Unambiguous Power System Dynamic Modeling and Simulation using Modelica Tools

L. Vanfretti, *Member, IEEE*, W. Li, *Student Member, IEEE*,
T. Bogodorova, *Student Member, IEEE*, P. Panciatici, *Member, IEEE*

***Abstract*—Dynamic modeling and time-domain simulation for power systems is inconsistent across different simulation platforms, which makes it difficult for engineers to consistently exchange models and assess model quality. Therefore, there is a clear need for unambiguous dynamic model exchange. In this article, a possible solution is proposed by using open modeling equation-based Modelica tools. The nature of the Modelica modeling language supports model exchange at the "equation-level", this allows for unambiguous model exchange between different Modelica-based simulation tools without loss of information about the model. An example of power system dynamic model exchange between two Modelica-based software Scilab/Xcos and Dymola is presented. In addition, common issues related to simulation, including the extended modeling of complex controls, the capabilities of the DAE solvers and initialization problems are discussed. In order to integrate power system Modelica models into other simulation tools (Matlab/Simulink), the utilization of the FMI Toolbox is investigated as well.**

***Index Terms*—Power system modeling and simulation, Modelica, Model exchange, Scilab/Xcos, Dymola, FMI**

I. Introduction and Motivation

Due to the complexity of power systems and the variety of dynamic phenomena they expose, different numerical and modeling approaches have been implemented as the core of power system modeling and simulation software to meet different simulation requirements [1]. However, there exists a challenging problem: dynamic modeling and simulation for power system is inconsistent across different simulation platforms. Methods that allow for unambiguous power systems modeling and model exchange among different simulation platforms would facilitate engineering work flow specially when using different software platforms.

There are several factors affecting consistent modeling and simulation across different platforms. On one hand, data formats are often platform dependent. On the other hand, dynamic models for different components are not consistent through platforms due to simplifications, modeling philosophy and assumptions. For example, conventional "block-diagram" modeling forces users to share only parameters of models with predetermined structure, the model's mathematical representation is therefore not shared explicitly [**?**]. This leaves open to interpretation how the actual implementation of the models is carried out. As a consequence, two different model implementations of the same block-diagram model can be inconsistent. Hence it becomes difficult to evaluate the correctness of the modeling of each component and to validate a power system model as a whole. Explicitly exchanging the equations of the model may aid in achieving consistency across different simulation platforms.

A possible solution can be found by using an open modeling equation-based approach. Modelica is an object oriented language developed for equation-based modeling of physical systems and its components [2], [3]. There are several advantages for using equation-based modeling and simulation approaches. First of all, the models of each component in such software type are open for modification. They allow for straightforward implementation of new elements and libraries in order to simulate the behavior of each component and a system as a whole. This helps users to make models for customer-defined components.

The second advantage is that Modelica-based tools use models defined in a common standard language, which allows having unambiguous model exchange among different modeling & simulation tools without loss of information about the model. This results in all Modelica simulation tools having the same model, not only the parameters but also their explicit equations. This moves the focus from putting questions on the "quality of the model" as seen from expected simulation results, to the "quality of the solvers" used by each simulation platform. If the model is well defined and the simulations carried out in different software do not match measured responses. This implies that while the model is correct the particular simulation platform giving unexpected results, this simulator might have difficulties in simulating the model, i.e. the solver is not capable to solve the model correctly.

From the users' perspective, most Modelica-based tools are transparent and flexible, and do not require substantial training in order to master it. Therefore, they offer convenient usage and one can quickly get started with simulations. In addition, the Modelica language is an object oriented modeling language which means efficiency and flexibility in codding. Comparing to procedural languages like Fortran or C, Modelica programs can be easily scalable and most parts of code can be reused.

The aim of this article is to explore the possibility of model exchange between two independent Modelica-based software and to evaluate their performances. In addition, several issues of using Modelica tools for power system modeling and simulation are addressed:

- Capability of modeling complex controls.
- The ability of the solvers to handle medium-sized power system models, including the effects of initialization, dimension of the system to be handled by the solver, extended modeling of different controls, etc.
- Capability of exploiting other mathematical solvers applied to the Modelica model.

The remainder of this article is structured as follows. Section II offers an example of model exchange between two independent Modelica-based software—Scilab/Xcos and Dymola—as

Manuscript submitted to the IEEE PES General Meeting 2013.

L. Vanfretti, W. Li and T. Bogodorova are with the Electric Power Systems Division, School of Electrical Engineering, KTH Royal Institute of Technology, Teknikringen 33, SE-100 44, Stockholm, Sweden. E-mail: luigiv@kth.se, wei3@kth.se, tetianab@kth.se. P. Panciatici is with RTE, DMA, Versailles, France. E-mail: patrick.panciatici@rte-france.com.

This work was supported by the European Commission through the European FP7 iTesla. L. Vanfretti is supported by the European Commission within the FP7 iTesla project and the STandUP for Energy collaboration initiative. W. Li is supported by the Swedish Energy Agency, Svenska Kraftnät, and ABB.T. Bogodorova is supported by the European Commission within the FP7 iTesla project.

a proof of concept of the unambiguous model exchange notion by using Modelica tools. Section III investigates the capability for simulating complex systems of different Modelica platforms, illustrating that while a model can be well-defined a solver might not be able to simulate it. Section IV explores the possibility of integrating power system Modelica models into Matlab/Simulink by using the FMI Toolbox, this aims to show how a model can be exchange between completely different simulation platforms with consistency. Finally, in Section V, conclusions are drawn and future work is outlined.

II. Proof of concept: model exchange between Modelica tools

Two independent Modelica-based software—Scilab/Xcos and Dymola—are chosen as the platforms for model exchange. Scilab [4] is a free and open source software for numerical computations providing a powerful computing environment for engineering and scientific applications. Xcos [5] is distributed with Scilab. It is a graphical editor to design hybrid dynamical systems models. Dymola, the Dynamic Modeling Laboratory, is a complete tool for modeling and simulation of integrated and complex systems for use within automotive, aerospace, robotics, process and other applications [6]. Bi-directional model exchange between both software will be investigated in the following two subsections.

A. Model exchange from Scilab/Xcos into Dymola

A simple power system model with two machines in Scilab/Xcos from [7], [8], as shown in Figure 1, will be used. This model can generate a Modelica description file for the model after compilation, initialization and simulation.

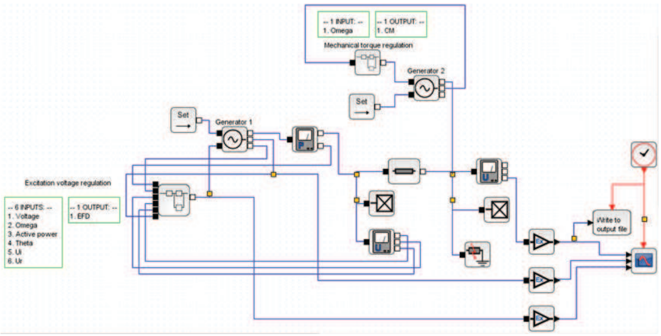
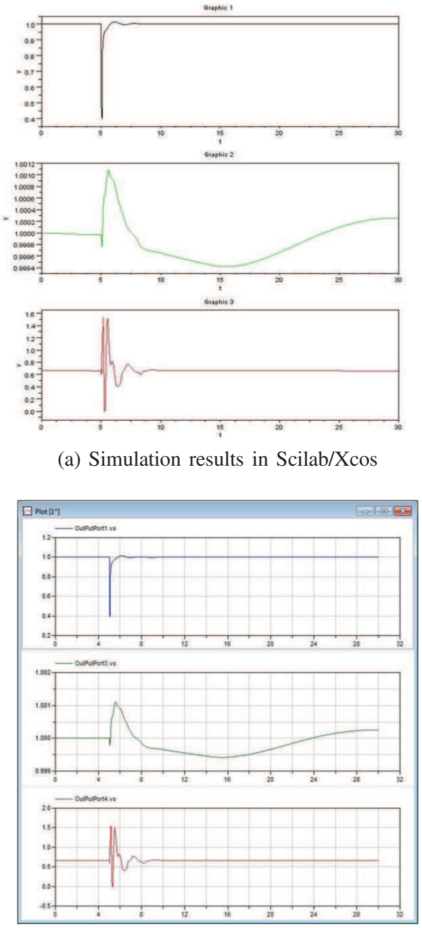


Fig. 1. Structure of the 2 generators model in Scilab/Xcos

Since some components in the model are defined in a custom power system modeling library [7], their Modelica definitions need to be added into the main Modelica description file, which is quite easy to handle. Another point to be noted is to change the initialization status from “fixed=true” to “fixed=false” for some specific variables when necessary (following the warnings in the translation log). After these small modifications and adjustments, this Modelica description file can be translated and simulated in Dymola. This simulation is consistent with the simulation results in Scilab/Xcos, as shown in the Figure 2.



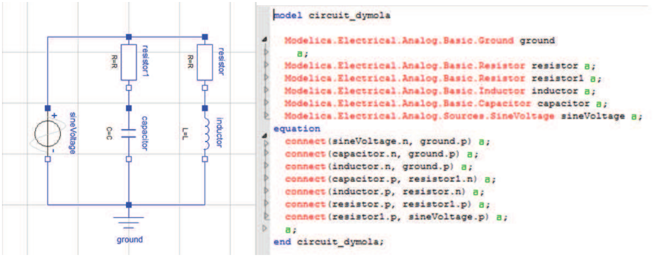
(a) Simulation results in Scilab/Xcos

(b) Simulation results in Dymola

Fig. 2. Simulation results of the 2 generators model in Scilab/Xcos and Dymola

B. Model exchange from Dymola into Scilab/Xcos

The Modelica description file is generated from a model in Graphical Editor of Dymola and applied into Scilab/Xcos to validate if the Modelica-based model exchange is bi-directional. A simple electrical circuit is drawn is Dymola as shown in Figure 3a. Since the electrical circuit model in Dymola directly takes use of the components from Dymola standard library, these components' Modelica definitions have to be declared when implementing the Modelica description file into Scilab/Xcos.



```
model circuit_dymola

  Modelica.Electrical.Analog.Basic.Ground ground
    a;
  Modelica.Electrical.Analog.Basic.Resistor resistor a;
  Modelica.Electrical.Analog.Basic.Resistor resistor1 a;
  Modelica.Electrical.Analog.Basic.Inductor inductor a;
  Modelica.Electrical.Analog.Basic.Capacitor capacitor a;
  Modelica.Electrical.Analog.Sources.SineVoltage sineVoltage a;
equation
  connect(sineVoltage.n, ground.p) a;
  connect(capacitor.n, ground.p) a;
  connect(inductor.n, ground.p) a;
  connect(capacitor.p, resistor1.n) a;
  connect(inductor.p, resistor.n) a;
  connect(resistor.p, resistor1.p) a;
  connect(resistor1.p, sineVoltage.p) a;
  a;
end circuit_dymola;
```

(a) Electrical circuit model

(b) Modelica description of the model

Fig. 3. Electric circuit model and its Modelica description in Dymola

Inside the Scilab/Xcos, a Modelica generic block (or MBlock) provides an easy way to build a Xcos block whose behavior is specified by a Modelica description. It can be imported in an Xcos diagram from the User-Defined Functions palette. The Modelica description associated to this block can be either given in a file or written in the window opened after double-clicking the block [7]. In this case, the Modelica definition of the electrical circuit is written in the opened window of the Modelica generic block and compiled. But since simulation results for variables can not be shown directly in Scilab/Xcos, Dymola has been used to show the simulation results of Scilab/Xcos. The Scilab/Xcos simulation results are identical with those from Dymola. As demonstrated by this simple electrical circuit, it is feasible to transfer models from Dymola to Scilab/Xcos with the Modelica description file.

From the two proof of concept examples above, we conclude that Modelica-based model exchange between Scilab/Xcos and Dymola is bi-directional and it also proves the feasibility of using Modelica models made in one Modelica platform by other platforms with minimum changes and adjustments. Through this model exchange instance, it is shown that if reliance is put on the internal translator of each platform from Modelica definitions onto C code, models can be exchanged between two independent simulation platforms without loss of consistency during the model exchange. This approach allows for the dynamic models and the solver for DAE solution to be completely independent and decoupled (which is not the case with most proprietary power system simulation tools), and in addition, for a preservation of the fidelity of simulation results for dynamic analysis (which is often difficult using proprietary power system simulation tools). Therefore, in principle, any complex or user-defined model can be exchanged consistently without loss of information of the model.

III. Simulation of complex power system models

Next, the limitations of specific Modelica-based tools for power system simulation are discussed in terms of the capability for extended modeling of complex controls, system dimension limits and initialization issues.

A. Extended modeling of complex controls

As shown in Figure 4, implementation of two sequential discrete events is not possible for simulation in Scilab/Xcos, in this case, a filter has to be added between two pulse blocks. However, in Dymola, by modifying the main Modelica-based file to connect two pulse blocks directly, it is possible to implement two sequential discrete events. This indicates that the solvers available in Dymola are more apt for simulation of complex controls used in power system plants by allowing for the solution of two sequential discrete components.

B. "Dimension" of the solver and initialization problem

To establish the system size limits for power systems simulation in Scilab/Xcos and Dymola, a medium-scale power system network was created by adding more generators and transmission lines in the Scilab/Xcos model in Figure 1 and transferred the generated Modelica description file into Dymola. This was done for three different cases by extending the original 2 generators model with 3, 4 and 22 generators, respectively.

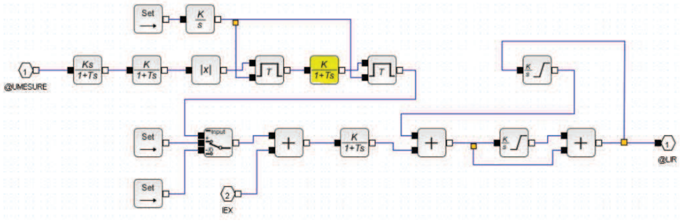


Fig. 4. Complex control model used in a power system plant

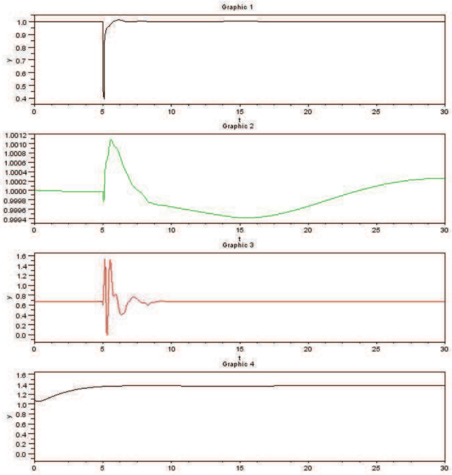


Fig. 5. Simulation results for three generators model in Scilab/Xcos

When adding the third generator in the Scilab/Xcos model, the excitation voltage (the forth sub-figure in Figure 5) of the third generator shows a distortion in the model's initialization. The initial values of the differential variables have to be provided by running the same model in other software's (e.g. EUROSTAG reference) power flow or by developing initialization routines that satisfy power flow constrains as typically used in power system simulation tools. Without correct starting values, the initialization can fail or steer the simulation into an incorrect operating point from the power systems point of view. In this case, the simulation of the Scilab/Xcos model with 4 or more generators cannot be executed without providing exact initialization.

Even though for either 4 or 22 generators models in Scilab/Xcos (Figure 6) the simulation results cannot be obtained, it is possible to generate the Modelica description file of the model, which can be implemented and simulated properly in Dymola. The simulation results in Dymola for the 22 generators model are shown in Figure 7.

As shown in the simulation results, the third generator's voltage signal "OutPutPort3" does not start from the equilibrium point. This is because the initialization in the Modelica description is not precise. However Dymola could simulate the model under this non ideal initial condition and determine the final equilibrium point. This indicates that the initialization routines in Dymola may need to be adapted for power system simulations.

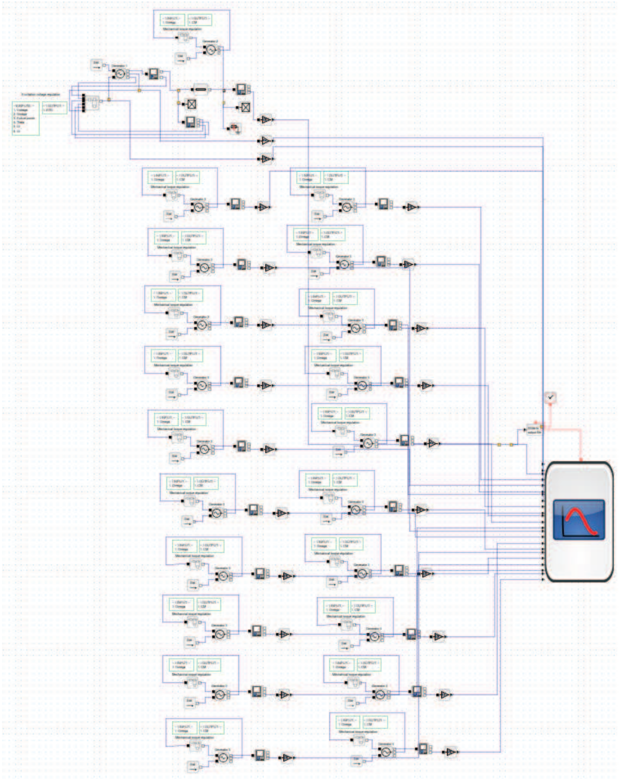


Fig. 6. Structure of the 22 generators model in Scilab/Xcos

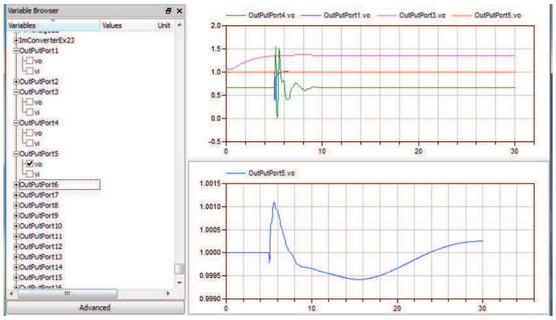
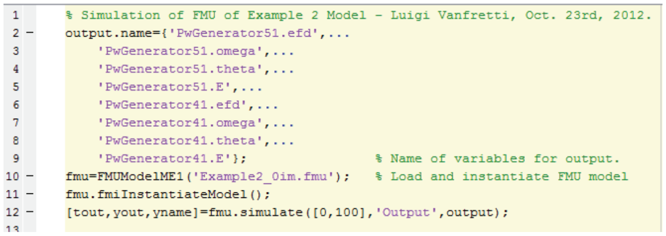


Fig. 7. Simulation results of the 22 generators model in Dymola

IV. Exploiting mathematical solvers using the FMI

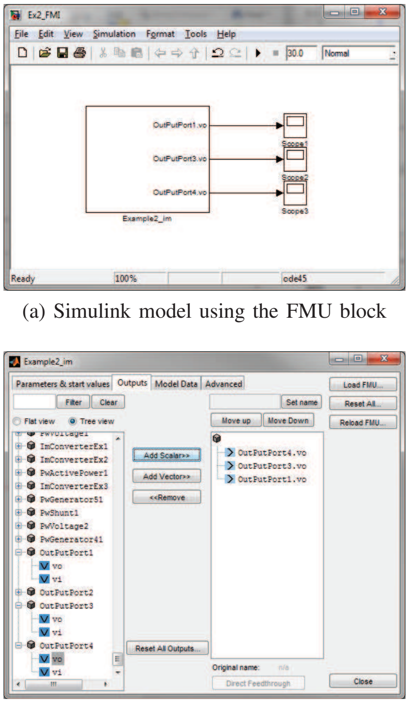
The functional mock-up interface (FMI) defines a standard interface for computer simulations of complex systems. This standard interface allows for model exchange between different simulation tools and for co-simulation. Details on the FMI can be found in [9] and [10]. The actual implementation of the FMI by a particular software environment enables the generation of a simulation model that can be interfaced with models of other simulation environments or allows for the creation of a software library called Functional Mock-up Unit (FMU).

This approach is attractive, as any simulation platform following the standardized FMI can be able to simulate and interface its own models to other models generated by a by a FMI-compliant tool. For example, Dymola is capable of exporting models that comply with the FMI by translation into a FMU. The FMI Toolbox for MATLAB/Simulink [11] provides tools for import and simulation of FMUs in Simulink including configuration interfaces. In addition it provides the flexibility of simulation of FMUs via MATLAB scripts. This toolbox supports the standard “FMI for Model Exchange 1.0” and “FMI for Co-Simulation 1.0” [11].



```
% Simulation of FMU of Example 2 Model - Luigi Vanfretti, Oct. 23rd, 2012.
output.name={'PwGenerator51.efd',...
    'PwGenerator51.omega',...
    'PwGenerator51.theta',...
    'PwGenerator51.E',...
    'PwGenerator41.efd',...
    'PwGenerator41.omega',...
    'PwGenerator41.theta',...
    'PwGenerator41.E'};                 % Name of variables for output.
fmu=FMUModelME1('Example2_0im.fmu');    % Load and instantiate FMU model
fmu.fmiInstantiateModel();
[tout,yout,yname]=fmu.simulate([0,100],'Output',output);
```

Fig. 8. MATLAB script for simulating the FMU of the 2 generators model



(a) Simulink model using the FMU block

(b) FMU block configuration for simulation outputs

Fig. 9. Simulink FMI block for simulating the FMU of the 2 generators model

Using the FMU generated in Dymola it is possible to carry out the same simulations directly in MATLAB or in Simulink. Figure 8 shows the MATLAB script used for simulating the FMU of the 2 generators model. As shown in Figure 9, the same FMU can also be implemented in a Simulink FMI block model, which needs to be properly configured by selecting the desired outputs of the simulation. Observe that the FMI block provides the flexibility of re-defining model parameters and initial conditions (start values). Both MATLAB script and Simulink FMI block methods allow to obtain the same simulation results when the solver has been properly chosen in Simulink, hence only Simulink FMI block simulation results are shown here.

The simulation results for the 2 generators model and the 22 generators model by using the Simulink FMI block are shown in Figure 10 and Figure 11, respectively. They are consistent with those obtained in Dymola (Figure 2, Figure 7). This shows how the FMI block allows for model exchange between two completely independent simulation platforms while preserving modeling and simulation consistency.

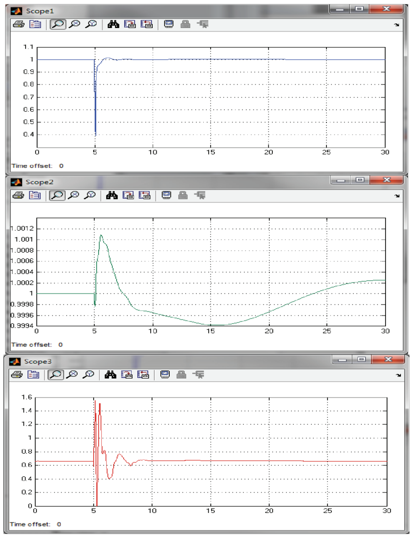


Fig. 10. Simulation results for the 2 generators model by using FMI block

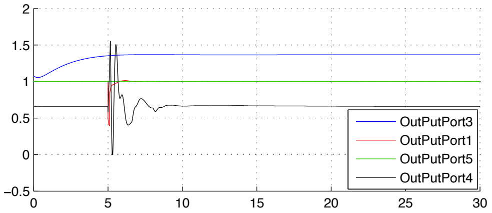


Fig. 11. Simulation results for the 22 generators model by using FMI block

V. Conclusion and Future Work

This article investigates the possibility of unambiguous power system modeling and simulation by using equation-based Modelica tools. The model exchange between two independent Modelica-based software—Scilab/Xcos and Dymola—shows consistent results. In addition, challenges for Modelica modeling tools simulation capabilities in terms of the complex controls, dimension of the power system and initialization problems were considered as well. At last, it is highlighted that the FMI Toolbox for Matlab allows for the integration of Modelica models into MATLAB/Simulink which shows the possibility of exploiting powerful mathematical solvers and other tools available in this software environment.

Future work will consider the following challenges:

- The use of equation-based modeling using Modelica tools and FMI still needs to be demonstrated for very large power system simulations. Initialization and the simulation of highly coupled power systems will prove challenging due to simulation performance requirements.
- Today, CIM [12] only defines the topology and parameters of the system; the addition of Modelica definitions of dynamic models may aid in guaranteeing consistency for model exchange using CIM instead of following current approaches adopted in CIM using fixed block-diagram representations of component models.

References

[1] F. Milano, *Power System Modelling and Scripting*. Springer, 2010.

[2] Modelica Design Group, *Modelica A Unified Object-Oriented Language for Physical Systems Modeling, Language Specification*, 1999.

[3] P. Fritzson, *Introduction to Modeling and Simulation of Technical and Physical Systems with Modelica*. John Wiley & Sons, 2011.

[4] Scilab description. [Online]. Available: http://www.scilab.org/products/scilab

[5] Xcos description. [Online]. Available: http://www.scilab.org/products/xcos

[6] Dymola product description. [Online]. Available: http://www.modelon.com/products/dymola/

[7] A. Chieh, P. Panciatici, and J. Picard, "Power system modeling in Modelica for time-domain simulation," *2011 IEEE Trondheim PowerTech*, pp. 1–8, June 2011.

[8] P. Panciatici and A. Chieh, "Equation-based hybrid modeling of power systems for time-domain simulation," *IEEE PES General Meeting*, pp. 1–9, July 2011.

[9] T. Blochwitz, et alt, "The functional mockup interface for tool independent exchange of simulation models," in *The 8th Modelica Conference*, 2011.

[10] Functional mockup interface. [Online]. Available: https://www.fmi-standard.org/tools

[11] Fmi toolbox for matlab. [Online]. Available: http://www.modelon.com/products/fmi-toolbox-for-matlab/

[12] E. Lambert, X. Yang, and X. Legrand, "Is CIM suitable for deriving a portable data format for simulation tools?" *IEEE PES General Meeting*, pp. 1–9, July 2011.

Luigi Vanfretti (Student Member '03, M'10) is an Assistant Professor at the Electric Power Systems Department, KTH Royal Institute of Technology, Stockholm, Sweden. He received the Electrical Engineering Degree from *Universidad de San Carlos de Guatemala* in June 2005. He was a visiting researcher at the Department of Electronics and Electrical Engineering of The University of Glasgow, Scotland in 2005. He received his M.Sc. in 2007 and Ph.D. in 2009, both in Electric Power Engineering, from Rensselaer Polytechnic Institute (RPI), Troy, NY, USA. For his research and teaching work towards his Ph.D. he was awarded the *Charles M. Close Award* from Rensselaer Polytechnic Institute. Since 2009, he has served as Secretary of the IEEE Task Force on Open Source Software and as WG Organizer for the IEEE PSDP Working Group on Power System Dynamic Measurements. His research interests are in the general area of modeling, dynamics, stability and control of power systems; while his main focus is on the development of applications of PMU data. He is an evangelist of Free/*Libre* and Open Source Software for power system engineering education.

Wei Li (Student Member '12) received the B.S. degree in Automation from Beijing Jiaotong University in 2010 and the M.S. degree in Electronics Design from Mid Sweden University in 2011. She carried out her master thesis on Hydro Turbine and Governor Modeling and Scripting for Small-Signal and Transient Stability Analysis of Power System at the Electric Power Systems Department of KTH from January to September 2011. She is currently a Ph.D. student within the at KTH. The main theme of her current research is the real-time wide-area state estimation of hybrid AC and DC grids.

Tetiana Bogodorova (Student Member '12) received her B.S. in Computerized Systems, Automatics and Control and her M.S. in Automatic and Control Systems from National Technical University of Ukraine—Kyiv Polytechnic Institute. For both degrees, she specialized in control theory. From 2007 to 2011, she was a systems engineer in System Analytics Group, R&D, NetCracker Technology Corp. She was involved into development process of the Operations Support System (OSS) for telecommunication industry. In 2012 she was enrolled as a PhD student at Electrical Power systems Department at KTH Royal Institute of Technology, Stockholm, Sweden. Her current research interests lie on intersection of system identification theory and power systems analysis, modeling and validation.

Patrick Panciatici (M11) received a degree from the French Ecole Supérieure d'Electricité in 1984. He joined EDF R&D in 1985, managing the EUROSTAG and CSVC projects. He joined RTE, Versailles, France, in 2003 and participated in the creation of the department "Methods and Support." He is the head of a team which develops real time and operational planning tools for RTE, and ensures operational support on the use of these tools. Mr. Panciatici is a member of CIGRE, SEE, and the ENTSO-E R&D Working Group, RTEs representative in PSERC and several European Projects including iTesla.