Exploiting Modelica and the OpenIPSL for University Campus Microgrid Model Development

Fernando Fachini[1] Srijita Bhattacharjee[1] Miguel Aguilera[2] Luigi Vanfretti[1] Giuseppe Laera[1]
Tetiana Bogodorova[1] Ardeshir Moftakhari[3] Michael Huylo[4] Atila Novoselac[4]

[1]Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, USA,
{fachif, bhatts10, vanfrl, laerag, bogodt2}@rpi.edu
[2]OPAL-RT Technologies, Canada, Miguel.Aguilera@opal-rt.com
[3]Architectural Engineering, Pennsylvania State University, USA, abm6934@psu.edu
[4]Department of Civil, Architectural, and Environmental Engineering, University of Texas at Austin, USA,
{mhuylo,atila}@utexas.edu

Abstract

The need for modeling different aspects of microgrid design and operation has seen the development of various tools over time for different analysis purposes. In this study, Modelica has been adopted as the language of choice to construct a University Campus Microgrid model, utilizing the Modelica Standard Library and the OpenIPSL library. This paper explores the advantages of utilizing Modelica for campus microgrid modeling, emphasizing its benefits and unique features. Modelica features, such as the use of record structures and replaceable templates prove to be particularly advantageous for the modeling task, enabling flexibility and efficiency in the modeling process. Furthermore, comprehensive validation tests are conducted to ensure the accuracy and reliability of sub-systems (e.g. specific power generator systems), before assembling the microgrid network model as a whole. The results demonstrate the efficacy of Modelica in accurately modeling and simulating microgrids, highlighting its potential for advancing microgrid research and development.

Keywords: Microgrid modeling, Modelica, OpenIPSL, record structures, replaceable templates

1 Introduction

1.1 Background and Related Works

The growing deployment of microgrids over the past few decades can be attributed to a multitude of factors that have collectively shaped their development. These include significant technological advancements, the growing trend towards decentralization in power generation and distribution, the increasing utilization of renewable energy sources, a heightened focus on grid resilience, and comprehensive studies on energy security. Microgrid modeling and simulation provide designers and operators the flexibility to explore diverse design options, and moreover, operational power dispatch scenarios and control strategies, without risking any disruptions in the actual operation of the physical system. This enables the identification of potential issues and facilitates the optimization of control strategies, ensuring the provision of a reliable and efficient power supply.

Over the years, numerous tools have been developed and employed specifically for the purpose of modeling microgrids. These tools serve as essential resources for a microgrid's stakeholders (e.g., operators, owners, etc.) seeking to accurately represent and analyze different aspects involved from the design, to deployment and operation of microgrid systems (Feng et al. 2018). The Distributed Energy Resource Customer Adoption Model (DER-CAM), pioneered by the Berkeley lab, integrates thermal and electrical storage sizing while optimizing equipment selection and operation to effectively reduce energy costs (Marnay et al. 2008).
The National Renewable Energy Laboratory (NREL) has developed the Hybrid Optimization Model for Multiple Energy Resources (HOMER), an optimization model for microgrids. This tool enables evaluating of various equipment options, accommodating various constraints and sensitivities (Mendes, Ioakimidis, and Ferrão 2011).
Diverging from the aforementioned tools, the Smart Grid Computational Tool (SGCT), created by the Electric Power Research Institute, takes a distinct approach to performing techno-economic analysis, providing valuable insights into the economic feasibility and advantages of smart grid implementations by considering a range of factors beyond energy balance (Xu et al. 2017).
The agent-based smart grid simulation software GridLAB-D developed by the Pacific Northwest National Laboratory offers comprehensive capabilities for simulating both the power flow of transmission networks and the performance of individual components within microgrids (Chassin, Schneider, and Gerkensmeyer 2008).

While most of these software tools focus on important microgrid design and analysis aspects, they do not provide adequate means to represent the system's dynamics, where the Modelica language excels. The primary motivation behind selecting Modelica as the modeling language for this work is its adoption by multiple software tools that

support the language. This permits the use of these models across several software platforms, enhancing flexibility and enabling wider accessibility for analysis features beyond power flow and time-series simulations. In particular, Modelica tools provide user-friendly environments for effortlessly linearizing models, which is challenging with traditional power system analysis tools. Meanwhile, dynamic simulation analysis, including transient and steady-state simulations, can be performed efficiently, providing valuable insights into the system's behavior under different operating conditions. Additionally, these features of the Modelica language allow to perform stability analysis, enabling the assessment of stability margins and the identification of potential stability issues within the microgrid's power system models.
In addition, Modelica-based libraries enable the utilization of diverse models that encapsulate the behavior and interactions of systems across different domains such as electrical, mechanical, thermal, and more. As highlighted in (Winkler 2017), Modelica exhibits substantial potential as a robust power system modeling tool when utilized in conjunction with the Open-Instance Power System Library (OpenIPSL) library (Baudette et al. 2018; Castro et al. 2023). The distinct features of the Modelica language, such as the ability to create and manage record structures, prove particularly advantageous for power system models that necessitate initialization with power flow data in various power dispatch scenarios. Furthermore, the inherent replaceable model structure and object orientation of Modelica greatly facilitate the modeling of a microgrid's sub-systems and components.

This paper presents the modeling of a University Campus Microgrid utilizing the Modelica language (Fritzson and Engelson 1998; Fritzson 2014) and the OpenIPSL (Baudette et al. 2018; Castro et al. 2023) and explores the benefits of utilizing specific unique features of Modelica for microgrid modeling. Through this work, valuable insights are gained into the potential of Modelica as a versatile modeling language for microgrid modeling, that can ultimately contribute to advancements in microgrid research, design, operation, and optimization.

1.2 Motivation

On the topic of microgrid modeling utilizing the OpenIPSL library, the author's previous work (Fachini et al. 2023) aimed to demonstrate the use of Modelica as a viable modeling language for microgrid, as well as the engineering rationale on how to translate document information into power system models using OpenIPSL. This work, on the other hand, is focused on sharing the exploitation of Modelica features and modeling details of a real-world microgrid system, that albeit similar in implementation as in paper (Fachini et al. 2023), represents a different facility in Texas. The topics explored in this paper address the utilization of records for both component parameter and initial condition instantiation, implementation of reusable templates (using `replaceable`) with a structure for generation unit implementation and characterization of variants, generation unit validation through simulation comparison, and description and simulation results of the microgrid model.

1.3 Contributions

The paper's main contribution is to illustrate how Modelica-specific features can be used in the implementation of a real-world microgrid. Thus, the contributions are:

- To describe the record structures used for initialization of the dynamic microgrid model implemented utilizing the OpenIPSL Modelica library, and component parameter instantiation.
- To describe the implementation of a synchronous generator-based replaceable model that allows the user to create variants by easily configuring a generation unit model and the use of inheritance.
- Description of the validation of the generation units utilized in the microgrid model using **csv compare** (Modelica-Tools n.d.) to contrast with results from the domain-specific tool.
- Description of the implementation of the microgrid model and closed-loop system stability through root locus analysis.

1.4 Paper Structure

This paper is structured as follows: Section 2 describes the university campus microgrid model and the advantages of using Modelica for microgrid modeling including the records structure proposed to handle the power-flow variables. In Section 3, we illustrate how this data container can be linked to OpenIPSL, validate the generation units developed, and benchmark the power flow values against the results obtained with commercial tools. Section 4 presents the simulation result. Finally, Section 5 concludes the work.

2 University Campus Microgrid Modeling utilizing the OpenIPSL Modelica Library

When modeling power systems dynamics, there are two distinct mathematical representations that one can choose: Electromagnetic Transient (EMT) based models, or Phasor domain (RMS) based models. EMT-based power system models provide a three-phase waveform representation of instantaneous values for currents and voltages, used for the analysis of high-frequency events in the grid, such as power electronic switching, lightning phenomena, etc (Mahseredjian, Dinavahi, and Martinez 2009). RMS-based power system models provide a positive sequence phasor representation of currents and voltages, used to model electromechanical oscillations in the system. This modeling representation considers only the fundamental frequency of the AC voltages and currents of the real-life electrical system. The OpenIPSL library components are validated against Siemens PTI PSSE (Siemens PTI 2017), therefore

the majority of the models within the library are RMS-type models.

Figure 1 displays the campus microgrid model, described in this paper, and implemented utilizing OpenIPSL components. This university campus microgrid is located in Texas and is connected to the local utility through four 69kV feeder lines as shown in Figure 1. The utility bus voltage level is reduced to 12kV through four step-down transformers, namely *T1* - *T4*. The microgrid also contains a 4.16kV portion in bus *B15*, stepped down from 12kV to 4.16kV through transformers *T5*, and *T6*.

The university campus microgrid is powered by two combustion turbo generators (CTs) and four steam turbo generators (STs). The two oldest steam turbine generation units rarely operate, typically online for a few days a year in extreme load conditions. For that reason, the model in Figure 1 contains two CTs and two STs, producing power at 12kV each. The maximum amount of power that the combined generation units can produce is approximately 127MW. The CTs and STs are composed of three essential components: the synchronous machine, which converts mechanical energy into electrical energy, the prime mover, which drives the synchronous machine, and the control system, which regulates and monitors the overall operation. Together, these components work in tandem to ensure the optimal performance of the CTs and STs within the power plant. Both the *CT1* and *ST1* utilize the *GENROU* synchronous machine, *IEEEVC* as the voltage compensator, a power system stabilizer (*PSS*), and the *ESST4B* and *ESST4A* as the excitation system respectively. The *CT2* and *ST2* models share a similar internal structure, with the distinction that they lack a voltage compensator. Notably, the *CT2* and *ST2* models employ the *AC7B* excitation system. The transformers are denoted as *T1*, *T2*, etc. The power transmission lines are denoted as *X1*, *X2*, and so on. Correspondingly, the buses within the network are labeled as *B01*, *B02*, etc, and the loads are symbolized by *L01*, *L02*, and so forth. The blocks labeled *Y* in the network represent the shunt component with conductance and susceptance as its parameters. The microgrid model also displays an electrical contingency block, namely *pwFault*, used for three-phase to ground fault testing. Lastly, the *UTI* component represents the utility bulk power system, modeled as an infinite source model.

This campus microgrid has a peculiarity to its operation: it is only allowed to purchase a limited amount of power from the utility grid in emergency situations, i.e., it operates to meet the campus demand on a continuous basis. Currently, the university satisfies the campus power demand from internal generation sources, however, with the increasing variability in the price of natural gas, the university management is studying the possibility of both purchasing more power and also selling excess power when lucrative. The development of this microgrid's model in OpenIPSL, utilizing the Modelica modeling language, will open the possibility of expanding the current model for multiple purposes, including the optimization of operation and revenue of both electrical and thermal domains of the combined heat and power system. This paper will focus on the electrical domain, while future work will expand it to represent the thermal domain, i.e., heat generation and distribution.

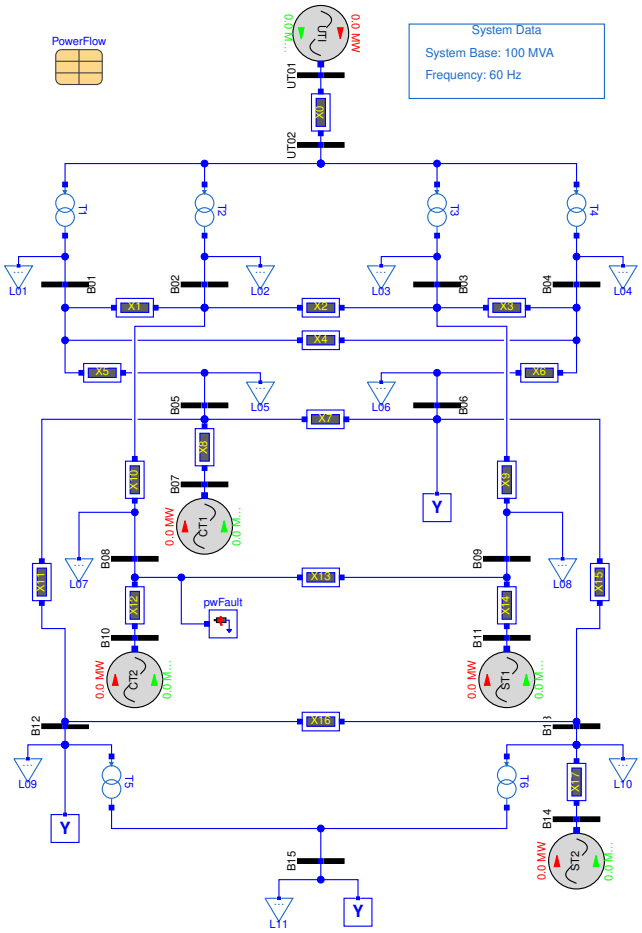


Figure 1. University Campus Microgrid "Virtual" Testbed Model.

2.1 Advantages of using Modelica for Microgrid Modeling

The Modelica modeling language was developed to provide a systematic approach to developing models of cyber-physical systems through mathematical equations (Fritzson and Engelson 1998). With that in mind, the OpenIPSL Library was implemented as a means to study power system dynamics with a modern and modular approach to power system modeling through the usage of the Modelica modeling language. The modeling paradigm used for the implementation of the OpenIPSL Library is the phasor-domain representation of power system components, meaning that the models are represented in terms of a set of differential-algebraic equations that represent the electromechanical transients in the power system (Kundur and Malik 2022). The general form of the differential-algebraic set of equa-

tions in power system studies is:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}(\mathbf{t}), \mathbf{y}(\mathbf{t}), \mathbf{t}), \\ 0 &= \mathbf{g}(\mathbf{x}(\mathbf{t}), \mathbf{y}(\mathbf{t}), \mathbf{t}), \end{aligned} \tag{1}$$

with initial conditions, $(\mathbf{x}_0, \mathbf{y}_0, \mathbf{t})$, i.e. $\mathbf{0} = \mathbf{f}(\mathbf{x}_0, \mathbf{y}_0, \mathbf{t})$ (Kundur and Malik 2022).

When implementing phasor-domain dynamic models, the assumption that is taken is that the balanced system is at its nominal system frequency, being it 50 or 60 Hz, depending on the system in study. This particular simplification allows representing passive system components, such as transmission lines, with algebraic equations that rely on lumped impedance values instead of a set of differential equations. Therefore, the differential equations in the phasor-domain models describe the states of generation units and their controls. The initial condition values utilized to generate different simulation starting points are derived from steady-state simulations of the electrical grid, known as power flow analysis (Powell 2004).

Unlike traditional power system tools for phasor-domain dynamic simulation, OpenIPSL was built utilizing the Modelica modeling language. With its object-oriented constructs, models can be reused and modeled conveniently, especially intricate systems containing multiple components (Fritzson and Engelson 1998; Fritzson 2014). Among several benefits of utilizing the Modelica language for power system simulation, two are of major interest and are implemented to be used in the microgrid models: (1) record structures, herein used for both system initialization and system characterization and (2) replaceable model templates, herein used for creating variants of generation units.

2.2 Initial Guess Record Structure

On the modeling front, the Modelica language utilizes its object-oriented paradigm to enable users to create models in a hierarchical manner. Similarly, one can also manage model parameter values and initial guess data through a hierarchical structure based on records. Figure 2 displays the *PfData* record structure that is used to initialize the dynamic power system model. The initial guess data necessary for the start of the dynamic simulation utilizing OpenIPSL components, which is obtained through power flow simulations, are voltage magnitude and angle of all buses, active and reactive power injection/consumption from the generation units, and the respective load profile of the microgrid.

The *PowerFlowTemplate* is a record template for the entire record structure which constrains the power flow records. The record files *BusTemplate*, *LoadTemplate*, and *MachineTemplate* define partial models that define the parameters that are extended in the data-filled record files. The record files *PfBus1*, *PfLoad1*, *PfTrafo1* and *PfMachine1* are extension records from the record templates, where the initial guess parameters from the templates are the attributed values. The *Pf1* record holds the template for all the record components. The numbering has been assigned to signify the initial record, creating room for incorporating multiple additional values for each component within their respective records across diverse power dispatch scenarios. This means that the defined record templates can have a multitude of different parameter values for multiple different initial guess values in the same system. The implementation of such a record structure can be done

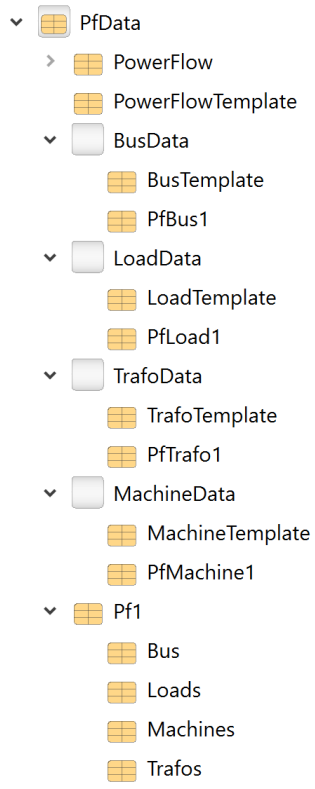


Figure 2. Record Structure for Dynamic Model Initial Guess

manually, but it can also be done automatically through an automation tool, namely the **pf2rec** Python script, that converts power flow simulation results into `*.mo` record file (Dorado-Rojas et al. 2021).

2.3 System Characterization Record Structure

Records can also be used for model parameter setup, which is useful when the user has the intention of modifying multiple system parameters to effectively study different grid or component models, without actually re-implementing the model from the ground up. This feature empowers users to easily customize and adapt the model based on specific requirements or scenarios by manipulating the record structure. This capability not only saves time and effort but also enhances model reusability and promotes iterative model refinement.

Figure 3 displays the *DynParamRecords* record structure, which serves as a repository for parameter values related to the model equipment, including the synchronous generator (Machine), exciter (ES), and the power system stabilizer (PSS). This record structure efficiently stores and organizes the data associated with each individual component, and

visually demonstrates the arrangement of the record structure, showcasing how the data for each machine or exciter model is maintained within this organized framework. *MachineDataTemplate*, *ESSTxATemplate, ESSTxBTemplate* and *ACxBTemplate* record files are also partial record models that define the parameters of the generation unit equipment. The set of record models, excluding the aforementioned ones, in the sub-packages *MachineData*, *ESData*, and *PSSData* extends their template files to attribute values to the defined component parameters. *CT1*, *CT2*, *ST1*, and *ST2* as denoted in the record structure as *CTG1*, *CTG2*, *STG1*, and *STG2* respectively, extend the *GUDynamicTemplate*, which is used as a `replaceable` in *GUDynamics*. This last record file is utilized in the models in order to (re)parametrize the components in the model.

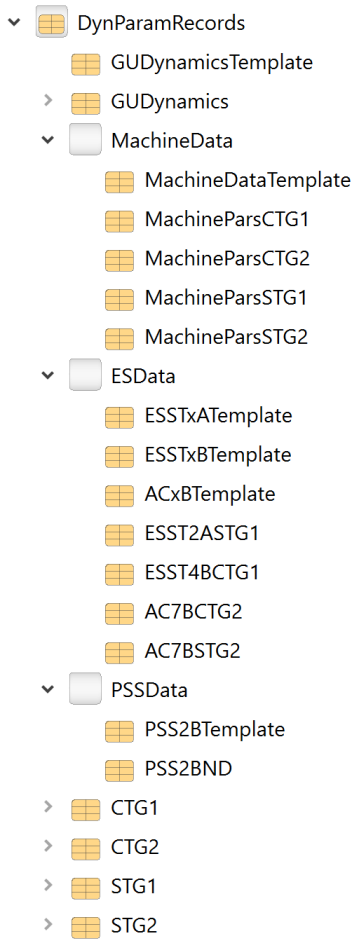


Figure 3. Data Record Structure for Generation Units

2.4 Replaceable Model Template for the Generation Units (CT and ST)

Because of the way the models are implemented in OpenIPSL, each type of generation unit component has multiple extensions to a base class. This means that it is possible to implement an "ALL-IN-ONE" generation unit model architecture that can be used to create variants that represent different generator units based on the different types of synchronous generator models, exciter models, and power system stabilizer models. For instance, Figure 4 displays an example of the *CT1* gas-turbine generation unit.

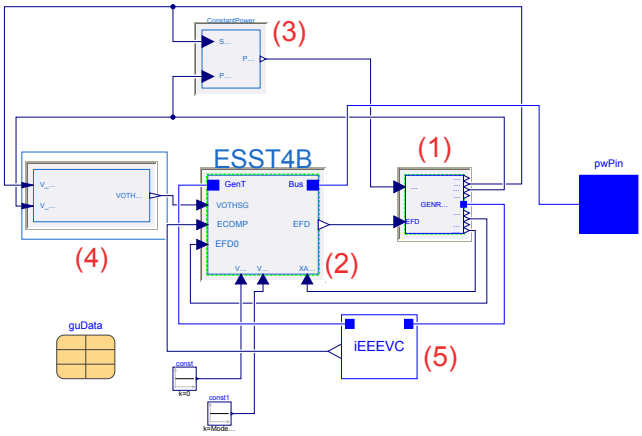


Figure 4. Replaceable Generation Unit Model Example

The sunken gray components are replaceable models for the synchronous generator (1), exciter (2), PSS (4), and turbine-governor model (3). Component (5) from Figure4 is a voltage compensation block. For the purpose of this work, the turbine-governor model is defined as a constant mechanical power to the generator, because of the lack of information on the turbine during the model implementation phase, however, this will be expanded in future work. Inspecting the generation unit component in the microgrid model, shown in Figure 5, one can observe the result of enabling the use of the records for parameter definition and model initialization (i.e. the use of the power flow data as an initial guess for the initialization problem).

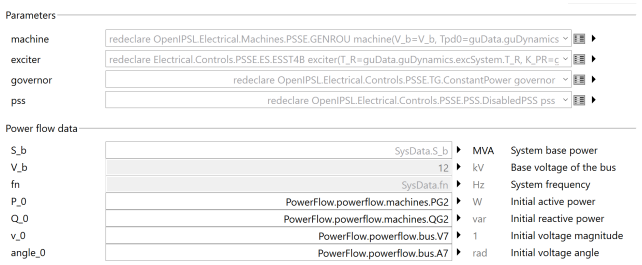


Figure 5. Parameters and Power Flow Data from the CT1 Model

The image displays the possibility of the user selecting different component models for the replaceable *machine*, *exciter*, *governor*, and *pss* components. On the Power flow data section, the user is able to select initialization values for *P_0, Q_0, v_0,* and *angle_0* based on values from the *PowerFlow* record file shown in Figure 2.

3 Validation System Models for the Generation Units

Before assembling the entire microgrid model, this work proposes the development of validation models that allows checking if the generator unit models have been charac-

terized correctly by reutilizing a partial model from the OpenIPSL library that is used for unit testing of the library. This helps in minimizing the complexity of debugging when assembling the microgrid system model, as the most crucial dynamic sub-systems have been verified in this step. To this end, the modeling and verification process is discussed next.

3.1 Generators

The campus microgrid power plant utilizes two gas-fired combustion turbo-generators (*CT1* and *CT2*) that produce steam at high pressure. This steam is then directed towards the two steam turbo-generators (*ST1* and *ST2*) in order to generate electricity. This section provides an in-depth explanation of the *CT* model, shedding light on the internal structure and components of the model.

Figure 4 provides a visual representation of the *CT1* model, offering a diagram view that highlights its key components. The central element is the *GENROU* synchronous machine, which is driven by a constant power component. The *IEEEVC* component acts as a voltage compensator, supplying the necessary voltage to the exciter, represented by the *ESST4B* component. The output of the exciter model i.e. the excitation voltage *EFD* is fed to the synchronous machine. Meanwhile, the power system stabilizer (*PSS*) continuously monitors the generator's shaft speed and electrical power output. To interface with the wider grid, the *pwPin* serves as the connection point between the generation unit and the rest of the system.

The *CT2* model, depicted in Figure 6, follows a similar approach, with the components following a similar numbering convention as the one in Figure 4. However, there are distinct differences, specifically in the exciter model, which is *AC7B* in this case. Additionally, the absence of the voltage compensator sets it apart from the previous model. The *guData* component within the model is used to parameterize the model components, including the machine, exciter, and power system stabilizer. By selecting the data record associated with *CT2*, the component parameters are automatically populated from the record structure, as depicted in Figure 3. This approach streamlines the process of configuring the model components with the appropriate values, enhancing efficiency and ease of use.

3.2 Single Machine Infinite Bus Test System

To ensure dynamic modeling accuracy, the generation units consisting of the two combustion turbo-generators (CTs) and the two steam turbo-generators (STs) undergo a validation process through the creation of individual test models. The single-machine infinite bus test system model is implemented utilizing the Modelica modeling language with the OpenIPSL Library components, simulated using Dymola (Brück et al. 2002), and in the Siemens PTI PSSE tool. The simulation results from both tools are then compared via **csv compare** tool. This comparison algorithm fits a tolerance tube around the data set, where the tube is linked to a tolerance value set by the user, and the simulation

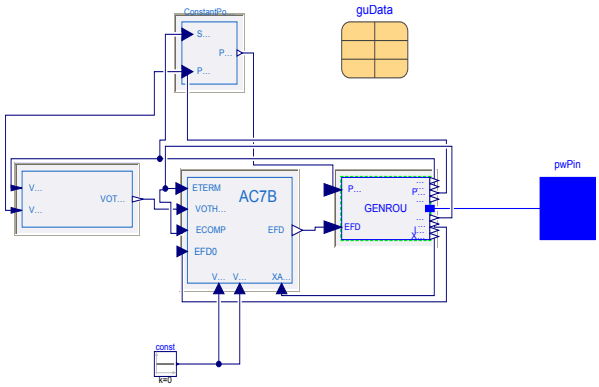


Figure 6. Combustion Turbo-Generator (CT2) Model

result values are then checked to see if they lie within the tube. In order to validate the Modelica-based model against the reference Siemens PTI PSSE model, a fault event is generated in order to evaluate the dynamic response of the generation unit and its controls.

Figure 7 depicts the test model for the *CT2*, in which the *pwFAULT* component from the OpenIPSL library is utilized to simulate a three-phase electrical fault contingency. This fault is applied to the designated *FAULT* bus, specifically from 2 to 2.15 seconds. The introduction of this fault condition results in a reduction in the voltage magnitude at the specified bus, thereby reflecting the ensuing impact on the system dynamics. Figure 8(a) and Figure 8(b) depict the active, and reactive power injection from the generation unit being validated (*CT2*).

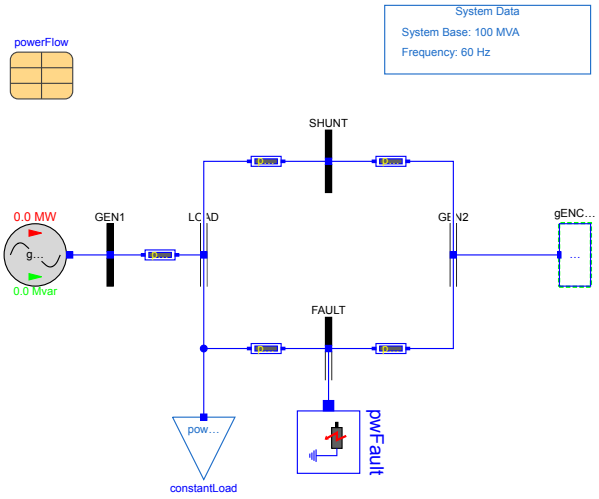
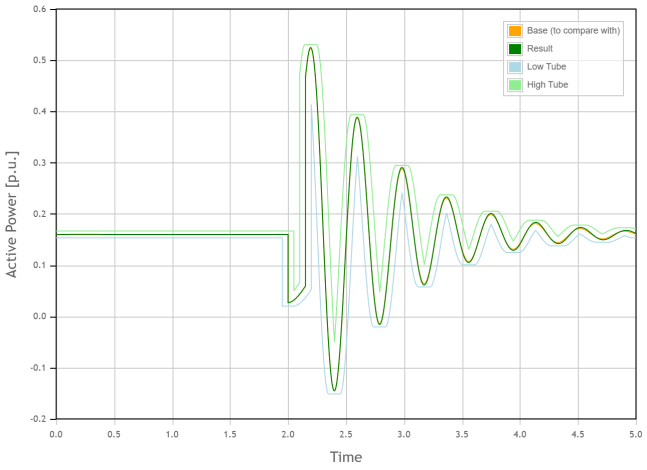


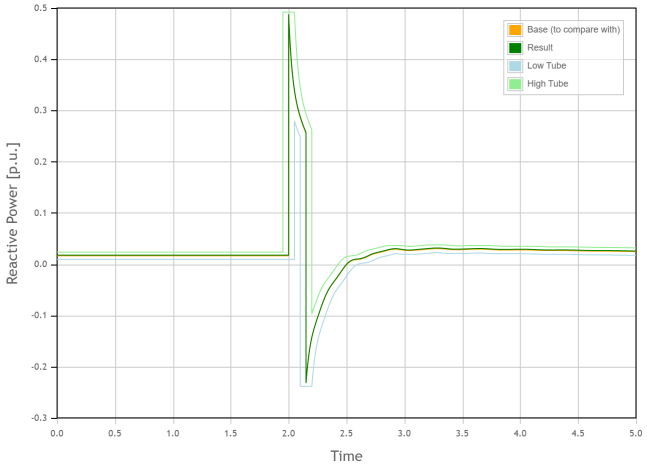
Figure 7. Single Machine Infinite Bus Validation Test Model with CT2

The orange curve is the simulation plot from Siemens PTI PSSE (Base), while the dark green curve is the simulation plot from Dymola (Result). The blue and light green curves are the lower and upper values of the tube mentioned earlier in the subsection, with a tolerance value chosen to be 0.01. From the comparison result, one can observe that the simulation in Dymola is practically identical to the simulation results from Siemens PTI PSSE and that both

curves are inside the tolerance tube. This means that the simulation passed the validation process and it is adequate to be used in the microgrid model. The same validation process is conducted in all the generation unit models implemented in this work, namely the two gas turbines and the two steam turbine units.



(a) Active Power Injection from CT2



(b) Reactive Power Injection from CT2

Figure 8. Active power injection comparison simulation results utilizing **csv compare** algorithm Figure 8(a), and reactive power injection comparison simulation results utilizing **csv compare** Figure 8(b).

4 Simulation Results

Paper (Fachini et al. 2023) discussed two simulation results for a similar microgrid model of another university campus in Colorado: an electrical contingency study in the microgrid, and an eigenvalue analysis of the microgrid. To expand on available Modelica capabilities in this work, using the newly developed model for a real-world microgrid in Texas, the authors explore the stability of the microgrid system when varying the proportional gain K_p from the exciter proportional-integrator (PI) controller in *CT1*. It is worth mentioning that the addressed PI controller takes in as input the error signal between a voltage reference and the terminal voltage.

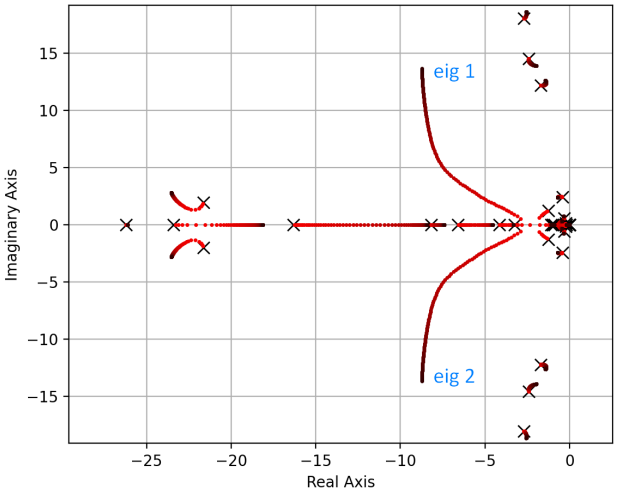


Figure 9. Root Locus Result for Increasing Values of K_p in the CT1 Generation Unit

The equilibrium solution to the system when increasing the proportional gain in increments can be used to perform a root-locus analysis, as an exploration of the system's eigenvalues with changing proportional gain values. Figure 9 displays the root-locus analysis done in the microgrid model from Figure 1, where K_p of the exciter from *CT1* generation unit is systematically for the range $[1, 500]$.

The X markers in Figure 9 define the eigenvalues from the first root locus simulation for $K_p = 1$. The subsequent iterations of the root locus plot are displayed in a gradient color spectrum, with vibrant red being values near the minimum of the K_p range and dark maroon being values near the maximum of the K_p range. The root locus plot displays a complex eigenvalues pair, labeled *eig 1*, and *eig 2*, which are associated with the exciter's voltage. As K_p increases, the complex eigenvalue pair present a reduction in the real components, which implies an increase of dampening.

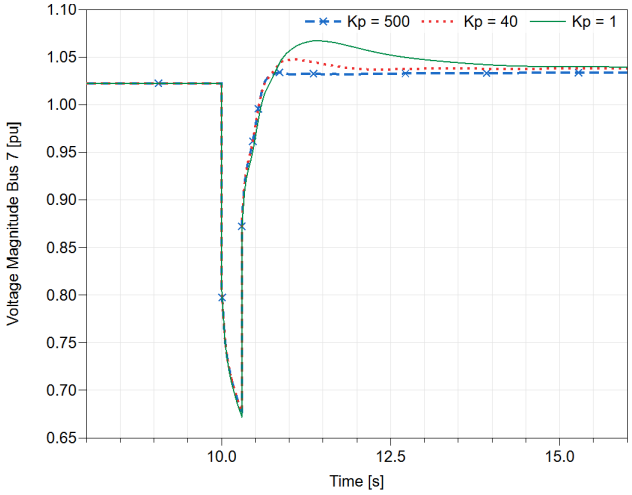


Figure 10. Time domain-simulation of Voltage in Bus 7 for different K_p values

This is verified in the plot shown in Figure 10, where a fault and a consequent line trip are applied to the system in Figure 1 to display the oscillatory behavior of the voltage magnitude at Bus 7. The voltage magnitude at Bus 7 for three different values for the proportional gain in the exciter: $K_p = 1$, $K_p = 40$, and $K_p = 500$, are shown. As expected, increasing the value of the gain results in a more aggressive dampening of the voltage magnitude oscillations, all due to a reduction in the real component the complex eigenvalue pair *eig 1*, and *eig 2*.

5 Conclusions

In this work, Modelica and the OpenIPSL library have been utilized to model a real-world university campus microgrid. The implementation of Modelica allowed leveraging its unique features, such as the record structures and replaceable templates, to effectively design and parameterize the generation units of the microgrid. To ensure reliable performance, each generation unit underwent a separate validation test before being integrated into the main grid. By adopting this approach, the advantages of utilizing Modelica in the development and validation of microgrid models have been demonstrated, bringing added value with multiple potential uses, such as linear-model-based analysis, which is challenging with domain-specific tools.

The study has been based on the documentation from the plant engineers, but unfortunately, there were not enough documents on the turbine governor (*TG*). Future work includes the TG and thermo-fluidic system used for the optimization of the heat-and-power model.

Acknowledgements

This paper is in part, based upon work supported by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under the Advanced Manufacturing Office, Award Number DE-EE0009139, and in part by the National Science Foundation Award No. 2231677.

References

Baudette, Maxime et al. (2018). "OpenIPSL: Open-instance power system library—update 1.5 to "iTesla power systems library (iPSL): A modelica library for phasor time-domain simulations"". In: *SoftwareX* 7, pp. 34–36.

Brück, Dag et al. (2002). "Dymola for multi-engineering modeling and simulation". In: *Proceedings of modelica*. Vol. 2002. Citeseer.

Castro, Marcelo de et al. (2023). "Version [OpenIPSL 2.0. 0]-[iTesla Power Systems Library (iPSL): A Modelica library for phasor time-domain simulations]". In: *SoftwareX* 21, p. 101277.

Chassin, David P, Kevin Schneider, and Clint Gerkensmeyer (2008). "GridLAB-D: An open-source power systems modeling and simulation environment". In: *2008 IEEE/PES Transmission and Distribution Conference and Exposition*. IEEE, pp. 1–5.

Dorado-Rojas, Sergio A et al. (2021). "Power flow record structures to initialize openipsl phasor time-domain simulations with python". In: *Modelica Conferences*, pp. 147–154.

Fachini, Fernando et al. (2023). "Developing a Campus Microgrid Model utilizing Modelica and the OpenIPSL Library". In: *2023 11th Workshop on Modelling and Simulation of Cyber-Physical Energy Systems (MSCPES)*. IEEE, pp. 1–6.

Feng, Wei et al. (2018). "A review of microgrid development in the United States–A decade of progress on policies, demonstrations, controls, and software tools". In: *Applied energy* 228, pp. 1656–1668.

Fritzson, Peter (2014). *Principles of object-oriented modeling and simulation with Modelica 3.3: a cyber-physical approach*. John Wiley & Sons.

Fritzson, Peter and Vadim Engelson (1998). "Modelica-a unified object-oriented language for system modeling and simulation". In: *ECOOP*. Vol. 98. Citeseer, pp. 67–90.

Kundur, Prabha S and Om P Malik (2022). *Power system stability and control*. McGraw-Hill Education.

Mahseredjian, Jean, Venkata Dinavahi, and Juan A Martinez (2009). "Simulation tools for electromagnetic transients in power systems: Overview and challenges". In: *IEEE Transactions on Power Delivery* 24.3, pp. 1657–1669.

Marnay, Chris et al. (2008). "Optimal technology selection and operation of commercial-building microgrids". In: *IEEE Transactions on Power Systems* 23.3, pp. 975–982.

Mendes, Gonçalo, Christos Ioakimidis, and Paulo Ferrão (2011). "On the planning and analysis of Integrated Community Energy Systems: A review and survey of available tools". In: *Renewable and Sustainable Energy Reviews* 15.9, pp. 4836–4854.

Modelica-Tools (n.d.). *Modelica-Tools/CSV-compare: Tool to compare curves from one CSV files with curves from other CSV files using an adjustable tolerance*. URL: https://github.com/modelica-tools/csv-compare.

Powell, Lynn (2004). *Power system load flow analysis*. McGraw Hill professional.

Siemens PTI (2017). "PSS®E 34.2.0 model library". In: *Siemens Power Technologies International, Schenectady, NY*.

Winkler, Dietmar (2017). "Electrical power system modelling in modelica–comparing open-source library options". In: *Proceedings of the 58th Conference on Simulation and Modelling (SIMS 58) Reykjavik, Iceland, September 25th–27th, 2017*. 138. Linköping University Electronic Press, pp. 263–270.

Xu, Liu et al. (2017). "A review of the ARRA smart grid projects and their implications for China". In: *January. LBNL-1007122*.