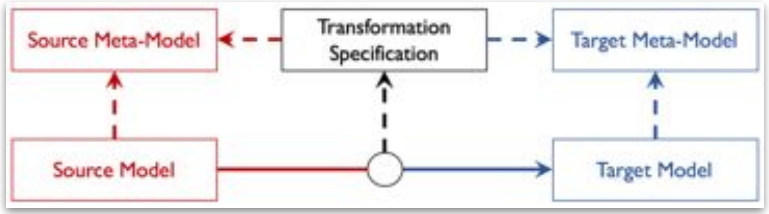
Rensselaer

ALSET lab why not change the world?®



Model Transformation for Smart Grid Applications

Marcelo de Castro[1†], Giuseppe Laera[1], Manuel Navarro[2], Luigi Vanfretti[1‡] and Glenn Halley[3].

[1]Rensselaer Polytechnic Institute, [2]Electric Reliability Council of Texas, [3]City Utilities of Springfield

E-mail: [†]decasm3@rpi.edu and [‡]luigi.vanfretti@gmail.com

Presentation Outline

- Introduction:
  - Motivation
  - Previous Efforts
  - Goals
- Methods:
  - Implementation in Modelica and Model Verification
  - Assessment of Models' Portability
  - Transformation Tool Design and Implementation
  - Assessment of Tool's Performance
- Final Considerations

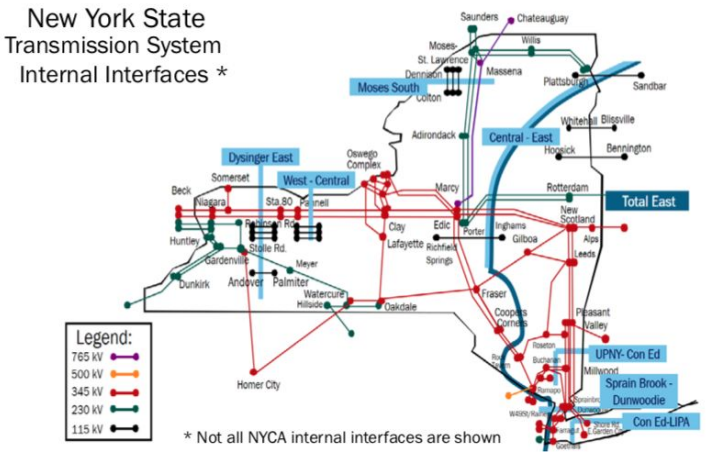
Motivation

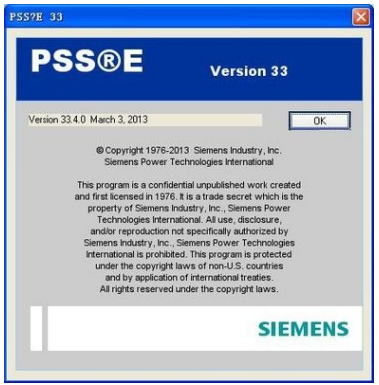
Future Energy Scenario

- Addition of large number of renewable energy sources in the grid
- New modeling and simulation capabilities and tools will be required
    - Clean technologies can be properly studied and models can be easily shared

Today's Scenario

- Lock-in: Few specific and proprietary software tools bound models to single tool.
- Lack of interoperability: Files retain parameter details but no information on equations
- Duplication and Inconsistency: Many different models are created to represent the same system, difficult to obtain the same results.





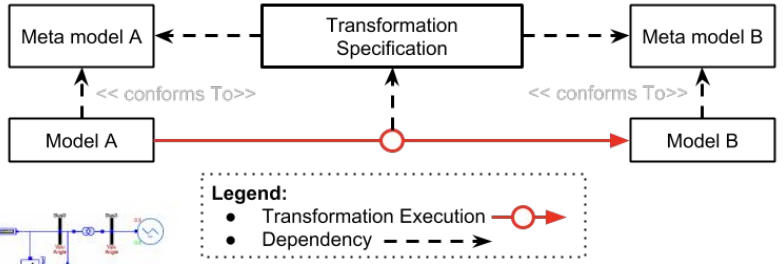
.RAW

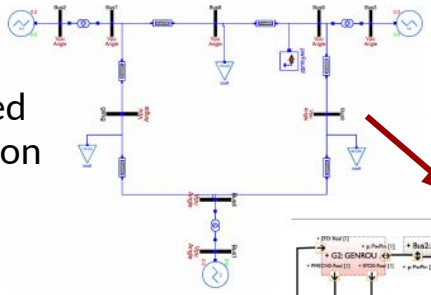
.DYR

Background

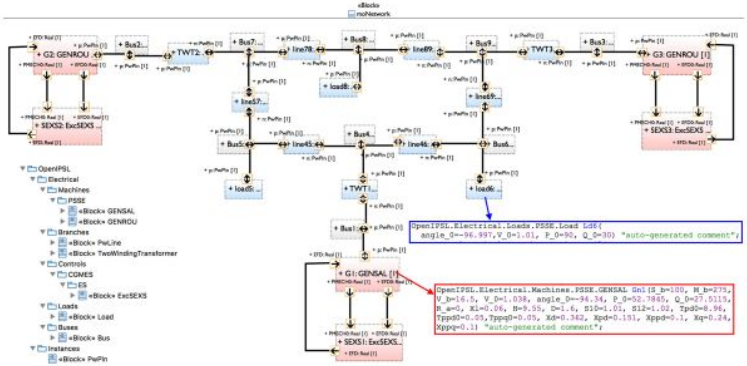
Model Transformation Tools

- Interpreter capable of transforming information and physical behavior of simulation models
- Computer program that:
  - Takes a model as input, also called source
  - Produces a model as output, or target
- Benefits of automatic translation:
  - No disparate number of models is needed
  - Automatic transformation avoids common implementation errors





SysML Representation and Modelica Model from M2M Transformation with CIM tools



Previous Works with Translation Tools

- Cim2Mod: Model-to-Model Transformation with CIM-compliant tool using Apache Jena and JABX solution for one-to-one translation
- CIM-verter: C++ template based approach for translating CIM models into Modelica

Proposed Solution and Presentation Scope

Our Solution: Open Source and Open Standards-based Interoperable Models

- Model Transformation Tool
  - Using PSSE and CIM as source models
  - Outputs Modelica (OpenIPSL) as target models
- Why Modelica?
  - We need model description to facilitate interoperability:
    - Strict mathematical representation: DAEs
  - Open access standard for object-oriented equation based modeling (language spec implemented by multiple tools).
- OpenIPSL Library:
  - Modelica-based power system dynamic models
  - In development for several years.
- FMI Standard Compliant: Supported by +100 tools
  - Models can be exported into many other different tools as FMUs

OpenIPSL

Scope

- *PSSE to Modelica:* template-based approach written in Python
- *CIM to Modelica:* XSLT-based approach for transforming XML files into Modelica

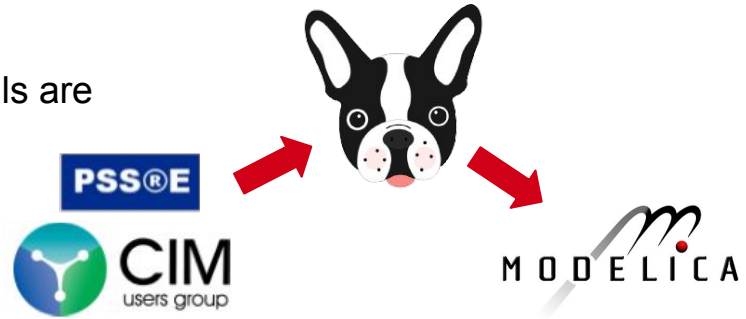
Implementation in Modelica and Model Verification

Main Challenges

- For a translation to work, it is necessary to be sure that models are consistent between tools
  - Mapping needs to be precise
  - Models need to be
    - Implemented in Modelica
    - Automatically tested
    - Verified against PSSE
  - Testing routine should be automatic:
    - Continuous Integration Framework

Overcoming Barriers

- Basic components (Machines, Exciters ...) are tested under different conditions in tiny models assembled using Modelica and OpenIPSL
- Regression testing to be implemented
- Continuous integration for library consistency

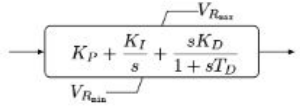


OpenIPSL is the means to represent PSSE and CIM models in Modelica

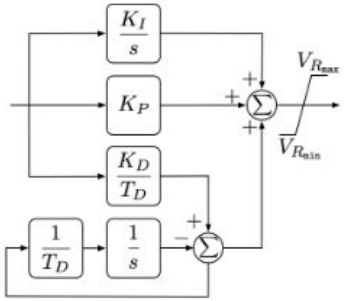
OpenIPSL must have the component being mapped and its behavior must be consistent with source

Implementation of Building Blocks

- Even simple blocks such as a limited output PID can have different implementations
  - When **only** block diagrams are used, the model can be ambiguous
  - Additional information is not always available or transparent
  - Different implementations can lead to different results

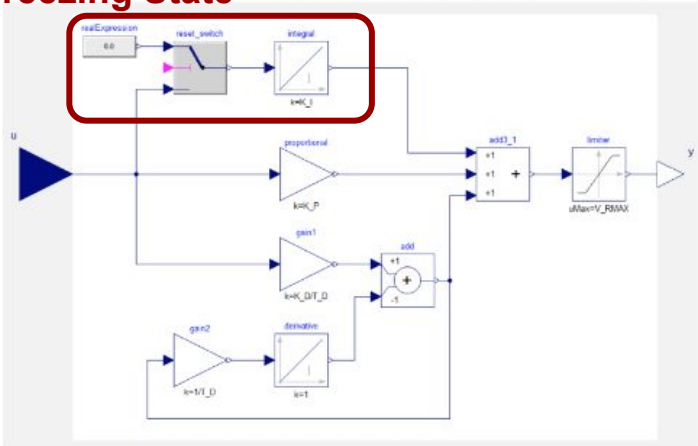


(a) Block diagram presenting an implementation for a limited output PID.

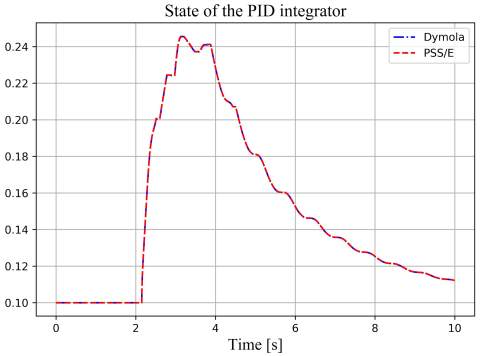


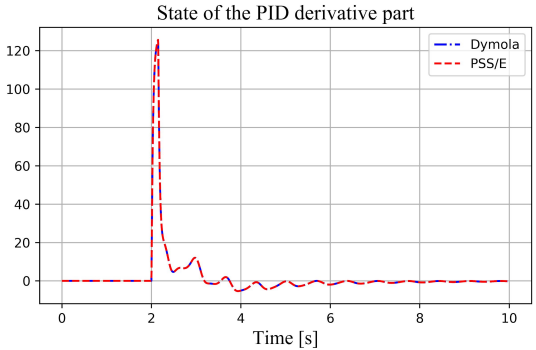
(b) Block diagram presenting an alternative and detailed implementation for the same limited output PID.

Freezing State



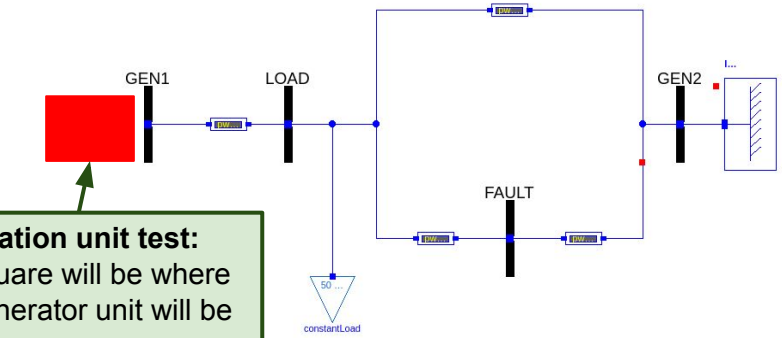
(c) Modelica implementation of the limited output PID block.





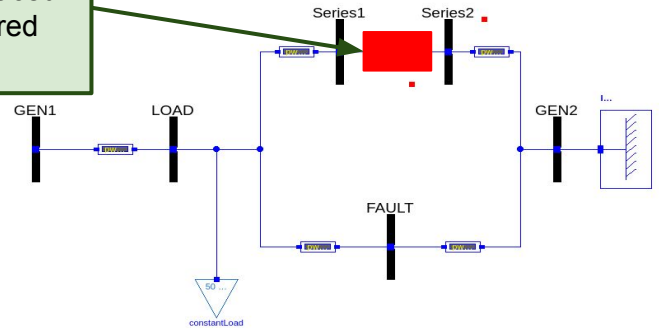
Templates: System Models for Verification of Basic Components

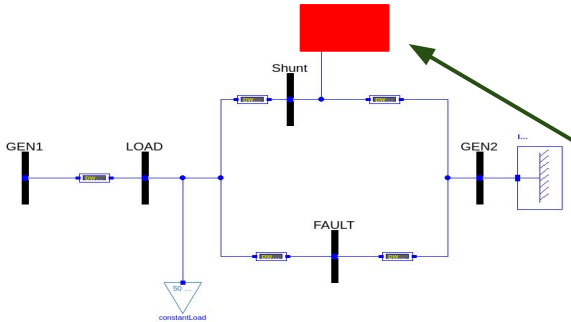
- A set of different small-scale systems was created for the testing and verification of basic components
  - Machines, Exciters, Stabilizers, Turbine Governors, etc.
  - Three different types of systems
  - Three different tests can be performed:
    - Fault (F)
    - Load Variation (LV)
    - Step in Reference value (SR)



Generation unit test: red square will be where the generator unit will be placed.

Series component test: Series component placed between two buses (red square).

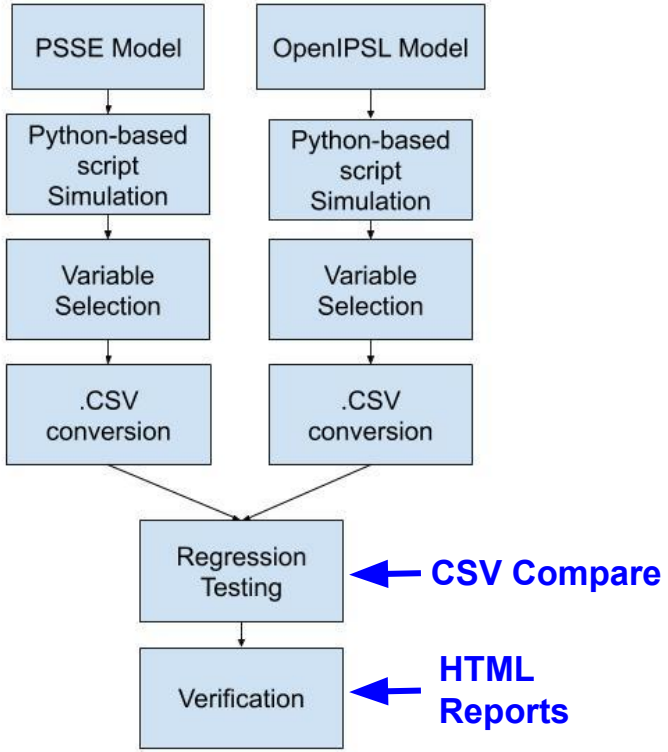




Shunt component test: Shunt component connected to a shunt bus (red square).

Regression Testing

- Manual verification of models is tiresome and can be subjective
  - How close the two curves should be so they are considered to be capturing the same behavior?
- Automatic procedures is needed for consistency:
  - Base results from PSSE are generated from Python scripts and saved into CSV
    - For each basic component
    - For each disturbance scenario
  - Results from OpenIPSL are obtained automatically via Python script and converted into CSV
  - CSV Compare tool is used
    - 1% Tolerance Tube



CSV Compare Report Example (1/2)

Meta report - CSV file comparison

Timestamp:	12/16/2019 3:25:40 PM [UTC]
Mode:	CsvTreeCompare
Base directory:	/home/marcelo/Dev/Gitted_Reps/NYPAModelTransformation/OpenIPSLVerification/VerficationRoutines/Dymola/Results/Generators/
Compare directory:	/home/marcelo/Dev/Gitted_Reps/NYPAModelTransformation/OpenIPSLVerification/VerficationRoutines/PSSE/Results/Generators/
Verbosity:	4
Tolerance:	1e-2
Compared files:	The compare directory contained 4 files. 4 files were tested. 1 file failed. Success rate is 75.0%.
Results	
	FAILED - At least one result failed its check with the base file. UNTESTED - No base file has been found or an exception occurred. SUCCEEDED - All results have been checked and are valid.
SUCCEEDED	GENROE.csv
SUCCEEDED	GENSAL.csv
SUCCEEDED	GENROU.csv
FAILED Ø0.24	GENSAE.csv

/home/marcelo/Dev/Gitted_Reps/Result/Generators/GENSAE_report.html

Meta report:	/home/marcelo/Dev/Gitted_Reps/Result/Generators/
Base file:	/home/marcelo/Dev/Gitted_Reps/NYPAModelTransformation/OpenIPSLVerification/VerficationRoutines/Dymola/Results/Generators/GENSAE.csv
Compare file:	/home/marcelo/Dev/Gitted_Reps/NYPAModelTransformation/OpenIPSLVerification/VerficationRoutines/PSSE/Results/Generators/GENSAE.csv
Tolerance:	0.01
Timestamp:	12/16/2019 3:25:40 PM [UTC]
Compared results:	The compare file contained 7 results. 7 results were tested. 1 result failed. Success rate is 85.7%.
Average relative error:	0.24
Failed tests:	GENSAE.gENSAE.delta

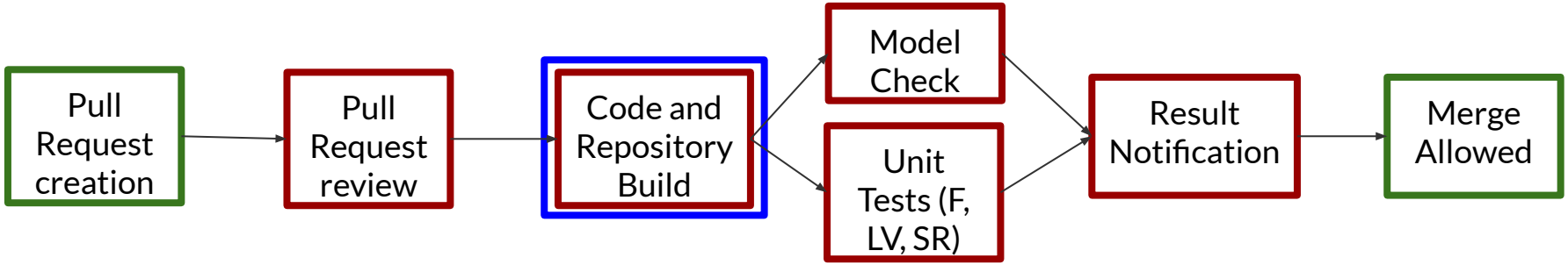
CSV Compare Report Example (2/2)

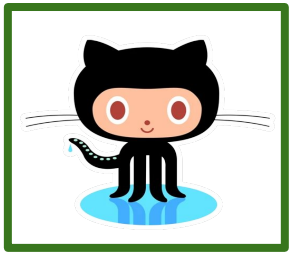
Offset between the reference and resulting signal!

The result signal is inside the ‘tolerance tube’ at all times.

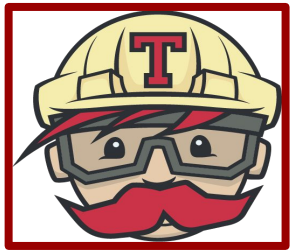
Continuous Integration

- OpenIPSL models are stored on a Github repository:
  - Need for an automatized procedure for code checking and model behavior verification
    - Continuous Integration - Software Engineering Solution

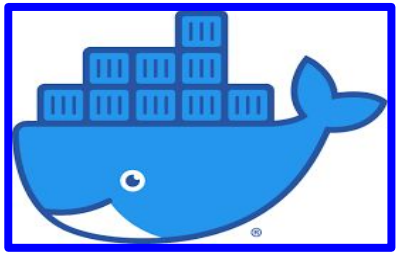




Github: repository where the code is located. Version control solution.



Travis CI: application to trigger and automate functions for the CI



Docker: container for our testing machine. Solution to the "it works on my machine" issue.

Assessment of Model's Portability

Main Idea

- Assess the notable features that explain why Modelica was chosen as target model in this project:
  - Object-oriented equation-based
    - Ideal for complete model description rather than just parameter tables
  - Use different Modelica-compliant tools:
    - Dymola, OMEdit, SystemModeler, Impact
  - Compliant with Functional Mock-up Interface standard
    - Models can be exported to different tools!

Methods

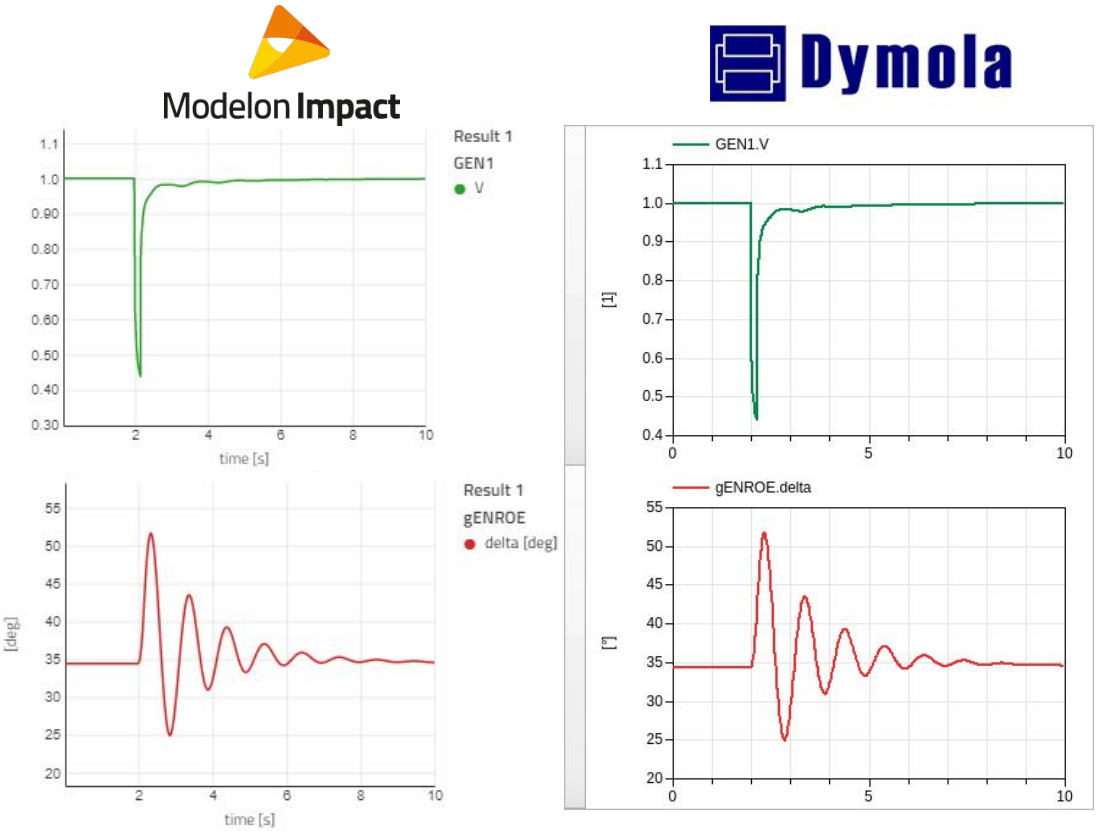
- Assess OpenIPSL in different modelica compliant tools
- Assess the simulation of Modelica models exported as FMUs in different tools

Models in OpenIPSL must work with maximum number of Modelica - compliant tools for interoperability

+

OpenIPSL models that are exported with FMUs must work in different simulation tools

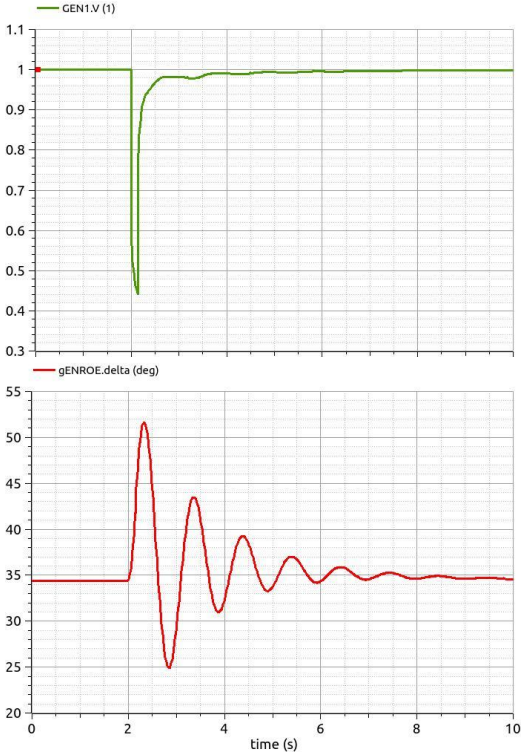
Interoperability Between Modelica Tools (1/2)



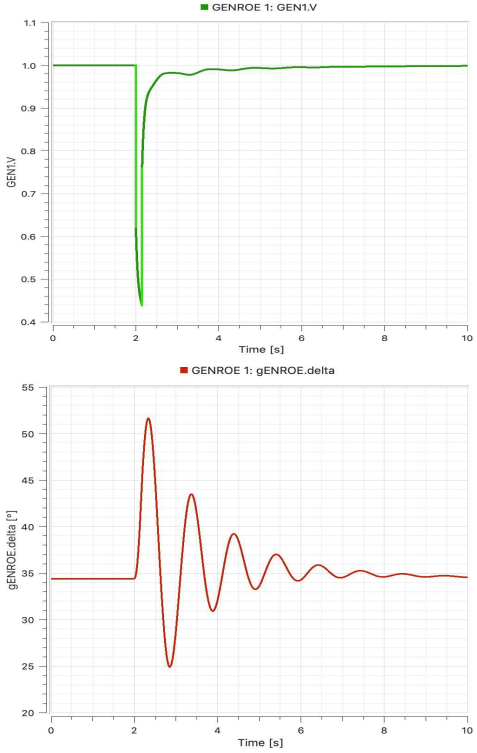
- OpenIPSL Library must maximize its compatibility with different Modelica-compliant software tools
- Library must be independent from tool
- Allows user to select their preferred tool to perform a study

Interoperability Between Modelica Tools (2/2)

OMEdit (OpenModelica)



Wolfram SystemModeler

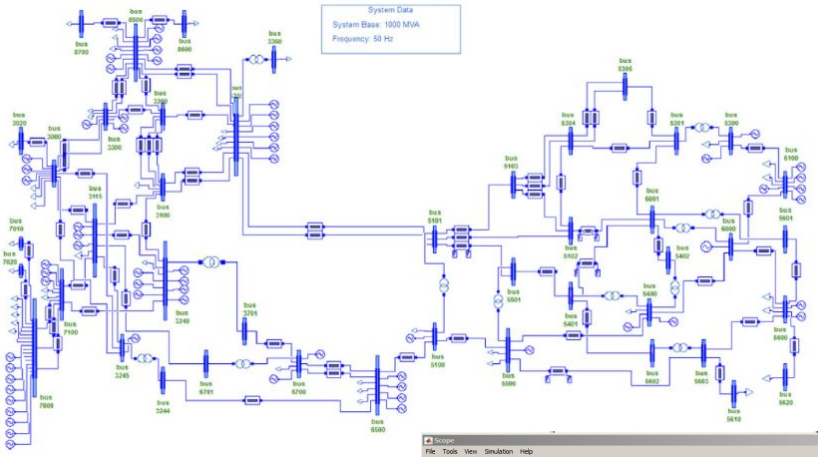


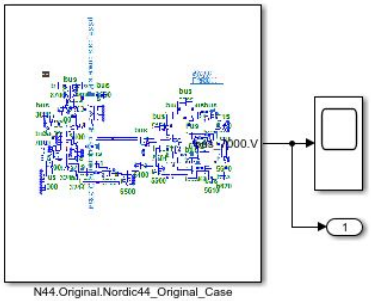
- OpenIPSL Library must maximize its compatibility with different Modelica-compliant software tools
- Library must be independent from tool
- Allows user to select their preferred tool to perform a study

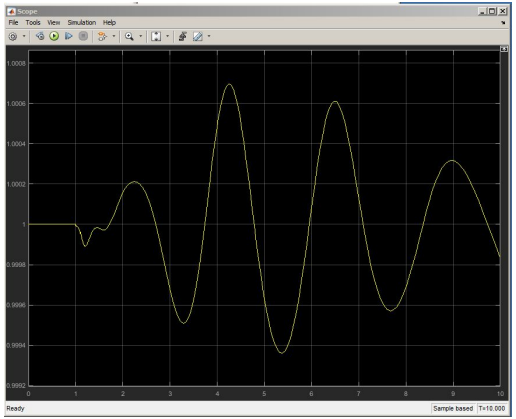
OpenIPSL and FMUs for Portability

Recomposing models for FMI-Export

- FMI export for different software tools:
  - Simulink
  - Python Libraries (FMPy PyFMU)
- FMUs for entire system:
  - Size is challenge



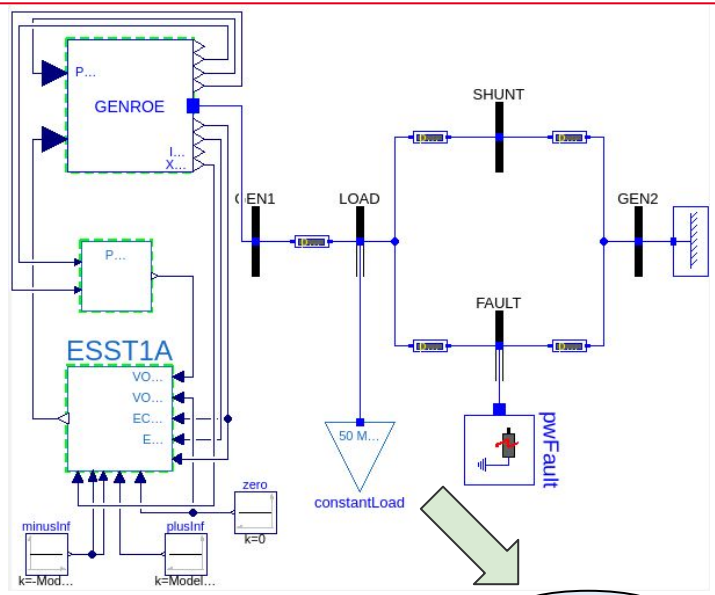




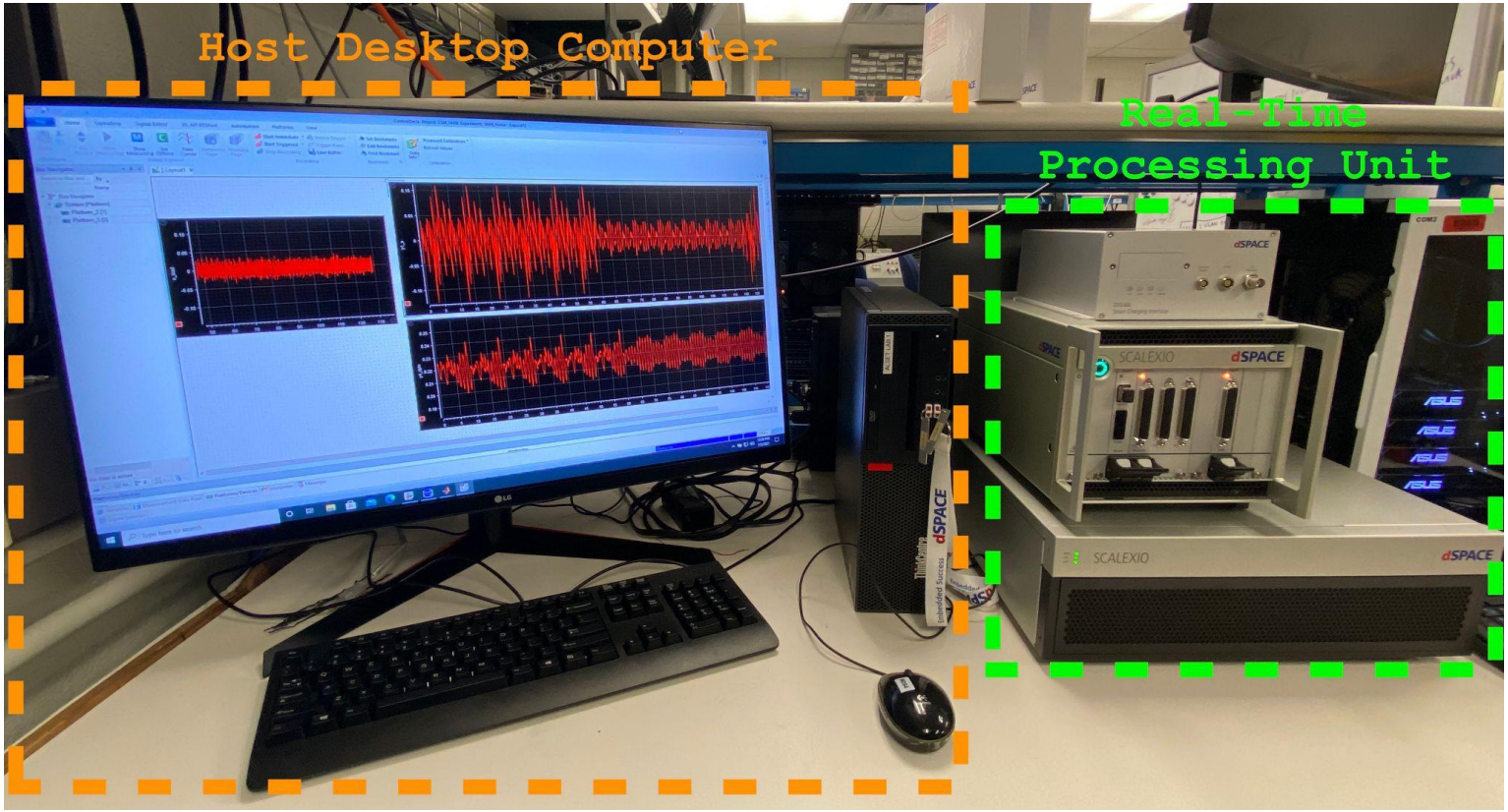
OpenIPSL and FMUs for Real Time Simulation

FMI-Export for Real Time Simulation

- Entire systems were used rather than different FMUs for each basic component
- Co-simulation FMU, with solver tolerance and simulation time already pre-defined
- Exported with source code via Dymola
- Loaded into dSPACE SCALEXIO:
  - Natively loads FMUs
  - No extra step is required



Real Time Simulation Setup



Real Time Simulation Result

Periodic Task 1/Overrun Count	Periodic Task 1/Task Call Counter	Periodic Task 1/Task Turnaround Time
13	30013	8.95870489417323E-05

Transformation Tool Design and Implementation

Main Idea

- Survey and analyze existing tools.
  - Existing Tools: Ditto, BIM2Modelica, CIM2Modelica
- Create the appropriate mappings from
  - PSSE to Modelica
  - CIM to Modelica
- Build Model Translation Tools
- Test and debug the tool with different systems
  - Single Machine Infinite Bus Systems
  - Components can be tested almost individually
    - Machines
    - Exciters
    - Governors
    - Stabilizers
    - Wind Machines
    - Compensators

OpenIPSL models can be exported and used in many tools

OpenIPSL models are shown to be consistent with PSSE

Need to design the basics of the translation tool with Modelica models as targets

Existing Tools - What's out there?

DiTTo (NREL)

- General idea: *many-to-one-to-many parser framework.*
  - Readers (inputs) and writers (outputs)
- Made for translating distribution systems and models from one data format to another.

BIM2Modelica

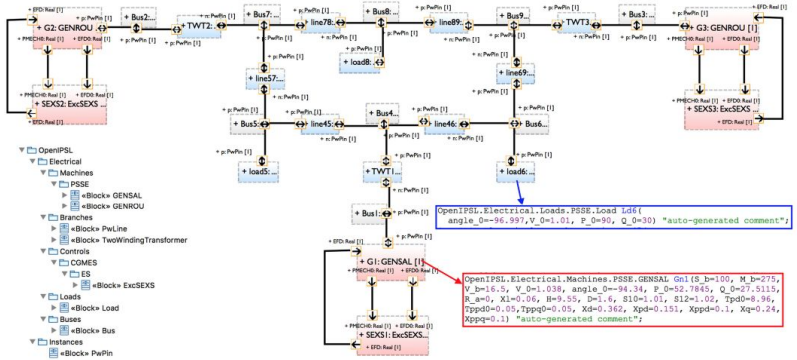
- Open source framework for generating and simulating thermal models by using data from BIM models.

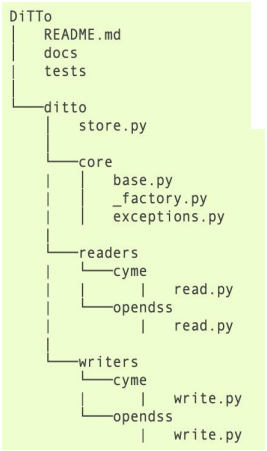
CIM2Modelica

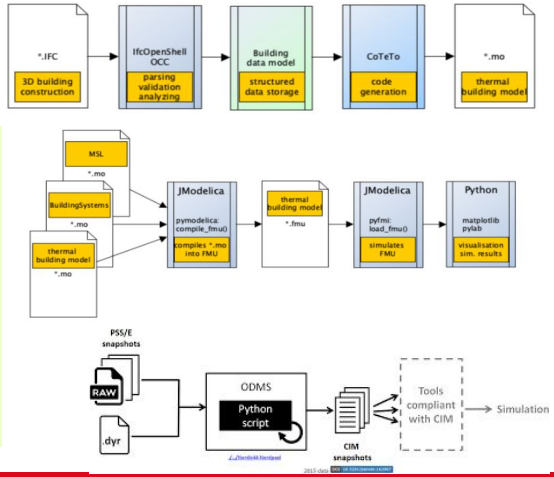
- Model-to-model transformation tool made for power systems
- XML schemas are used together with model-driven engineering concepts and paradigms

PSS@ODMS

- Proprietary tool with PSSE to CIM translation



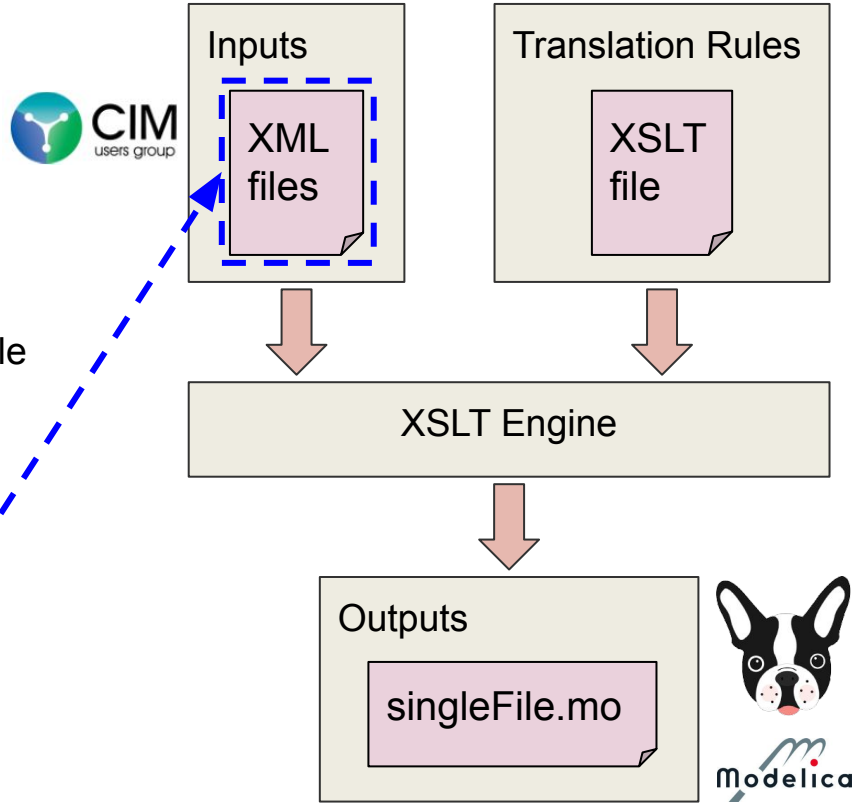




CIM to Modelica

Overview

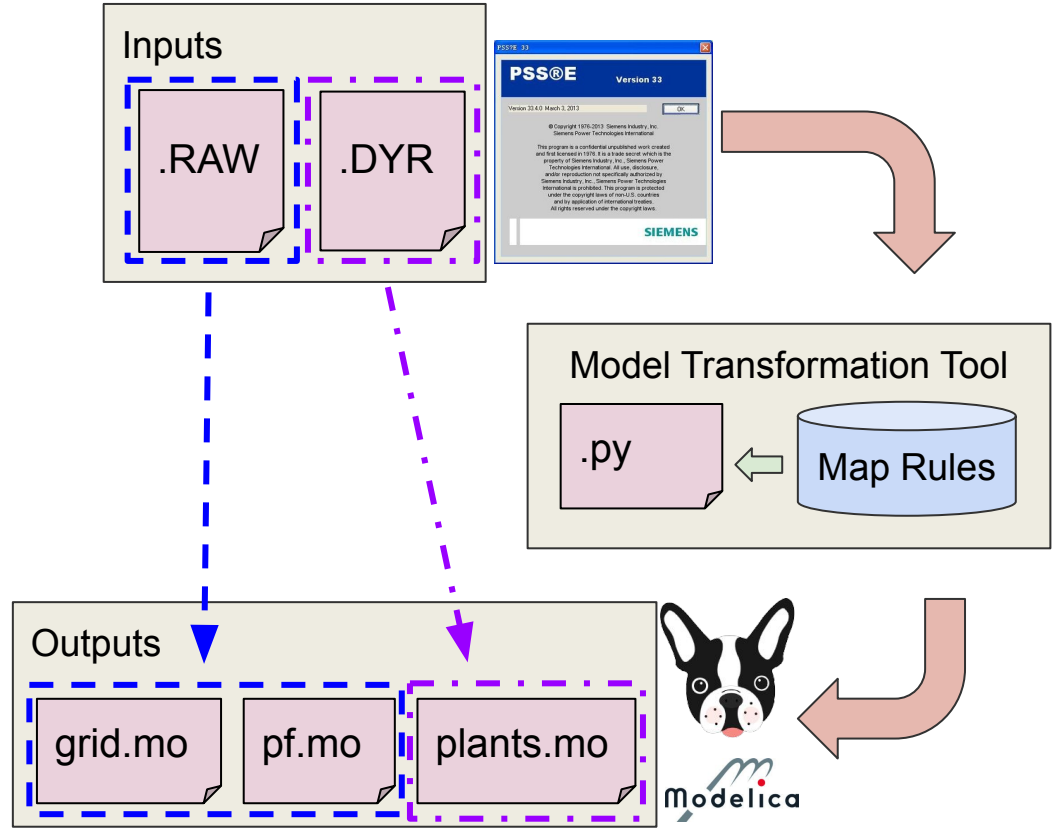
- Uses XSLT to translate CIM to Modelica
  - Proprietary tool - Windows Engine
- Common Information Model:
  - Readable (eXtensible Markup Language - XML).
  - Normalized - common data used in multiple types, linked using keys
  - Multiple CIM files including:
    - State Variables (SV)
    - Dynamic (DY)
    - Equipment (EQ)
    - Topology (TP)
    - Steady-state Hypothesis (SSH)



PSSE to Modelica

Overview

- Tool's general structure
  - Parsers, Readers and Writers
  - Based on templates
  - Written in Python
  - PSSE to Modelica
    - Raw files are translated into modelica network:
      - Lines, buses, shunts, loads
    - Dyr files are translated into dynamic models:
      - Machines, exciters, turbine-governors



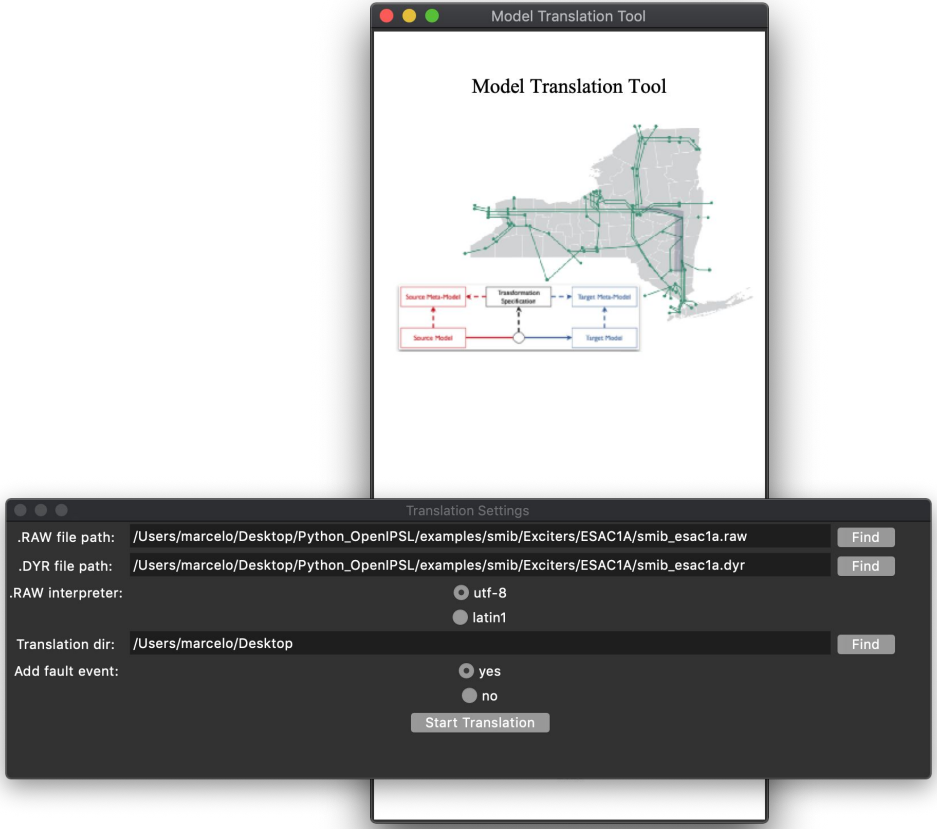
Assessment of Tool's Performance

Main Idea

- Test translations performed by tool
- Test tool's scalability with different systems:
  - Brazilian 7-bus System, IEEE14, IEEE23, Nordic 44, Icelandic System, REN System, NYPA 500

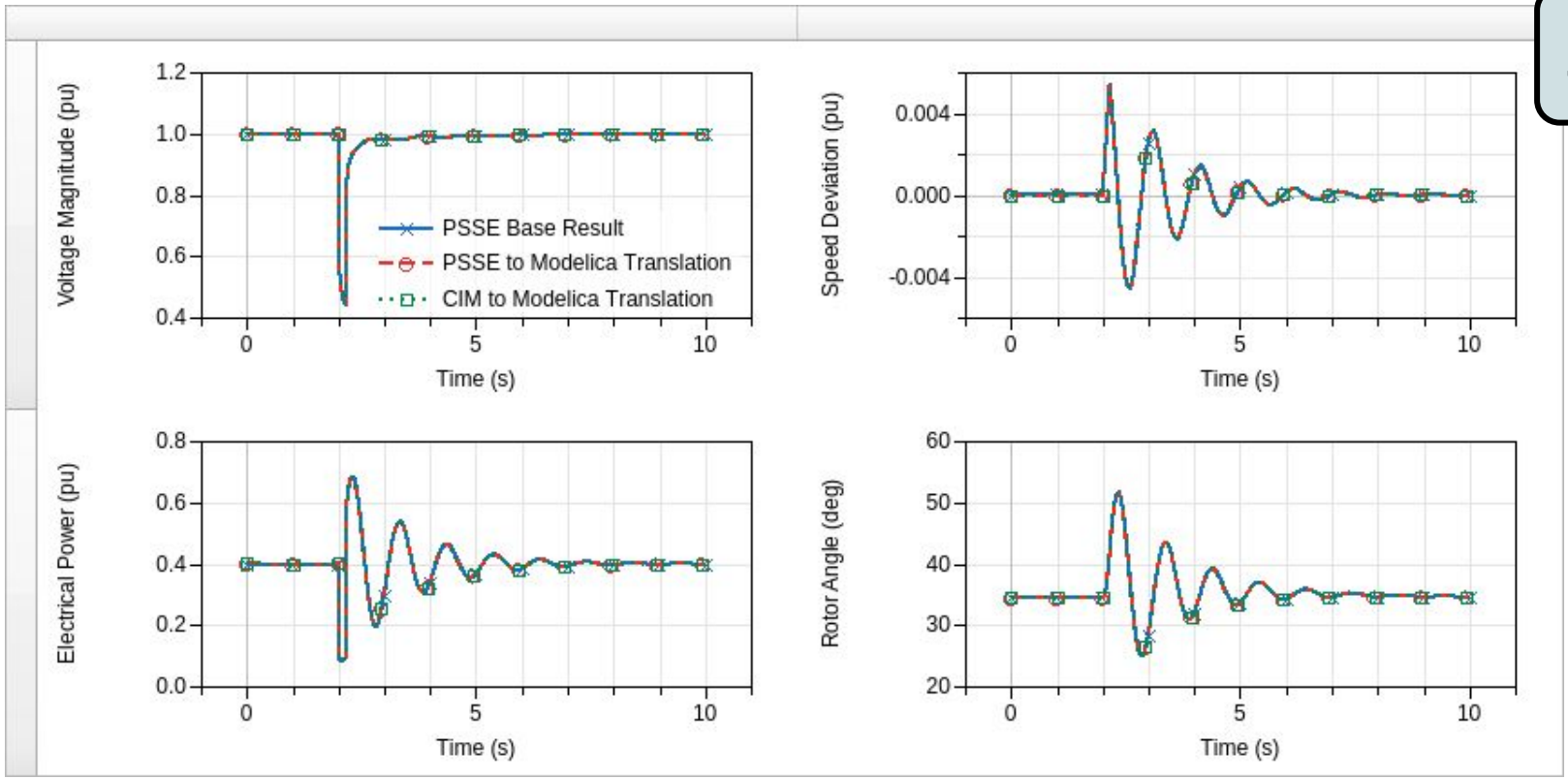
Methods

- Test system for many different testing:
  - Assessing results
    - Check if translated systems simulate and if their results are the same of translated versions
  - Timing translation for metrics (PSSE to Modelica)
    - Test tool in different machines, translating different systems.



Simulation of Translated Models Using MT Tools

IEEE 14-Bus



PSSE to Modelica Performance Test: Settings

Tests with different machines:

- **Goal is not to compare** the translation between different operating systems/machines
- **Goal is to assess** tool's **scalability** in different operating systems/machines

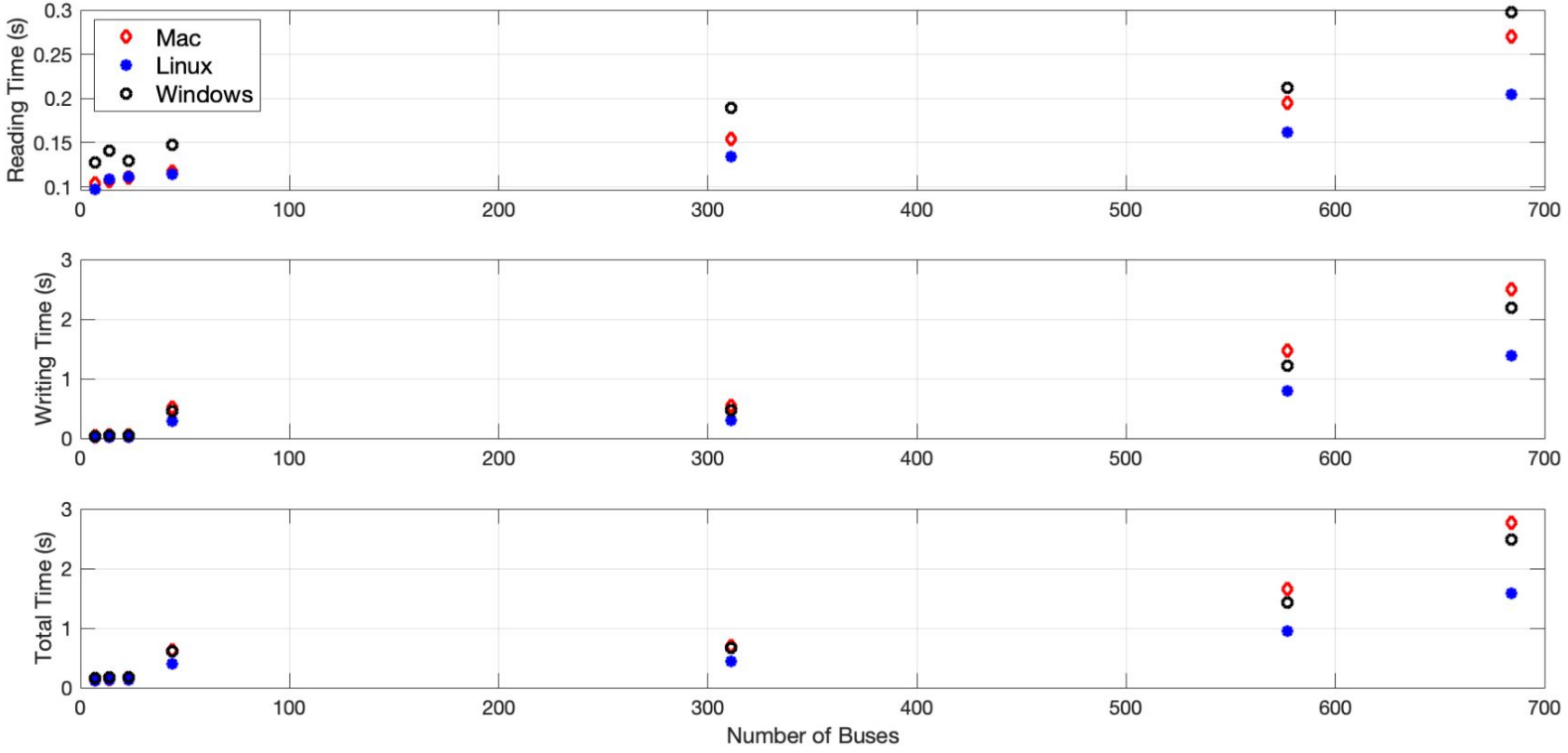
	Linux-based	**Mac**	**Windows**
Operating Software	Ubuntu 18.04.5 LTS	MacOS Mojave	Windows 10 20H2
Processor	i7 2.9 GHz x8	i7 2.8 GHz	AMD Ryzen 7 PRO 2700 3.20 GHz x8
Memory	16GB 1.6GHz DDR3	16GB 1.6GHz DDR3	64GB 1.6GHz DDR3

PSSE to Modelica Performance Test: Test Systems

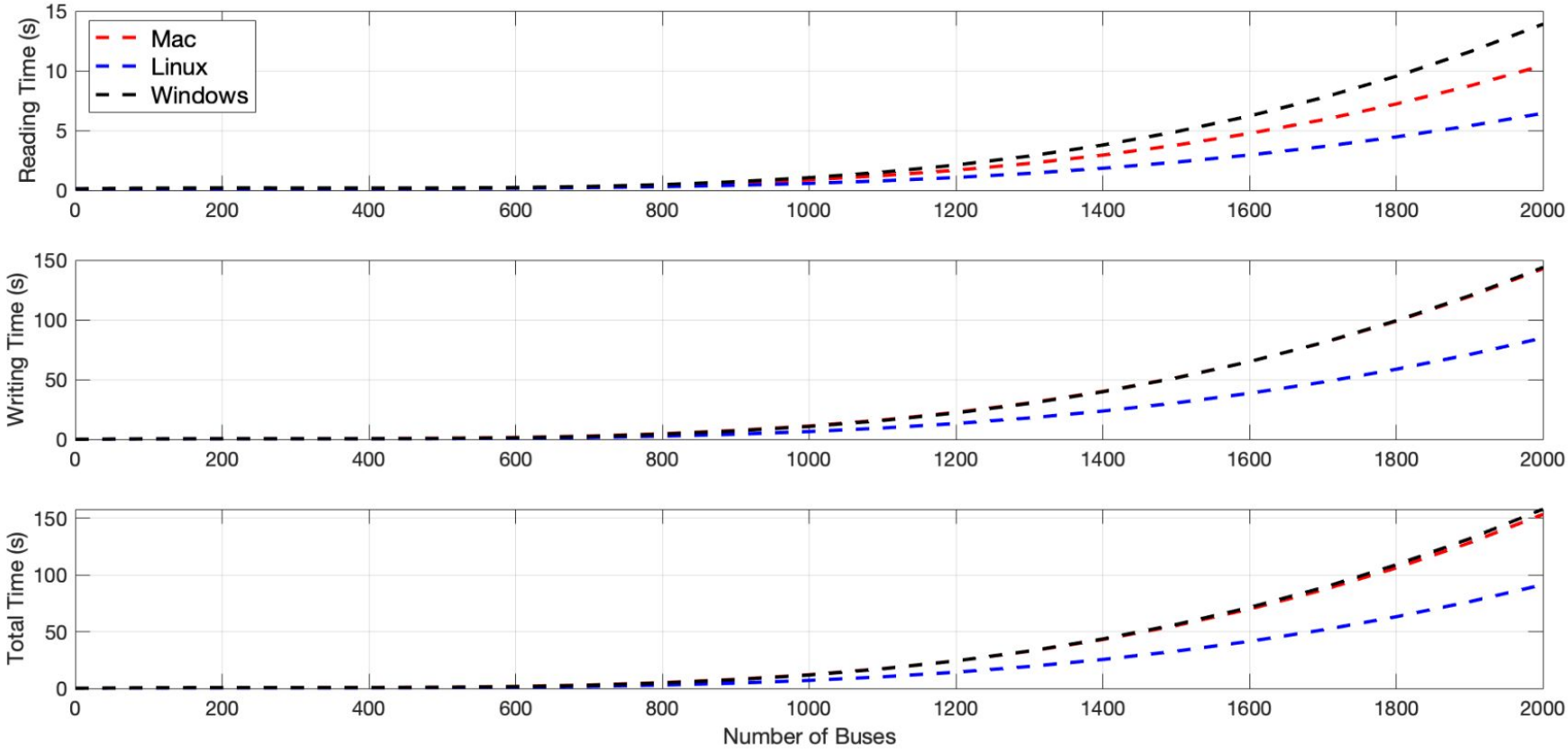
Different test systems

	Number of Buses	Number of Machines	Total Number of DAE
Brazilian 7-Bus	7	5	1032
IEEE 14	14	5	1346
IEEE 23	23	6	1619
Nordic 44	44	80	15976
Icelandic System	311	61	20342
REN System	577	138	34878
NY System	684	234	63094

PSSE to Modelica - Performance Results

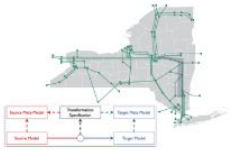


PSSE to Modelica - Performance Projection



Public Website

Model Transformation Tool - BabelGrid



Model Transformation for High Performing Grid Applications.

View the Project on GitHub
marcelofcastro/Python_OpenIPSL

This is the website for the 'Model Transformation for High Performing Smart Grid Applications' project. The project has been on going since Fall 2018. Joint project between New York Power Authority and Rensselaer Polytechnic Institute.

Project Overview

With the upcoming changes in the energy landscape of New York State, new modeling capabilities and simulation tools will be required in order for renewable and distributed energy technologies to be properly studied and incorporated into the electricity grid. In today's environment, grid modeling and operation in New York State are limited to few specific and proprietary software tools traditionally used for modeling and simulation over many decades. The ability of introducing new tools has been impractical, due to the way models have been developed and maintained. By using interoperable standards, and technologies, this project will create a new basis for developing and transforming grid models that could facilitate the utilization of multiple tools to advantage of the best features each tool has to offer in order to support the engineering and decision making process driven by the NYS Energy Plan and NY Reforming the Energy Vision (REV).

Main Resources

In this website you can find the following:

Pages	Links
Overview	About the Project
Model Validation	CSV Compare Results
Tool Description	About the Tool
Tool Guide	Using the Tool
About us	Contact Info

Website:
https://marcelofcastro.github.io/Python_OpenIPSL/index

- General information about the project is available there.

Dymola Reports

General results for Dymola tests can be found in the table below.

Model Type	General Reports
1. Machines	(a)Fault Report (b)Load Variation Report
2. Exciters	(a)Fault Report (b)Load Variation Report (c)Reference Step Report
3. Turbine Governors	(a)Fault Report (b)Load Variation Report
4. Power System Stabilizers	(a)Fault Report (b)Load Variation Report
5. Wind Machines	(a)Fault Report

All validation reports are available here

Conclusion and Final Remarks

- The presentation showed that:
  - A valid mapping between CIM/PSSE and Modelica is possible, allowing models from one tool (language) to be exported to another tool without loss of information and consistency of simulation results.
  - The library of Modelica models developed in this project is compatible with many different Modelica-compliant tools and models can be exported via FMI standard
    - Modelica model is a good target model (open access specification supported by multiple tools natively)
    - Target Model can be exported to various tools using FMI (+100!) for different applications
  - Both translation strategies are shown to be:
    - Effective
      - Results are consistent between translations and across different platforms
    - Scalable for larger models and more complex grids:
      - 2000 buses is expected to take a couple minutes
      - Room for improvement!

Acknowledgements

New York Power Authority Contributors

George Stefopoulos

Rahul Kadavil

Behnam Khaki

External Contributors

Dietmar Winkler
U. of South-Eastern Norway

Svein Harald Olsen
Statnett SF, Norway

THANK YOU

Rensselaer

why not change the world?®

Q/A