the base load (see Figure 5).

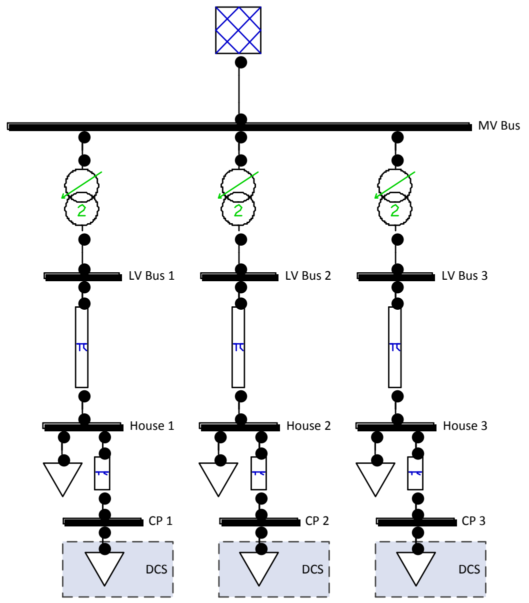

Fig. 4. Simple network model of three households with charging points.

V. Conclusions

This work presents a prototype platform for the co-simulation of components, controls and power systems based on open source software. The core of the framework is provided by GridLAB-D, OpenModelica is used to model complex components, power systems are simulated using PSAT and controls are introduced via 4DIAC. This prototype intends to be an important development step towards a mature open source software environment suited for the simulation-driven development of controllers for realistic applications.

By applying this co-simulation approach to a smart charging case study in the context of electric vehicles, its possibilities have been demonstrated.

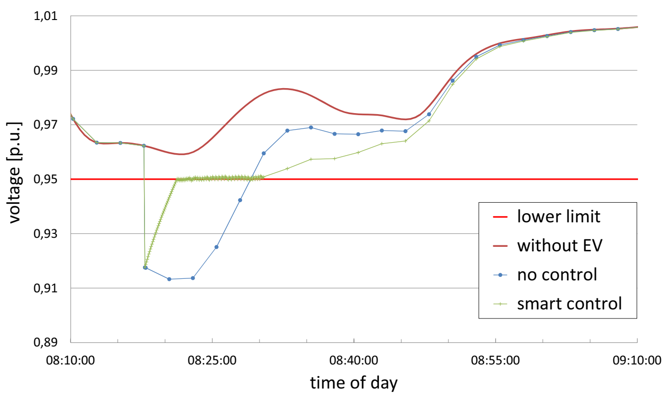

Fig. 5. Evolution of voltage levels with time.

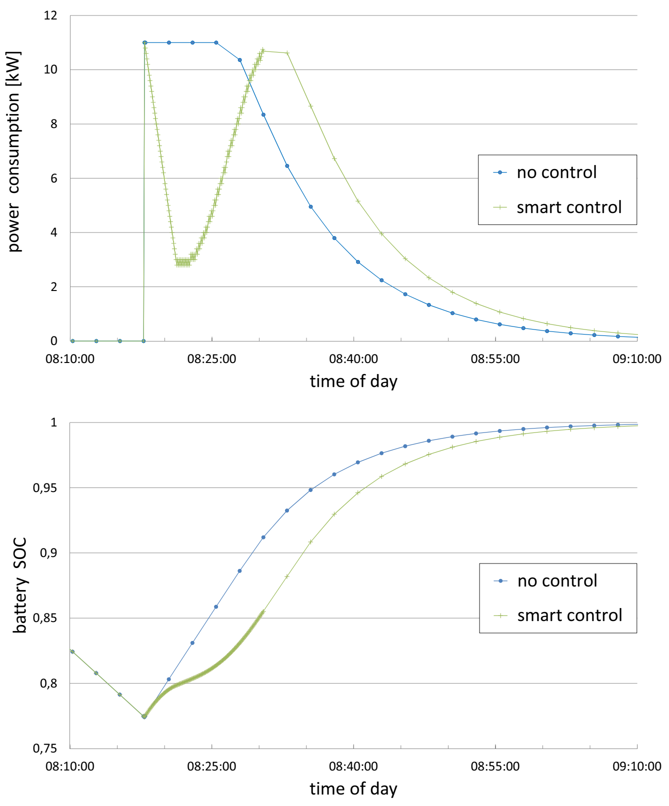

Fig. 6. Evolution of the battery's power consumption and SOC with time.

References

[1] P. Palensky, E. Widl, and A. Elsheikh, "Simulating cyber-physical energy systems: challenges, tools and methods," *submitted to IEEE Transactions on Systems, Man and Cybernetics*, 2012.

[2] E. Widl, P. Palensky, and A. Elsheikh, "Evaluation of two approaches for simulating cyber-physical energy systems," in *Proceedings of the 38th IEEE Conference on Industrial Electronics IECON 2012*, Montreal, Canada.

[3] H. Georg, S. C. Müller, C. Rehtanz, and C. Wietfeld, "A HLA Based Simulator Architecture for Co-simulating ICT Based Power System Control and Protection Systems," in *3rd IEEE International Conference on Smart Grid Communications (SmartGridComm 2012)*, Tainan City, Taiwan, November 2012.

[4] P. Palensky, E. Widl, A. Elsheikh, and M. Stifter, "Modeling intelligent energy systems: Lessons learned from a flexible-demand EV charging management," *submitted to IEEE Transactions on Smart Grid.*

[5] M. Galus, R. Waraich, F. Noembrini, K. Steurs, G. Georges, K. Boulouchos, K. Axhausen, and G. Andersson, "Integrating power systems, transport systems and vehicle technology for electric mobility impact assessment and efficient control," *IEEE Transactions on Smart Grid*, vol. 3, no. 2, pp. 934 –949, Jun. 2012.

[6] T. Strasser, M. Stifter, F. Andren, D. Burnier, and W. Hribernik, "Applying open standards and open source software for smart grid applications: Simulation of distributed intelligent control of power systems," in *Power and Energy Society General Meeting, IEEE*, 2011.

[7] J. Martinez, V. Dinavahi, M. Nehrir, and X. Guillaud, "Tools for analysis and design of distributed resources - Part IV: Future trends," *IEEE Transactions on Power Delivery*, vol. 26, no. 3, pp. 1671–1680, 2011.

[8] P. Fritzson, P. Aronsson, H. Lundvall, K. Nystrm, A. Pop, L. Saldamli, and D. Broman, "The OpenModelica Modeling, Simulation, and Software Development Environment," *Simulation News Europe*, vol. 44, no. 45, Dec. 2005.

[9] A. Elsheikh, E. Widl, and P. Palensky, "Simulating complex energy systems with modelica: A primary evaluation," in *DEST'2012, the 6th IEEE International Conference on Digital Ecosystems and Technologies*, Campione d'Italia, Italy, 2012.

[10] F. Milano, "An open source power system analysis toolbox," *Power Systems, IEEE Transactions on*, vol. 20, no. 3, pp. 1199 – 1206, aug. 2005.

[11] V. Vyatkin and G. Zhabelova, "Multi-agent smart grid automation architecture based on IEC 61850/61499 intelligent logical nodes," *Industrial Electronics, IEEE Transactions on*, vol. PP, no. 99, p. 1, 2011.

[12] *IEC 61499: Function blocks*, Intern. Electrot. Commission Std., 2005.

[13] "The German Standardisation Roadmap E-Energy/Smart Grid," German Commission for Electrical, Electronic & Information Technologies of DIN and VDE, Frankfurt, Germany, Tech. Rep., 2010.

[14] *IEC 61850: Communication networks and systems for power utility automation*, International Electrotechnical Commission Std., 2010.

[15] A. Zoitl, T. Strasser, and A. Valentini, "Open Source Initiatives as basis for the Establishment of new Technologies in Industrial Automation: 4DIAC a Case Study," in *IEEE International Conference on Industrial Electronics (ISIE 2010), Bari, Italy*, July 4-7 2010.

[16] T. Blochwitz and et al., "The functional mockup interface for tool independent exchange of simulation models," in *Modelica'2011: The 8th International Modelica Conference*, Dresden, Germany, 2011.

[17] H. Erdélyi, W. Prescott, S. Donders, and J. Anthonios, "Fmi implementation in LMS Virtual.Lab motion and applications to a vehicle dynamics case," in *Modelica'2012, the 9th International Modelica Conference*, Munich, Germany, 2012.

[18] M. Gräber, C. Kirches, D. Scharff, and W. Tegethoff, "Using functional mocku-up units for nonlinear model predictive control," in *Modelica'2012, the 9th International Modelica Conference*, Munich, Germany, 2012.

[19] A. Abel, T. Blochwitz, A. Eichberger, P. Hamann, and U. Rein, "Functional mock-up interface in mechatrronic gearshift simulation for commercial vehicles," in *Modelica'2012, the 9th International Modelica Conference*, Munich, Germany, 2012.

[20] M. Einhorn, F. V. Conte, C. Karl, C. Niklas, H. Popp, and J. Fleig, "A modelica library for simulation of elecric energy storages," in *Modelica'2011: The 8th International Modelica Conference*, Dresden, Germany, 2011.