Aspects of Power System Modeling, Initialization and Simulation using the Modelica Language

Gladys León, Milenko Halat
Marc Sabaté
Aplicaciones en Informática Avanzada, S.L.
Sant Cugat del Vallés, Barcelona, Spain
leonge@aia.es, milenkoh@aia.es
sabatem@aia.es

Jean-Baptiste Heyberger
RTE Résau de Transport d'Électricité
Paris, France
jean-baptiste.heyberger@rte-france.com

Francisco José Gómez[1]
Luigi Vanfretti[1,2]
[1]KTH Royal Institute of Technology
Stockholm, Sweden
[2]Statnett SF, Oslo, Norway
fragom@kth.se, luigiv@kth.se
luigi.vanfretti@statnett.no

***Abstract*—A medium sized power system, with a large number of devices, is fully modeled and simulated in Modelica. For this, new models for electrical components and mathematical blocks were developed in Modelica and the full system topology and parameters were taken from a reference system made in Eurostag. Initialization and time-domain simulations are performed in a Modelica simulation environment and the results obtained for the steady-state and the response to time-events are compared and validated with simulations performed in Eurostag.**

***Index Terms*—OpenSource Software, Simulation Software, Modelica, Midsize Power System Models**

I. Introduction

One of the major concerns in the field of power systems nowadays is the need of a common language for dynamic components models that could allow different TSOs to have a common understanding the dynamic behaviour of any particular device or control system. Substantial efforts have been devoted in the recent years to investigate the modelling of power system components using Modelica [1], an open modelling language that has proven to be suitable for the development of a common language allowing the exchange of mathematical models used in time-domain simulations [3], [4]. The European project iTESLA [5], on deployment until 2015, took over the developments of the first version of a Power System Library (PSL) started in [4]. A full Power System Library has now been developed in Modelica, extending the first library with a large number of new models, giving it structure and hierarchy, adding models created by different development teams, and ensuring compatibility with the standard Modelica library. This work allows to test the modelling and simulation of power systems growing in size and therefore in complexity [6] using Modelica. This power system library is being developed with the clear purpose of providing users with an open library compliant with many Modelica tools that facilitates the development and easy exchange of models, as well as the assembly and simulation of medium to large size power systems. Moreover, one of the main objectives of the iTesla project is to model parts of the European network in Modelica, and to perform simulations using different Modelica simulation environments. For the system studied here we focused in two environments: OpenModelica [7], which is an open source simulation environment developed and supported by *Linköping University* and the *Open Source Modelica Consortium*, and Dymola which is a commercial simulation environment from *Dassault Systemes* [9]. In this paper we discuss a system composed of 22 buses containing numerous lines, transformers and loads, as well as various synchronous machines with complex regulations. The system was built in Modelica language using models existing in the iTesla power system library and new models specifically developed for this system and later included in the library.

The paper is structured as follows: In Section II different models of power system components developed in Modelica are described. These models have been used specifically for developing the 22-buses system. In Section III, the validation process of power system components is explained. In Section IV, the development of the full mid-size model is presented. Initialization issues and pros and cons of using Open Modelica [7] and Dymola [9] simulation environments are also described. Section V is dedicated to analyse the simulation results of the full system.

II. Modelling power system components in Modelica

Modelica is an object-oriented equation-based programming and modelling language, which allows the representation of cyber-physical systems using a strict mathematical representation, for the purpose simulations using digital computers. This is possible thanks to the standardized Modelica language definition [8]. The Modelica compiler compiles the Modelica model into machine code and executes it together with a numerical solver of choice, capable of solving sets of differential and algebraic equations (DAE), discrete equations or their combination. Thus, a Modelica model is not coupled to any specific solver , and can be easily exchange between software

This work was supported in part by the EU-funded FP7 iTESLA project and by RTE, the French Transmission System Operator and Statnett SF, the Norwegian Transmission System Operator, under grant agreement n283012. iTESLA (Innovative Tools for Electrical System Security within Large Areas), online: www.itesla-project.eu

environments or specific applications capable of compiling Modelica code.

In this section, basic electrical models, specifically developed for the construction of the 22-buses system, are described. These models are included in the Power Systems Library (PSL) developed in the context of iTESLA Project. The PSL has been designed to fulfil the requirement of

1) Creating a user-friendly Power System library.
2) Creating a library compatible with the Modelica models existing in the *Modelica Standard Library* (MSL) [1].
3) Having a library composed by electrical components and non-electrical components allowing the assembly and simulation of power systems of small, medium and large size.

The structure of the MSL has been taken as a reference for the structure and the hierarchy of the different types of models in the PSL. The PSL is hierarchically organized as follows:

- Examples
- Electrical models
  - Machines.
  - Branches
  - Loads
  - Banks
  - Sensors
  - Events
  - Solar devices
  - Buses
  - Controls
  - Wind
- Non Electrical models
  - Mathematical models
  - Continuous models
  - Non linear models
  - Logical models
- Connectors
- Interface

where each one of the levels and sub-levels shown above contain the different models and classes developed for power systems modeling. The description of some of the first models added can be found in [4] [6].

A. Connector Model

The development of all electrical devices in the PSL required a connection point, allowing the flow of *complex* variables type. However, this *complex* type is not present in the MSL. Thus, a new type of connection was developed, where the voltage and current values are split into real and imaginary parts. The connector model `PwPin` allows the connection between Modelica models for electrical components.

```
connector PwPin
    "connector for electrical blocks treating
        voltage and current as complex variables"
    Real vr; // real part of the voltage
    Real vi; // imaginary part of the voltage
    flow Real ir; // real part of the current
    flow Real ii; // imaginary part of the current
end PwPin;
```

The `flow` prefix precedes the variable declaration, the connected variables sum zero, whereas an equality rule is applied otherwise. Thus, the `PwPin` connector satisfies both Kirchoff's laws for voltage and current at every connection point. An additional connector, `ImPin` connector is used to connect non-electrical models with the electrical models, allowing the PSL to be compatible with the models from MSL.

```
connector ImPin = Real "connector for non-electrical
     blocks"
```

B. Bus model

The bus model is simply formed by a `PwPin` connector that satisfies Kirchoff's laws. The real and imaginary parts of the current in the Bus are set to zero in order to ensure that the sum of all currents, coming from the components connected to the Bus, is zero. The voltages of all components connected to the Bus will be equated trough the connection of their respective `PwPin` to the same Bus.

```
class Bus
  PwPin p(vr(start = Vo_real), vi(start = Vo_img));
  Real v;
  Real anglev;
  parameter Real Vo_real = 1;
  parameter Real Vo_img = 0;
equation
  v = sqrt(p.vr ^ 2 + p.vi ^ 2);
  anglev = atan2(p.vi, p.vr);
  p.ir = 0;
  p.ii = 0;
end Bus;
```

C. Open-line model

The implemented model follows the π-equivalent model of a transmission line [10]. The Modelica model takes as parameters the series resistance R, the reactance X, the shunt half conductance G and the susceptance B due to leakage current between the phases and ground. When either the receiving or the sending end of the line is open, the current at this end is null, and the equations of the line are simplified:

```
class PwOpenLine
  PwPin p;
  PwPin n;
  parameter Real R "Resistance p.u.";
  parameter Real X "Reactance p.u.";
  parameter Real G "Shunt half conductance p.u.";
  parameter Real B "Shunt half susceptance p.u.";
  parameter Boolean OpenR_end
    "Open at Recieving end (true), Open at Sending
        end (false)";
  Real Zr;
  Real Zi;
equation
  Zr = R*G - X*B;
  Zi = R*B + X*G;
  if (OpenR_end == true) then // Open at Receiving
      node
```

```
    p.vr*(2.0*G + G*Zr - B*Zi) - p.vi*(2.0*B + Zr*B
        + Zi*G) = p.ir*(1.0 + Zr) - p.ii*Zi;
    p.vr*(2.0*B + Zr*B + Zi*G) + p.vi * (2.0*G + G*
        Zr - B*Zi) = p.ir*Zi + p.ii*(1.0 + Zr);
    n.vr - p.vr*(1.0 + Zr) + p.vi*Zi = - p.ir*R +
        p.ii*X;
    n.vi - p.vi*(1.0 + Zr) - p.vr*Zi = - p.ir*X -
        p.ii*R;
  else  // Open at Sending node
    n.vr*(2.0*G + G*Zr - B*Zi) - n.vi*(2.0*B + Zr*B
        + Zi*G) = n.ir*(1.0 + Zr) - n.ii*Zi;
    n.vr*(2.0*B + Zr*B + Zi*G) + n.vi*(2.0*G + G*Zr
        - B* Zi) = n.ir * Zi + n.ii * (1.0 + Zr);
    p.vr - n.vr * (1.0 + Zr) + n.vi * Zi = - n.ir *
        R + n.ii * X;
    p.vi - n.vi * (1.0 + Zr) - n.vr * Zi = - n.ir *
        X - n.ii * R;
  end if;
end PwOpenLine;
```

D. Voltage Dependent Load model

Load model used in the 22-buses system is represented with the static P and Q load model [10]. They are expressed in terms of *algebraic functions* of the bus voltage magnitude and frequency.

$$P = P_0 \left(\frac{V}{V_0}\right)^{\alpha} \left(\frac{\omega}{\omega_0}\right)^{\gamma} \quad (1)$$

$$Q = Q_0 \left(\frac{V}{V_0}\right)^{\beta} \left(\frac{\omega}{\omega_0}\right)^{\delta} \quad (2)$$

In our case the exponents δ and γ are zero because we do not consider frequency dependency. For exponents α and β we use the most accepted model where the active power is represented as constant current ($\alpha = 1$) and reactive power as constant impedance ($\beta = 2$).

```
class PwLoadVoltageDependence "Load with voltage
    dependence."
  PwPin p(vr(start=Vo_real), vi(start=Vo_img), ir(
      start=1), ii(start=0))
  parameter Real P "Active Power p.u.";
  parameter Real Q "Reactive Power in p.u.";
  parameter Real Vo_real=1.0 "Initial voltage at
      node in p.u. (Real part)";
  parameter Real Vo_img=0.0 "Initial voltage at node
       in p.u. (Imaginary part)";
  parameter Real vo = sqrt(Vo_real^2 + Vo_img^2);
  parameter Real omega0=1;
  Real v(start=vo);
  Real a "auxiliary variable. Voltage division";
  parameter Real alpha=1;
  parameter Real beta=2;
equation
  a=v/vo;
  (P*a^alpha) = (p.vr*p.ir + p.vi*p.ii);
  (Q*a^beta) = (-p.vr*p.ii + p.vi*p.ir);
  v=sqrt(p.vr^2 + p.vi^2);
end PwLoadVoltageDependence;
```

E. Regulators

Three different types of complex regulator models have been developed in Modelica and connected to various synchronous machines used in the 22-buses system: a speed regulator, a voltage regulator and a vibrations regulator. These regulators

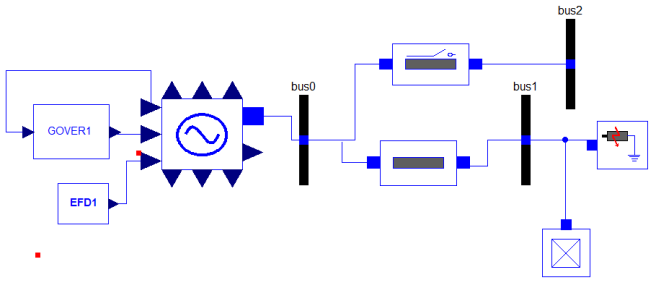

Fig. 1. Modelica diagram of one of the test systems, the `Simple_Test_Model` used to validate the `Bus` model, the `OpenLine` model and the `Voltage Dependent Load` model.

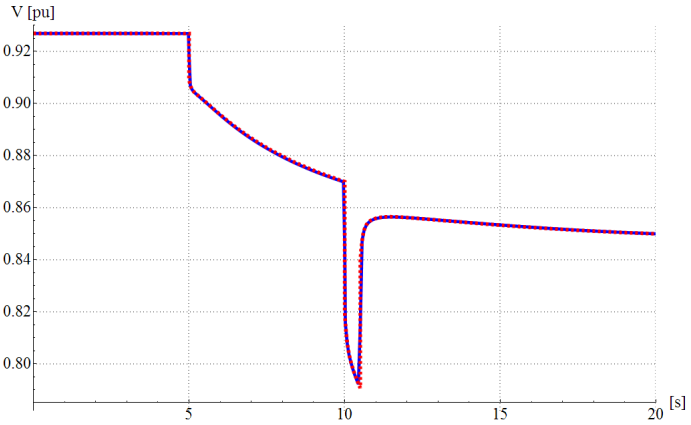

Fig. 2. Voltage variation at node `bus1` of `Simple Test Model` due to the load modification at $t = 5$ s and the fault occurring at time $t = 10$ s and during $0.5s$. The blue dashed curve corresponds to Dymola results and the red continuos line corresponds to results from Eurostag.

were previously built in Eurostag and are used as reference models for their implementation in Modelica. Due to confidentiality issues, the speed and voltage regulations cannot be detailed here, but the vibrations regulator, designed for reducing and stabilising torsional vibrations for turbine-type AC generators has two public patents associated (see [11] and [12]).

III. Validation of Power System Components

The Modelica model of a power system network needs to be initialized with proper steady-state values for the correct behaviour of the corresponding time-domain simulation. However, power flow calculation have not yet developed using Modelica. This is a remarkable characteristic of the benefits of using Modelica language for power system modelling so, the model is decoupled from an specific solver and can be easily exchanged between simulations tools that support the Modelica compiler. In this case, the power flow computation was performed using Eurostag and the same values have been used to initialize the equivalent Modelica model. Thus, the process of model validation with its Eurostag model reference becomes reliable since both Modelica and Eurostag models have the same initial conditions in all the components of

the network: buses, loads, lines, synchronous machines and regulators. For the validation of the electrical components, a small network model was built (see Fig. 1). The system contains three buses, a synchronous generator connected to bus0, a transmission line between bus0 and bus1, a speed governor (`GOVER1`) controlling the angular velocity of the generator, a voltage set point (`EDF1`) maintaining a constant excitation voltage of the generator during the simulation. The open line is connected between bus0 and bus1, and the voltage dependant load is connected at bus 1. The configuration of all the components has been carried out inside the design of the model and there is no need to set the initial conditions, from a power flow solution, because any mathematical solver is able to rapidly find the steady-state condition of this small system. By default, the `dassl` integrator solver used in Dymola sets the initial conditions to 0. Two events were introduced to study the time-response of the system: a load modification at $t = 5s$ and a fault at bus1 occurring at $t = 10s$ during $0.5s$. The comparison between Eurostag simulation and Modelica simulation is shown in Fig. 2

IV. Mid-Size Power System Model Implementation

A. System Implementation

The reference model was originally built and simulated using Eurostag tool. This network model represents a sub-network of the French grid. The initial conditions of the model have been taken from the power flow solution calculated in Eurostag, as well as the topology and parameters of each component, which are taken from the real grid. Figure 3 shows all the Modelica components in the 22-buses system and the connections between them. Due to confidentiality issues, the system does not provide geographical location information. Moreover, the dynamic parameters for machines and regulators have been modified from the original values. The system is composed by 40 buses with 27 of them connected by coupling devices and one disconnected bus. Thus the actual number of buses is reduced to 22. There are 26 lines with one line open at the sending node (explained in section II-C), two fixed tap transformers [4] and 11 transformers with variable tap; one of them is a phase shifting transformer. The system has 6 generators with four of them modelled as synchronous machines defined by their external parameters, based on the M2S machine model form Eurostag (developed in Modelica during the Pegase project [4]), and three different types of regulators. The remaining two generators did not have dynamic data in the original model built in Eurostag and thus are modelled as constant power injection. These are modelled as a constant impedance loads, in Modelica. The system also contains 23 voltage dependent loads. The resulting Modelica model of this 22-buses system contains more than 2500 unknowns and equations. Once all component models were developed and validated against Eurostag, the next step was to include them in the system following the topology given in Eurostag. Once the model was built in Modelica, with all the steady-state and dynamic parameters, simulations in different Modelica tools, Dymola and OpenModelica, have been performed.

B. Initialization of the Modelica model

To initialize a Modelica model, initial guess for all algebraic, continuous and discrete variables need to be provided [14]. These values should either be written manually when designing the model or computed through an external routine. In Modelica language, the initial values for variables and parameters can be specified with an equation through the initial equation construct or by setting the $(fixed = true, start = x0)$ properties of the instance variables. If nothing is specified, the default value would be zero $(start = 0)$ with the property $(fixed = false)$, then the value will be viewed as a guess value. Setting the property $(fixed = false)$, the initial value of the parameter and variables can be computed during initialization. However, not assigning an initial value to algebraic states or using default values (0 or no value) may result in divergences of the initialization algorithm. Following these considerations, in the 22-bus system, voltage for buses, lines, loads and and starting values for machines and regulators are initialized with the steady-state solution from the power flow solution, from Eurostag, as well as P and Q values for loads. Their values are treated as a guess value $(fixed = false)$ for the initialization of the simulation. Only some regulator attributes are treated as parameters with value $(fixed = true)$, which means the its value does not change after the initialization of the simulation. The current values in buses, lines and loads are set to zero and are determined by the static analyser at the initial step of the simulation. Once the model is built with all the starting values, the first step of the Modelica compiler is to correctly determine the initial steady state without initial values for currents. Dymola and OpenModelica manipulate symbolically the initialization problem and generate analytic Jacobians for non-linear problems. Author have experienced different behaviour with these Modelica tools. Using the Modelica compiler within the OpenModelica editor, the initialization calculation can be deactivated, forcing the solver to use the initial conditions from the power flow computation. This simulation configuration ensures the use a power flow solution before any time-domain simulation. Using the Modelica compiler in the Dymola editor, there is no direct method for deactivating the initialization phase in the solver. But the use of guess values for all variables but current values improved the computation speed of the initialization phase.

V. Simulation Results of the Mid-Size Model

The 22-bus system was simulated with the presence of two different dynamic events: a line opening and closing at the sending node, and a fault connected to one node during $1s$ of simulation. As stated in the previous section, both OpenModelica and Dymola have been used. Using OpenModelica, the solver is able to perform the flattening of the model and the initialization, but the time-domain simulations were not executed by the compiler, thus no analysis could be performed using this editor. Using Dymola, the system is flattened, is correctly initialized and correctly simulated, thus the rest of this section is devoted to explain the results obtained with Dymola.

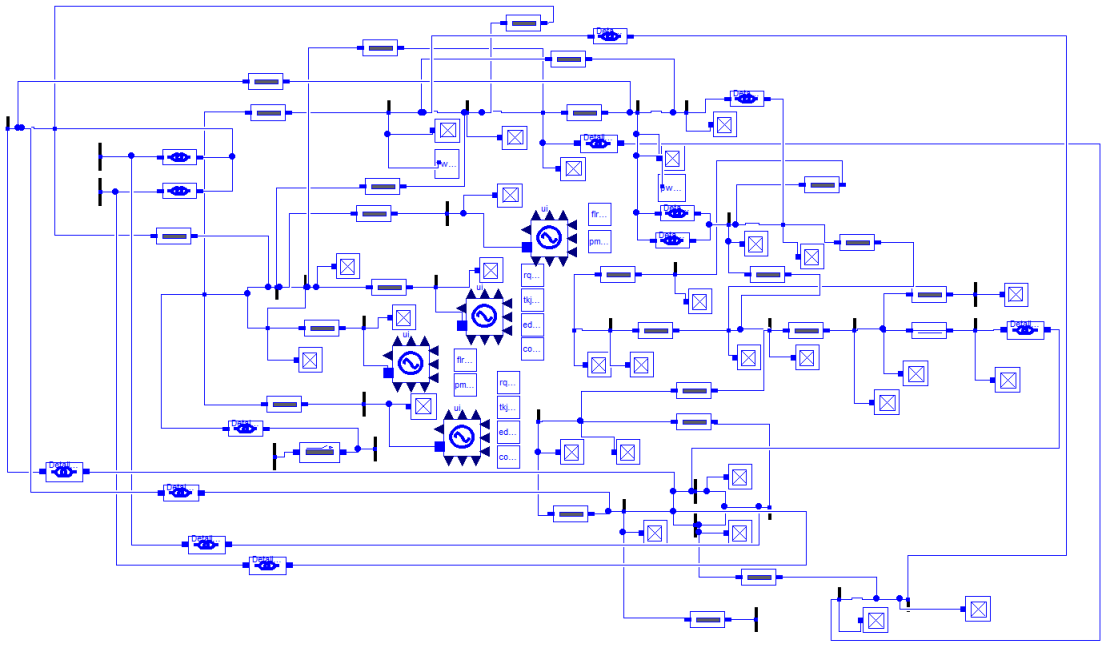

Fig. 3. Power System composed by 22 buses, 26 lines, 11 transformers, 4 synchronous machines with regulators, 23 voltage dependent loads and 2 constant impedance loads.

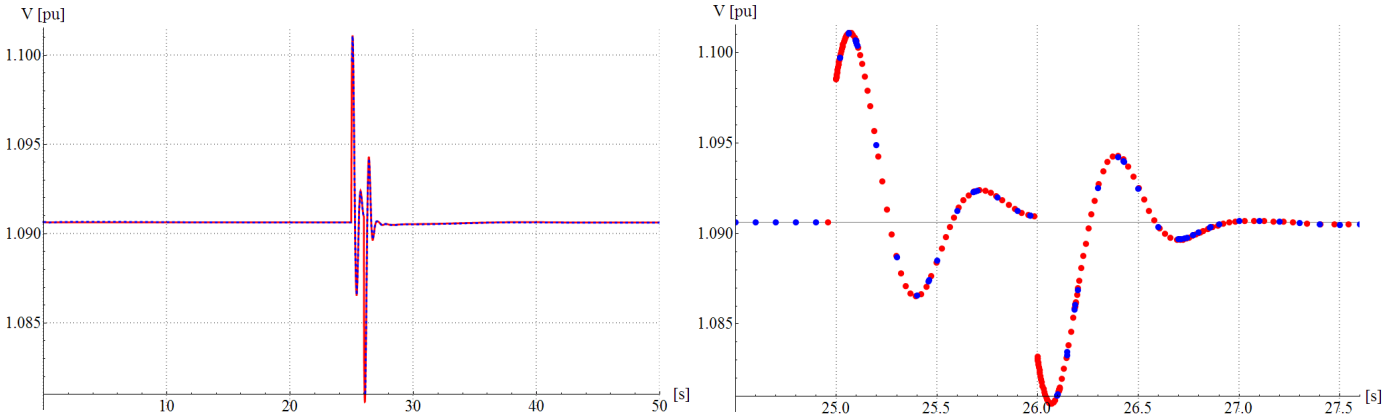

Fig. 4. System response to the opening of a line at the sending node at time $t = 20$ s. The left figure shows the voltage at one specific bus (in p.u.) on the full time-scale, while the figure of the right zooms the plot at the instance when the opening-closing event occurs. Blue lines/dots correspond to results obtained with Dymola simulation environment and red lines/dots correspond to results obtained with Eurostag.

Figures 4 and 5 show the system response to a line opening and a fault in a bus, respectively, using the `dassl` solver with a tolerance of 1×10^{-04}. The results are compared with simulation results from Eurostag. Comparing the different integrator solvers available in Dymola, `dassl` and `lsodar` gave the best results. Smaller tolerance values, increased largely the simulation time without improving the results. Analysing the plots, the system steady-state obtained with Dymola solver coincides with the Eurostag outputs, with a precision of 1×10^{-05} in per unit scale.

VI. Conclusions

A power system library is being developed in Modelica to give users the possibility to build and perform power system time-domain simulations in power systems in an open language, allowing the sharing of models, and providing a common understanding of each of the component's modelling. The use of Modelica language in modelling mid-size power system models proofs that open-source software is also reliable for power system dynamic simulations, as in comparison with proprietary tools. The development of this library is the result of the collaboration between different partners of the iTesla project. The library will undergo a continuous enlargement with new electrical and non-electrical models until the end of the iTesla project in 2015. For building a medium size power system composed by 22 buses and multiple devices, component models existing in the library have been used. Moreover,

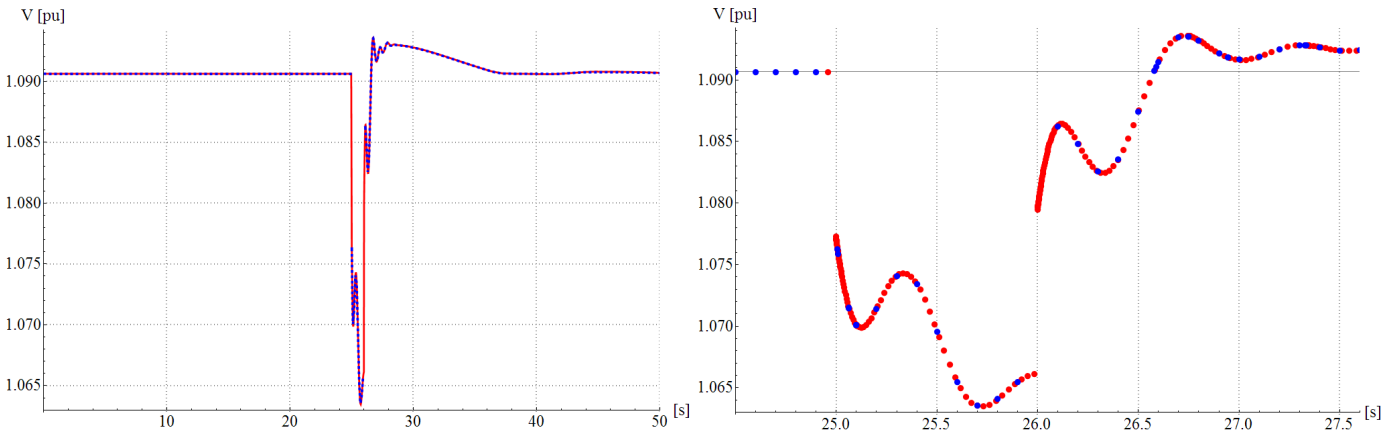

Fig. 5. System response to a fault with impedance $R = 0.5$ p.u. and $X = 0.5$ p.u. in one bus during $1s$. The left figure shows the voltage at one specific bus (in p.u.) on the full time-scale studied, the right figure zooms the plot at the instance when the fault occurs. Blue lines/dots correspond to results obtained with Dymola simulation environment and red lines/dots correspond to results obtained with Eurostag.

new models for electrical and non-electrical components were developed for building and simulate this mid-size system. The system topology was built manually in Modelica following a system previously made in Eurostag, a load flow computation was performed as well as the computation of initial values for machines and regulations, and then all these start variables were introduced in the system in order to initialize and simulate it in Dymola. The time-domain simulations yielded the expected results when compared with the reference system made in Eurostag. However, further analysis and simulations of mid-size power system models should be performed with open source tools like OpenModelica, to prove the complete capabilities of Modelica compiler for the simulation of power system models in different Modelica tools.

REFERENCES

[1] Modelica® and the Modelica Association. http://www.modelica.org/

[2] P. Fritzson. *Principles of Objected-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach* . June 2014.

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

[4] Pan European Grid Advanced Simulation And State Estimation (PEGASE) Project. *D5.2: Proof of concept for an open simulation and model sharing framework validated on a medium size power system.* September 2010. Available at Downloads in http://www.fp7-pegase.com/.

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

[6] T. Bogodorova, M. Sabate, G. León, L. Vanfretti, M. Halat, J.B. Heyberger and P. Panciatici. *A Modelica Power Sytem Library for Phasor Time-domain Simulation*. Innovative Smart Grid Technologies Europe (ISGT EUROPE), 2013 4th IEEE/PES.

[7] OpenModelica: open-source Modelica-based modeling and simulation environment. http://www.openmodelica.org

[8] P. Fritzson and P. Bunus, Modelica, A General Object-Oriented Language for Continuous and Discrete-Event System Modeling, In Proceedings of the 35th Annual Simulation Symposium, 2002, pp. 1418.

[9] Dymola: Tool for modeling and simulation of integrated and complex systems. A comertial product from *Dassault Systemes*. http://www.3ds.com

[10] P. Kundur. *Power System Stability and Control*. McGraw-Hill, Inc. 1994

[11] European Patent FR2475213 (A1). E. Irving and C. Charcossey. *Procede de Regulation auto-adaptative du fonctionnnement d'un groupe turbo-alternateur et regulateur meetant en oeuvre le procede*, issued 1981-08-07.

[12] European Patent FR2654231 (A1). G. Michel and L. Michel. *Device for reducing and stabilising torsional vibrations for turbine-type AC generators*, issued 1991-05-10.

[13] Modelica® - *A Unified Object-Oriented Language for System Modeling. Language Specification*. Version 3.3. Modelica Association, May 2012.

[14] S.E. Mattsson, H. Elmqvist, M. Otter and H.Olsson, *Initialization of Hybrid Differential-Algebraic Equations in Modelica 2.0*, Proceedings of the 2nd International Modelica Conference, Oberpfaffenhofen, Germany, Mar 18-19, 2002