CURENT

CENTER FOR ULTRA-WIDE-AREA RESILIENT ELECTRIC ENERGY TRANSMISSION NETWORKS

CURENT Seminar Series

The University of Tennessee, Knoxville, August 22nd 2017

Modeling and Simulation of Electrical Power Systems using OpenIPSL.org and GridDyn

OpenIPSL

GRIDDYN

Prof. Luigi Vanfretti

Rensselaer Polytechnic Institute, ECSE, Troy, NY

Web: ALSETLab.com

Email: vanfrl@rpi.edu - luigi.vanfretti@gmail.com

Rensselaer

Dr. Phillip Top

Lawrence Livermore National Lab

Web: https://software.llnl.gov

Email: top1@llnl.gov

Lawrence Livermore National Laboratory

Agenda

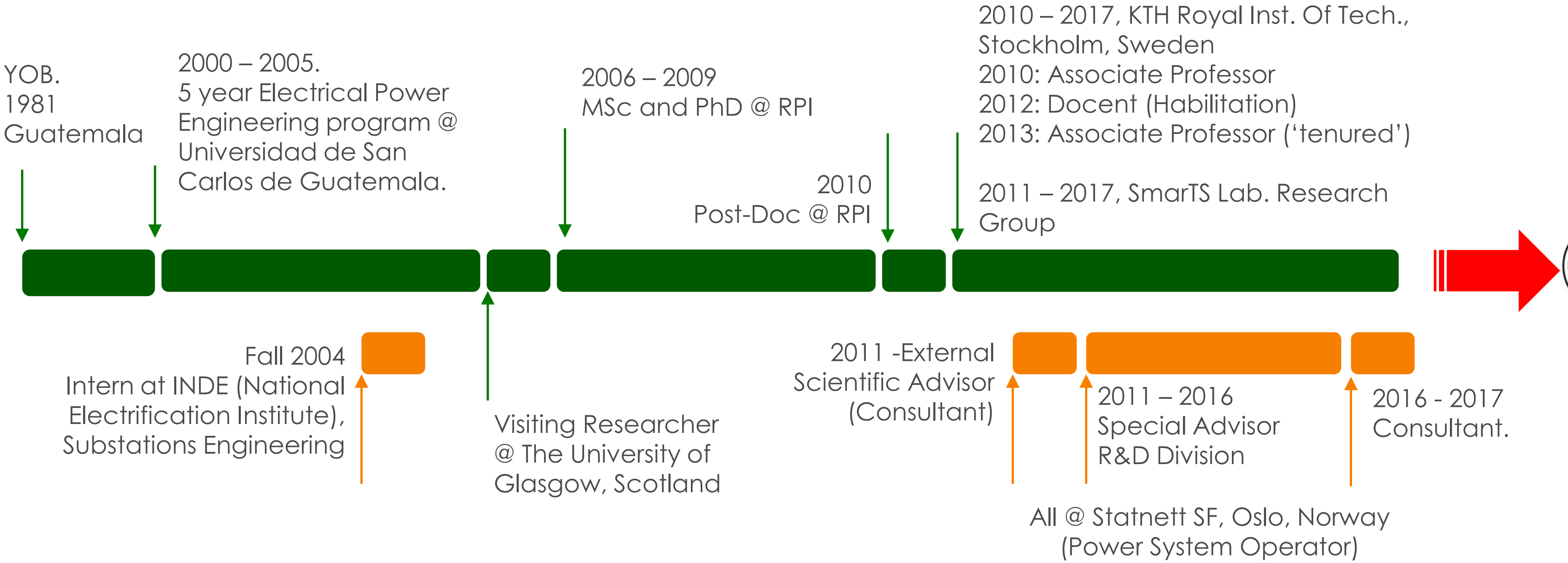
- About Me (Luigi)
- My Research.
- Recruiting for: ALSETLab

- **Part 1: (Luigi Vanfretti)**
  - Introduction:
    - Modeling and Simulation Generalities and Modelica
  - The OpenIPSL
  - Recent developments

- **Part 2: (Phillip Top)**
  - Applications of the OpenIPSL library and the FMI in
    - GRIDDYN

- Latter today: Tutorial: ***(Me + You!)***
  - Hands-on-Tutorial:
    - Overview of the OpenIPSL library
    - Overview of the OpenModelica environment
    - Hans-on-Example
    - Do the "preparatory work" so that everything is ready to go in your computer!

About Me - http://ALSETLab.com - Dr. Luigi!



Rensselaer

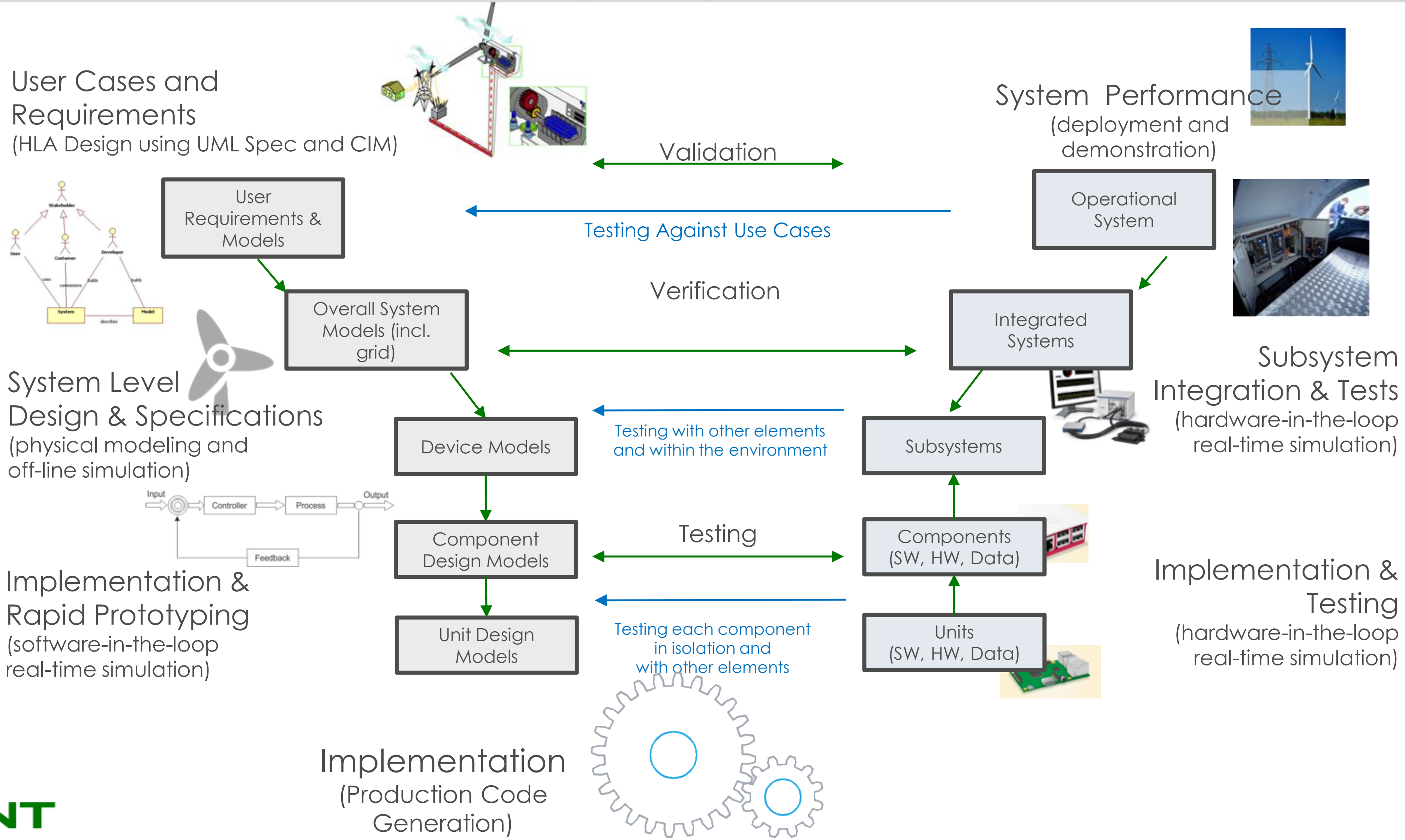
Other facts and numbers:

- Guatemalan and Italian Citizenships.
- Speak/write Spanish (native), English, Italian (spoken, poorly written), Norwegian (Basic)
- 36 years, married (March 4th, 2017) - no kids yet… but really want a dog!
- Close family, brother and wife, live in Woodstock, NY; run Dolce Caffe in Kingston's Historical Roundout
- *Lived in 4 countries, worked in 5…*

CURENT

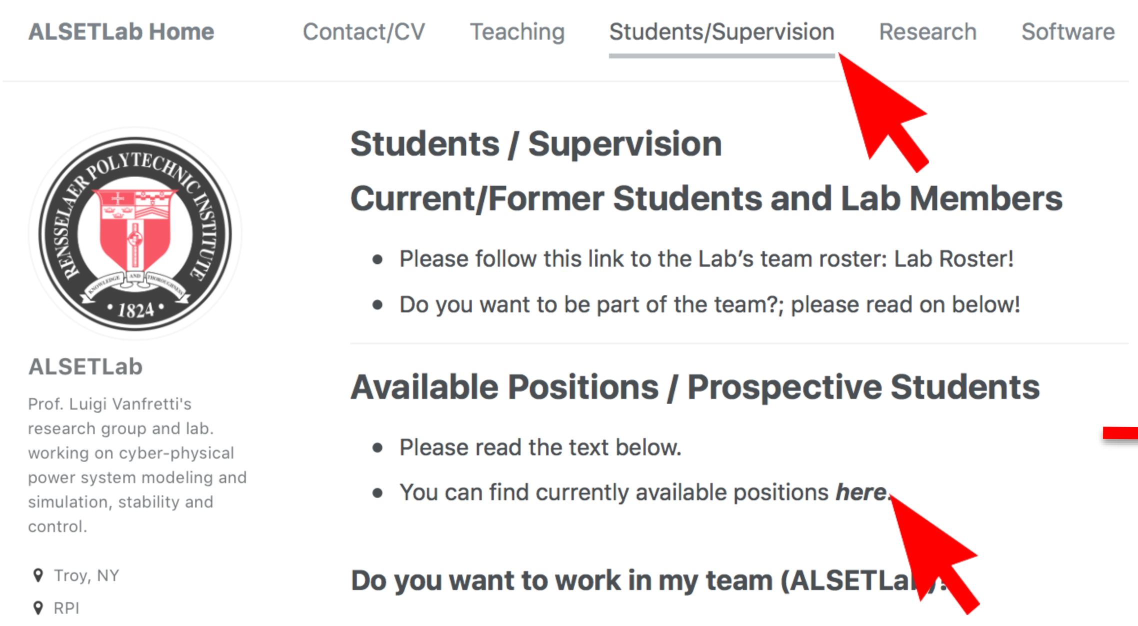
My Research – Cyber-Physical Power Systems

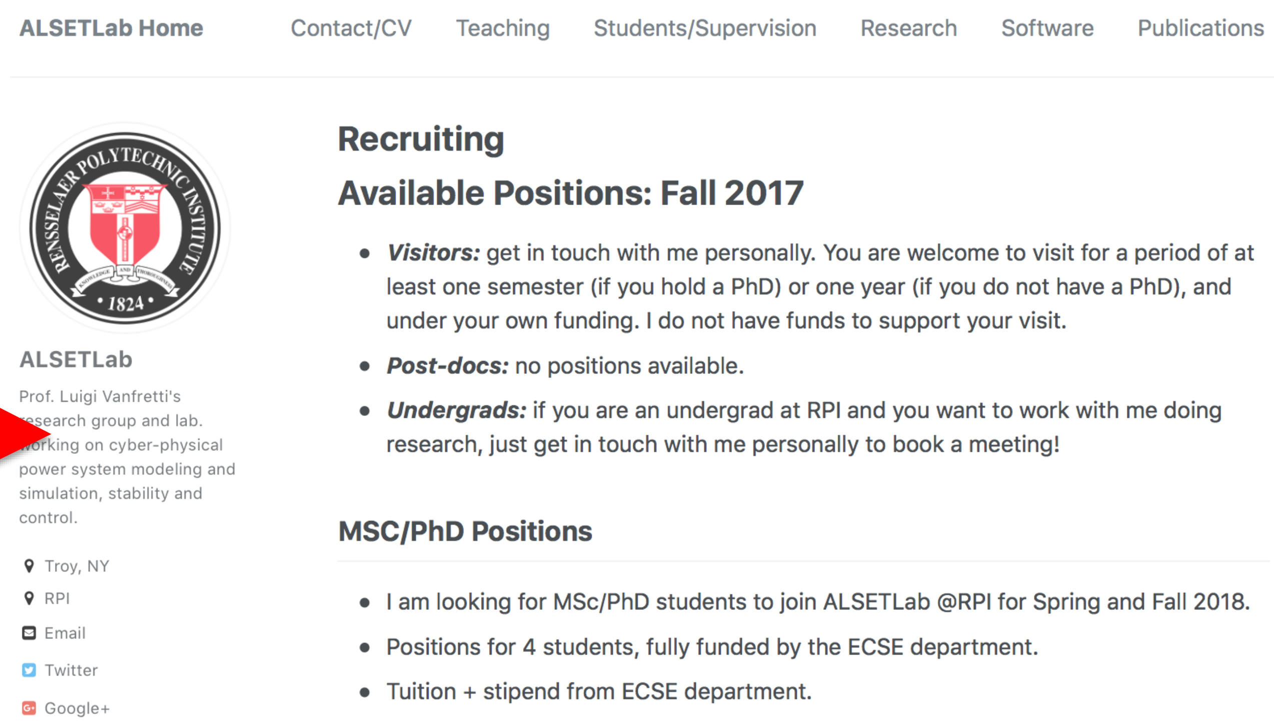
Model-and-Measurement-Based Systems Engineering of Power System and Synchrophasor Technologies



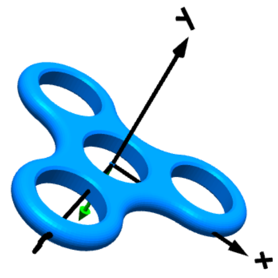
Recruiting @ ALSETLab

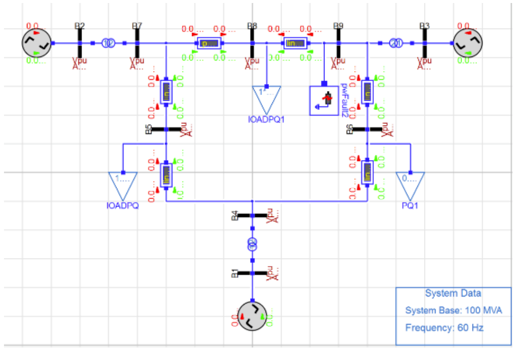
- I'm looking for graduate students to join my team!
- If you know someone that would be interested, please tell them to check my website
- See: http://ALSETLab.com

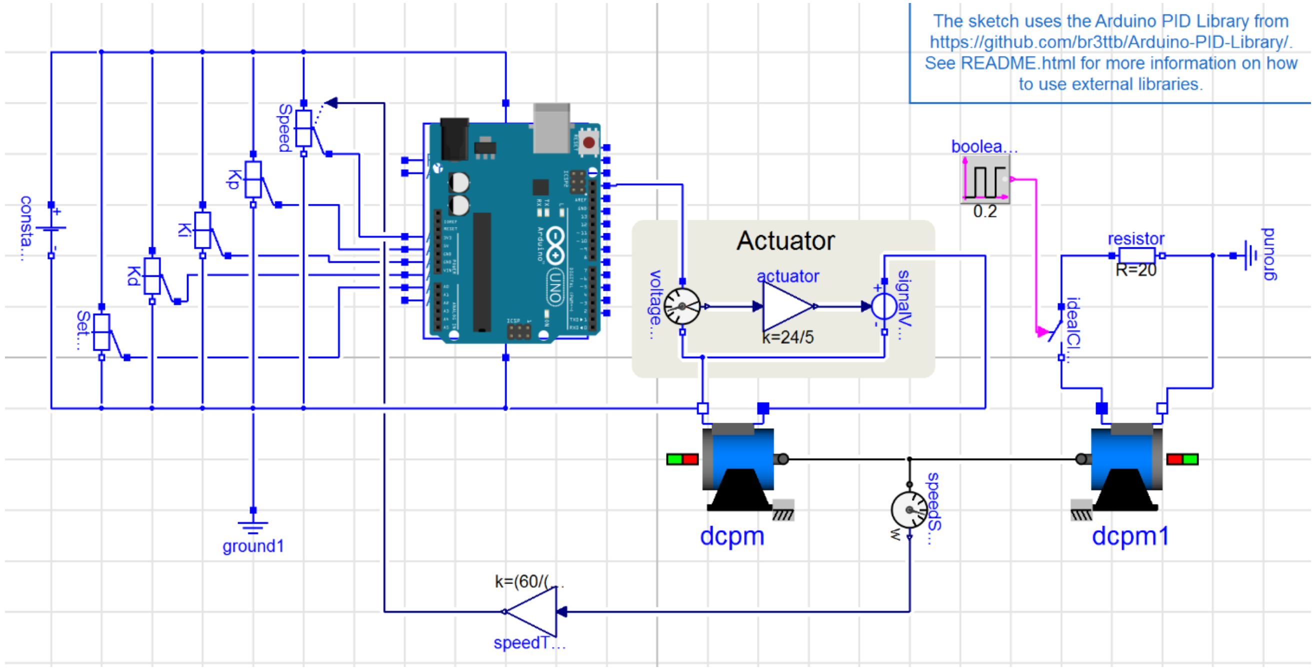




New Course! Spring 2018 - CPS Modeling, Simulation and Analysis



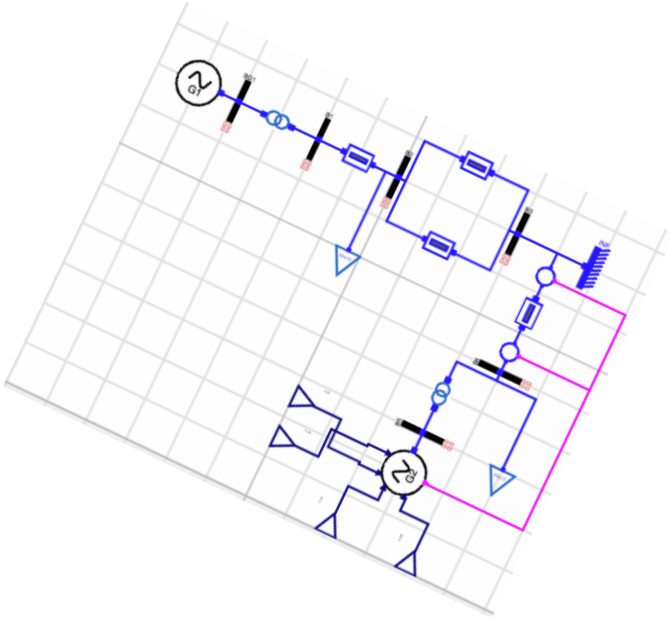




- Understand cyber-physical systems and how to model them.
- Lean about **standardized modeling languages and compliant tools**
- Learn and become proficient with the Modelica language
- Learn and apply model-based systems engineering concepts and tools (UML, SysML using Papyrus RT)
- Apply identification, control and optimization techniques to CPS systems
- Apply its use for analysis of:
  - Power systems
  - Energy efficient building automation
  - Multi-domain energy systems
  - Cyber-physical systems design and analysis

CENTER FOR ULTRA-WIDE-AREA RESILIENT ELECTRIC ENERGY TRANSMISSION NETWORKS

Modeling and Simulation of Electrical Power Systems using *OpenIPSL.org* and *OpenModelica.org*



Part 1: Introduction

Prof. Luigi Vanfretti
Rensselaer Polytechnic Institute, ECSE, Troy, NY
Web: ALSETLab.com
Email: vanfrl@rpi.edu luigi.vanfretti@gmail.com

Rensselaer

Outline

- The role of models and simulation
  - Generalities
  - In power electrical systems
- Modelica and power systems
- The OpenIPSL Project
- The OpenIPSL Library
- Continuous Integration
- On-going developments

A Fundamental Question: *Why do we develop models and perform simulations?*

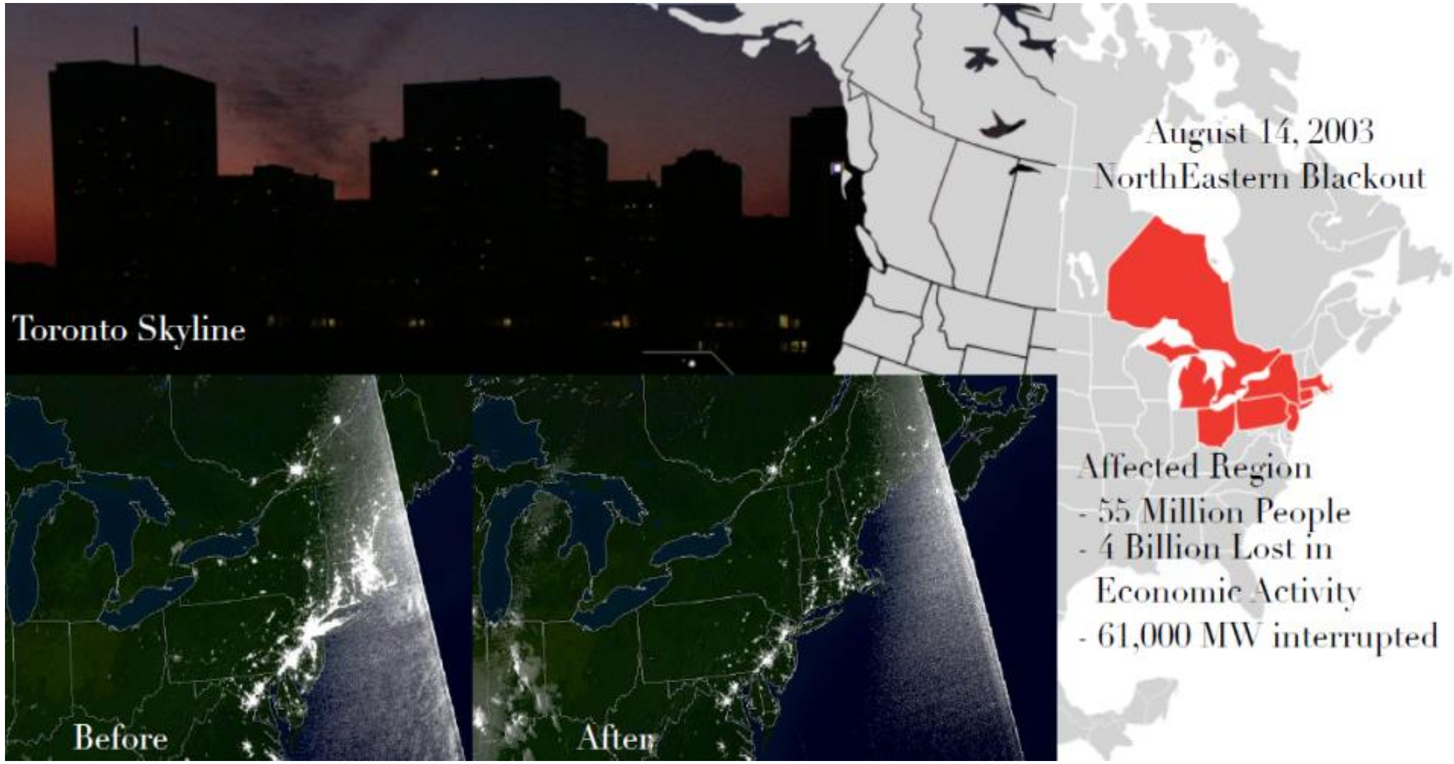
- The prospective pilot sat in the top section of this device and was required to line up a reference bar with the horizon. 1910.
- More than half the pilots who died in WW1 were killed in training.

- To reduce the lifetime cost of a system
  - ***In requirements:*** trade-off studies
  - ***In test and design:*** fewer proto-types
  - ***In training:*** avoid accidents
  - ***In operation:* anticipate problems!**
    - Crucial for electrical power systems!

CURENT

A *Failure* to Anticipate → Huge Costs!

There are many examples of failures to anticipate problems in power system operation!



Failure!

Existing modeling and simulation (and associated) tools were unable to predict this (and other) events.

Others: WECC 1996 Break-up, European Blackout (4-Nov.-2006), London (28-Aug-2003), Italy (28-Sep.-2003), Denmark/Sweden (23-Sep.-2003)

CURENT

The Multiple Roles of Modeling and Simulation in building:
Complex Cyber-Physical "Systems-of-Systems"

Simulating Success

How do modeling and simulation activities, capabilities benefit Boeing? Let us count the ways—9 of them

By Debby Arkell

Hands-on experience often can be the best way to tackle complex problems or master challenging skills. But when it comes to navigating intricate, variable-laden scenarios, or combat situations involving complex military maneuvers using expensive equipment, "on-the-job training" often is not a prudent approach.

That's why Boeing Integrated Defense Systems, Commercial Airplanes and Phantom Works engage in a wide variety of modeling and simulation activities, designed to provide ever more realistic simulations to internal customers across the enterprise—and to external customers as well.

"There is a tremendous amount of diversity in modeling and simulation being worked on at Boeing, encompassing very complex issues within a very broad spectrum," said Ron Fuchs, director of Modeling and Simulation for IDS. "Right now there are more than

U.S. Navy legacy platform and deliver a comprehensive and credible concept of operations.

Boeing analysts have a variety of tools available—or under development—that can demonstrate concepts and provide significant cost savings by exploring ideas, developing systems, testing and manufacturing within a virtual environment before committing to specific approaches.

4 Product or system testing

Models and simulations are used to test

M&S used to test prototypes in variety of environments.

networked computer systems in the FCS system of systems will be tested in a large-scale distributed simulation facility called the FCS System of Systems Integration Lab. The SoSIL provides a

Training systems and maintenance

M&S are used to *train users in the operational environment* – enhancing learning. Simulation *costs 1/10* of running actual scenarios.

Network communications

Tactical military communications networks—such as Joint

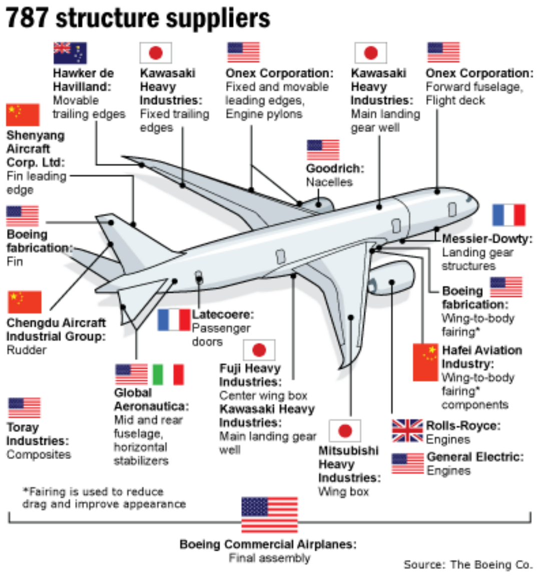
Scale of networks: cost-prohibitive or technically impossible for field tests.

M&S used to test and validate networking protocols in laboratory - environment acting as a test bed.

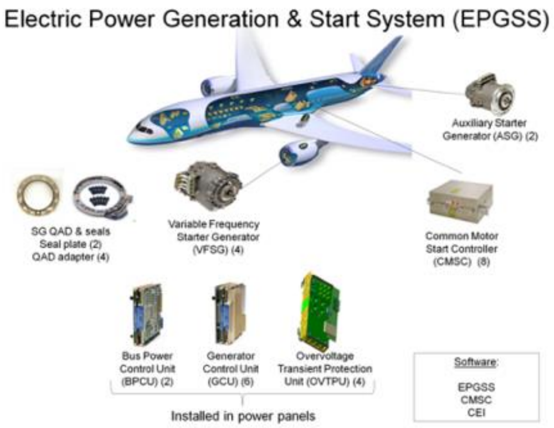
8

acts as a distributed virtual test bed, and the Boeing Transformational Communications Laboratory in El Segundo, Calif., performs a similar function for satellite communications.

Large Number of Vendors for the Final System



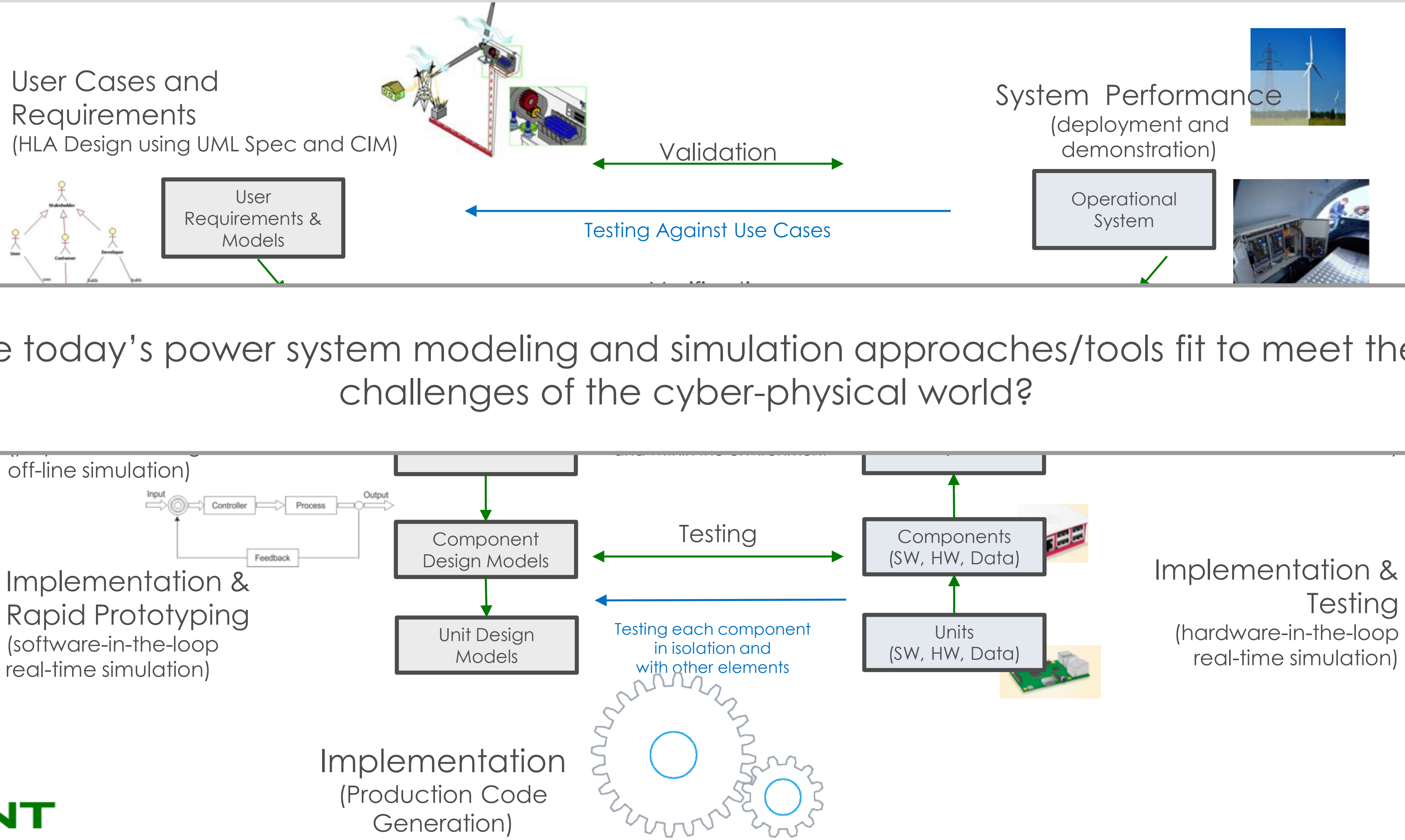
A Flying Micro-Grid!



CURENT

The Multiple Roles of Modeling and Simulation to develop
Cyber-Physical Power Systems (aka 'smart grids')

Conceptual Application for the Development of a WT Synchro phasor-Based Controller



Are today's power system modeling and simulation approaches/tools fit to meet the challenges of the cyber-physical world?

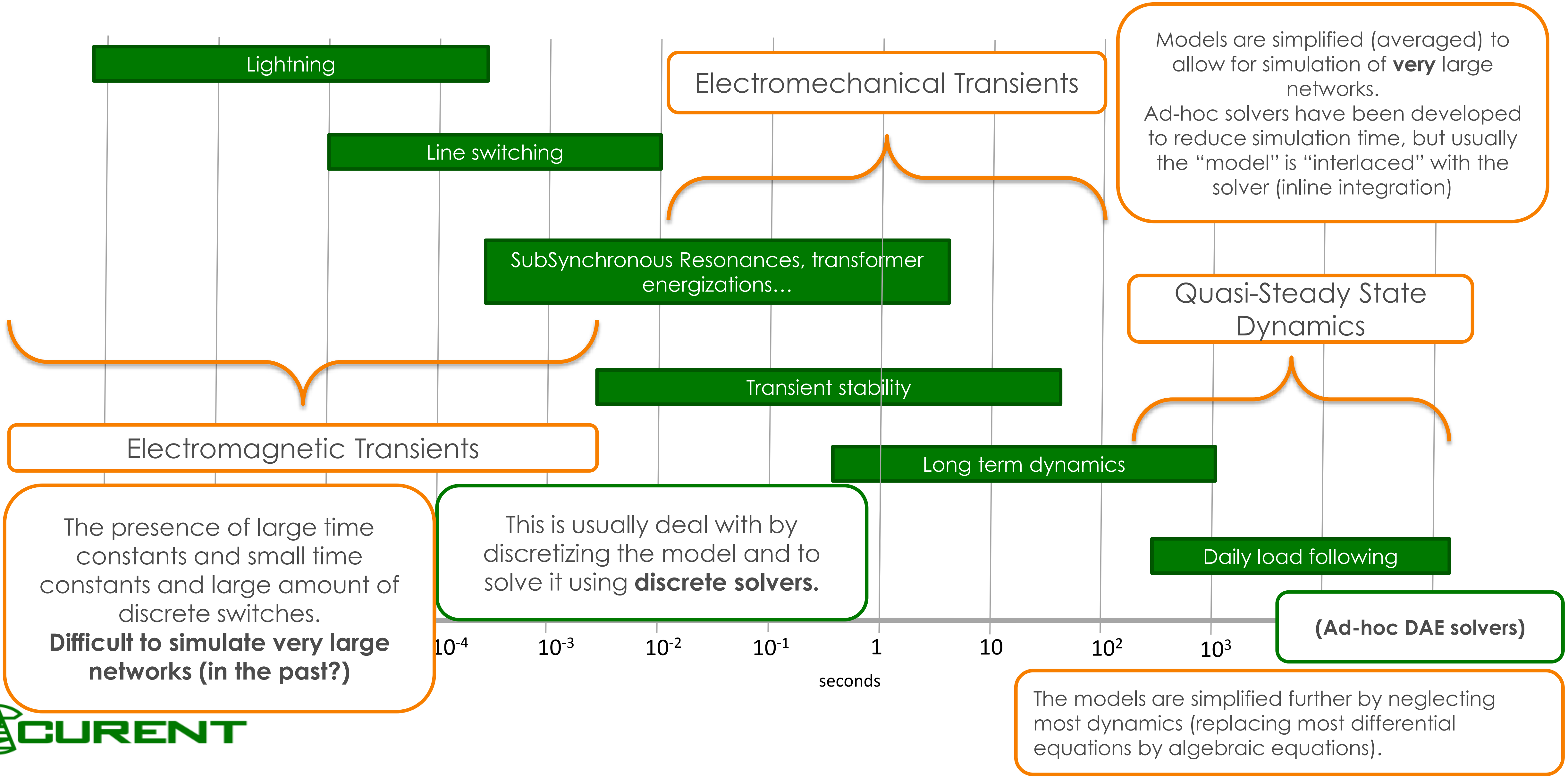
Dangers of Models and Simulation

- **Falling in love with a model**
  The Pygmalion effect (forgetting that model is not the real world)
  - From the Greek myth of Pygmalion, a sculptor who fell in love with a statue he had carved.
- **Forcing reality into the constraints of a model**
  The Procrustes effect (e.g. economic theories)
  - Procrustes: "the stretcher [who hammers out the metal]", a rogue smith from Attica that physically attacked people by cutting/stretching their legs, so as to force them to fit the size of an iron bed.
  - A **Procrustean bed** is an **arbitrary standard** to which exact conformity is forced.
- **Forgetting the model's level of accuracy**
  Simplifying assumptions forgotten more than yesterday's pudding…

CURENT

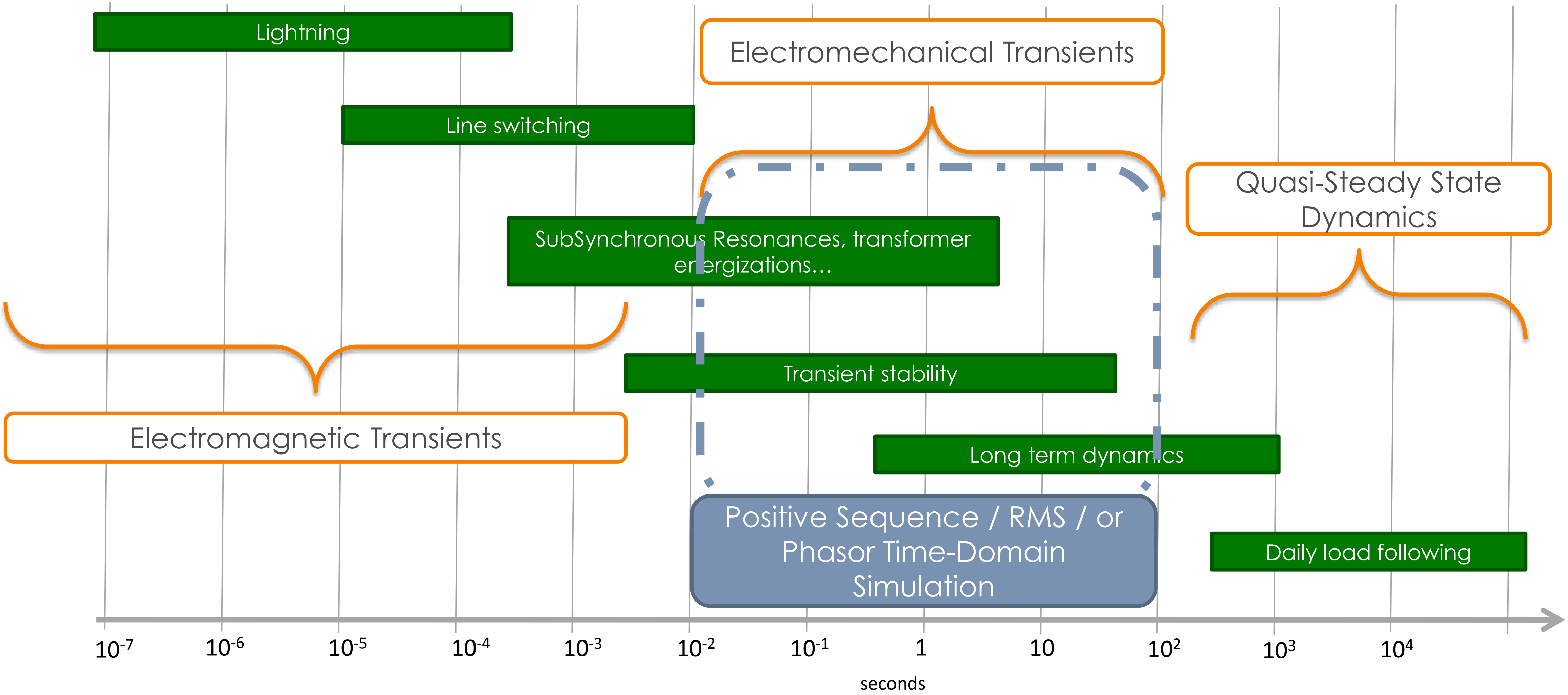
Power system dynamics challenges *for* simulation
the tyranny of *multiple time-scales*

Generally there are no discrete events.
(Ad-hoc DAE solvers)



Power System Phenomena Modeled and Discussed from this point on:

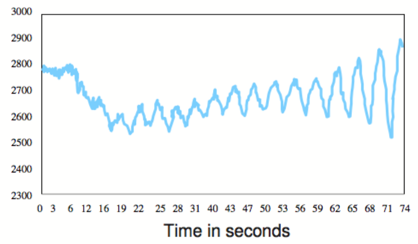
power system *electromechanical dynamics*

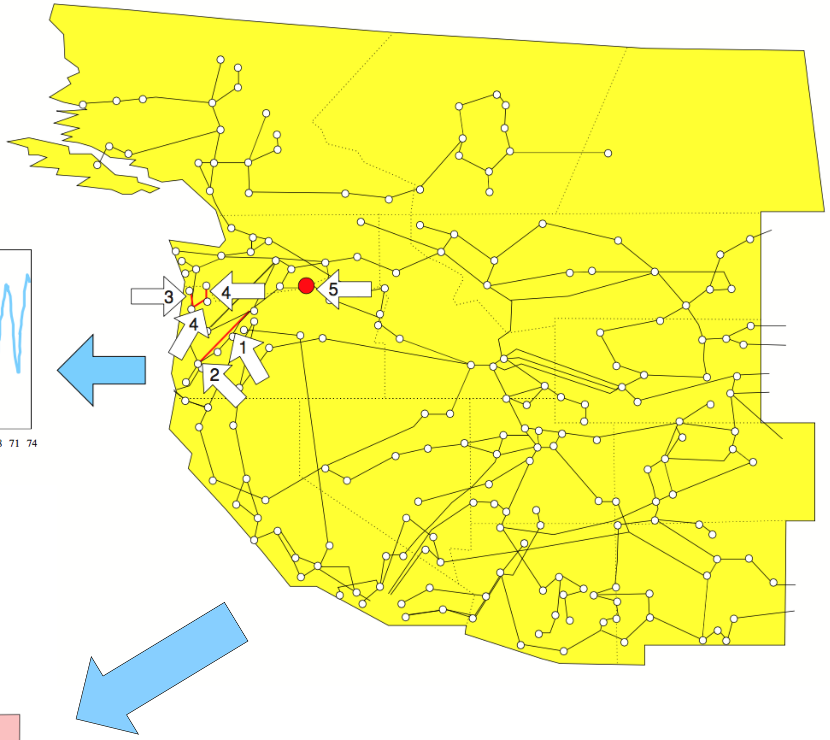


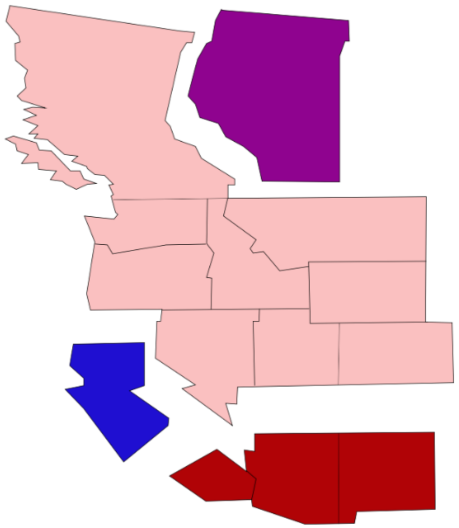
CURENT

Electromechanical Dynamics in the Western US (1996)

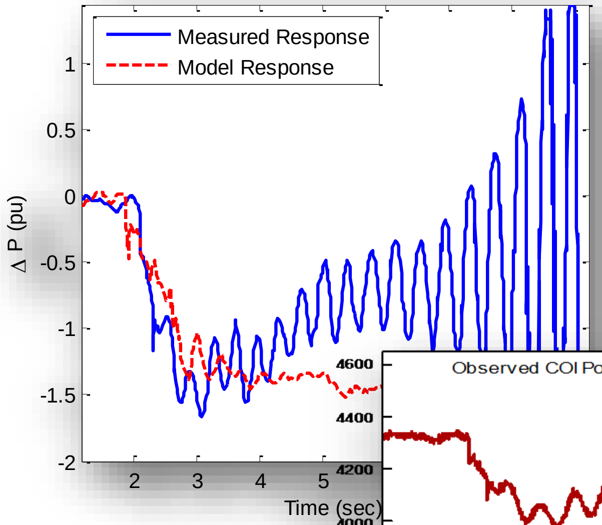
- What was measured:



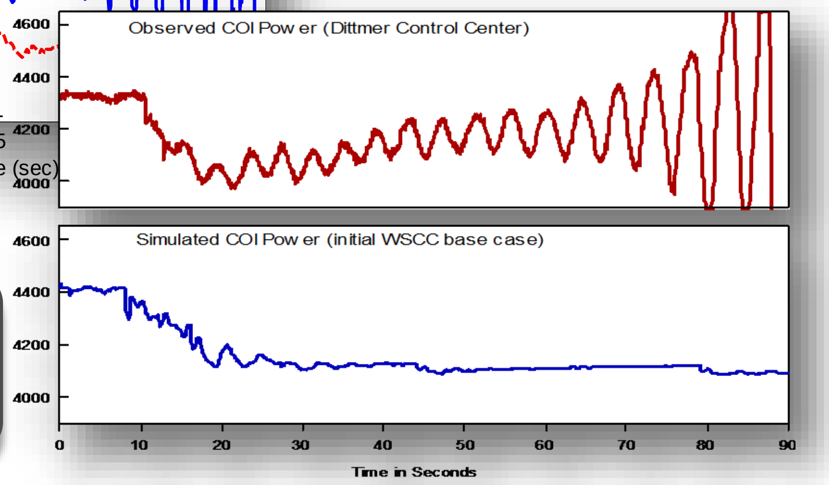




- What was simulated:



Before Model Calibration



After Model Calibration

Electromechanical dynamic modeling help to capture ”wide-area” behavior across geographical sparse interconnected networks such as these type of oscillations. Good models are crucial for planning and operation of electrical power networks.

Power Systems General DAE Model

$$\dot{\mathbf{x}} = f\left(\mathbf{x}, \mathbf{y}, \boldsymbol{\eta}, \tilde{\mathbf{u}}, t\right),$$

$$\mathbf{0} = g\left(\mathbf{x}, \mathbf{y}, \boldsymbol{\eta}, \tilde{\mathbf{u}}, t\right).$$

- $\mathbf{x}$ – is the vector of state variables, $\mathbf{x} \doteq \tilde{\boldsymbol{\xi}}_i$
- $\mathbf{y}$ – is the vector of algebraic variables, $\mathbf{y} \doteq \tilde{\boldsymbol{\xi}}_f$
- $\boldsymbol{\eta}$ – is the vector of parameters, from discarding $\tilde{\varphi}_s$ and letting $\tilde{\boldsymbol{\xi}}_s = \boldsymbol{\eta}$
- $\tilde{\mathbf{u}}$ – is the vector of discrete variables.
- $f(\cdot)$ – are the differential equations, $f(\cdot) \doteq \tilde{\varphi}_i(\cdot)$
- $g(\cdot)$ – are the algebraic equations, $g(\cdot) \doteq \tilde{\varphi}_f(\cdot)$

[Ref.] F. Milano, Power System Modeling and Scripting, Springer, 2010.

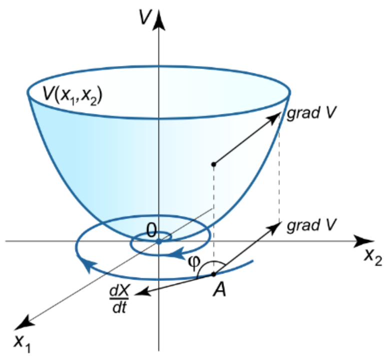
Finding the *"Power Flow"* Steady State "Equilibria"

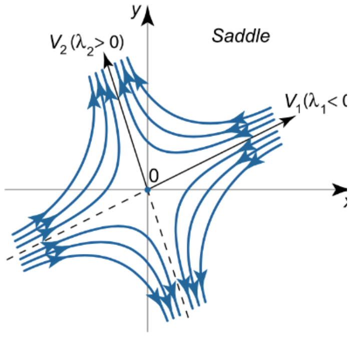
- The power system needs to be in balance, i.e. after a disturbance it must converge to an equilibrium (operation point).
  - Q: How can we find this equilibrium?
  - A: Set derivatives to zero and solve for all unknown variables!

Modelica –compliant tools attempt to solve this problem

$$0 = f(\mathbf{x}, \mathbf{y}, \boldsymbol{\eta}, \tilde{\mathbf{u}}, t),$$

$$0 = g(\mathbf{x}, \mathbf{y}, \boldsymbol{\eta}, \tilde{\mathbf{u}}, t).$$





- Some observations that can be made:
  - The algebraic equations in corresponded to having the fast differential equations at equilibrium all the time (in the model and in the timescale considered).
  - Finding the equilibrium when most of the variables are unknown is very difficult if when we try to solve this equation system simultaneously.
  - **NB: power system tools do not generally do this!**
  - Hence, we attempt to sequentially solve the equation system for each *t*.
  - First, we need to solve the algebraic equations g that only depend on the algebraic variables… this is were power systems deviates from other fields.

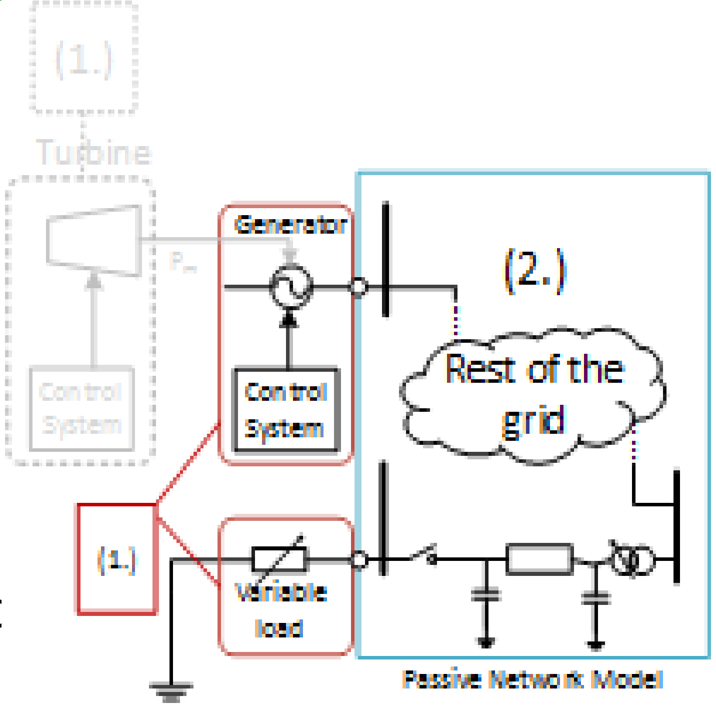
CURENT

Power System Modeling and Simulation Approach

- Equation set ***g*** is separated in two sets of algebraic equations:

$$\left.\begin{aligned}\dot{\mathbf{x}} &= f\left(\mathbf{x},\mathbf{y},\boldsymbol{\eta},\tilde{\mathbf{u}}\right),\\ \mathbf{0} &= g_1\left(\mathbf{x},\mathbf{y},\boldsymbol{\eta},\tilde{\mathbf{u}}\right),\end{aligned}\right\} \quad (1.)$$

$$\mathbf{0} = g_2\left(\mathbf{y},\boldsymbol{\eta},\tilde{\mathbf{u}}\right)\} \quad (2.)$$

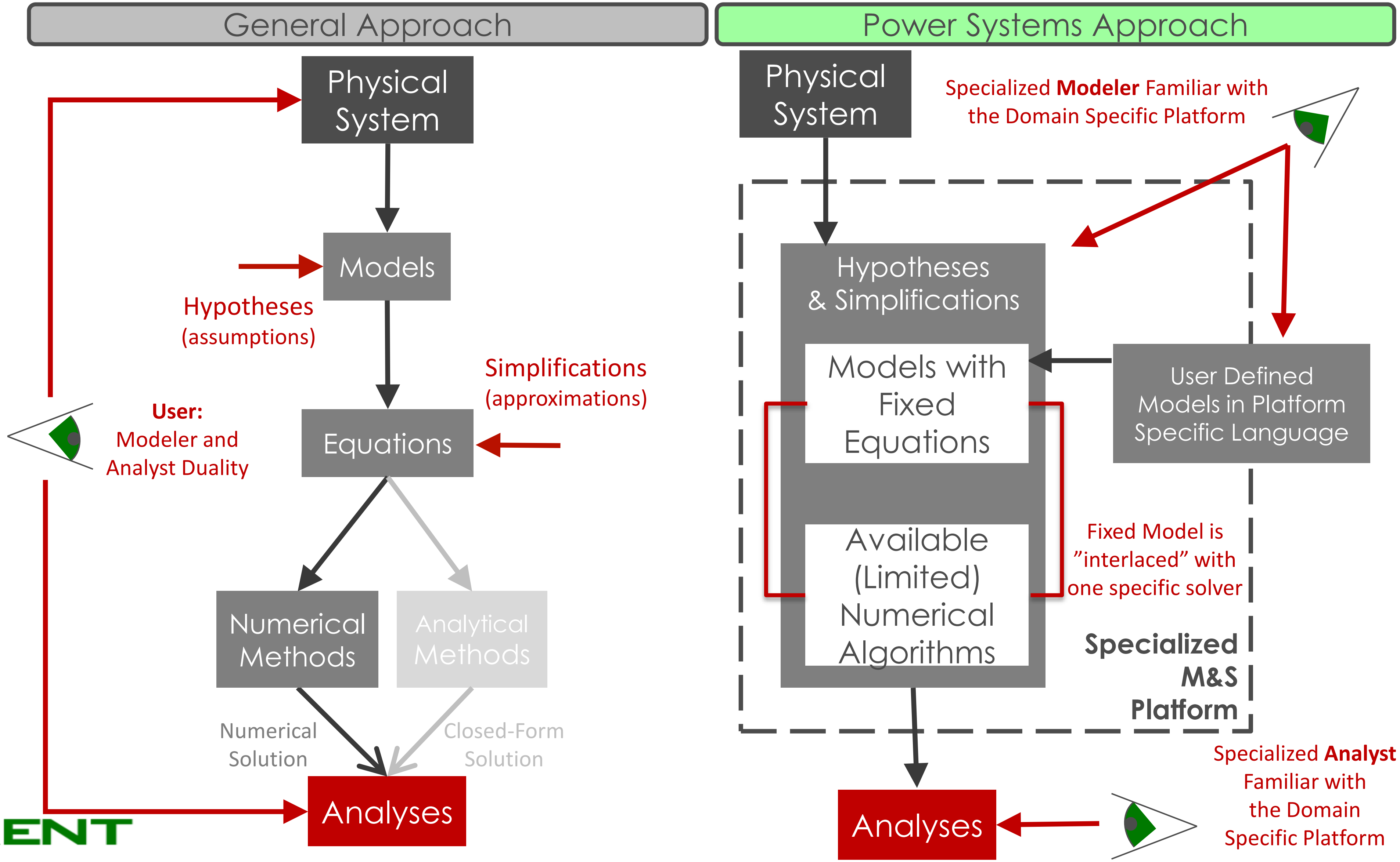


(1) Is the part which governs how dynamic models will evolve, sinc both **x** and **y** , e.g. generators and their control systems.

(2) Is the network model, consisting of transmission lines and other passive components which only depends on algebraic variables, **y**.

Simulation: Starting from a solution of (2) **only**, equations (1) are solved at equilibrium **individually;** to compute the starting guess of an ad-hoc DAE solver that iterates for (1)-(2) at each time step.

Fundamental Implications of the Conventional Power Systems Approach



Practical Implications of the Conventional Power Systems Approach

Status Quo:
Multiple simulation tools, with their own interpretation of different model features and data “format”.
Implications of the Status Quo:
- Dynamic models can ***rarely be shared in a straightforward manner without loss of information on power system dynamics.***
- ***Simulations are inconsistent without drastic and specialized human intervention.***

Beyond general descriptions and parameter values, a **common and unified modeling language** would **require a formal mathematical description of the models** – but this is not the practice to date.

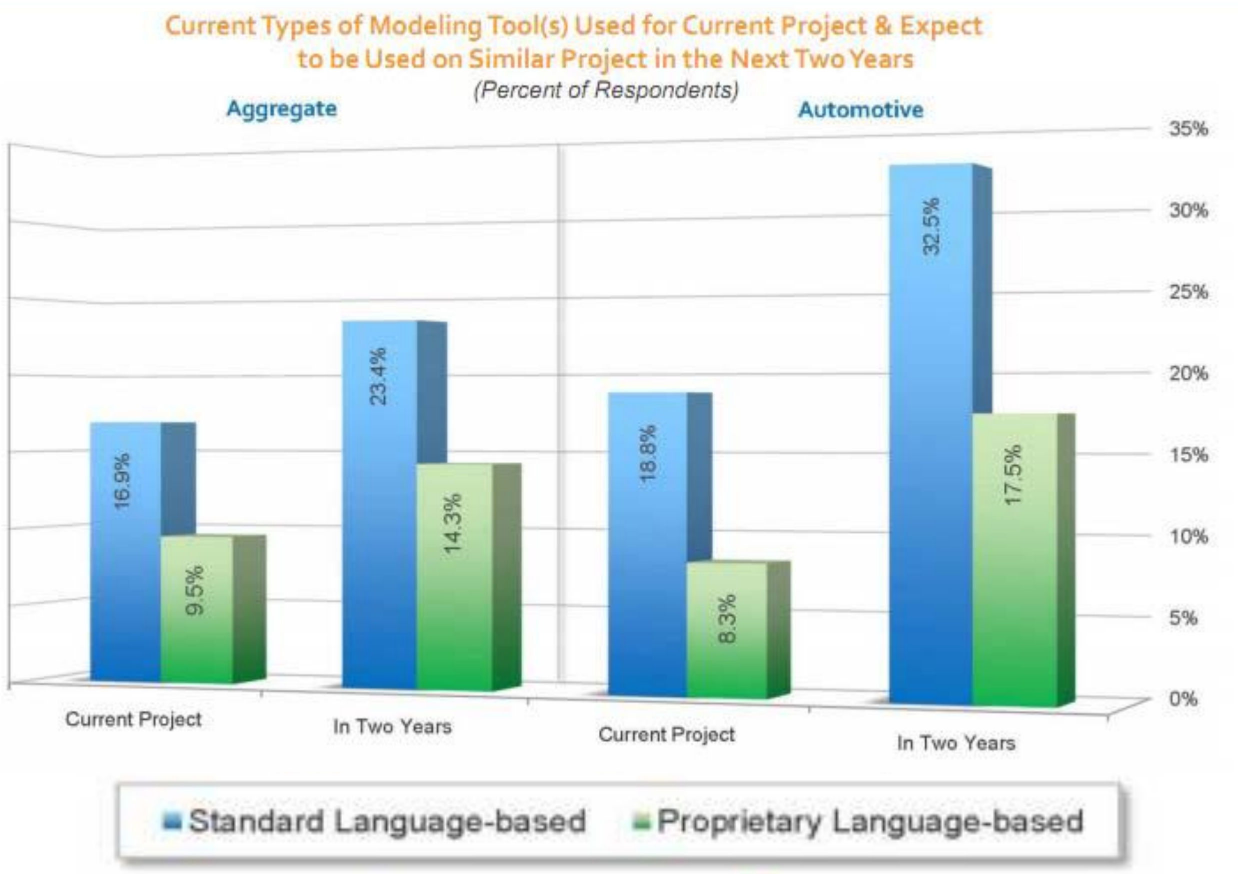
http://openprtspecs.blogspot.com

Why open standard-based modeling languages?

- Modeling tools first gained adoption as engineers looked for ways to simplify SW development and documentation.

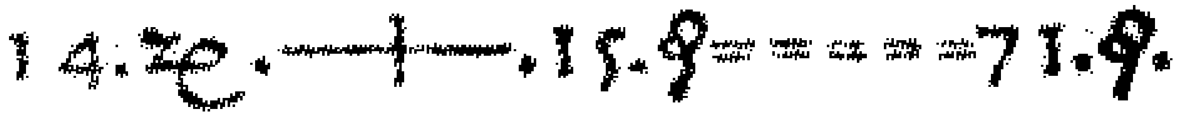
- **Today's modeling tools and their use cases have evolved.**

- ***Now:*** need for addressing both system level design and SW development/construction.



vdcresearch.com (2015)

Why equation-based modeling?



Equation—based modeling:

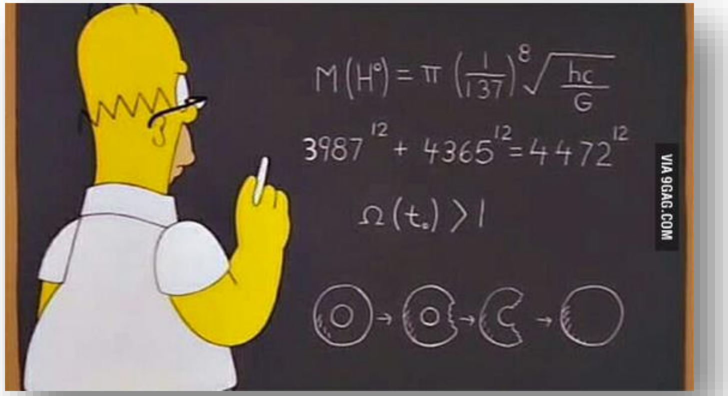
- Defines an implicit (not explicit) relation between variables.
- **The data-flow between variables is defined right before simulation** of the model (not during the modelling process!)

- A system can be seen as a complete model or a set of individual components.
- **The user is (in principle) only concerned with the model creation,** and does not have to deal with the underlying simulation engine (only if desired).
- It also allows decomposing complex systems into simple sub-models easier to understand, share and reuse

CSSL (1967) introduced a special form of “equation”:

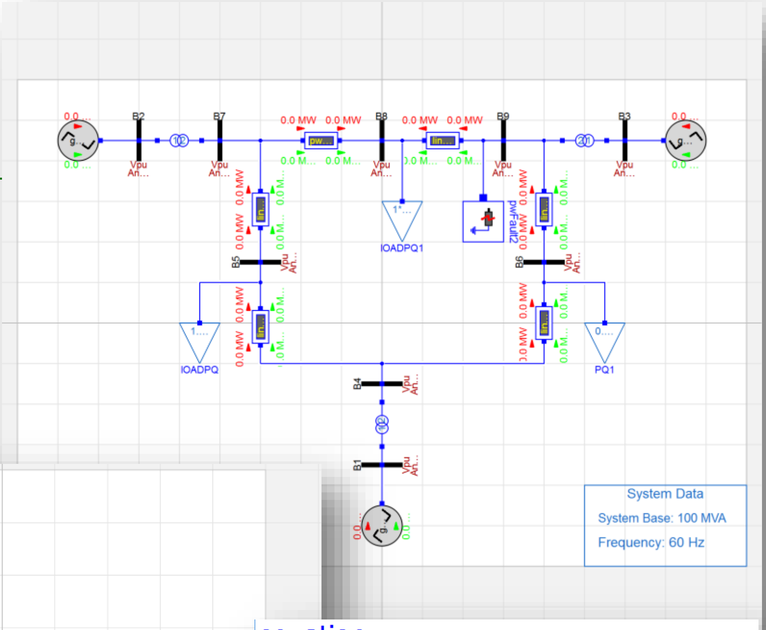
```
variable = expression
v = INTEG(F)/m
```

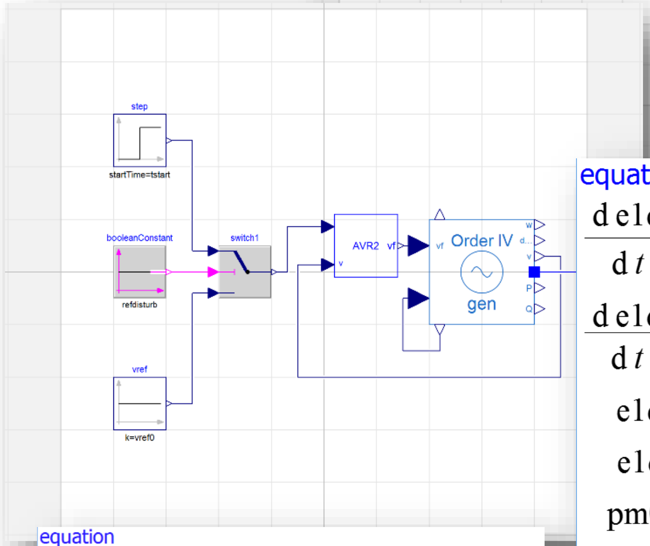
Programming languages usually do not allow equations!



Graphical Equation-Based Modeling

- Each icon represents a physical component (i.e. a generator, wind turbine, etc.)
- Composition lines represent the actual interconnections between components (e.g. generator to transformer to line to …)
- Physical behavior of each component is described by equations.
- There is a hierarchical decomposition of each component.





equation

$$\frac{d\,e1q}{d\,t} = \frac{(-e1q) - (Xd - x1d)\,id + vf}{Td10}$$

$$\frac{d\,e1d}{d\,t} = \frac{(-e1d) + (Xq - x1q)\,iq}{Tq10}$$

$$e1q = vq + Ra \cdot iq + x1d \cdot id$$

$$e1d = vd + Ra \cdot id - x1q \cdot iq$$

$$pm0 = pm00$$

$$vf0 = vf00$$

end Order4

equation

$$\frac{d\,vm}{d\,t} = \frac{v - vm}{Tr}$$

$$u = vref - vm - vr2 - \frac{vf \cdot Kf}{Tf}$$

$$\frac{d\,vr2}{d\,t} = -\frac{\frac{vf \cdot Kf}{Tf} + vr2}{Tf}$$

$$simpleLagLim.u = u$$

$$\frac{d\,vf}{d\,t} = -\frac{vf \cdot (Ke + Se) - simpleLagLim.y}{Te}$$

$$Se = Ae \cdot e^{Be \cdot |vf|}$$

end AVRTypeII

MODELICA is a (computer) modeling language, *it is <u>not</u> a tool!*

- Modelica is a free/*libre* object-oriented modeling language with a textual definition to describe physical systems using differential, algebraic and discrete equations.
- A Modelica modeling environment is needed to edit or to browse a Modelica model graphically in form of a composition diagram (= schematic).
- A Modelica translator is needed to transform a Modelica model into a form (usually C-code) which can be simulated by standard tools.
- A Modelica modeling and simulation environment provides both of the functionalities above, in addition to auxiliary features (e.g. plotting)

MODELICA

Modelica® - A Unified Object-Oriented Language for Systems Modeling

Language Specification

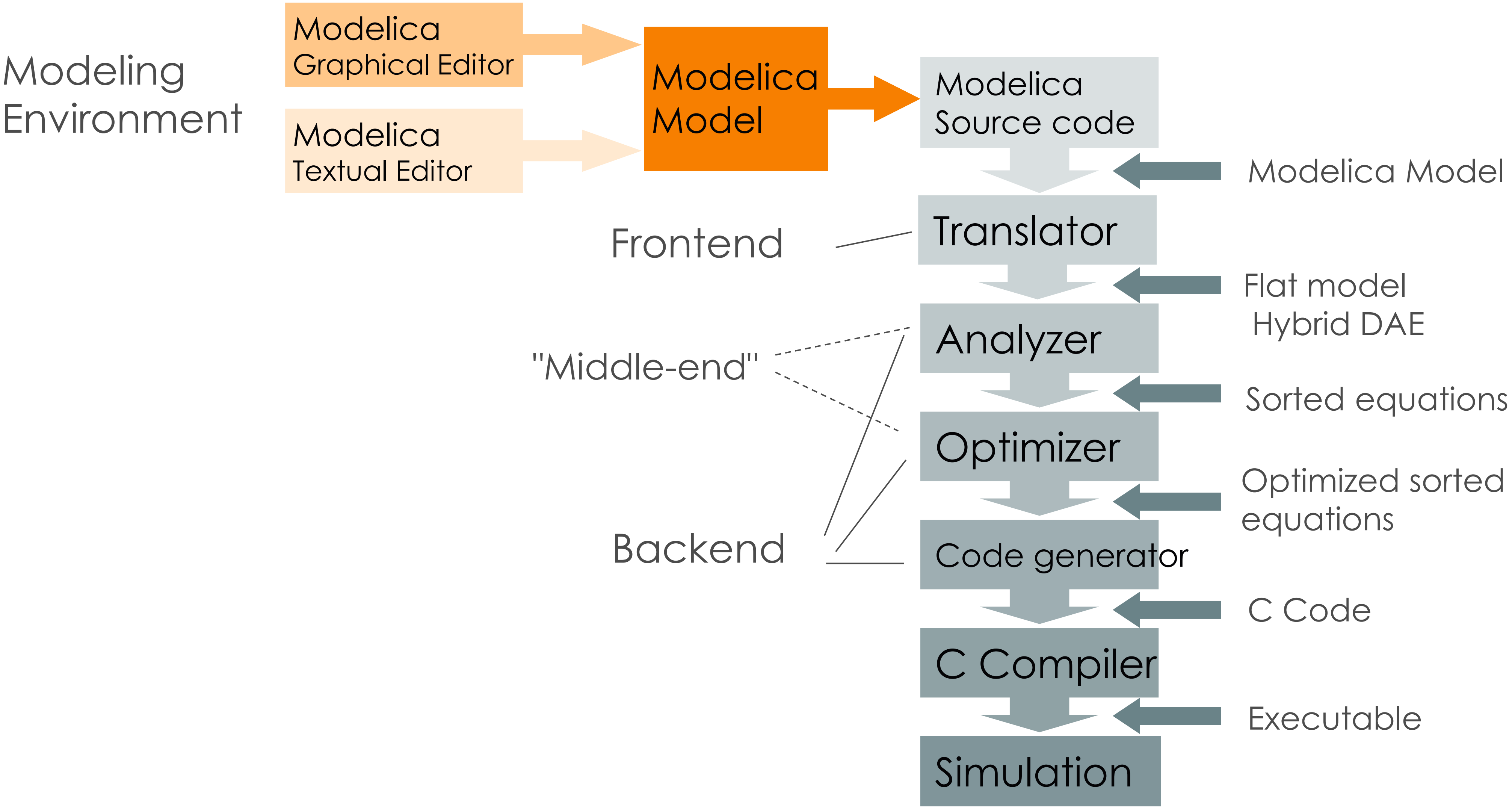
Version 3.3 Revision 1

July 11, 2014

http://modelica.readthedocs.io/en/latest/#

Key: standardized and open language specification

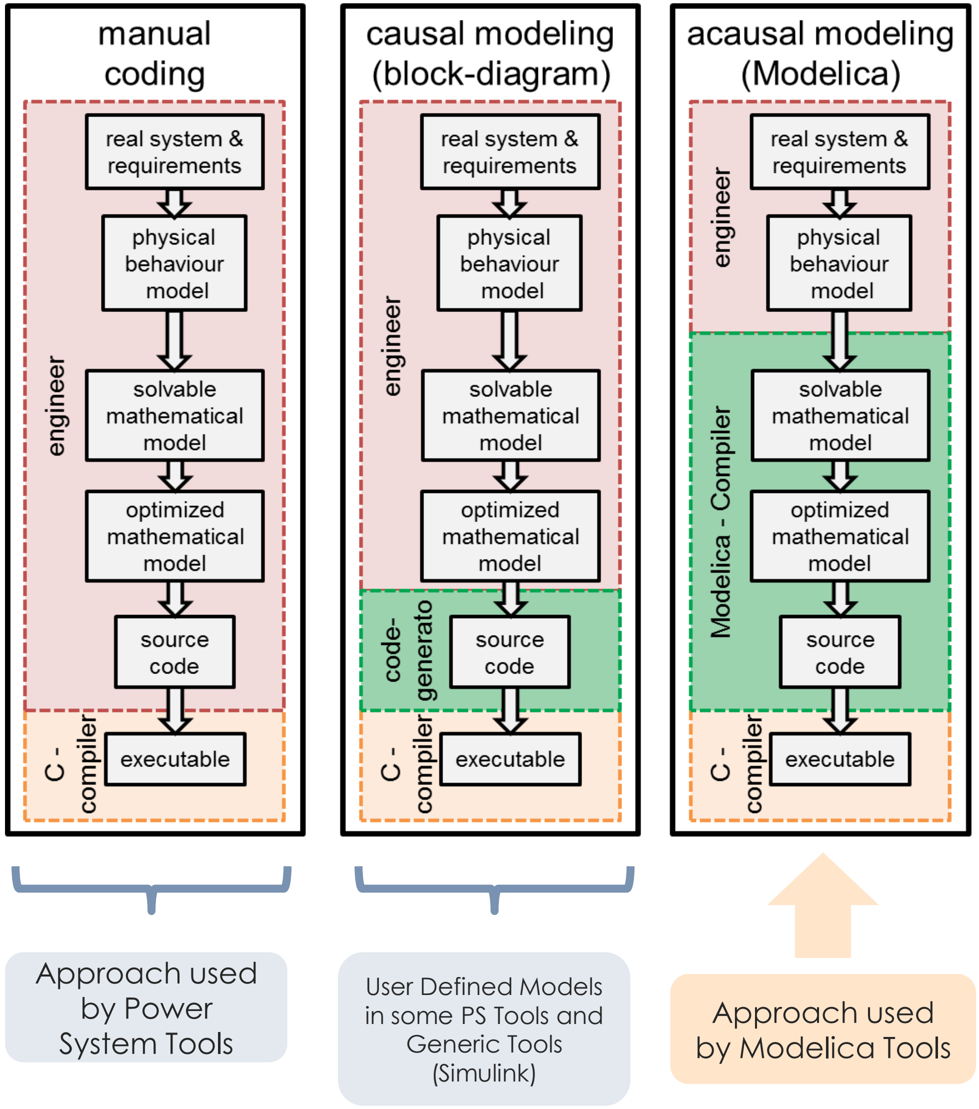
modeling and simulation environment (tool) tasks



Acasual Modeling and it's Implications on Model/Tool Development

- Acasual Modeling implicitly leads to **faster development and lower maintenance** for models (and even tools)
- The acausality makes Modelica library classes more reusable than traditional classes containing assignment statements where the input-output causality is fixed.

- Modelica Compiler performs **Causalization**
  - It flattens the model and then transform and sort all equations that give the model description.
  - Aim is to match each equation to a variable, hence the term "matching" process, doing an Index reduction of the DAEs and transform them to ODEs



Why MODELICA for power systems?

- The **order of computations is decided at modelling time**

Acausal	Causal

`R*I = v;`	`i := v/R;`
	`v := R*i;`
	`R := v/i;`

- Most tools make **no difference between “solver” and “model”** – in many cases solver is implanted in the model

- There is **no guarantee** that the same standardized model is implemented in the same way across different tools

- Even in Common Information Model (CIM) v15, **only block diagrams** are provided instead of equations

- Models are **black boxes** whose parameters are shared in a **specific “data format”**

- For large models this **requires translation** into the internal data format of each program

and Power Systems

- *Previous and Related Efforts*
  - Modelica for power systems *was first attempted* in the early 2000's (Wiesmann & Bachmann, Modelica 2000) - "electro-magnetic transient (EMT) modeling" approach.
    - SPOT (Weissman, EPL-Modelon) and its close relative PowerSystems (Franke, 2014); supports multiple modeling approaches –i.e. 3phase, steady-state, "transient stability", etc.
  - Electro-mechanical modeling or "transient stability" modeling:
    - Involves electro-mechanical dynamics, and neglects (very) fast transients
    - For system-wide analysis, easier to simulate/analyze - domain specific tools approach
  - ObjectStab (Larsson, 2002; Winkler, 2015) adopts "transient stability" modeling.
  - The PEGASE EU project (2011) developed a small library of components in Scilab, which where ported to proper Modelica in the FP7 iTesla project (2012-2016).
  - The iPSL - iTesla Power Systems Library (Vanfretti et al, Modelica 2014, SoftwareX 2016), was released during 2015. Most models validated against typical power system tools.

OpenIPSL takes iPSL as a starting point and moves it forward (this presentation).

  - F. Casella (OpenModelica 2016, Modelica 2017) presents the challenges of dealing with large power networks using Modelica, and a dedicated library to investigate them using the Open Modelica compiler.

CURENT

MODELICA and Power Systems

Why another library for power systems?

Social Aspects (Vanfretti et al, Modelica 2014)

- **Resistance to change: an irrational and dysfunctional reaction of users (and developers?)**
    - Users of conventional power system tools are skeptical about any other tools different to the one they use (or develop), and are averse about new technologies (slow on the uptake)
- Change agents contribute (+/-) to address resistance through actions and interactions.

(1) Strategy do not impose the use of a specific simulation environment (software tool), instead,

(2) Propose
a **common human and computer-readable mathematical "description":** use of Modelica for unambiguous model exchange.

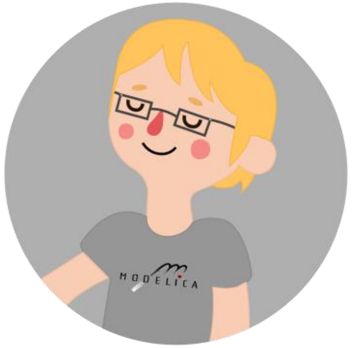
(3) Decrease of avoidance forces
- SW-to-SW validation gives quantitatively an similar answer than domain specific tools.
- Accuracy (w.r.t. to *de* facto tools) more important than performance

A never-ending effort!
- The library has served to bridge the gap between the Modelica and power systems community by:
- Addressing resistance to change (see above)
- Interacting with both communities – different levels of success…

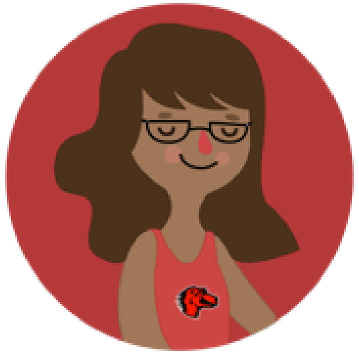
CURENT

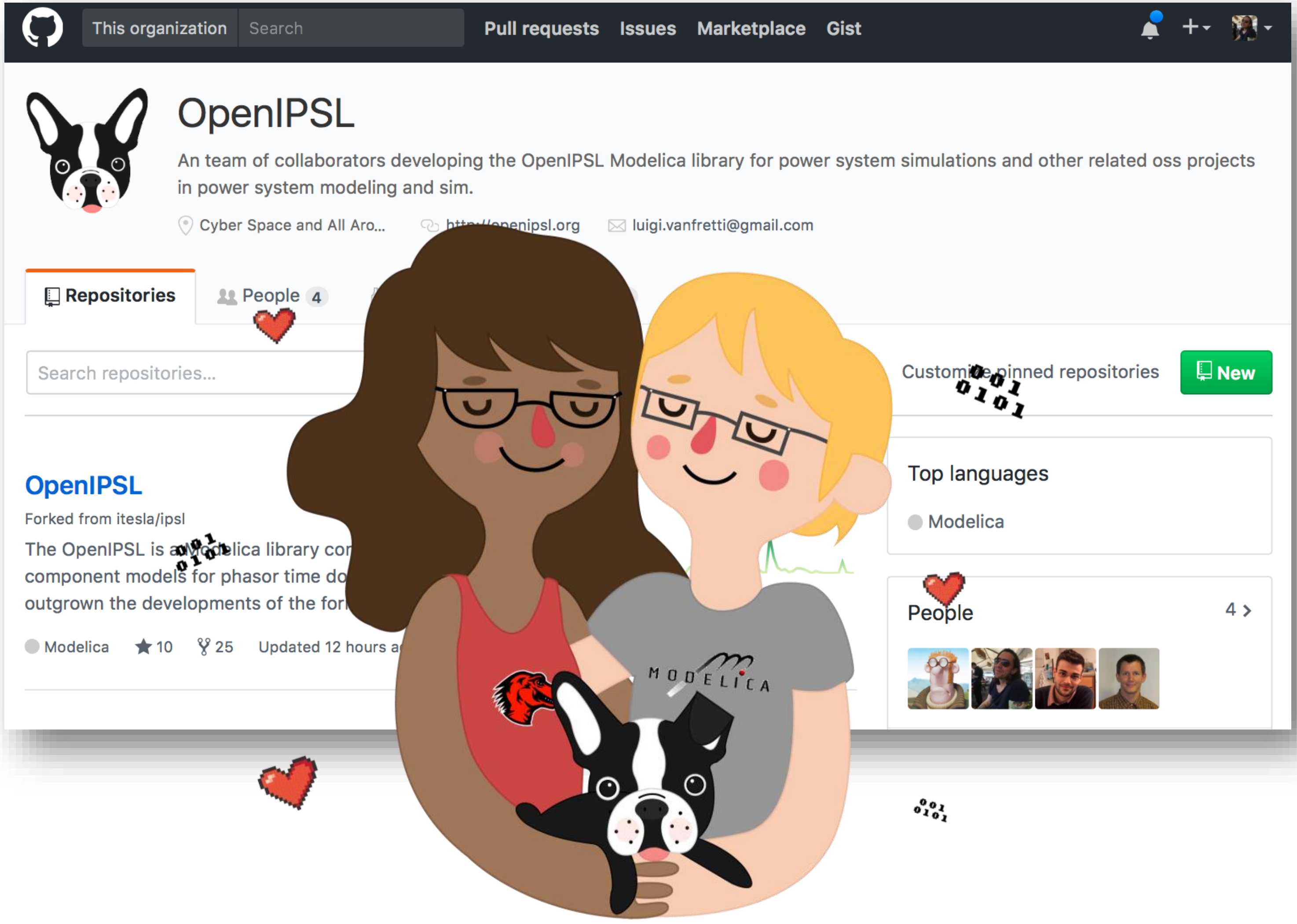
The *OpenIPSL* Project

- http://openipsl.org
- Built using the Modelica language:



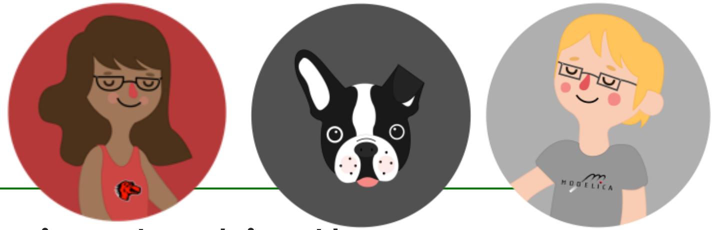
- Distributed with the MPL2 license:





Free as in Puppy!
Needs a lot of **your** love and care to grow and be happy!

The *OpenIPSL* Project - Origins



- **KTH SmarTS Lab** (my former research team) actively participated in the group or partners developing iPSL until the end of the *iTesla* project (March 2016)
- **iPSL** is a nice prototype, ***but we identified the following issues:***
    - **Development:** Need for compatibility with OpenModelica, (better) use of object orientation and proper use of the Modelica language features.
    - **Maintenance:** Poor harmonization, lack of code factorization, etc.
    - **Human issues:** The development workflow was complex
        - Different parties with disparate objectives, levels of knowledge, philosophy, etc.

New research requirements and the experiences from previous effort indicated:
- a clear *need for a different development approach* –
one that should address a complex development & maintenance workflow!

- OpenIPSL *started as a* ***fork*** of iPSL in 2016, and has now largely evolved!
- OpenIPSL is hosted on GitHub at http://openipsl.org
- OpenIPSL is actively developed by **ALSETLab** (formerly SmarTS Lab) members and friends, as a research and education oriented library for power systems → ***it is ok to try things out*** !

Fork: copy of a project going *in a different development direction*

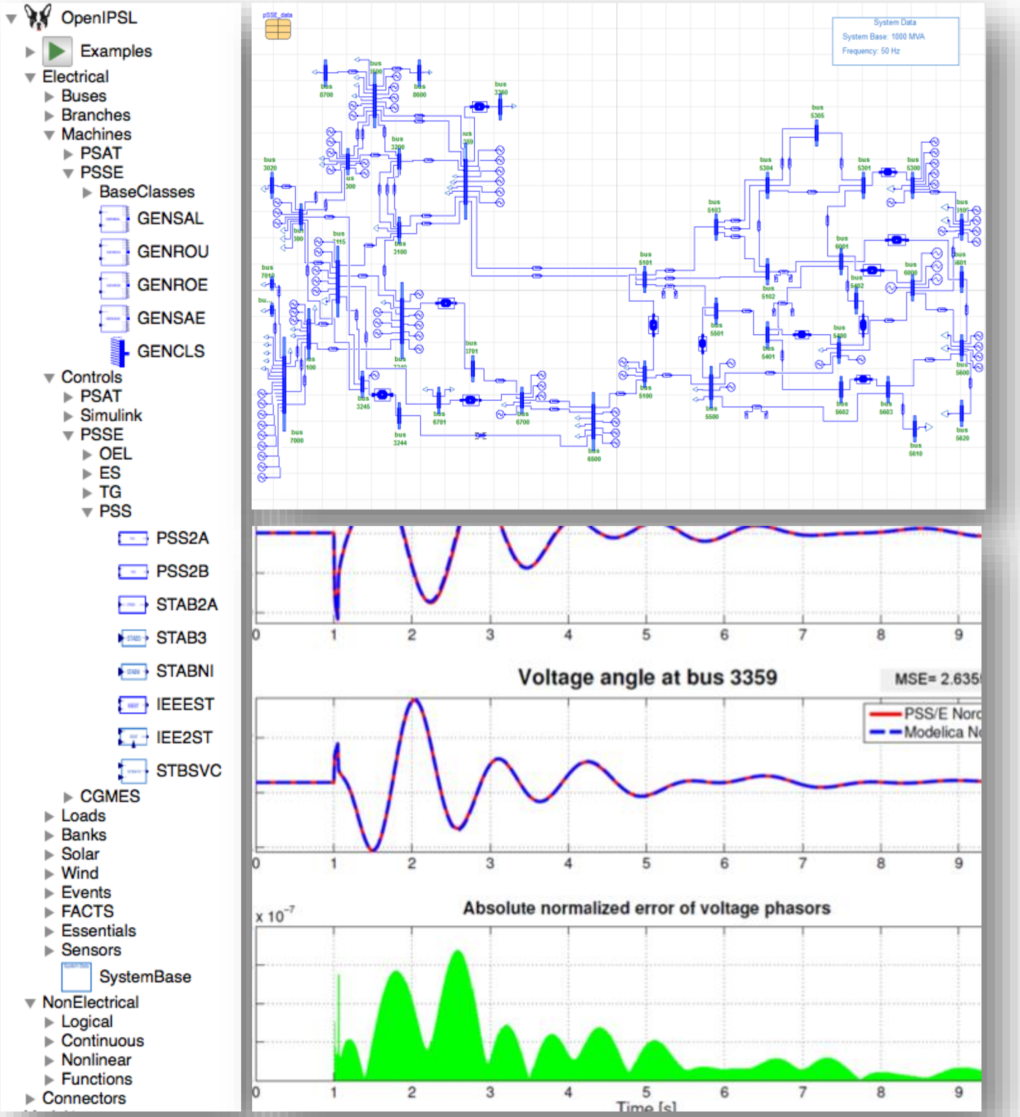
The *OpenIPSL* Library – Key Feautures

OpenIPSL is an open-source Modelica library for power systems

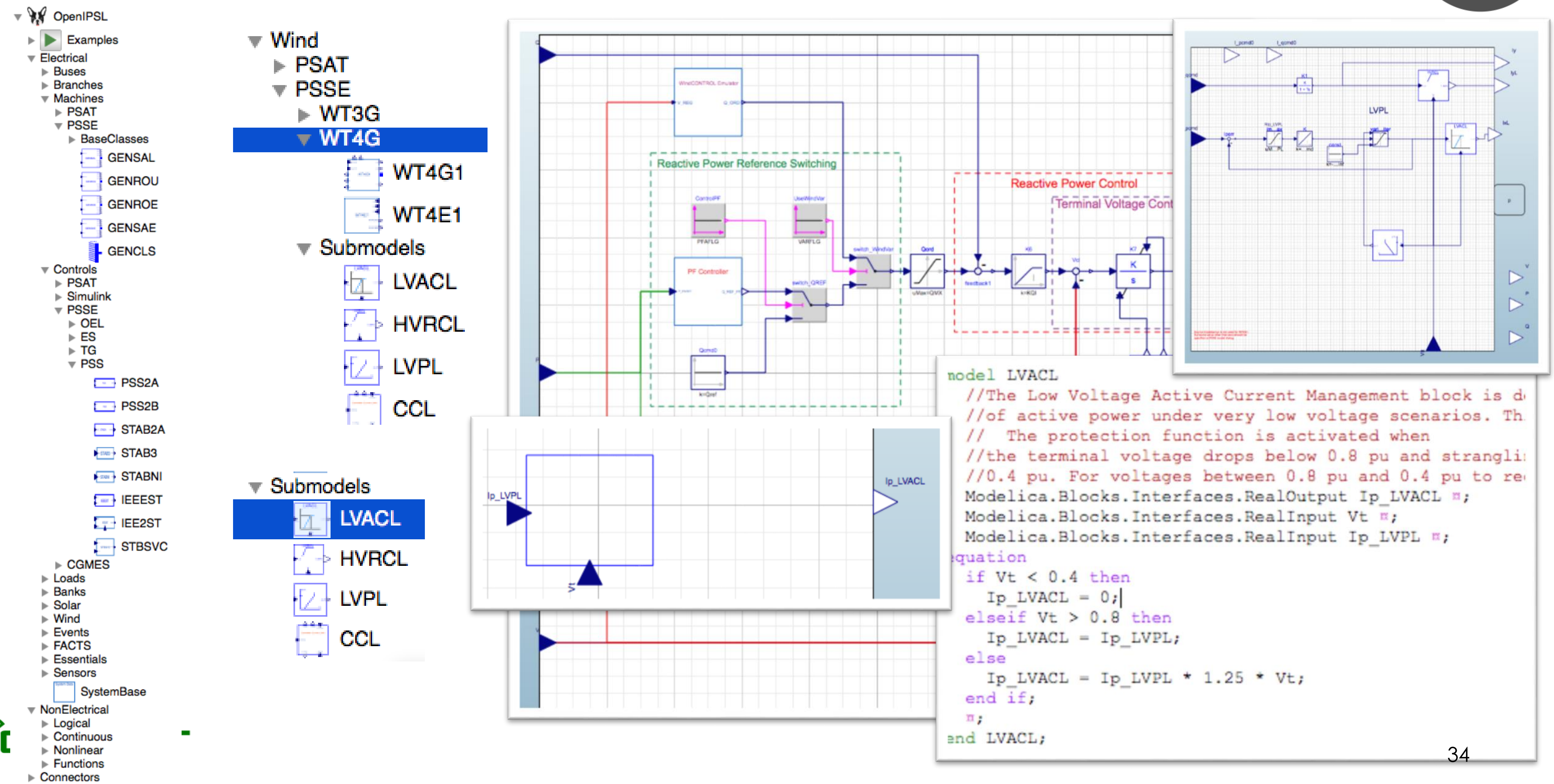
- It contains a set of **power system components** for **phasor time domain** modeling and simulation
- Models have been **validated** against a number of reference tools (mainly PSS/E)

OpenIPSL enables:

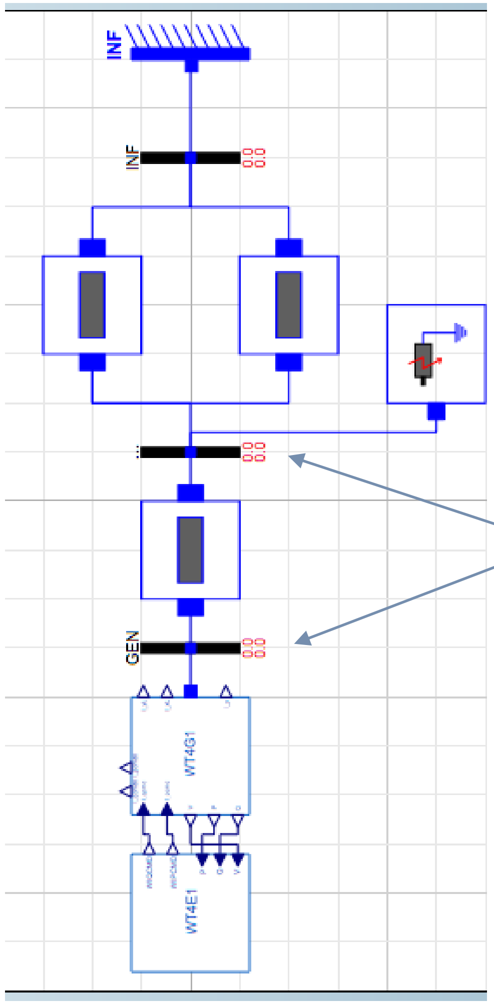
- **Unambiguous** model exchange
- Formal **mathematical description** of models
- **Separation** of **models** from tools/IDEs and **solvers**
- Use of **object-oriented** paradigms



The *OpenIPSL* Library – WT Example



The *OpenIPSL* Library – Network Example



- OpenIPSL
  - Examples
    - Machines
  - Wind
    - PSSE
      - WT4G
        - WT4G1_WT4E1
        - WT4G1

Class Connections

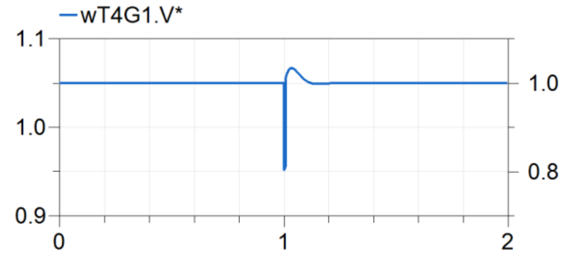
```
equation
  connect(wT4G1.p, GEN.p) ;
  connect(GEN.p, pwLine2.p) ;
  connect(pwLine2.n, BUS1.p) ;
  connect(BUS1.p, pwLine.p) ;
  connect(pwLine1.p, pwLine.p) ;
  connect(pwFault.p, BUS1.p) ;
  connect(pwLine.n, INF.p) ;
  connect(pwLine1.n, INF.p) ;
  connect(INF.p, gENCLS2_1.p) ;
  connect(wT4E1_1.WIQCMD, wT4G1.I_qcmd)
  connect(wT4E1_1.WIPCMD, wT4G1.I_pcmd)
  connect(wT4G1.P, wT4E1_1.P) ;
  connect(wT4G1.V, wT4E1_1.V) ;
  connect(wT4G1.Q, wT4E1_1.Q) ;

end WT4G1_WT4E1;
```

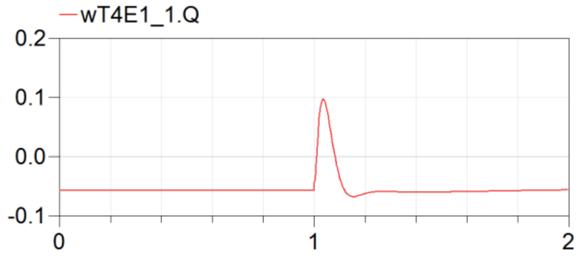
Resulting Parameter Declaration

```
model WT4G1_WT4E1
  extends Modelica.Icons.Example;
  constant Real pi=Modelica.Constants.pi;
  parameter Real V1=1.00000;
  parameter Real A1=-1.570655e-005;
```

Resulting Class Instantiation

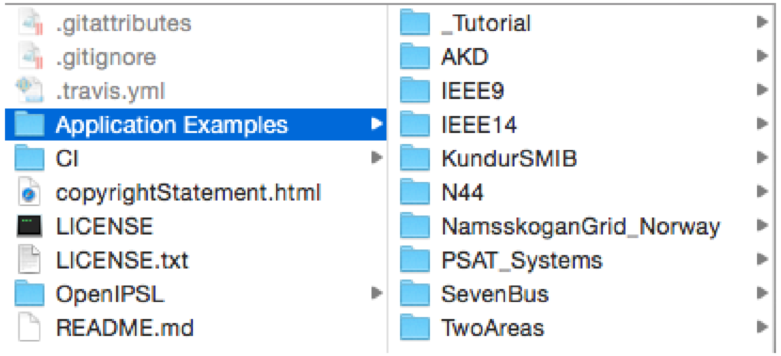
```
OpenIPSL.Electrical.Branches.PwLine pwLine2(
  G=0,
  B=0,
  R=2.50000E-3,
  X=2.50000E-3) ;
OpenIPSL.Electrical.Wind.PSSE.WT4G.WT4G1 wT4G1(
  V_0=V3,
  angle_0=A3,
  M_b=100,
  P_0=P3,
  Q_0=Q3,
  T_IQCmd=0.02,
  T_IPCmd=0.02,
  V_LVPL1=0.4,
  V_LVPL2=0.9,
  G_LVPL=1.11,
  V_HVRCR=1.2,
  CUR_HVRCR=2,
  RIp_LVPL=2,
  T_LVPL=0.02) ;
```



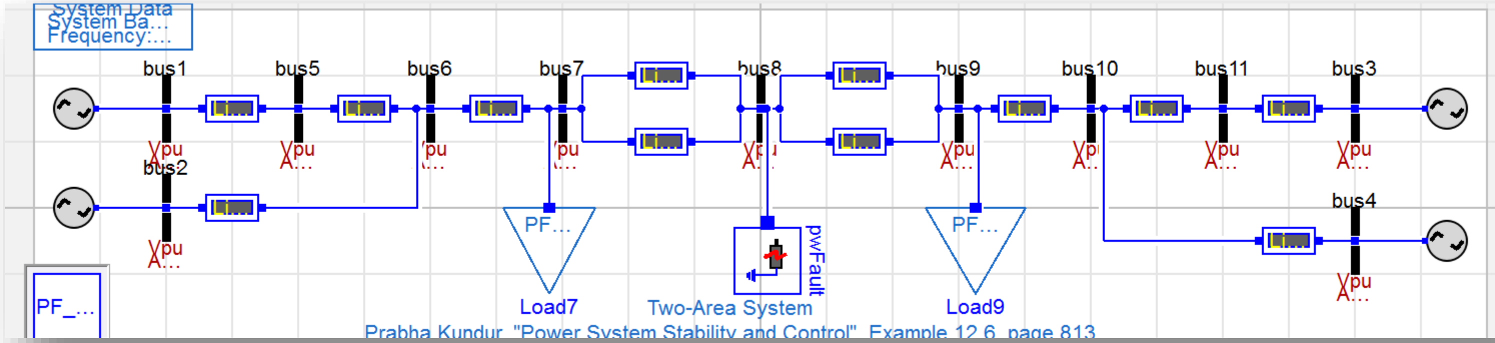


The *OpenIPSL* Library – Application Examples

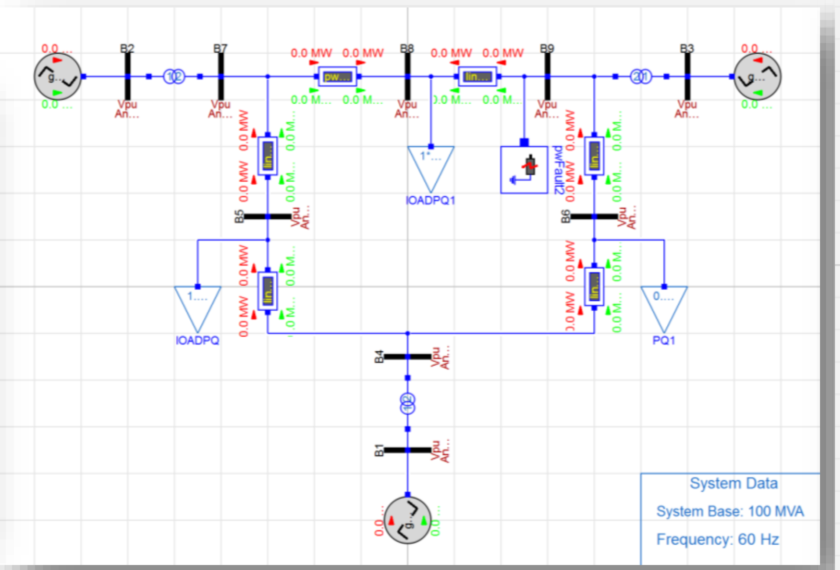
Many Application Examples Developed!!!



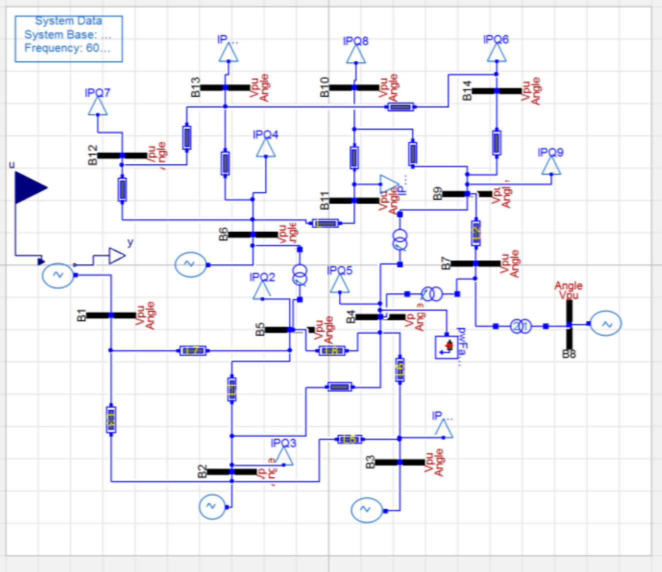
Klein-Rogers-Kundur 2-Area 4-Machine System



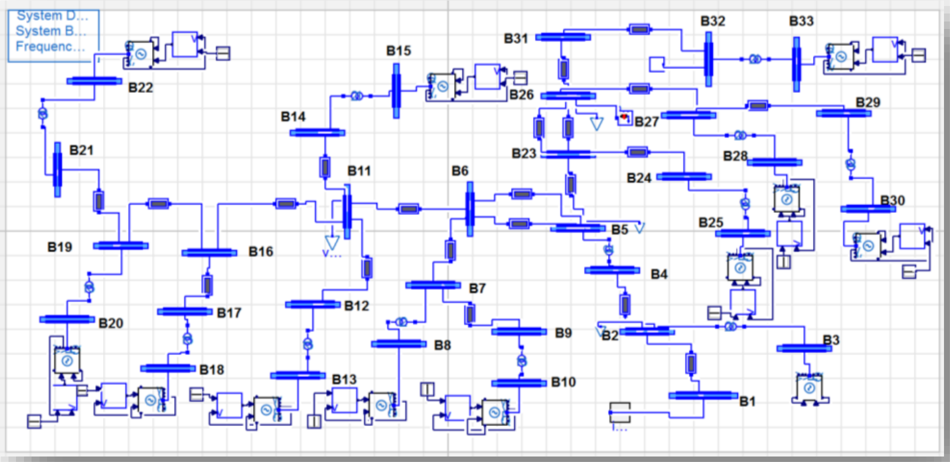
IEEE 9 Bus



IEEE 14 Bus



Namsskogan Distribution Network



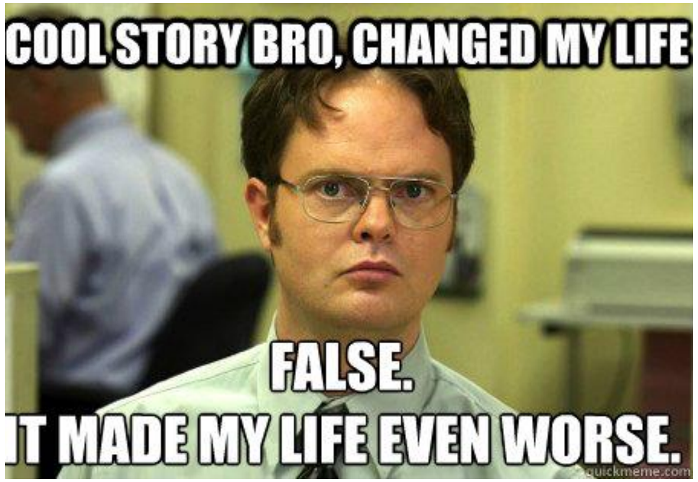
The *OpenIPSL* Library – providing a good "initial guess"

- **An initial guess for all algebraic, continuous and discrete variables need to be provided to solve a numerical problem!**
- When solving differential equations, one needs to provide the **initial value** of the state variables at rest.

- In Modelica, **initial values can be either solved or specified** in many ways, we use the following
  - Using the "initial equation" construct:
    - **`initial equation`**
      - **`x = some_value OR x = expression to solve`**
  - Setting the (**`fixed=true, start=x0`**) attribute when instantiating a model when the start value is known (or possible to calculate)
  - If nothing is specified, set the default would be a guess value (**`start= 0, fixed=false`**).

- In the OpenIPSL models we do the following:
  - The initial guess value is set with (**`fixed = false`**) for initialization.
  - Model attributes are treated as parameters with value (**`fixed = true`**),

- In OpenIPSL we use a power flow solution from an external tool (e.g. PSAT or PSS/E) as a starting point to compute initial guess values through parameters within each model.
  - The power flow solution is NOT the initial guess value itself.
  - Aim is to provide a better "initial guess" to find the initial values of the DAE system.



CURENT

The *OpenIPSL* Library – “initial guess” example

Power flow data

V_b	400	Base voltage of the bus (kV)
V_0	1	
angle_0	0	
P_0	1	
Q_0	0	
S_b	SysData.S_b	
fn	SysData.fn	

Initialization

w	1		Rotor speed (pu)
v	V_0	true	Generator terminal voltage (pu)
P	P_0 / S_b		Active power (pu)
Q	Q_0 / S_b		Reactive power (pu)
anglev	angle_0 / 180 * pi		Bus voltage angle
e1q	e1q0		q-axis transient voltage (pu)

Third order model from PSAT implemented in OpenIPSL

```
model Order3 "Third Order Synchronous Machine with Inputs and Outputs"
  import Modelica.Constants.pi;
  extends BaseClasses.baseMachine(delta(start = delta0), pe(start = pm00), pm(start = pm00)

    Real e1q(start = e1q0) "q-axis transient voltage (pu)";
  protected
    parameter Real Xd = xd * CoB "d-axis reactance, p.u.";
    parameter Real x1d = xd1 * CoB "d-axis transient reactance, p.u.";
    parameter Real Xq = xq * CoB "q-axis reactance, p.u.";
    parameter Real m = M / CoB2 "Machanical starting time (2H), kWs/kVA";
    parameter Real c1 = Ra * K "CONSTANT";
    parameter Real c2 = x1d * K "CONSTANT";
    parameter Real c3 = Xq * K " CONSTANT";
    parameter Real K = 1 / (Ra * Ra + Xq * x1d) "CONSTANT";
    parameter Real delta0 = atan2(vi0 + Ra * ii0 + Xq * ir0, vr0 + Ra * ir0 - Xq * ii0) "Initialitation";
    parameter Real vd0 = vr0 * cos(pi / 2 - delta0) - vi0 * sin(pi / 2 - delta0) "Initialitation";
    parameter Real vq0 = vr0 * sin(pi / 2 - delta0) + vi0 * cos(pi / 2 - delta0) "Initialitation";
    parameter Real id0 = ir0 * cos(pi / 2 - delta0) - ii0 * sin(pi / 2 - delta0) "Initialitation";
    parameter Real iq0 = ir0 * sin(pi / 2 - delta0) + ii0 * cos(pi / 2 - delta0) "Initialitation";
    parameter Real pm00 = (vq0 + Ra * iq0) * iq0 + (vd0 + Ra * id0) * id0 "Initialitation";
    parameter Real vf00 = e1q0 + (Xd - x1d) * id0 "Initialitation";
    parameter Real e1q0 = vq0 + Ra * iq0 + x1d * id0 "Initialitation";
  initial equation
    der(e1q) = 0;
  equation
    der(e1q) = ((-e1q) - (Xd - x1d) * id + vf) / Td10;
```

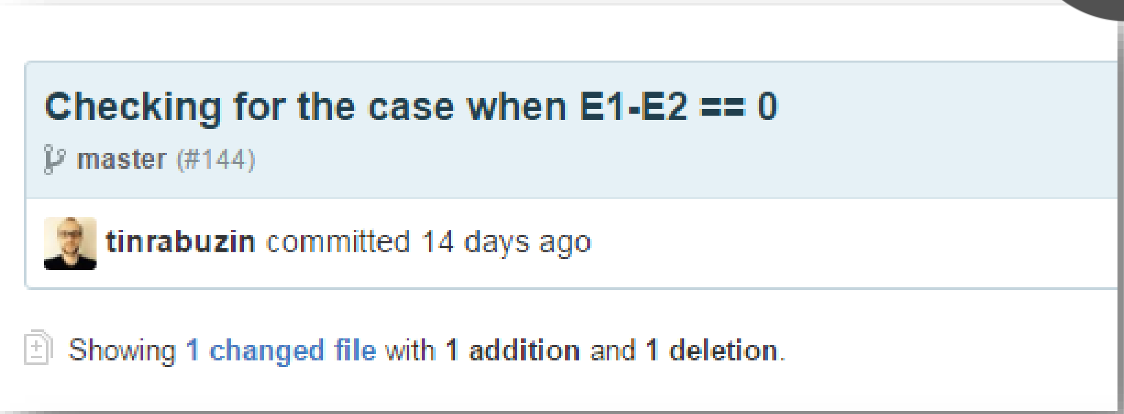
The *OpenIPSL* Project Documentation

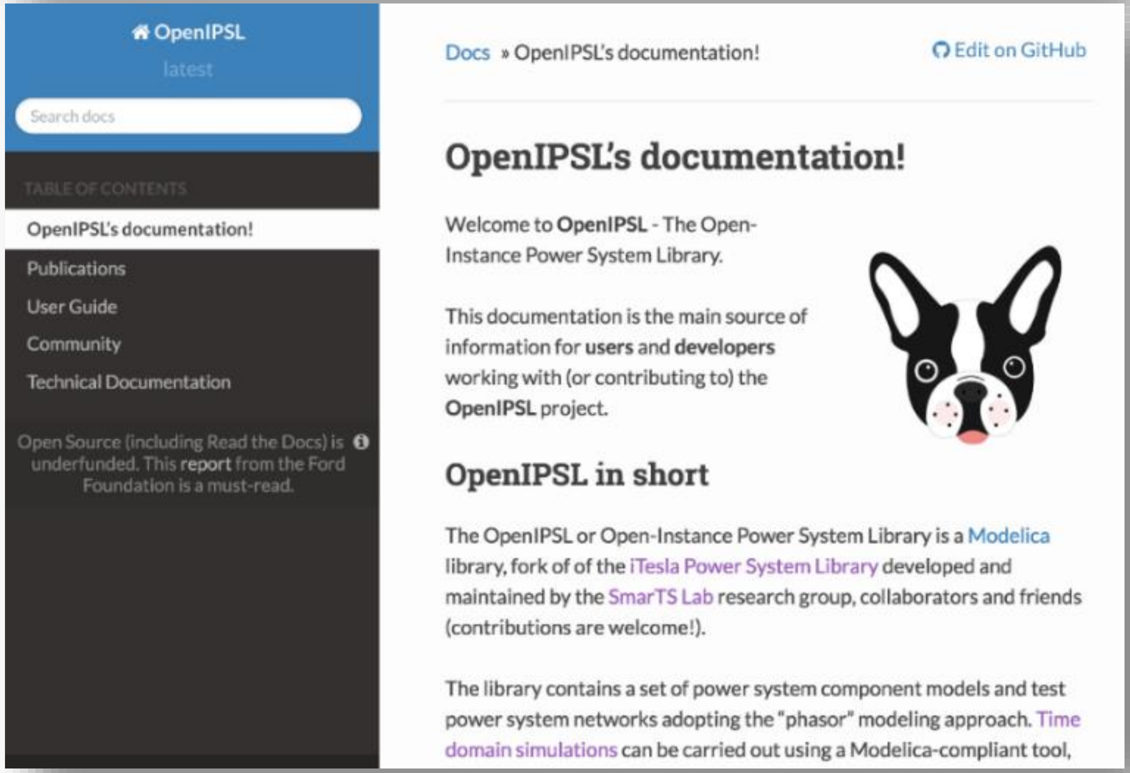
- Documentation of the code changes:
  - Explicit messages in ***commits*** and ***pull-requests***

- Documentation of the project
  - Presentation
  - User guide
  - Dev. guidelines & How to contribute

→ The documentation is written in ***reStructuredText*** (reST) hosted on http://openipsl.readthedocs.io/

- *Note:* Model documentation is not included, users are referred to literature, textbooks and the proprietary documentations.

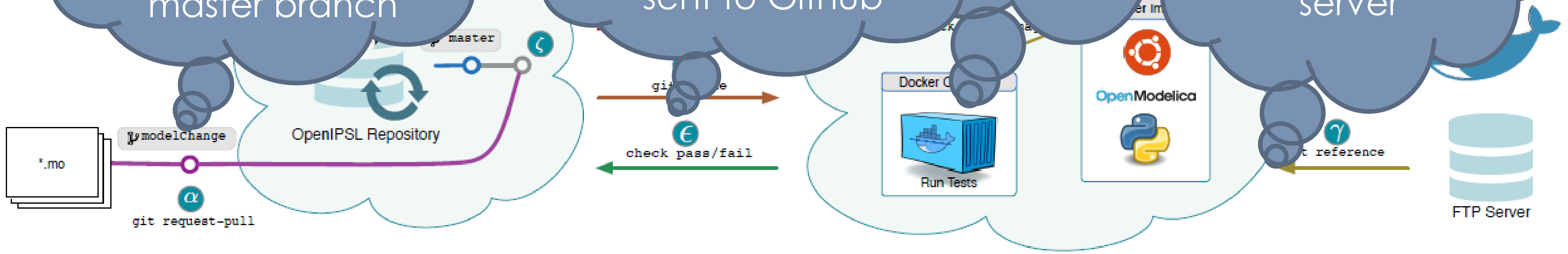




The *OpenIPSL*'s Project - Continuous Integration (CI) Service

- A CI service was implemented and integrated to the repository. The Modelica support was achieved with the following architecture:
- ***Travis*** as CI service provider
- ***Docker*** as the "virtualization" architecture
- ***DockerHub*** to host a Docker image with ***Python*** & ***Open*** ...***lica***

- The CI performs automated ***sy***...



The *OpenIPSL*’s CI Commit Output (Syntax Check Workflow)

Hide log | Raw log

```
1   Worker information                                                   worker_info
6   Build system information                                             system_info
340
341 $ git clone --depth=50 https://github.com/OpenIPSL/OpenIPSL.git      git.checkout  3.45s
359 $ sudo service docker start                                          services      0.02s
362 $ bash -c 'echo $BASH_VERSION'
363 4.3.11(1)-release
364 $ docker pull smartslab/ci_openipsl                                  install       78.20s
370 $ docker run -i -t -v $(pwd):/OpenIPSL smartslab/ci_openipsl sh /OpenIPSL/CI/changeUs  247.62s
371 2017-07-31 15:04:09,726 - OMCSession - INFO - OMC Server is up and running at
    file:////tmp/openmodelica.smartslab.objid.09f7f46af99b49c5af25991f610391bd
372 /OpenIPSL/package.mo is successfully loaded.
373 ==== Check Summary for OpenIPSL ====
374 Number of models that passed the check is: 267
375 Number of models that failed the check is: 0
376 /ApplicationExamples/KundurSMIB/package.mo is successfull
377 ==== Check Summary for KundurSMIB ====
378 Number of models that passed the check is: 7
379 Number of models that failed the check is: 0
```

Travis

un when setting
);``
t under Dymola

Results from CI testing on syntax check on several “Application Examples”

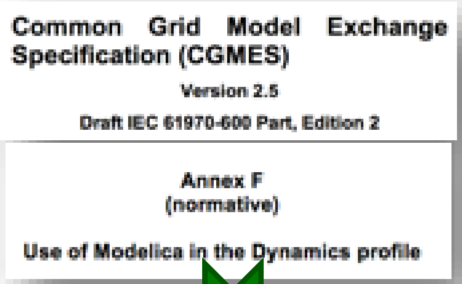
The *OpenIPSL* – Ongoing Developments and Future Work!

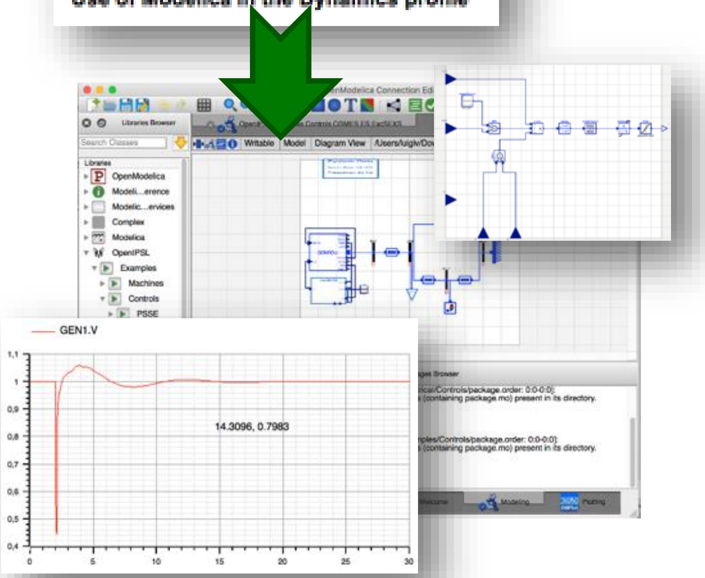
- **Library Improvements (Tin Rabuzin, Maxime Baudette)**

100% Compatibility with OM (100% Syntax Check, ~100% Simulation for components) through efforts in Continuous Integration adoption

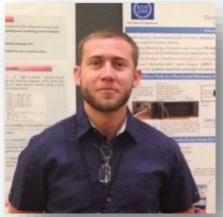
Change in the models to include inheritance (code factorizing)

Fixing and validating network models – application examples (thanks to CI)





- **ENTSO-E IOP Models (Francisco Gomez Lopez)**
  - Proof of concept and test model
  - Excitation system and small network model



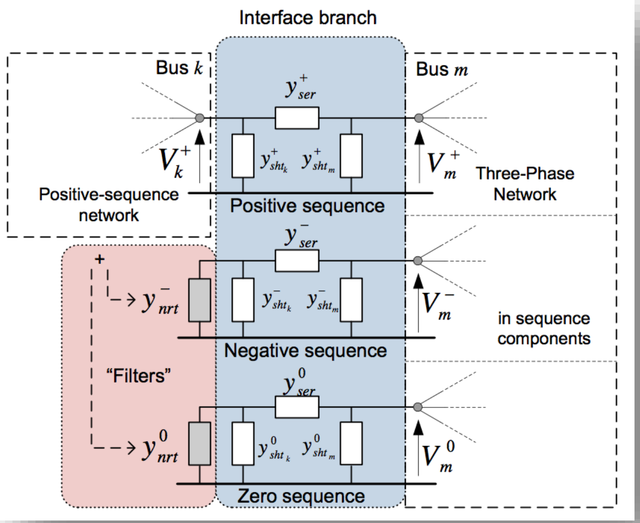
- **OpenIPSL.org organization in Github! (Prof. Dietmar Winkler)**
  - **Website will be also hosted there!**

CURENT

The *OpenIPSL* – Ongoing Developments and Future Work!

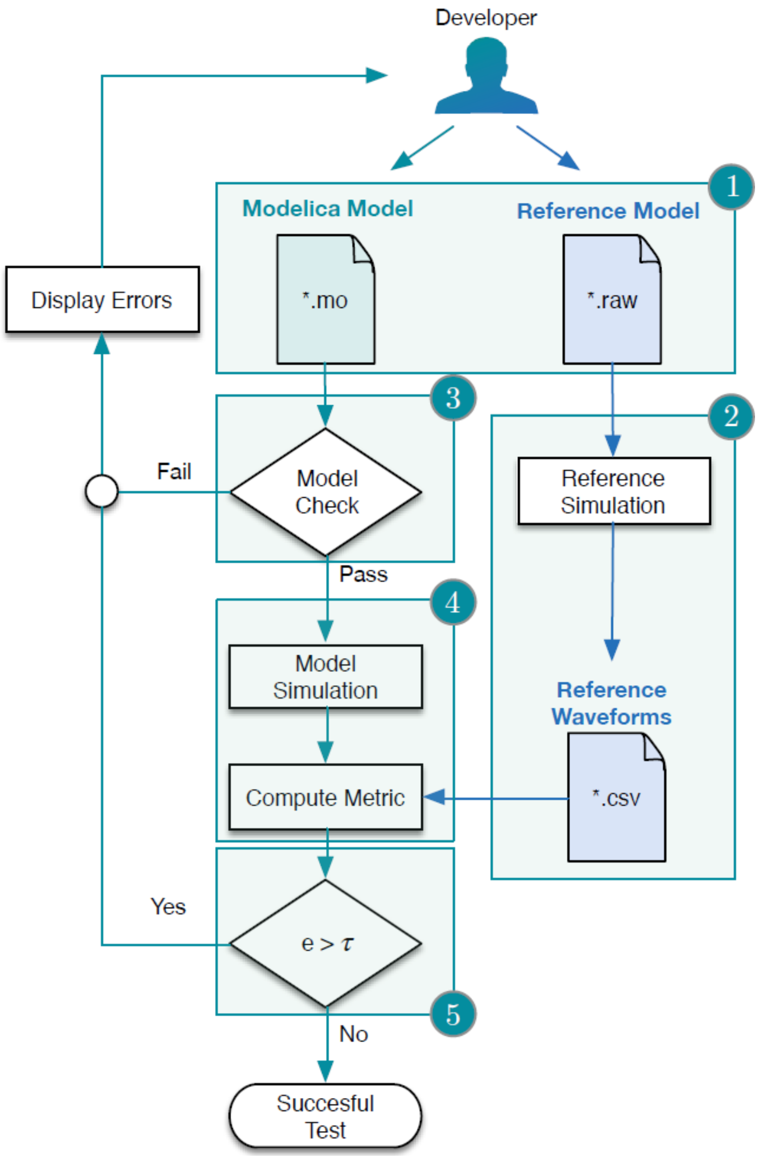
- **Efforts within the openCPS project**
- **New Component Models**
  - New components for the OpenIPSL being developled:
  - Process noise (stochastic) **pdf-based** load models
  - Frequency estimation models
  - PMU “Container” Block (frequency estimation + packaging)
  - Control systems for islanded operation and automated resynch
- **Multi-domain modeling for gas turbines and power systems**
  - Based on ThermoPower and OpenIPSL
  - Miguel Aguilera, et al.

- ***More!***
- *Joint* modeling and simulation of transmission (positive sequence) and distribution (three-phase) power networks
  - Marcelo de Castro Fernandez and Prof. Janaina Gonçalvez (UFJF, Brazil)

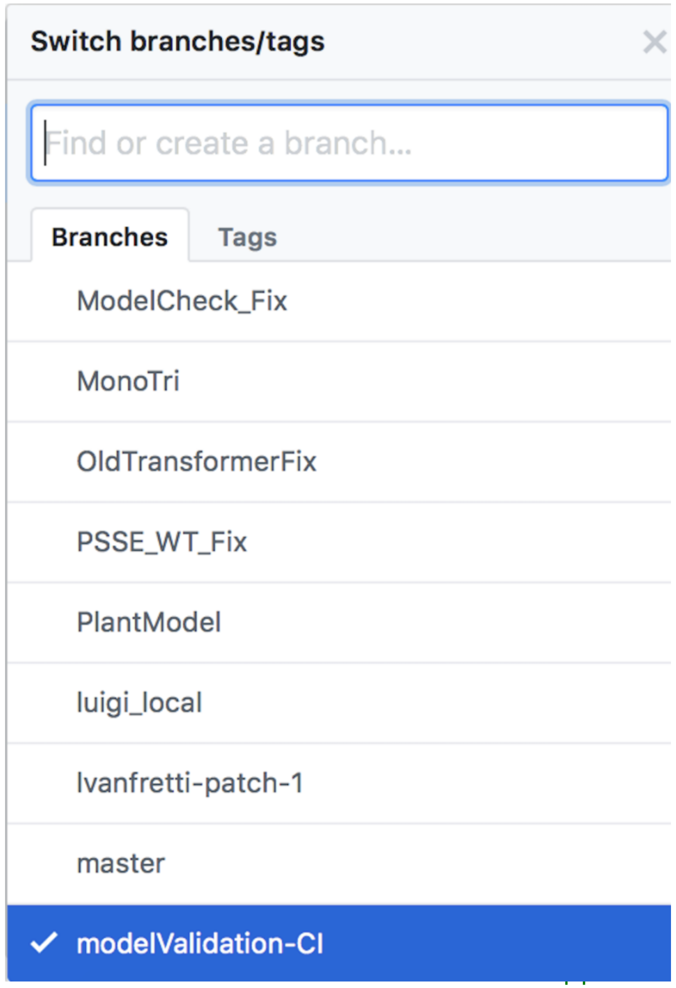


The *OpenIPSL* – Ongoing Developments (CI+R)

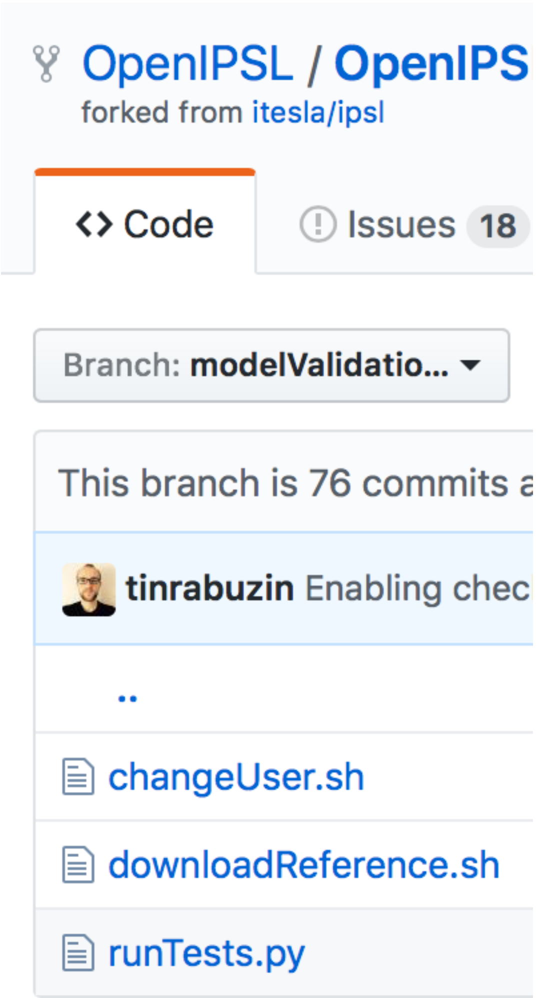
- CI + Regression Testing
- A two-stage process
  - Modelica ***syntax*** check (against Modelica language implementation in OM)
  - Model ***validation*** check (against reference simulation results of "trusted" model)
- Fully automated through online CI services
- Diagnostic help to the developers to ***locate the error!***



Prototype Implementation in the "modelValidation-CI" branch



The *OpenIPSL* – Ongoing Developments (CI+R)

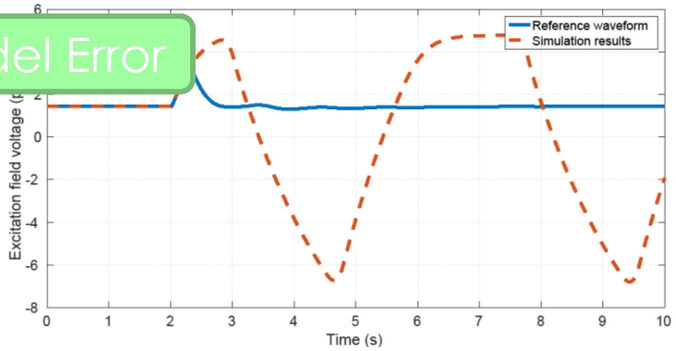


Syntax Error

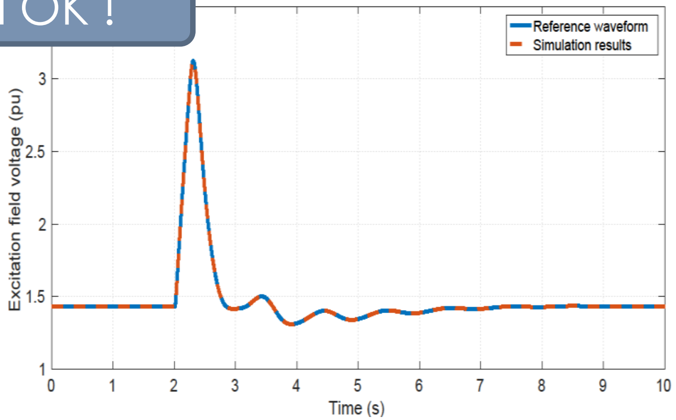
```
penIPSL/Examples/Controls/PSSE/ES/IEEEX1.mo
:80:readonly] Error: Variable iEEEX1_1: In
modifier (KA = 75), class or component KA not found
    in <OpenIPSL.Electrical.Controls.PSSE.ES.
    IEEEX1$iEEEX1_1>.
Error: Error occurred while flattening model OpenIPSL.
    Examples.Controls.PSSE.ES.IEEEX1
```

OR

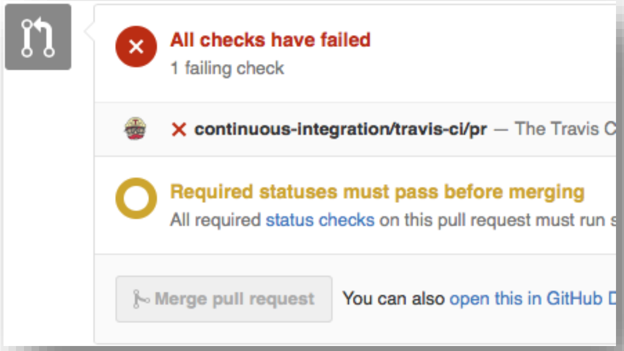
Model Error

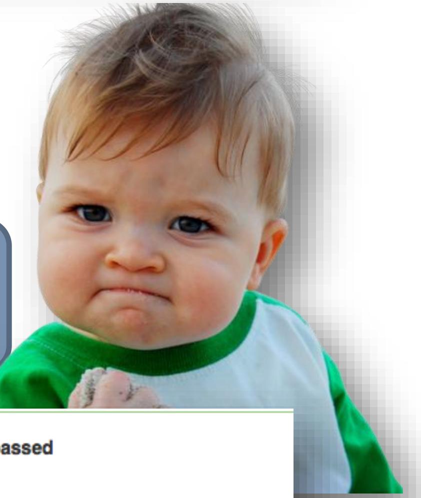


All OK !

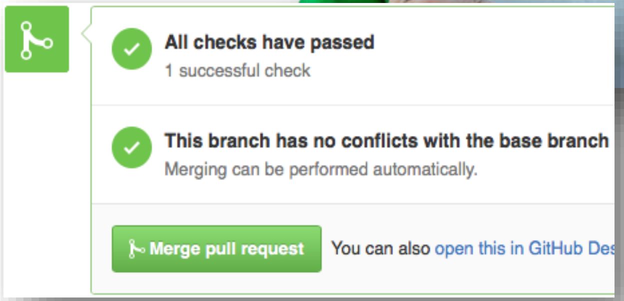


Merging Blocked



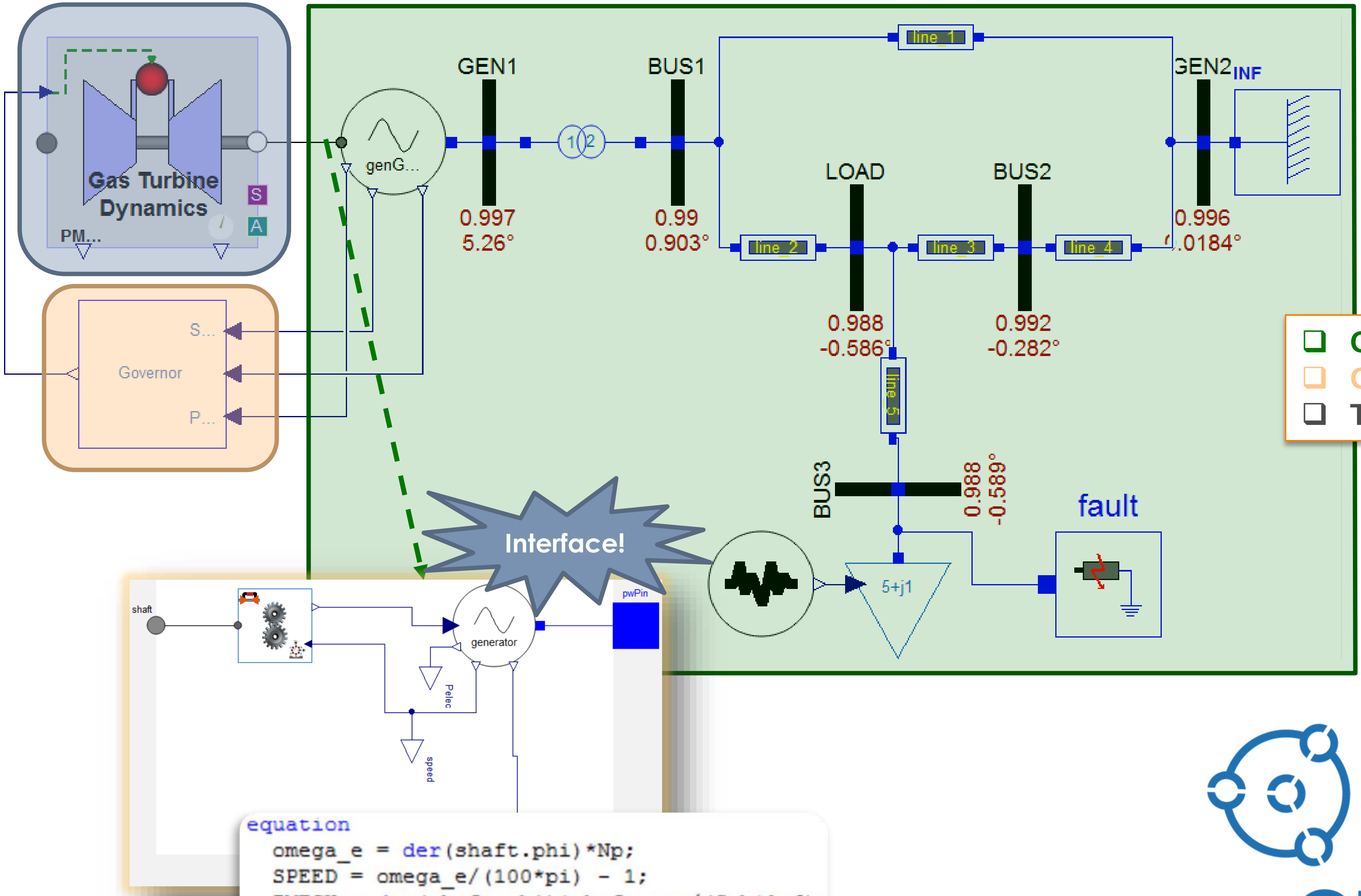


Merging Allowed



CURENT

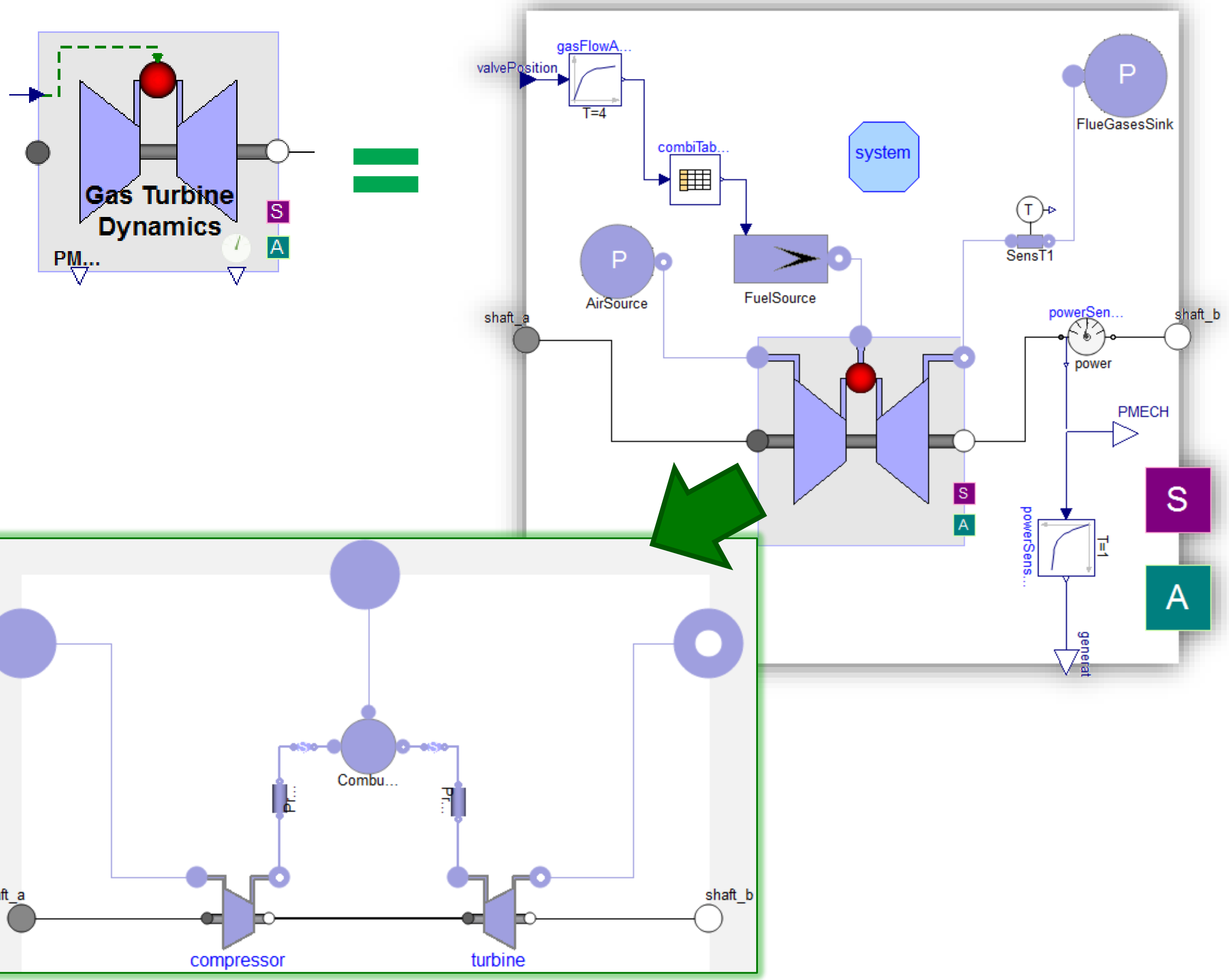
Multi-domain modeling for gas turbines and power systems




```
equation
  omega_e = der(shaft.phi)*Np;
  SPEED = omega_e/(100*pi) - 1;
  PMECH = der(shaft.phi)*shaft.tau/(S_b*1e6);
```

- OpenIPSL
- OpenIPSL
- ThermoPower

Multi-domain modeling for gas turbines and power systems



- OpenCPS_D53B
  - TurboMachineryDomain
    - GTArrangements
    - GTModels
    - Tests
  - PowerSystemDomain
    - Generation_Groups
    - Controls
    - NoiseInjections
    - Networks
    - Breakers
    - Loads
    - Tests
  - MultiDomain
    - Common
    - Generation_Groups
    - SMIB
    - Sync
    - OpenLoopTests

CURENT

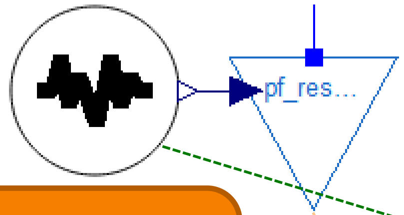
openCPS

ITEA 3

EUREKA

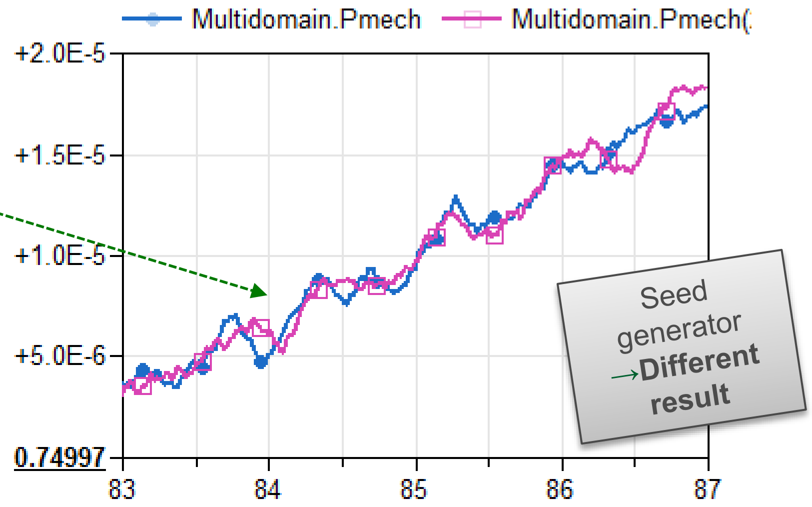
innovation across borders

Stochastic Load Model and Influence in Single-Domain and Multi-Domain Model Response

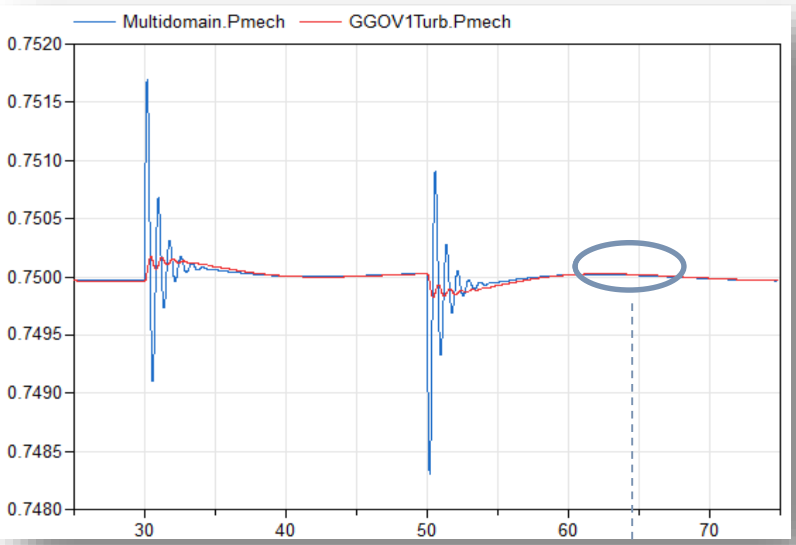


Noise Model
- Expectation value
- Standard deviation
- Sample Period

Sine wave or **ramp** containing the noise can be used to model the “normal” load variation

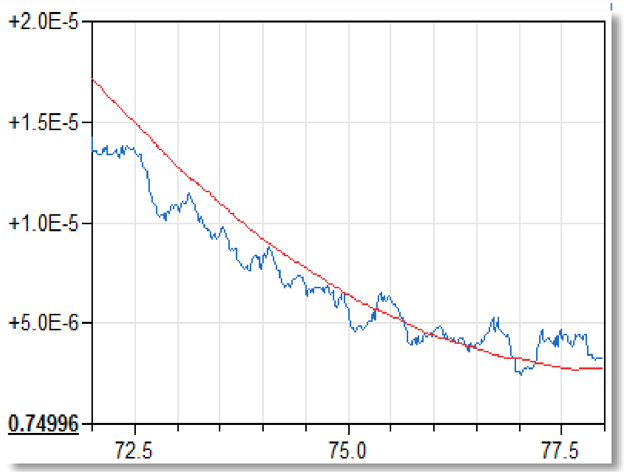


Mechanical Power




```
model VariableLoad "PSS/E Load with variation"
…
  parameter Real d_P "Active Load Variation
…
public
  Modelica.Blocks.Interfaces.RealInput u;
equation
  if time >= t1 and time <= t1 + d_t then
    kI*S_I.re*v + S_Y.re*v^2 + kP*(S_P.re + d_P) + u = p.vr*p.ir + p.vi*p.ii;
    kI*S_I.im*v + S_Y.im*v^2 + kP*(S_P.im + d_Q) = (-p.vr*p.ii) + p.vi*p.ir;
  else
    kI*S_I.re*v + S_Y.re*v^2 + kP*S_P.re + u = p.vr*p.ir + p.vi*p.ii;
    kI*S_I.im*v + S_Y.im*v^2 + kP*S_P.im = (-p.vr*p.ii) + p.vi*p.ir;
  end if;
end VariableLoad;
```

Load varies **d_P** in the time interval between **t1** and **t1+d_t**.
Noise model can be added as real input **u**.

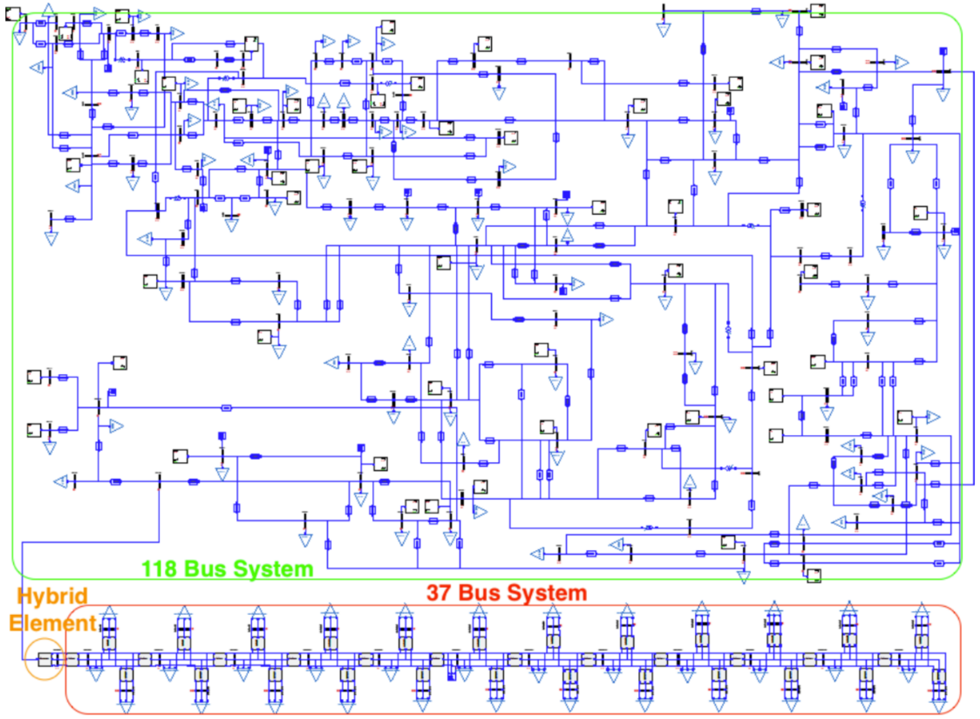


CURENT

Joint modeling and simulation of transmission and distribution power networks

- Work together with Marcelo de Castro Fernandez , Prof. Janaina Gonçalvez (UFJF, Brazil) and Maxime Baudette


- Hybrid single-phase three- phase model for power flow simulation using Modelica as modeling language. The formulation of such model was proposed by Jose Mauro Marinho and Glauco Nery Taranto in the paper:
  - [ref] Jose Mauro T. Marinho and Glauco Nery Taranto. A Hybrid Three-Phase Single-Phase Power Flow Formulation Published in: IEEE Transactions on Power Systems (Volume: 23, Pages: 1063:1070, Issue: 3, Aug. 2008)



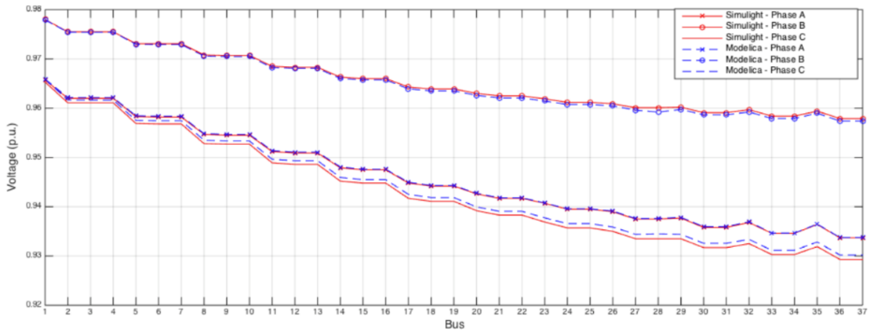
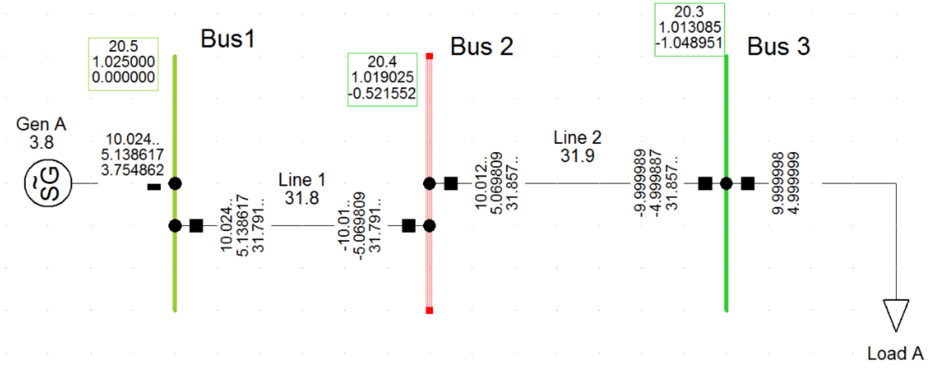


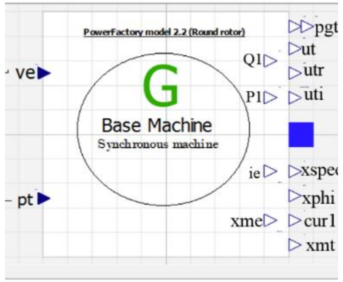
Figure 4.4: Results for the 118 plus 37 bus test using OpenModelica and Simulight.

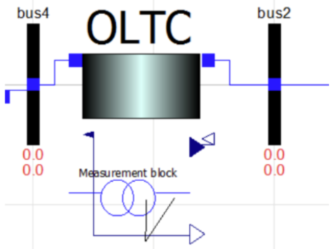
Implementation of DigSilentPowerFactory Component Models

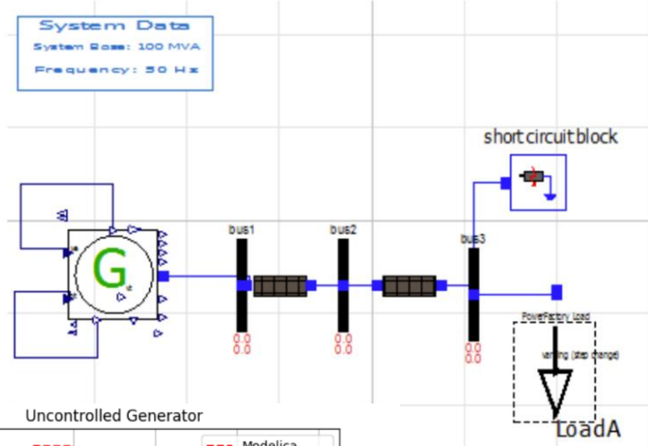
MSc Thesis work by Harish Krishnappa
H.Krishnappa@student.tudelft.nl
IEPG, TU Delft

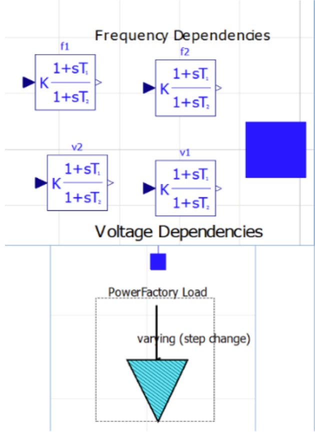


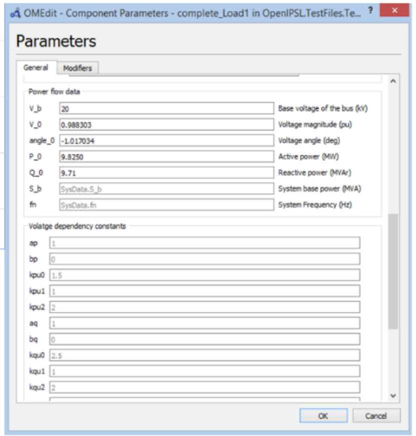
- Models include:
  - Synchronous Gen.
  - Loads
  - Transformer
  - Transmission Line
  - Exciter
  - OEL
  - Speed governor
  - Steam turbine
  - OLTC
  - Induction motor

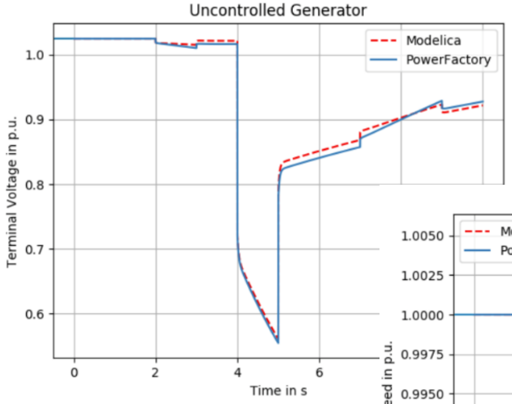


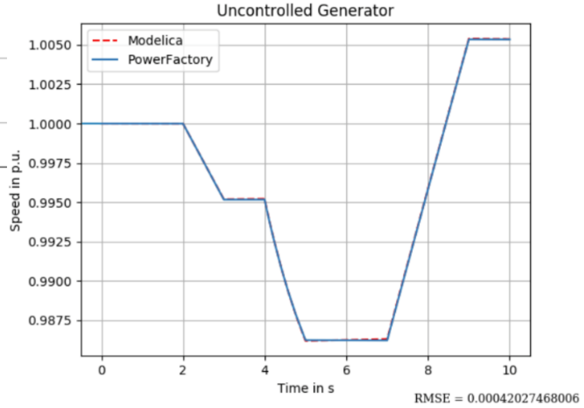










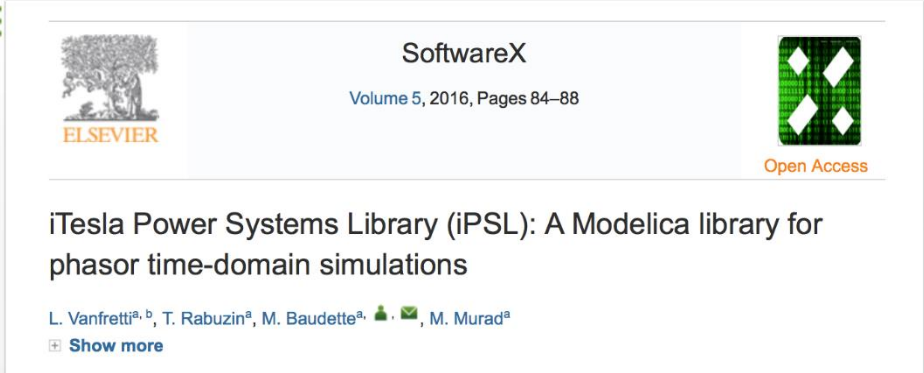


The **OpenIPSL** can be found online

- http://openipsl.org

Our work on **OpenIPSL** has been published in the SoftwareX Journal:

- http://dx.doi.org/10.1016/j.softx.2016.05.001



ELSEVIER

SoftwareX

Volume 5, 2016, Pages 84–88

Open Access

iTesla Power Systems Library (iPSL): A Modelica library for phasor time-domain simulations

L. Vanfretti[a, b], T. Rabuzin[a], M. Baudette[a], M. Murad[a]

Show more

ELSEVIER

SoftwareX

Available online 25 August 2016

In Press, Corrected Proof — Note to users

Open Access

RaPId: A modular and extensible toolbox for parameter estimation of Modelica and FMI compliant models

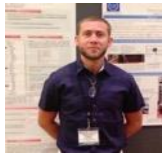
RaPId, a **system identification** software that uses OpenIPSL can be found at:

- https://github.com/ALSETLab/RaPId
- http://dx.doi.org/10.1016/j.softx.2016.07.004

Luigi Vanfretti

Achour Amazouz

Mohammed Ahsan Adib Murad



Francisco José Gómez

Giusseppe Laera

Tin Rabuzin

Jan Lavenius

Le Qi

Maxime Baudette

Mengjia Zhang

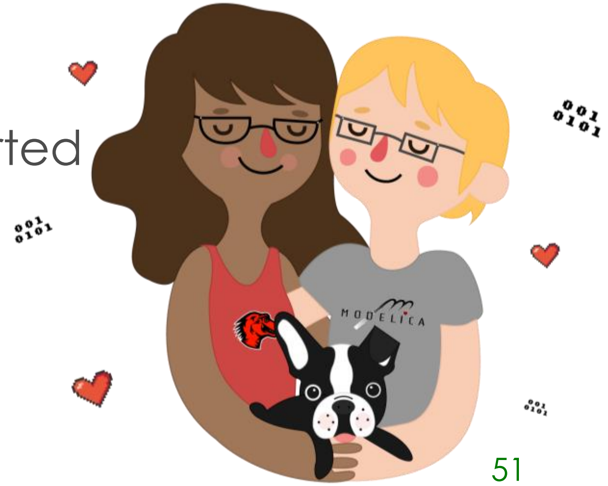
Tetiana Bogodorova

Joan Russiñol Mussons

Marcelo Castro

Miguel Aguilera

Thanks to all my current and former students, friends and developers that have supported the effort!



Part 2: (Phillip Top)

Applications of the OpenIPSL library and the FMI in GRIDDYN

GRIDDYN