SoftwareX 21 (2023) 101277

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Software update

Version [OpenIPSL 2.0.0] - [iTesla Power Systems Library (iPSL): A Modelica library for phasor time-domain simulations]

Marcelo de Castro [a,*], Dietmar Winkler [b], Giuseppe Laera [a], Luigi Vanfretti [a], Sergio A. Dorado-Rojas [a], Tin Rabuzin [c], Biswarup Mukherjee [d], Manuel Navarro [e]

[a] Rensselaer Polytechnic Institute, Troy, NY, United States
[b] University of South-Eastern Norway, Porsgrunn, Norway
[c] Siemens Gamesa, Stockholm, Sweden
[d] MINES Paris - PSL, Centre PERSEE, Sophia Antipolis, France
[e] Electric Reliability Council of Texas, Austin, TX, United States

ARTICLE INFO

Article history:
Received 15 November 2022
Accepted 17 November 2022

Keywords:
Modelica
Power systems
Simulation
Power system modeling
Power system dynamics

ABSTRACT

This paper describes Open-Instance Power System Library (OpenIPSL) version 2.0.0 and its most recent enhancements. This new version brings upgrades that include more robust models that are better documented, and example systems that illustrate certain functionalities to users. Repository and library documentations have been enhanced and expanded, and the library is now released under a new license. Changes are meant to foster user and developer communities, while providing more attractive frameworks for collaborative work to be carried out with the library.

Code metadata

Current code version	v2.0.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-22-00373
Code Ocean compute capsule	none
Legal Code License	3-Clause BSD license
Code versioning system used	git
Software code languages, tools, and services used	Modelica (code language), Github (services)
Compilation requirements, operating environments & dependencies	Modelica Standard Library (MSL) 3.2.3, Modelica IDE, and corresponding C compiler
If available Link to developer documentation/manual	https://doc.openipsl.org
Support email for questions	luigi.vanfretti@gmail.com

1. Introduction

Following the completion of the iTesla project, the original iTesla Power Systems Library (iPSL) [1] was forked into a new project, the Open-Instance Power System Library (OpenIPSL). Its primary goal is to provide a robust set of models that can be used for research and teaching activities involving power system phasor-domain stability studies, while providing a transparent library development framework. Since the release of version 1.5.0 [2], the OpenIPSL has also focused its efforts towards developing a user community, feature that can be of tremendous advantage for the library. Therefore, many changes that have been made to this new version of OpenIPSL are developed to foster the user and developer communities. Version 2.0.0 also brings a more robust library base, with structural changes that enhance numerical simulation. Furthermore, the library brings new models, while already-present models have been revisited for

DOI of original article: https://doi.org/10.1016/j.softx.2016.05.001.
* Corresponding author.
E-mail address: marcelo.decastrof@gmail.com (Marcelo de Castro).

https://doi.org/10.1016/j.softx.2022.101277

cleaning and updating implementations, and fixing bugs. These new modifications are discussed in Section 2.

In addition, a tool that creates a bridge between the OpenIPSL and other power system simulation tools has been developed. In this new library version, a Python tool has been created to automate processes needed to provide starting guess values used by Modelica tools in the initialization of power system's dynamic models. This new tool and its valuable usage to the simulation of power system models assembled with OpenIPSL are briefly described in Section 3.

2. New features added to the library

2.1. Licensing, facilitating contributions and documentation

The previous Mozilla Public License 2.0 was replaced by entity and individual versions of the BSD 3-clause license. This modification facilitates features developed in different collaborations to be securely added to the library.

In addition, to facilitate community contributions, the library now has an automated procedure for verification and storage of signatures on contributor agreements, allowing transparency in library development with multiple contributors. This has been implemented using "Contributor License Agreement assistant" (github.com/cla-assistant), and is automated through GitHub services.

The GitHub repository now has more succinct documentation that directly informs users about the library itself. A code of conduct together with contribution guidelines have been added to enable better interactions between users and the multiple developers. Moreover, documentation was added to the new and revisited Modelica models. This, together with the recently-added `User's Guide`, allows users to be informed about functionalities and characteristics while using the library in a Modelica-compliant tool.

2.2. Stable base version

The library now uses Standard International (S.I.) and per-unit units in quantities, allowing a better compliance with the Modelica language, and facilitating the development of consistent models. It also enables the `Types` feature to be used throughout the library, allowing the scaling of differential equations [3]. Hence, it culminates previous efforts to date in a robust and stable base version that is compatible with multiple Modelica-compliant tools, such as Dymola®, OpenModelica®, Modelon Impact® and SystemModeler®. Fig. 1 depicts simulation results obtained using different Modelica-compliant tools for the same OpenIPSL model.

A software-to-software validation procedure has been applied to many models in the library, allowing bugs and inefficient implementations to be identified and corrected. Models that underwent this process have a been graphically annotated using a green line around their icons, indicating the verified component model to the user. The validation results are now available publicly for use of third parties, see: https://alsetlab.github.io/NYPAModelTransformation/reports. In addition, new models have been also added to the library. Examples of such models are the `ULTC` (Under-Load Tap-Changing) transformer, the `GGOV1DU` a variant of the general purpose turbine-governor models `GGOV1` and the `SoftPMU` which may allow users to develop and test applications that need estimated bus frequency values for control design [4].

2.3. Examples and tests

Originally a set of `Application Examples` that illustrated how the library can be used to model both common test models and real-world grids was located outside the library itself as an independent `package`, which created confusion for new users. Hence, this has been updated and it is now available within OpenIPSL's main package under `Examples`. As a consequence, the set of examples is now maintained together with the main library, resulting in more efficient continuous development. Furthermore, the library's previous version had a sub-package called `Examples` that was renamed in this new version to be `Tests`, because it consists of simulation models that are used for unit testing as the one used for software-to-software validation.

3. Power flow to records

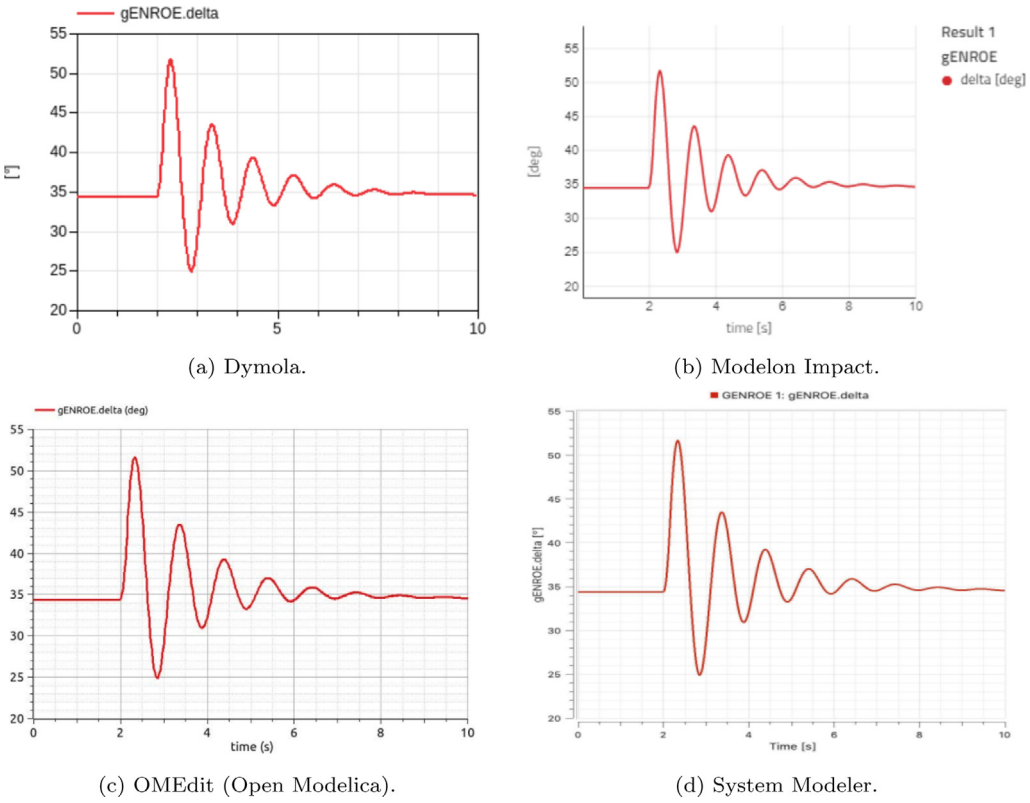
Since its first version, the OpenIPSL has focused on dynamic simulation and its models are initialized with results from power flow computations. The library, however, does not provide the means for power flow calculations to be performed natively within a Modelica-tool. As a matter of fact, this was done in previous versions of OpenIPSL by using tedious procedures involving calculations in third-party proprietary tools and manually inputting data into models for simulation to be carried out from a particular operation point. This, of course, was a great caveat of the library that needed to be addressed.

To overcome this issue, a new tool has been developed in Python [5]. The tool is able to carry power flow calculations using GridCal (), an open-source Python library. The results are automatically translated to Modelica records which are used to provide starting guess values used by Modelica-tools to initialize the model for numerical simulations to be carried out. The tool also provides useful documentation and tutorials that ease the tool's integration with the OpenIPSL for new and experienced users and is available publicly at https://github.com/ALSETLab/SMIB_Tutorial.

4. Conclusion

The OpenIPSL continues to pursue its main goal of developing a robust library for research and teaching of phasor-domain power systems models as it is shown in this new OpenIPSL release. Many changes have been made to allow a more efficient interaction with the user and developer's community. As a result, contributions to the library have been made a more transparent and simple process, establishing an appropriate framework for the continuous development of the library.

In addition, the complete adoption of S.I. and per-unit units and the usage of `types` have made individual component models in the library more consistent between each other. The different component models that have been revisited, together with models that have been recently added to the library, also contribute to a more robust version of the OpenIPSL. In fact, version 2.0.0 can be seen as a more stable basis from which the OpenIPSL development can be carried on without significant structural changes. Moreover, a complimentary Python tool, that sets up power flow solutions and automatically populates Modelica records to be used in model initialization has been developed. It greatly benefits OpenIPSL models by bridging a gap between Modelica-based tools that are used mainly for dynamic analysis with those that are used for steady-state analysis.



(a) Dymola. (b) Modelon Impact.

(c) OMEdit (Open Modelica). (d) System Modeler.

Fig. 1. Simulation results for the machine's rotor angle in an Single Machine Infinite Bus power system model simulated in different software tools using the plot representation of the respective tools.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Marcelo de Castro Fernandes reports financial support was provided by Dominion Energy Inc. Giuseppe Laera reports financial support was provided by Dominion Energy Inc.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work was supported in part by Dominion Energy through sponsored research projects at Rensselaer Polytechnic Institute, United States; in part by the New York State Energy Research and Development Authority (NYSERDA), United States through the Electric Power Transmission and Distribution (EPTD) PON 3770 High Performing Grid Program together with the New York Power Authority (NYPA); and in part by the ERC Program of the National Science Foundation, United States and DOE under NSF Award, United States Number EEC-1041877 and by the CURENT Industry Partnership Program, United States.

References

[1] Vanfretti L, Rabuzin T, Baudette M, Murad M. iTesla power systems library (iPSL): A modelica library for phasor time-domain simulations. SoftwareX 2016;5:84–8. http://dx.doi.org/10.1016/j.softx.2016.05.001.

[2] Baudette M, Castro M, Rabuzin T, Lavenius J, Bogodorova T, Vanfretti L. Openipsl: Open-instance power system library—update 1.5 to "iTesla power systems library (iPSL): A modelica library for phasor time-domain simulations". SoftwareX 2018;7:34–6.

[3] Casella F, Braun W. On the importance of scaling in equation-based modelling. In: Proceedings of the 8th international workshop on equation-based object-oriented modeling languages and tools. 2017, p. 3–7.

[4] Mukherjee B, de Castro Fernandes M, Vanfretti L. A PMU-based control scheme for islanded operation and re-synchronization of DER. Int J Electr Power Energy Syst 2021;133:107217.

[5] Dorado-Rojas SA, Laera G, de Castro Fernandes M, Bogodorova T, Vanfretti L. Power flow record structures to initialize openipsl phasor time-domain simulations with python. In: Sjölund M, Buffoni L, Pop A, Ochel L, editors. Proceedings of the 14th international modelica conference. Linköping electronic conference proceedings, (181):Modelica Association and Linköping University Electronic Press; 2021, p. 147–54. http://dx.doi.org/10.3384/ecp21181147.