# A Modelica Power System Library for Phasor Time-Domain Simulation

T. Bogodorova, *Student Member, IEEE*, M. Sabate, G. León,
L. Vanfretti, *Member, IEEE*, M. Halat, J. B. Heyberger, and P. Panciatici, *Member, IEEE*

***Abstract*— Power system phasor time-domain simulation is often carried out through domain specific tools such as Eurostag, PSS/E, and others. While these tools are efficient, their individual sub-component models and solvers cannot be accessed by the users for modification. One of the main goals of the FP7 iTesla project [1] is to perform model validation, for which, a modelling and simulation environment that provides model transparency and extensibility is necessary. [1] To this end, a power system library has been built using the Modelica language. This article describes the Power Systems library, and the software-to-software validation carried out for the implemented component as well as the validation of small-scale power system models constructed using different library components. Simulations from the Modelica models are compared with their Eurostag equivalents. Finally, due to its standardization, the Modelica language is supported by different modelling and simulation tools. This article illustrates how Modelica models can be shared across different simulation platforms without loss of information and maintaining consistency in simulation results.**

***Index Terms*—Power system simulation, model validation, Modelica**

## I. Introduction and Motivation

Several attempts of power systems modeling using the Modelica language have been made [2], [3]. Nevertheless, several of these libraries (e.g. [3]) have become proprietary software and closed for modifications. The latter means one can no longer determine which modelling simplifications have been implemented by the authors during model development. On the other hand, available open source libraries [4] are not maintained in order to be compliant with the current Modelica language standard. Such libraries require dramatic changes from end users to support each new version of the language. Due to these reasons the authors were urged to make a new power system library that offered simplicity for maintenance and extensibility. In general, an equation-based modelling language is the most appropriate choice because one will know the exact model of the system independently of the software in which it is modelled. In the case of the Modelica language, it allows for consistent model sharing between different simulation platforms supporting the Modelica language.

Another reason for modelling of power systems using Modelica is that it offers the possibility of exploiting powerful mathematical or engineering tools as MATLAB/Simulink, Mathematica and JModelica.org, through the Flexible Mock-up Interface (FMI) or inherently [5]. Within MATLAB/Simulink, Modelica models can be used through the FMI Toolbox; while for Mathematica, SystemModeler links Modelica models directly with the computation kernel. The aims of this article is to demonstrate the value of Modelica as a convenient language for modeling of complex physical systems containing electric power subcomponents, to show results of software-to-software validations in order to demonstrate the capability of Modelica for supporting phasor time-domain power system models, and to illustrate how power system Modelica models can be shared across different simulation platforms. In addition, several aspects of using Modelica for simulation of complex power systems are addressed:

- Design and implementation of new electrical and non-electrical component blocks
- Connection of electrical and non-electrical components
- Modeling of different synchronous machines matching Eurostag's models
- Performing phasor time-domain simulations, studying initialization for models and analysing system response to time events.

The remainder of this article is organized as follows. Section II offers a description of the power system elementary component models for electrical and non-electrical domains included into the library. Section II is dedicated to validation of the components incorporated into two power systems. Section III shows a software-to-software validation of the new library through the simulation of two test systems and comparison with their Eurostag equivalents. Section IV demonstrates how the use of Modelica allows to exploit different modeling and simulation environments without ambiguity on the model itself. In Section V, the results are summarized, conclusions and the future work are presented.

## II. A Modelica Power Systems Library

This section describes the power system library models developed using the Modelica language. A previous version of the library [2] was developed in Scilab/Xcos simulation environment during the European project PEGASE (see [2]). Scilab/Xcos has several limitations in the support of Modelica language specification [6]. Therefore, for the *i*Tesla project each component block inherited from PEGASE into the Power Systems library was converted from Scilab/Xcos models to the most recent Modelica language definition by using Dymola. The conversion involves creation of the Modelica classes into Dymola including their graphical representation and utilizing differential and algebraic equations corresponding to each electrical or a block in the PEGASE library. The Power System

[1]This research was supported by the *i*Tesla Collaborative R&D project co-funded by the European Commission (7th Framework Programme) [1].

library has also been improved adding new electrical and mathematical blocks. With these new components it is possible to build and simulate power systems of small and medium size using Modelica language, to perform time-domain simulations and to validate results using Eurostag models as reference.

In the following section, the authors present the list of the components (electrical and non-electrical) of the library with a short description of their functionality, paying special attention to new models that have been recently developed and omitting descriptions of simple blocks. The next section (Section III) shows two examples of power networks: a system with one machine ("System A"), and another ("System B") containing two machines (with different levels of model complexity). Simulations for both systems are carried out in Dymola and compared against their corresponding simulations in Eurostag.

### A. Electrical Blocks

New electrical blocks which are most frequently used for power systems modeling were created. Several blocks were translated from Scilab/Xcos [7] to Dymola with a list of modifications presented below:

1) Creation of blocks for connector's pin (PwPin) in Dymola. This connector connects components electrically (real and imaginary part of voltage and current)
2) Introduction of equations inside a Class for each component model, indicating the corresponding pins to be used and defining Icons for each block
3) Adaptation of connector blocks to Modelica format
4) Putting all the new blocks in a single package named Power System Library
5) Reviewing that parameters and variables were in accordance to Modelica formats
6) Checking all block models in Dymola to see if the number of equations and number of variables were correct, implying that the translation was done correctly.

The modeled library now contains the following electrical blocks:

- PwLine. The model of the transmission line is based on the equivalent pi-model. It is represented by a series impedance and two shunt impedances located at the two terminal nodes.
- PwLoad. The load is modeled by an impedance between it's connection node and the ground.
- PwTransformer. The model is based on the transformer equivalent circuit, composed of an ideal transformer, a series impedance taking into account losses due to the load current and a shunt admittance representing no-load losses.
- PwGenerator. The modeling of synchronous machine is done according to Park's classical theory. The developed model corresponds to Eurostag's full model, given by its internal parameters. For more detailed information see [8], p.57.
- PwLinewithOpening. Power line with opening event during certain period of time at a specific time $t$.
- PwLoadWithVariation. This block describes a model which similar to the PwLoad block from the PEGASE library, but in this case the input values are the initial voltage at the node and the active and reactive power consumed by the load. This class allows a variation on the active and/or reactive power at a specific time $t$.
- PwCapacitorBank. The capacitor bank model is composed of a set of capacitors connected in parallel on a node called steps, with the possibility to switch on (+) or switch off (-) a certain number of steps at a certain time $t_1$.
- PwGeneratorM1S. A synchronous machine may be given either by its internal parameters or by its external parameters. Given the external parameters, the record PwExtInt-Parameters computes the internal parameters, using the Canay model [9] (if the Canay mutual inductance, and the damper circuit time constant are not equal to 0).
- PwGeneratorMS2. Once the internal parameters are computed, the developed model PwGeneratorMS2 is used.This model corresponds to Eurostag's full model ( [8], p.74) given by its external parameters.

### B. Non-Electrical Blocks

A collection of non-electrical blocks have been created in order to model synchronous generator controls (AVR or PSS) in a block-oriented fashion. Most of these models have been adapted from the PEGASE library to Dymola, and some of them have been developed for the modeling of the turbine governor and the voltage regulator models in Eurostag [8]. These non-electrical blocks were designed to be compatible with the input/output blocks from the Modelica standard library.

## III. Software-to-Software Validation of Component Models

The first system built to test the library, shown in (Fig. 1), is called System A. It is composed by a 1000 MW unit, a synchronous machine, feeding a 600 + $j$200 MVA load through a 24/400 kV step-up transformer, two 400 kV lines and a 400/158 kV transformer. Two events are simulated at specific times to study the systems' behavior after:

- An instantaneous increase in the load of 50 + $j$25 MVA at time $t$ = 10 s.
- Switching off one of the two 380 kV lines at the receiving node at time $t$ = 20 s.

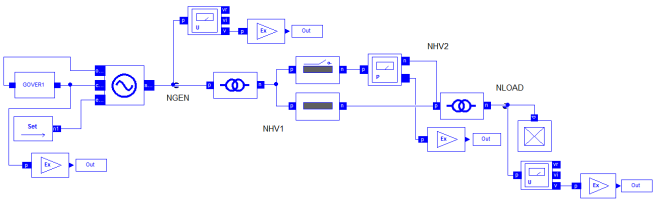



Fig. 1: Dymola Diagram of System A

The synchronous machine (Generator M2S in Eurostag) is given by its external parameters and the PwExtIntParameters block (not shown in the figure) computes the internal parameters. It is equipped with a voltage regulator chain that consists of a constant excitation voltage, and a governor chain.

In order to validate System A, the same system has been built and simulated in Eurostag. The start values necessary for time simulations for the synchronous machines models, voltage regulators, voltage governors and power loads are taken from the result of the power flow computations and dynamic initialization in Eurostag. Once these values are known, the system can be simulated using Dymola.

### A. System B

The second test system is called System B (Fig. 2) and is composed by nodes NGEN, NHV1, NHV2 and NLOAD similar to System A, and a new node NHV3 where a new synchronous machine is connected. The 1000 MW machine connected to node NGEN is defined by its internal parameters (Eurostags machine M1S) and is regulated with the speed governor GOVER3 and the voltage regulator AVR3. The node NHV2 is linked by a 380kV line to the node NHV3 where a second synchronous machine is connected. This machine is also defined by its internal parameters, has an internal transformer and is regulated by GOVER3 and AVR3 controller models.

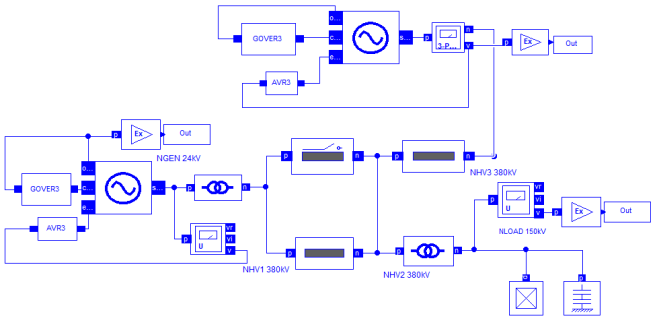



Fig. 2: Dymola Diagram of System B

### B. Validation against faults

*a) System A:* After running the simulation in Eurostag and Dymola, the voltage magnitude for the NGEN and the NLOAD nodes (Fig. 1) have been chosen as output for validation. As can be seen from Fig. 3, there is an expected voltage drop and the subsequent voltage stabilization after an instantaneous increase in the load of $50 + j25$ MVA at time $t$ = 10 s, and after switching off one of the two 380 kV lines at the receiving node at time $t$ = 20 s. Obtaining the same results for the time-simulations performed in the two different environments demonstrates high accuracy of models built using Modelica and validity of choice of the equation-based language for construction of models of small and medium size. The results serve as evidence of correctness of the new component models in particular for the synchronous

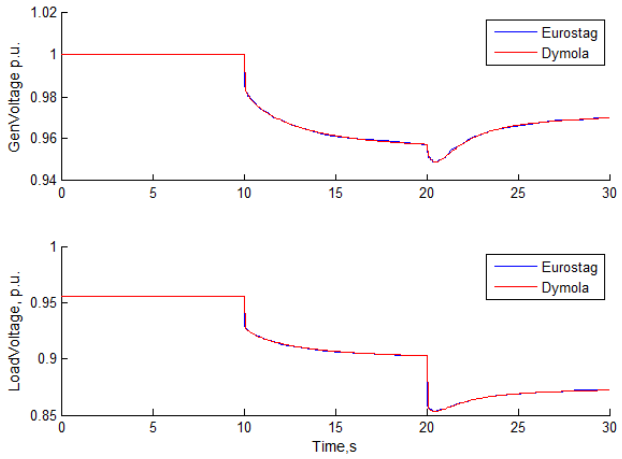



Fig. 3: Results of System A model validation

machine given by external parameters as well as the line opening event and load with modification.

*b) System B:* After running the simulation in Eurostag and Dymola, the voltage magnitude for the NGEN and the NLOAD nodes obtained are shown in Fig. 4. The voltage drop and the subsequent voltage stabilization after a change in the compensation level of the capacitor bank can be observed.

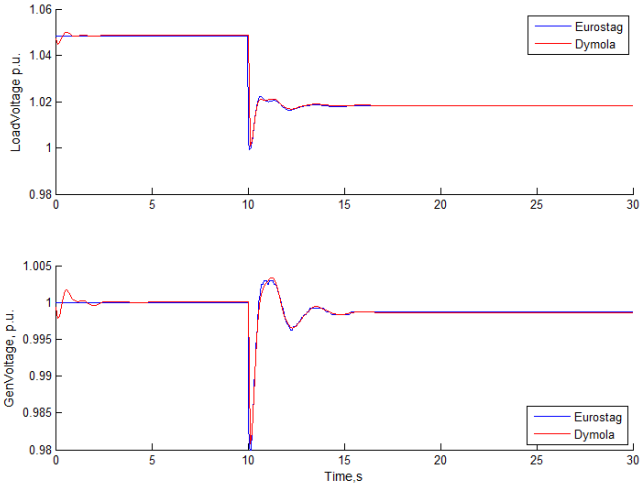



Fig. 4: Results of System B model validation

Analyzing the result in Fig. 4, one can observe small oscillations of Dymola model appeared at the initialization phase that are not seen in Eurostag. This is probably due to initialization of regulators from Eurostag values or other parameters of System B. In order to solve this problem, tests are performed in Dymola and Eurostag. One of the possible solutions for the initial oscillation problem is to start the simulation 3 seconds before to allow the Modelica solver reach the steady-state. On the other hand the systems response to the bank modification is similar in both tools, thus indicating that the Modelica models are valid.

## IV. Modelica Model Exchange and Simulation across Different Simulation Environments

This section demonstrates the flexibility and benefits of using Modelica for model exchange and simulation across different software platforms supporting the Modelica language. Four different simulation environments have been chosen in order to test feasibility and compatibility of Modelica models for exchange. SystemModeler can interpret the Modelica models directly, while the FMI Toolbox for MATLAB uses Flexible Mock-Up Units (FMUs) which is a software component used to exchange Modelica models through the FMI standard [10]. This allows the user to benefit from the numerical computation, visualization and modeling capabilities of Mathematica, MATLAB, or other tools. OpenModelica and JModelica.org are open source software which also offer modeling and simulation capabilities compliant with the Modelica language.

### A. FMI Toolbox & MATLAB/Simulink

FMI Toolbox was designed for model exchange purposes via an individual executable model representation called Functional Mock-up Unit (FMU) [11]. FMUs facilitate model exchange through two options:

- FMUs of specific devices are generated, and then incorporated into the overall model in MATLAB/Simulink
- FMUs of a complete model can be generated and shared without revealing the model's internal structure/equations

As an example we explore the second option in this paper. For this purpose the model of System A was chosen. FMUs can be generated using Dymola, JModelica.org, and other tools. The results in this paper were achieved using Dymola as FMU compiler and the FMI Toolbox in MATLAB/Simulink as recipient of the FMU (Fig. 5). The simulation results obtained

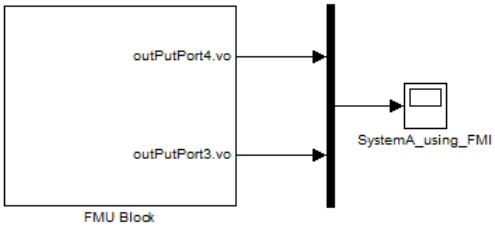



Fig. 5: MATLAB/Simulink model using FMI Toolbox

with MATLAB/Simulink, Dymola, and Eurostag, respectively, coincide with the results shown in Fig. 3.

### B. System Modeler & Mathematica

Wolfram SystemModeler is a physical modeling and simulation tool. It can process the Modelica language, which takes an advantage of the strengths from equation-based modeling, where the flow in components is modeled. This presents significant advantages compared to block-based modeling [12]. In addition System Modeler is capable to interpret Modelica models directly without conversion into FMU. This ability makes it similar to Dymola, but the major and decisive advantage of SystemModeler on Dymola is the capability to use Mathematica - a well known computational tool. The result of modelling (Fig. 6) and simulation (Fig. 7)

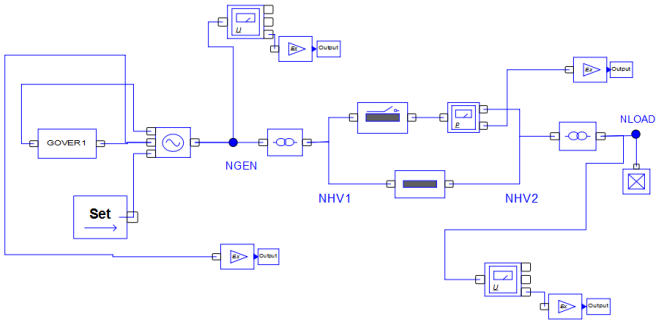



Fig. 6: SystemModeler model using the model in Fig. 1

is the same as using FMI Toolbox in MATLAB/Simulink. The only configuration that must be carried out is to load the Power Systems Library in SystemModeler.

### C. OpenModelica

OpenModelica is an open source Modelica-based modeling and simulation environment. The goal with the OpenModelica effort is to create a comprehensive open source Modelica modeling, compilation and simulation environment based on free software distributed in binary and source code form for research, teaching, and industrial usage [13]. OpenModelica is Open Source Software, and thus, it becomes attractive for non-profit organizations (for example, universities) to exploit it. In order to use models developed in Dymola in OpenModelica, one should pay attention to any empty annotations. It may generate errors, so just removing these annotations solves the problem. As expected, the simulation result in OpenModelica (Fig. 7) is identical to Dymola output (Fig. 3). It proves that the model was transferred from Dymola to OpenModelica without loss of information.

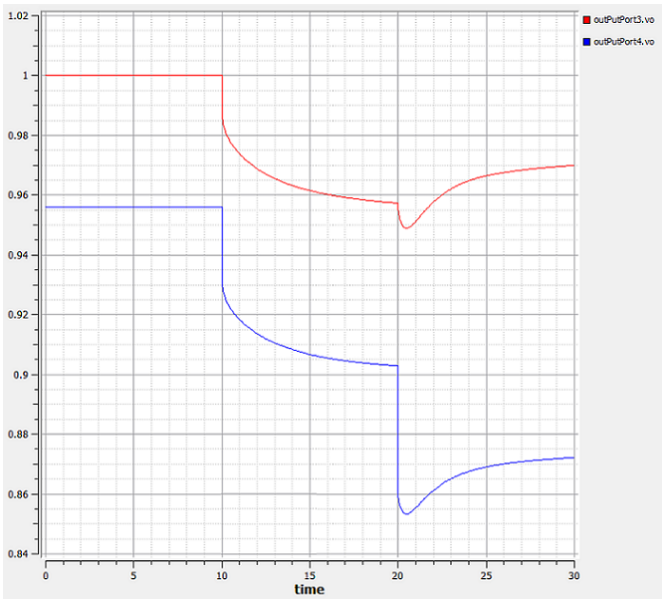
### D. JModelica.org

JModelica.org is an Open Source Software whose main purpose is not only modelling and simulation, but also optimization [5]. Therefore the user can be interested in exploiting it as free software for optimization purposes. The programming language used by JModelica.org is Python.

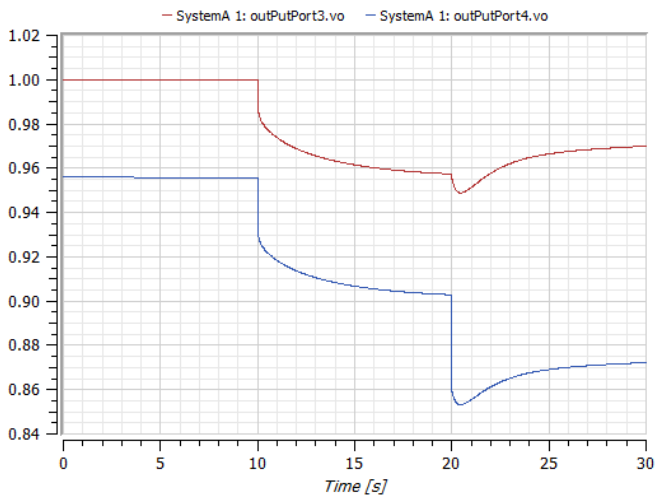
JModelica.org allows users to simulate their own model or use FMUs compiled in another software. In this paper the second option was demonstrated, as shown in Fig. 8, one can see the simulation results of System A.

## V. Conclusion and Future Work

This article described a power system model library developed using the Modelica language, and the software-to-software validation of the library components with respect to models in Eurostag. The possibility to build




(a) OpenModelica simulation results




(b) SystemModeler simulation results

Fig. 7: OpenModelica and SystemModeler simulation results

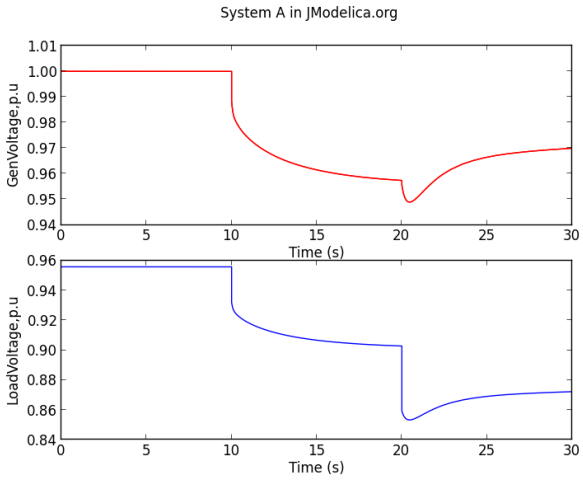



Fig. 8: JModelica.org simulation results

power systems in different simulation environments was investigated. An example model was transferred from Dymola to MATLAB/Simulink, SystemModeler/Mathematica, OpenModelica, and JModelica.org (Figs. 3, 5, 7, 8). The results of software-to-software validation have shown that the model is interpreted correctly by each of the software tested. The decision on which simulation environment to use is left for the end user, thereby providing flexibility to use the power system model in different computational environments. Future work will consider the challenges enumerated below:

- Validation of Eurostag's induction machine model in Modelica using a simple system.
- Include initial equations to improve initialization.
- Build and test new electrical and non-electrical components in order to simulate bigger systems.
- Continue with the development of new components models to enlarge the Power System Library.

At the time of preparation of this paper, the library is undergoing a continuous development of models. The iTesla consortium partners are discussing the possibilities of publicly releasing the library under an open source license. The release of the library will most likely take place at the end of the FP7 iTesla project in 2015.

## REFERENCES


[1] iTesla: Innovative Tools for Electrical System Security within Large Areas. [Online]. Available: http://www.itesla-project.eu/

[2] 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.

[3] Power Systems library SPOT. [Online]. Available: https://www.modelica.org/libraries/spot

[4] ObjectStab Free library for power systems voltage and transient simulation. [Online]. Available: https://www.modelica.org/libraries/ObjectStab

[5] JModelica.org - an extensible Modelica-based open source platform for optimization, simulation and analysis of complex dynamic systems. [Online]. Available: http://www.jmodelica.org/

[6] L. Vanfretti, W. Li, T. Bogodorova, and P. Panciatici, "Unambiguous Power System Dynamic Modeling and Simulation using Modelica Tools," *IEEE PES General Meeting 2013*.

[7] *PEGASE deliverable D5.2: Proof of concept for an open simulation and model sharing framework validated on a medium size power system.*

[8] *Tractebel RTE. Eurostag Theory Manual.*

[9] I.M.Canay, "Determination of model parameters of synchronous machines," *Electric Power Applications*, vol. 130, pp. 86–94, 1983.

[10] Functional mock-up interface. [Online]. Available: https://fmi-standard.org/

[11] W. Chen, M. Huhn, and P. Fritzson, "A Generic FMU Interface for Modelica," in *4th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools*. ETH Zurich, Switzerland, 2011.

[12] Wolfram System Modeler. [Online]. Available: http://www.wolfram.com/system-modeler/

[13] OpenModelica. [Online]. Available: https://www.openmodelica.org/