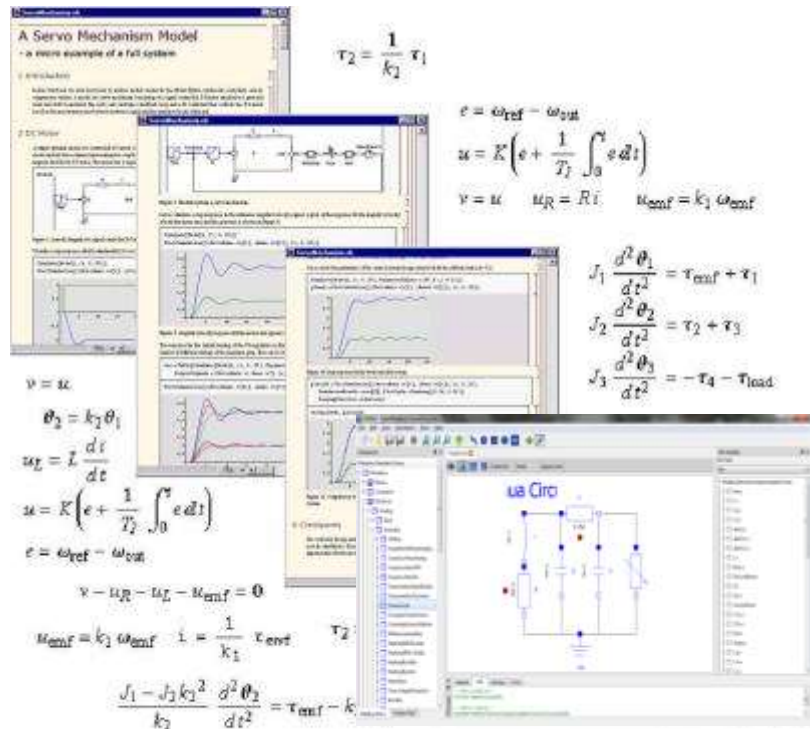


Introduction to Object-Oriented Modeling and Simulation with Modelica and OpenModelica



Lecture 1

Peter Fritzson

Professor em. at Linköping University, peter.fritzson@liu.se

Research Director at Programming Environments Lab

Vice Director of the Open Source Modelica Consortium

Vice Director of the MODPROD Center for Model-based Development

Slides

Based on book and lecture notes by Peter Fritzson

Contributions 2004-2005 by Emma Larsdotter Nilsson, Peter Bunus

Contributions 2006-2018 by Adrian Pop and Peter Fritzson

Contributions 2009 by David Broman, Peter Fritzson, Jan Brugård, and Mohsen Torabzadeh-Tari

Contributions 2010 by Peter Fritzson

Contributions 2011 by Peter F., Mohsen T., Adeel Asghar,

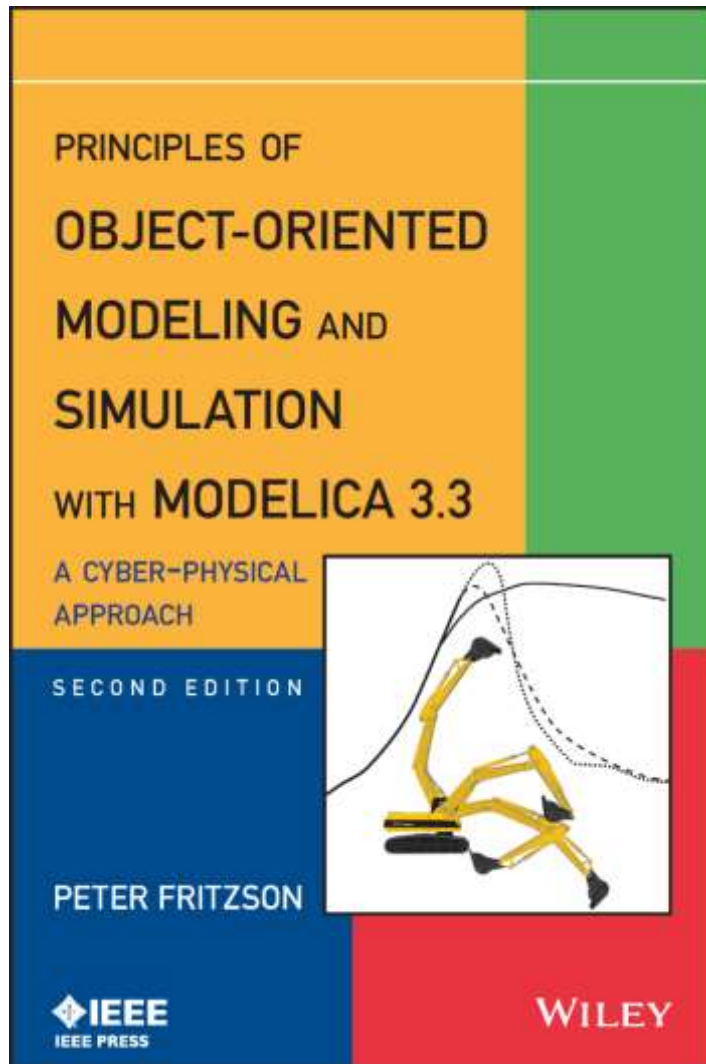
Contributions 2012-2018 by Peter Fritzson, Lena Buffoni, Mahder

Gebremedhin, Bernhard Thiele, Lennart Ochel

Contributions 2019-2023 by Peter Fritzson, Arunkumar Palanisamy, Bernt Lie, Adrian Pop

Tutorial Based on Book, December 2014

Download OpenModelica Software



Peter Fritzson

Principles of Object Oriented Modeling and Simulation with Modelica 3.3

A Cyber-Physical Approach

Can be ordered from Wiley or Amazon

Wiley-IEEE Press, 2014, 1250 pages

- OpenModelica
 - www.openmodelica.org
- Modelica Association
 - www.modelica.org

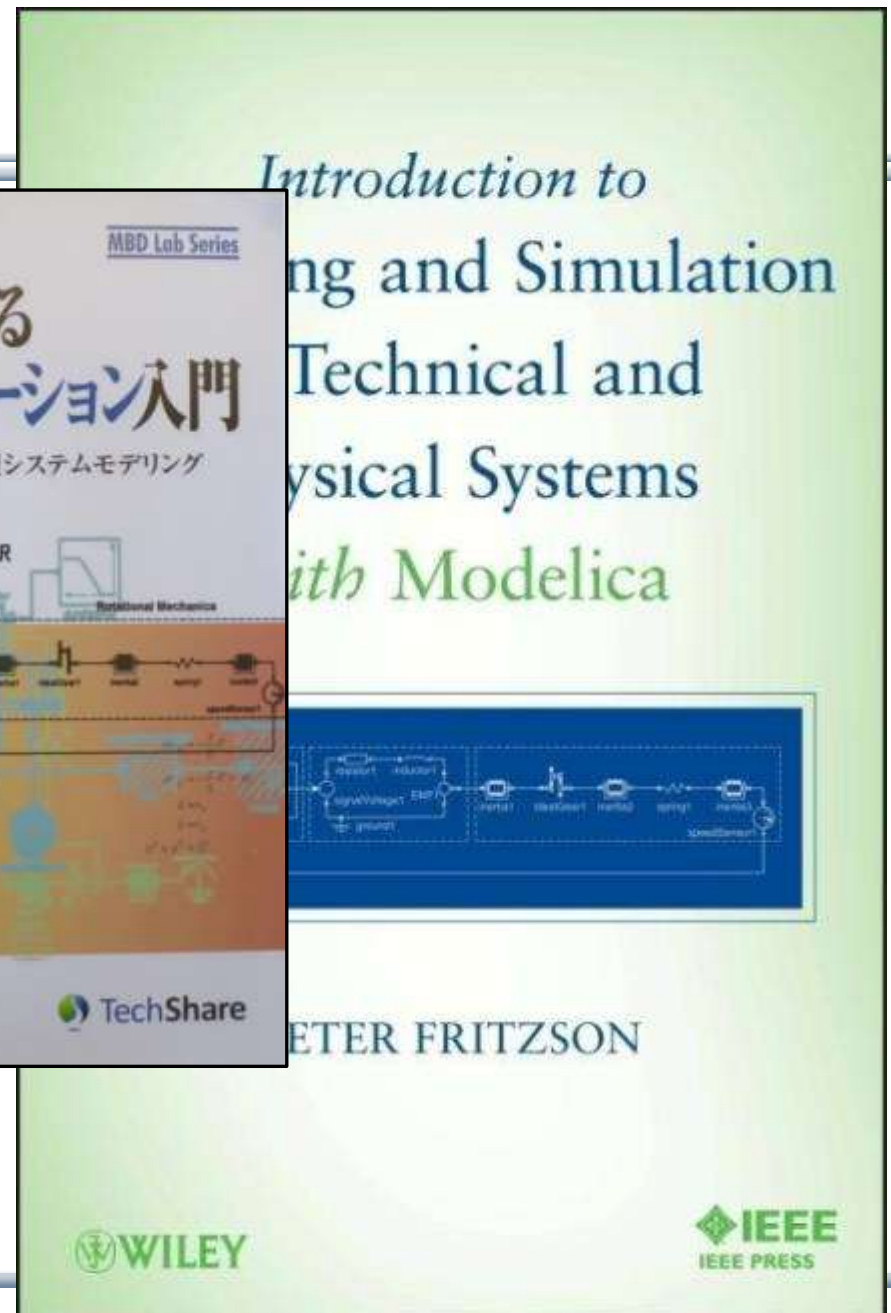
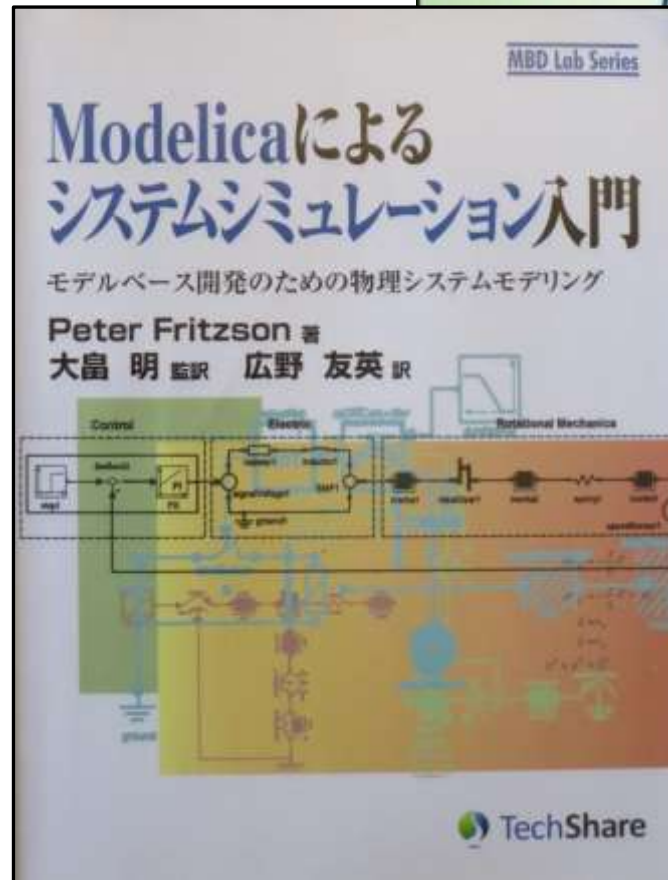
Introductory Modelica Book

September 2011
232 pages

Translations
available in
Chinese,
Japanese,
Spanish

Wiley
IEEE Press

**For Introductory
Short Courses on
Object Oriented
Mathematical Modeling**



Acknowledgements, Usage, Copyrights

- If you want to use the Powerpoint version of these slides in your own course, send an email to: peter.fritzson@ida.liu.se
- The slides are free to use under Creative Commons with attribution CC-BY
- Thanks to Emma Larsdotter Nilsson, Peter Bunus, David Broman, Jan Brugård, Mohsen-Torabzadeh-Tari, Adeel Asghar, Lena Buffoni, etc., for contributions to these slides.
- Most examples and figures in this tutorial are adapted with permission from Peter Fritzson's book "Principles of Object Oriented Modeling and Simulation with Modelica 2.1", copyright Wiley-IEEE Press
- Some examples and figures reproduced with permission from Modelica Association, Martin Otter, Hilding Elmqvist, Wolfram MathCore, Siemens
- Modelica Association: www.modelica.org
- OpenModelica: www.openmodelica.org

Detailed Schedule Day 1

Day 1, 9.00-16.00 (including lunch and short breaks)

12.00 -13.15 lunch break.

10.00 -10.15; 11.00 -11.15 and 14.00 -14.15, 15-15.15 small breaks.

Lecturers: Peter Fritzson

Lecture 1 Introduction to Modeling and Simulation with Modelica and OpenModelica

Lecture Introduction to Modelica and OpenModelica

- Demo+short exercise: Graphic modeling with OMEdit
- OpenModelica OMNotebook and OMWebbook usage
- Introduction to textual modeling
- Demo+Exercise: OMNotebook, DrModelica, OMWebbook, Spokentutorial

Lecture Modelica Classes, Inheritance and Equations

- Lecture+Exercises: classes and inheritance
- Exercise01-classes-simple-textual.onb
- Lecturing on Modelica equations.

Lecture Debugging and Performance Analysis

- Lecture and exercises

Lecture – Modelica Connectors, Packages and Libraries

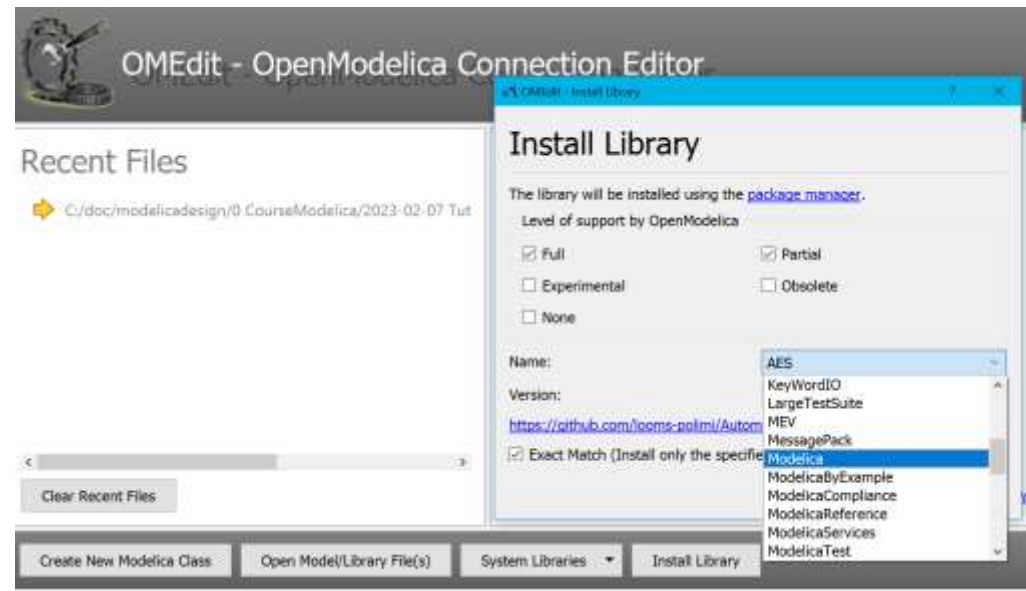
- Lecturing+Exercises: Component connectors and connections, graphical modeling
- Exercise02-graphical-modeling.onb
- Lecturing on Modelica packages and libraries

Lecture – 3D Visualization and animation

- Lecturing+Exercises: MSL Multi-Body 3D visualization using OMEdit

Software Installation - Windows

- Start the software installation
- Install OpenModelica-1.20.0 or later Download from www.openmodelica.org (takes about 20min)
- You also need to load the Modelica standard library if not already loaded:
(push the load library button and select Modelica)



Software Installation – Linux (requires internet connection)

- Go to <https://openmodelica.org/index.php/download/download-linux> and follow the instructions.

Software Installation – MAC (requires internet connection)

- Go to <https://openmodelica.org/index.php/download/download-mac> and follow the instructions or follow the instructions written below.
- The installation uses MacPorts. After setting up a MacPorts installation, run the following commands on the terminal (as root):
 - *echo rsync://build.openmodelica.org/macports/ >> /opt/local/etc/macports/sources.conf # assuming you installed into /opt/local*
 - *port selfupdate*
 - *port install openmodelica-devel*

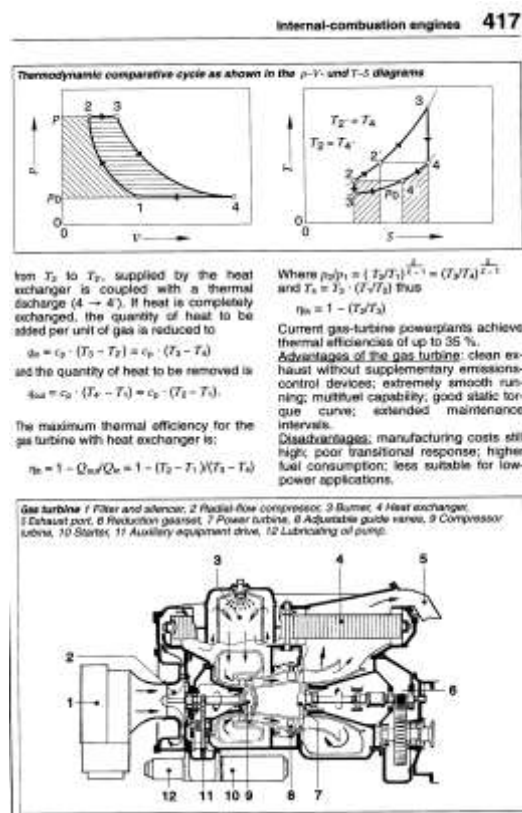
Part I

Introduction to Modelica and a demo example



Modelica Background: Stored Knowledge

Model knowledge is stored in books and human minds which computers cannot access



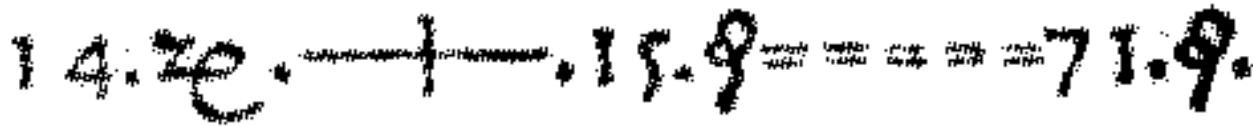
“The change of motion is proportional to the motive force impressed”
– Newton

Lex. II.

Mutationem motus proportionalem esse vi motrici impressae, & fieri secundum lineam rectam qua vis illa imprimitur.

Modelica Background: The Form – Equations

- Equations were used in the third millennium B.C.
- Equality sign was introduced by Robert Recorde in 1557



Newton still wrote text (Principia, vol. 1, 1686)

“The change of motion is proportional to the motive force impressed”

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

variable = expression

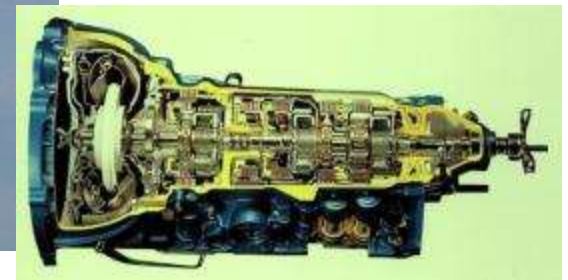
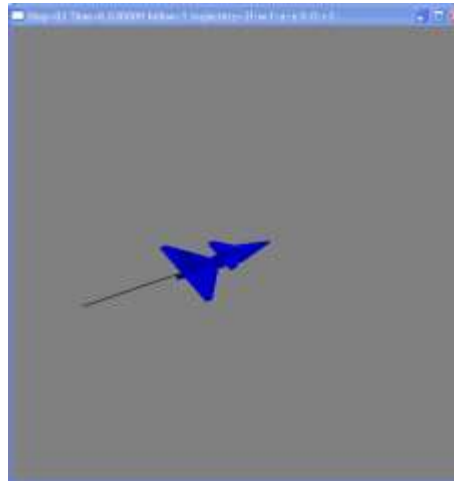
$v = \text{INTEG}(F) / m$

Programming languages usually do not allow equations!

What is Modelica?

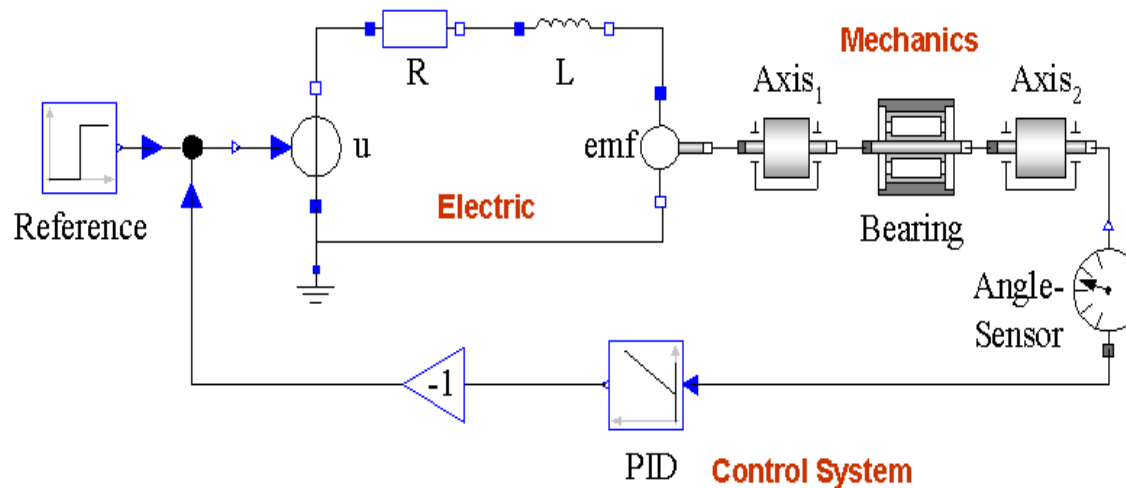
A language for modeling of **complex cyber-physical systems**

- Robotics
- Automotive
- Aircrafts
- Satellites
- Power plants
- Systems biology



What is Modelica?

A language for **modeling** of complex cyber-physical systems



Primary designed for **simulation**, but there are also other usages of models, e.g. optimization.

What is Modelica?

A language for modeling of complex cyber-physical systems

i.e., Modelica is not a tool

Free, open language
specification:



There exist one free and several commercial tools, for example:

- **OpenModelica from OSMC**
(in ABB Optimax, Bosch-Rexr Control Edge Designer, Mike DHI)
- Dymola from Dassault systems
- Wolfram System Modeler from Wolfram MathCore
- SimulationX from ITI, part of ESI Group
- MapleSim from MapleSoft
(also in Altair solidThinking Activate)
- AMESIM from LMS
- Optimica Toolkit from Modelon
(also in ANSYS Simpler, Rickardo tool, etc.)
- MWORKS from Tongyang Sw & Control
- IDA Simulation Env, from Equa

•

Available at: www.modelica.org

*Developed and standardized
by Modelica Association*

Modelica – The Next Generation Modeling Language

Declarative language

Equations and mathematical functions allow acausal modeling, high level specification, increased correctness

Multi-domain modeling

Combine electrical, mechanical, thermodynamic, hydraulic, biological, control, event, real-time, etc...

Everything is a class

Strongly typed object-oriented language with a general class concept, Java & MATLAB-like syntax

Visual component programming

Hierarchical system architecture capabilities

Efficient, non-proprietary

Efficiency comparable to C; advanced equation compilation, e.g. 300 000 equations, ~150 000 lines on standard PC

Modelica Acausal Modeling

What is *acausal* modeling/design?

Why does it increase *reuse*?

The acausality makes Modelica library classes *more reusable* than traditional classes containing assignment statements where the input-output causality is fixed.

Example: a resistor *equation*:

$$R \cdot i = v;$$

can be used in three ways:

$$i := v/R;$$

$$v := R \cdot i;$$

$$R := v/i;$$

What is Special about Modelica?

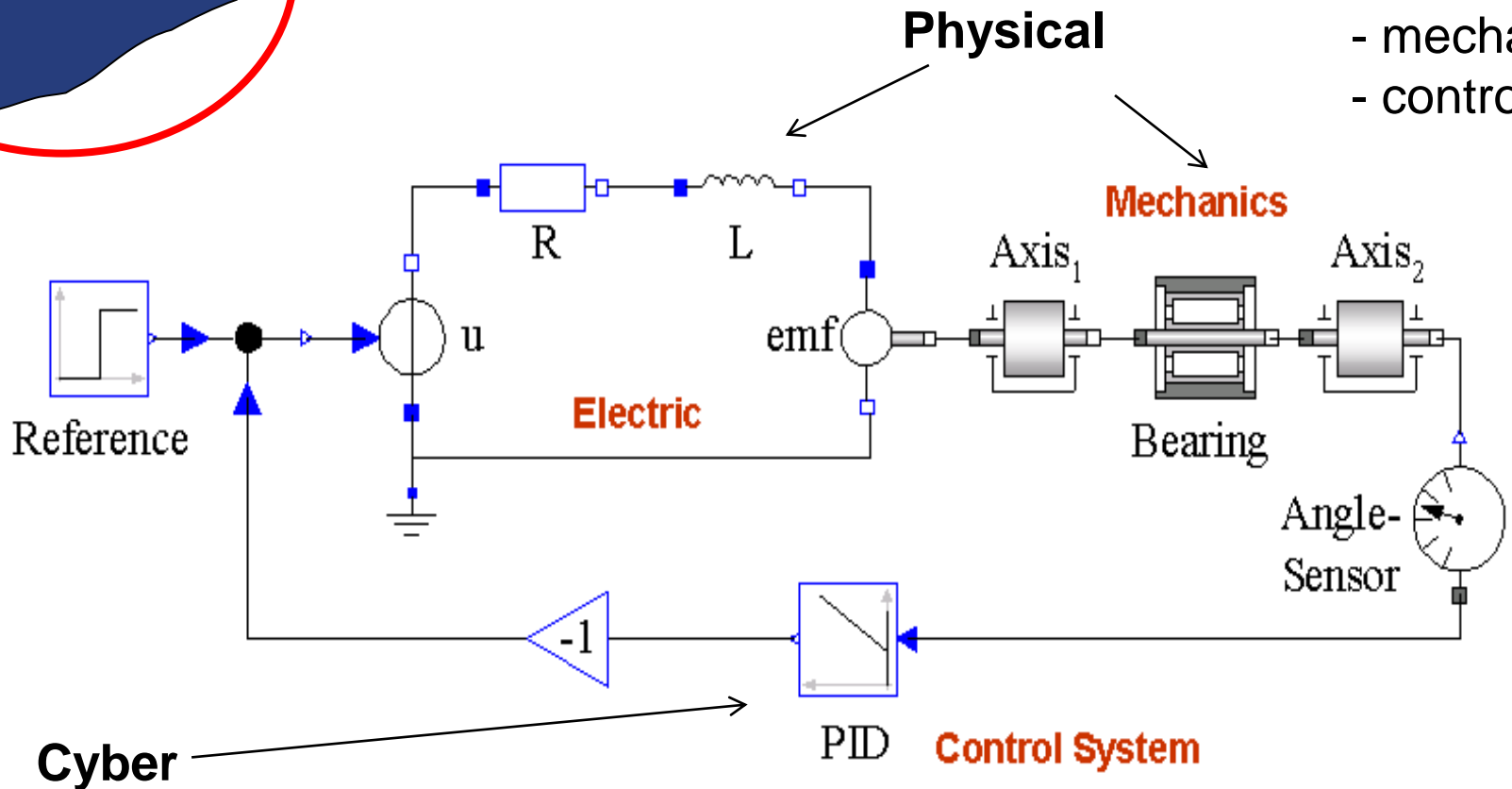
- Multi-Domain Modeling
- Visual acausal hierarchical component modeling
- Typed declarative equation-based textual language
- Hybrid modeling and simulation

What is Special about Modelica?

Multi-Domain
Modeling

Cyber-Physical Modeling

3 domains
- electric
- mechanics
- control



What is Special about Modelica?

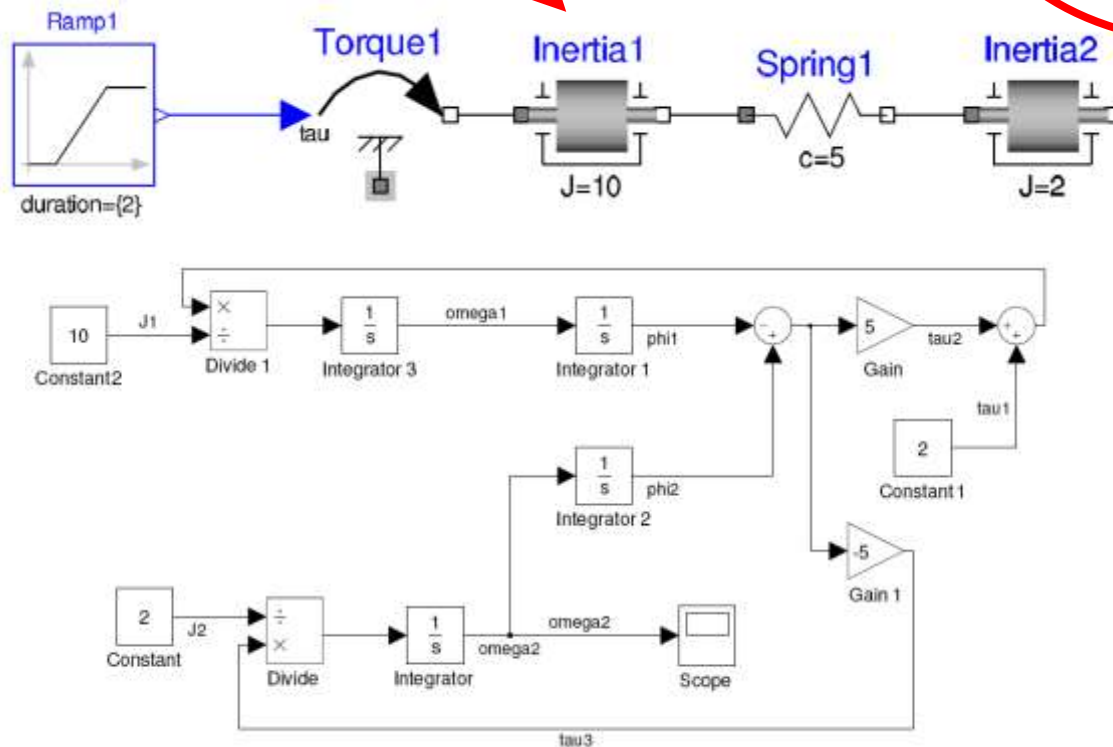
Multi-Domain
Modeling

Acausal model
(Modelica)

Causal
block-based
model
(Simulink)

Keeps the physical
structure

Visual Acausal
Hierarchical
Component
Modeling

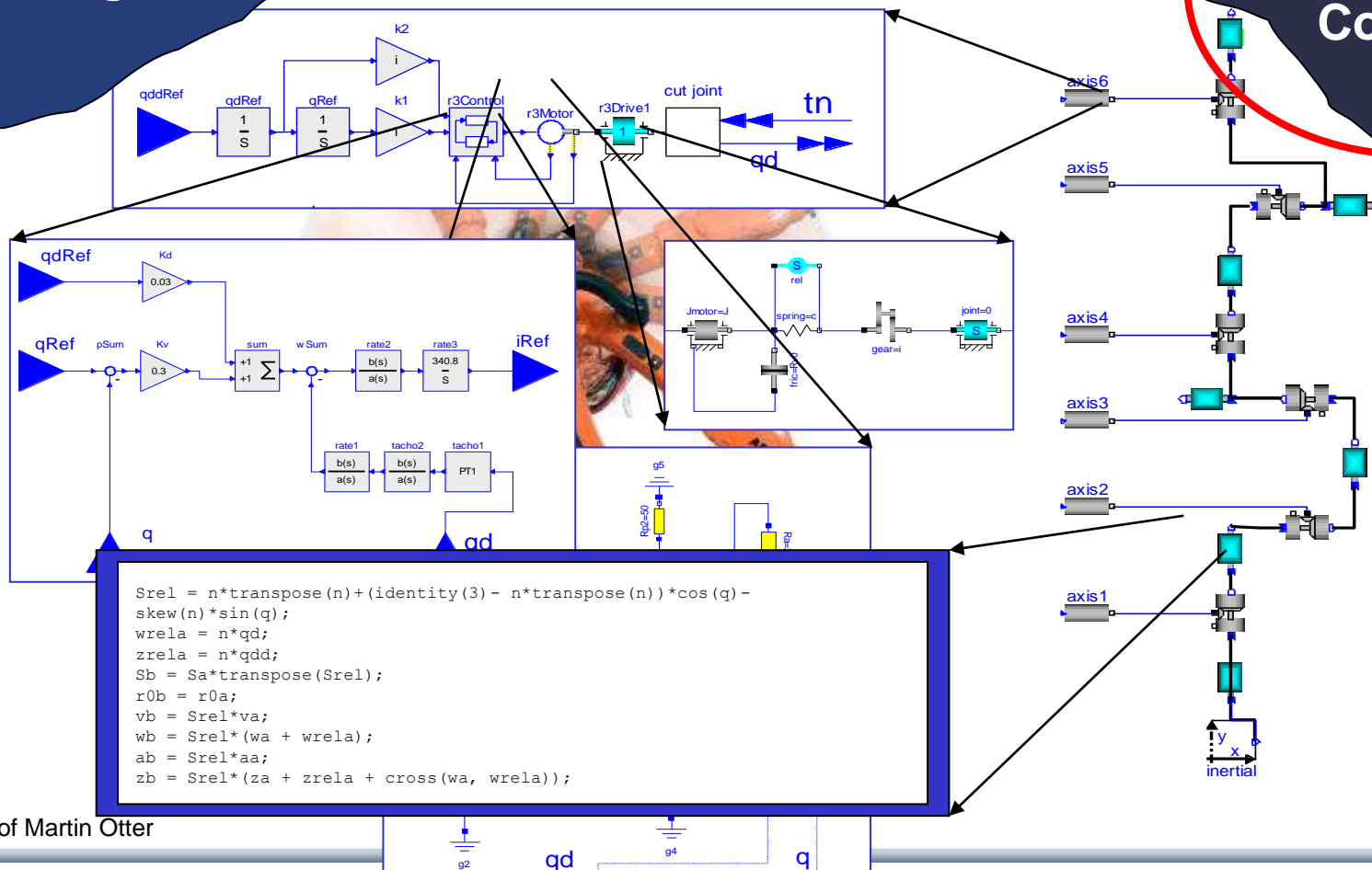


What is Special about Modelica?

Multi-Domain
Modeling

Hierarchical system
modeling

Visual Acausal
Hierarchical
Component
Modeling



Courtesy of Martin Otter

What is Special about Modelica?

Multi-Domain
Modeling

A textual *class-based* language
OO primary used for as a structuring concept

Visual Acausal
Hierarchical
Component
Modeling

Behaviour described declaratively using

- Differential algebraic equations (DAE) (continuous-time)
- Event triggers (discrete-time)

Variable
declarations

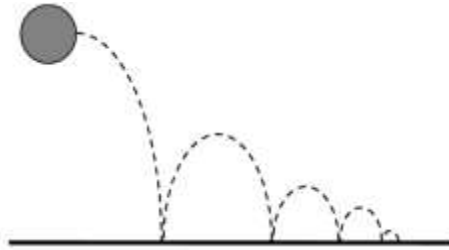
```
class VanDerPol "Van der Pol oscillator model"  
  Real x(start = 1) "Descriptive string for x";  
  Real y(start = 1) "y coordinate";  
  parameter Real lambda = 0.3;  
equation  
  der(x) = y;  
  der(y) = -x + lambda*(1 - x*x)*y;  
end VanDerPol;
```

Differential equations

Typed
Declarative
Equation-based
Textual Language

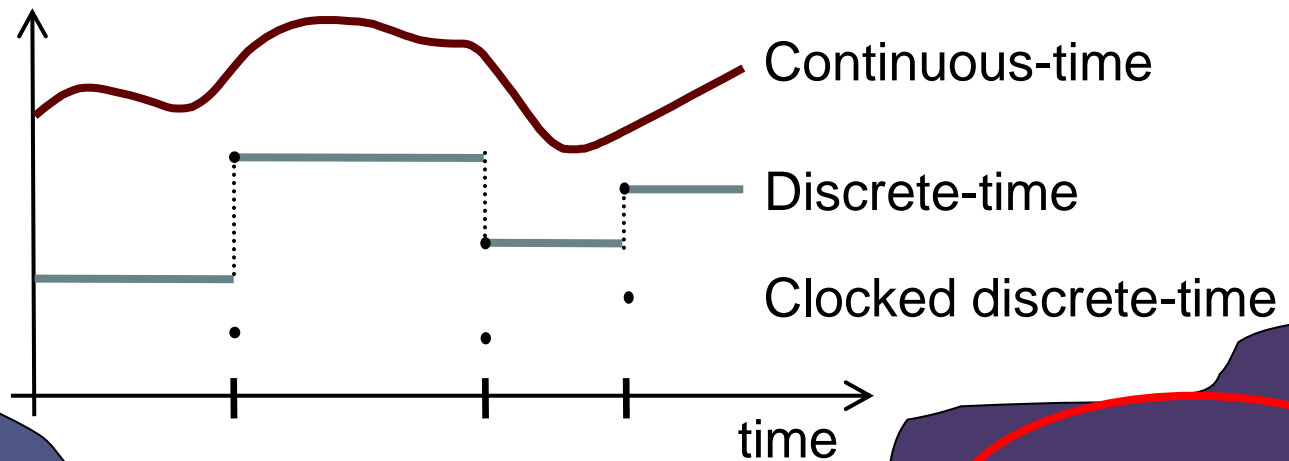
What is Special about Modelica?

Multi-Domain
Modeling



Visual Acausal
Component
Modeling

Hybrid modeling =
continuous-time + discrete-time modeling

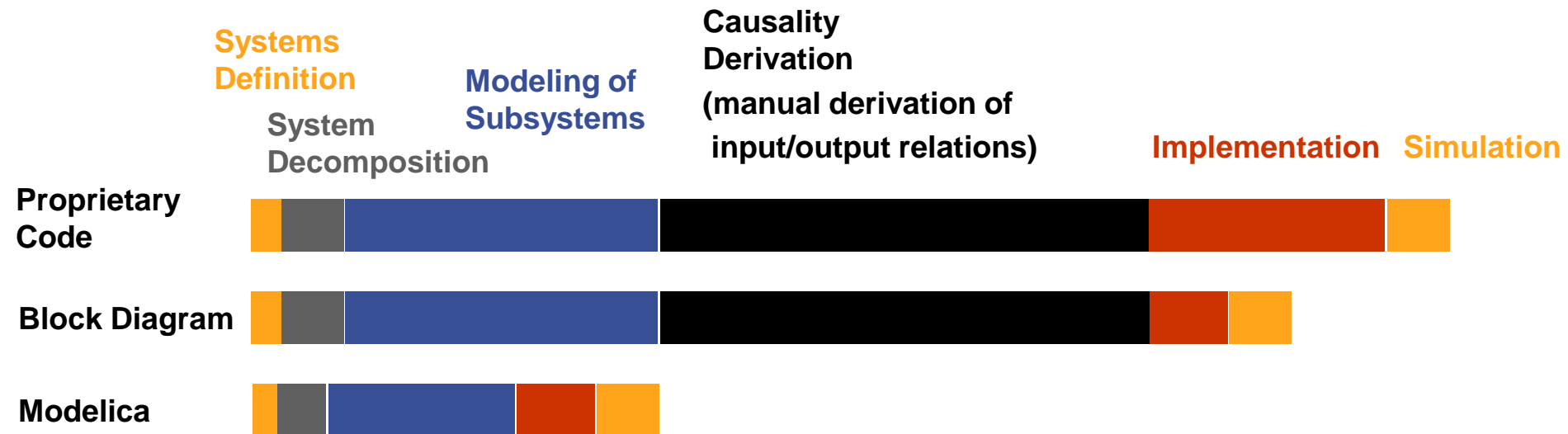


Typed
Declarative
Equation-based
Textual Language

Hybrid
Modeling

Modelica – Faster Development, Lower Maintenance than with Traditional Tools

Block Diagram (e.g. Simulink, ...) or
Proprietary Code (e.g. Ada, Fortran, C,...)
vs Modelica

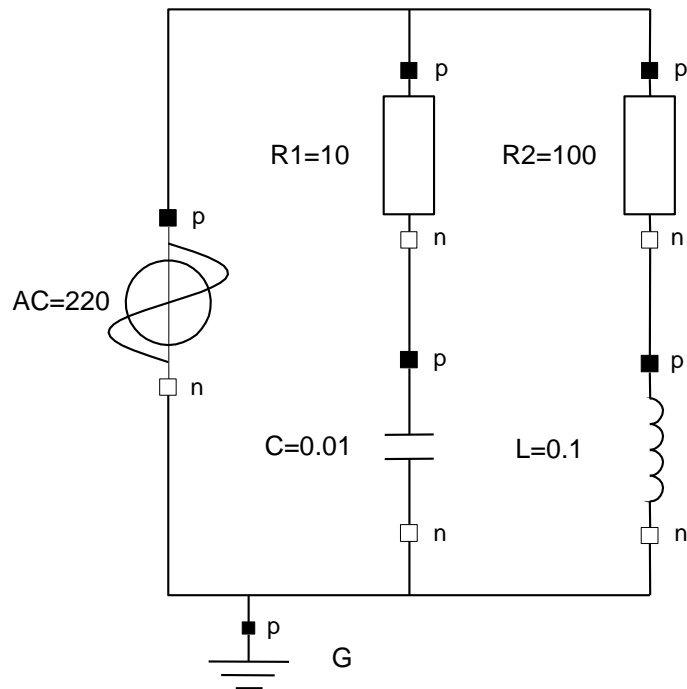


Modelica vs Simulink Block Oriented Modeling

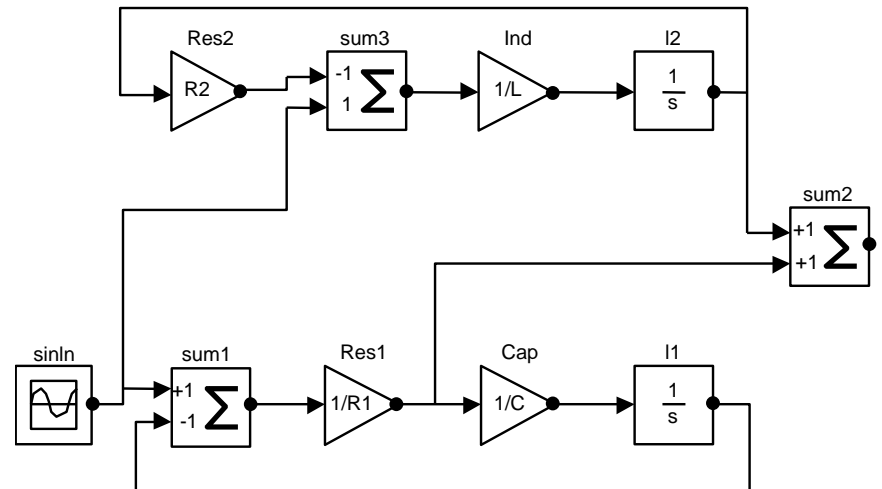
Simple Electrical Model

Modelica:
Physical model –
easy to understand

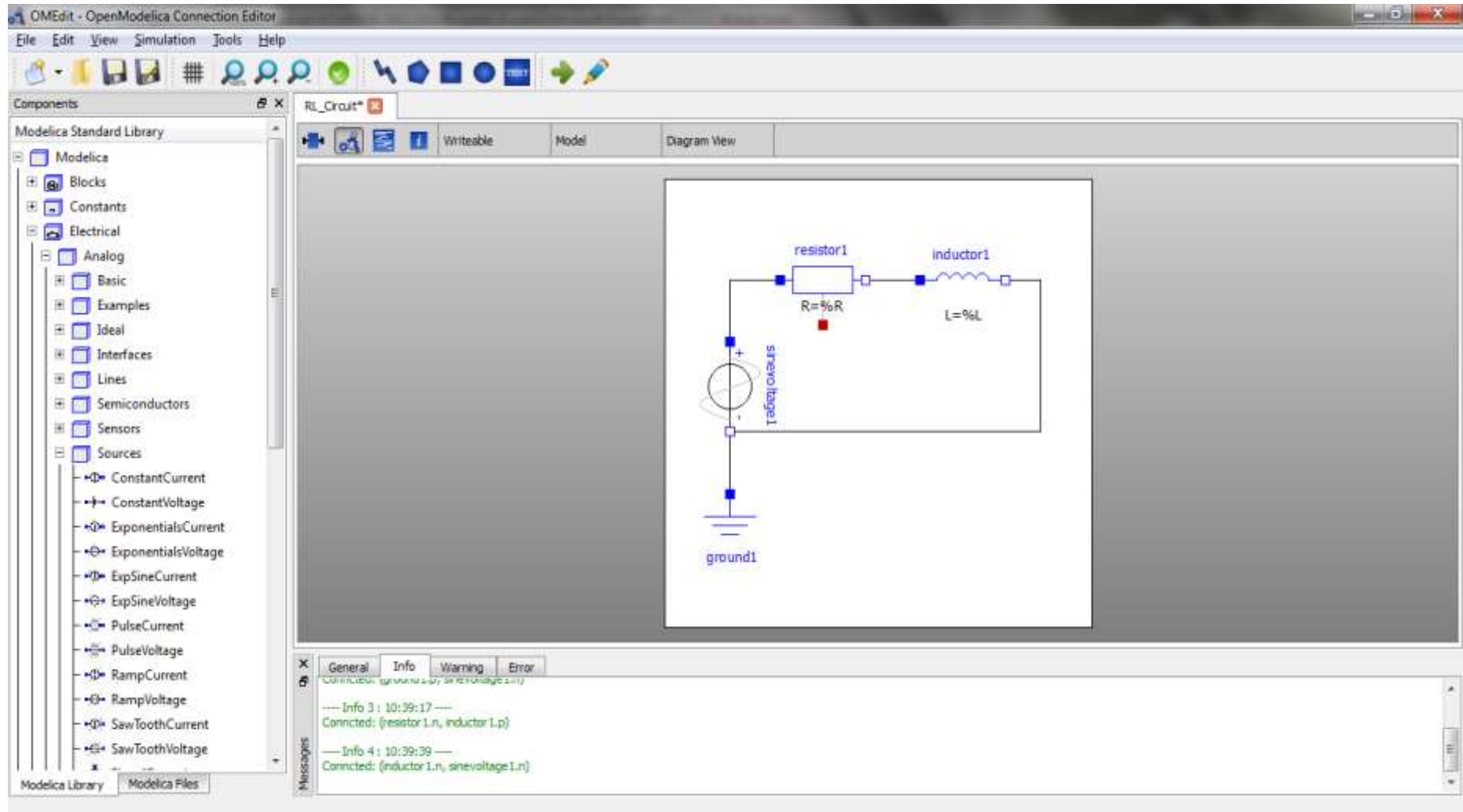
Keeps the
physical
structure



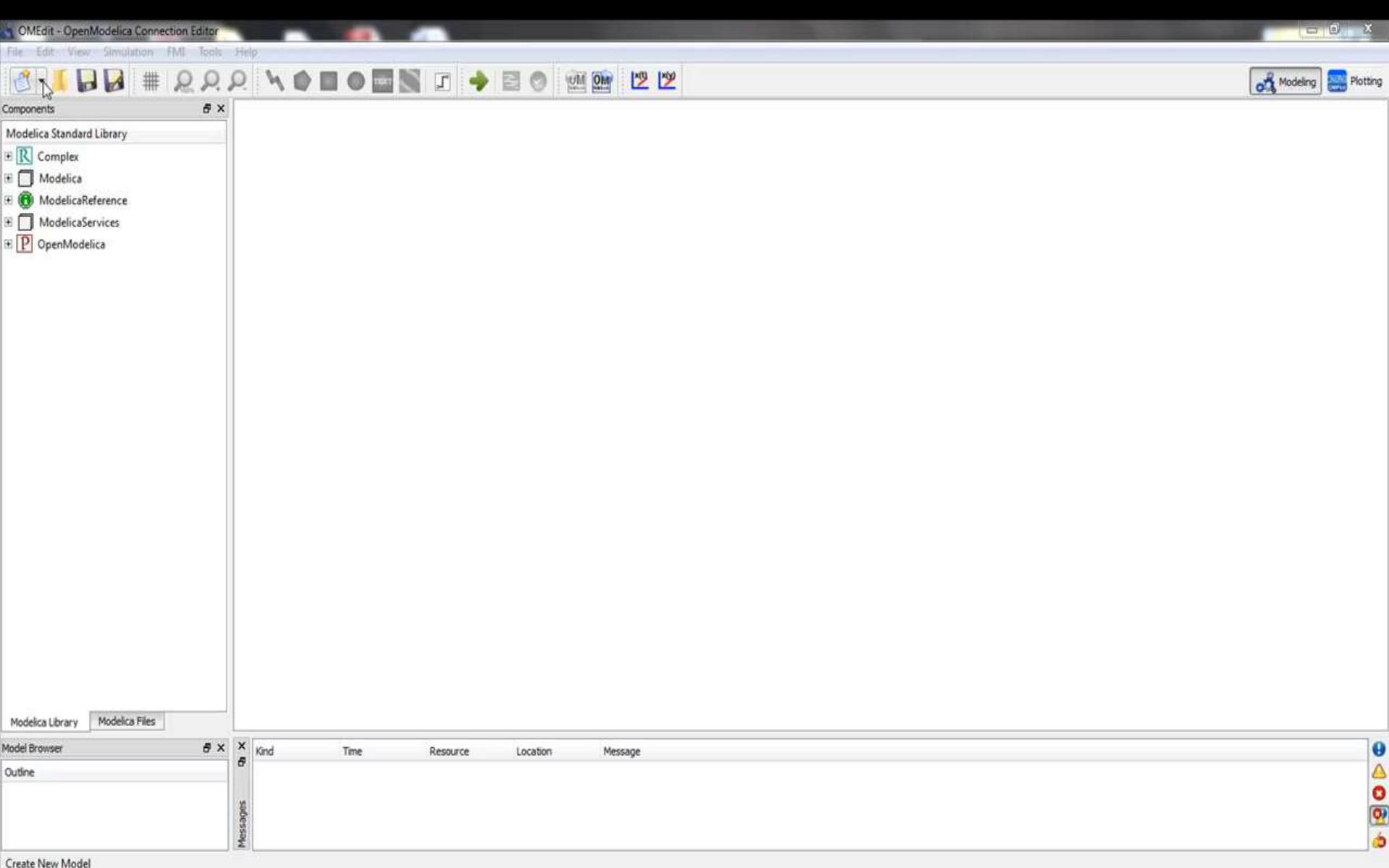
Simulink:
Signal-flow model – hard to
understand



Graphical Modeling - Using Drag and Drop Composition



Graphical Modeling with OpenModelica Environment



Multi-Domain (Electro-Mechanical) Modelica Model

- A DC motor can be thought of as an electrical circuit which also contains an electromechanical component

model DCMotor

Resistor R(R=100);

Inductor L(L=100);

VsourceDC DC(f=10);

Ground G;

ElectroMechanicalElement EM(k=10,J=10, b=2);

Inertia load;

equation

connect (DC.p,R.n);

connect (R.p,L.n);

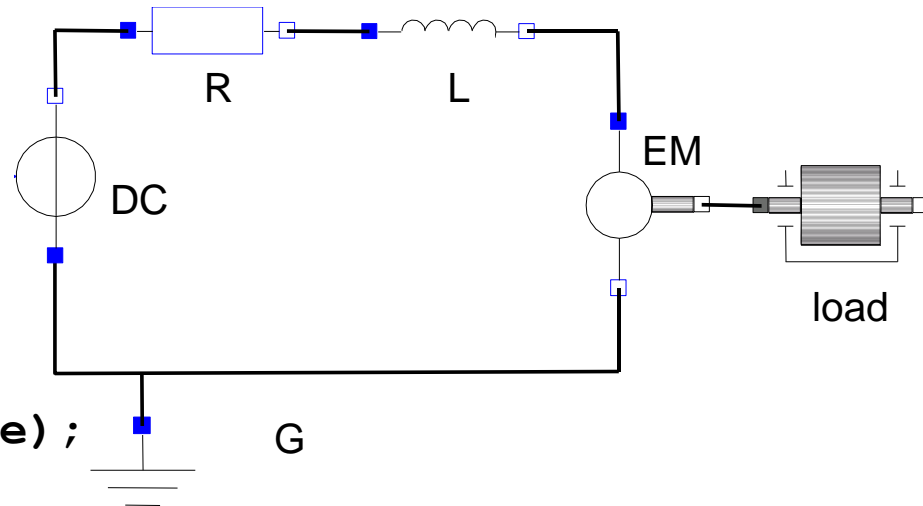
connect (L.p, EM.n);

connect (EM.p, DC.n);

connect (DC.n,G.p);

connect (EM.flange,load.flange);

end DCMotor



Corresponding DCMotor Model Equations

The following equations are automatically derived from the Modelica model:

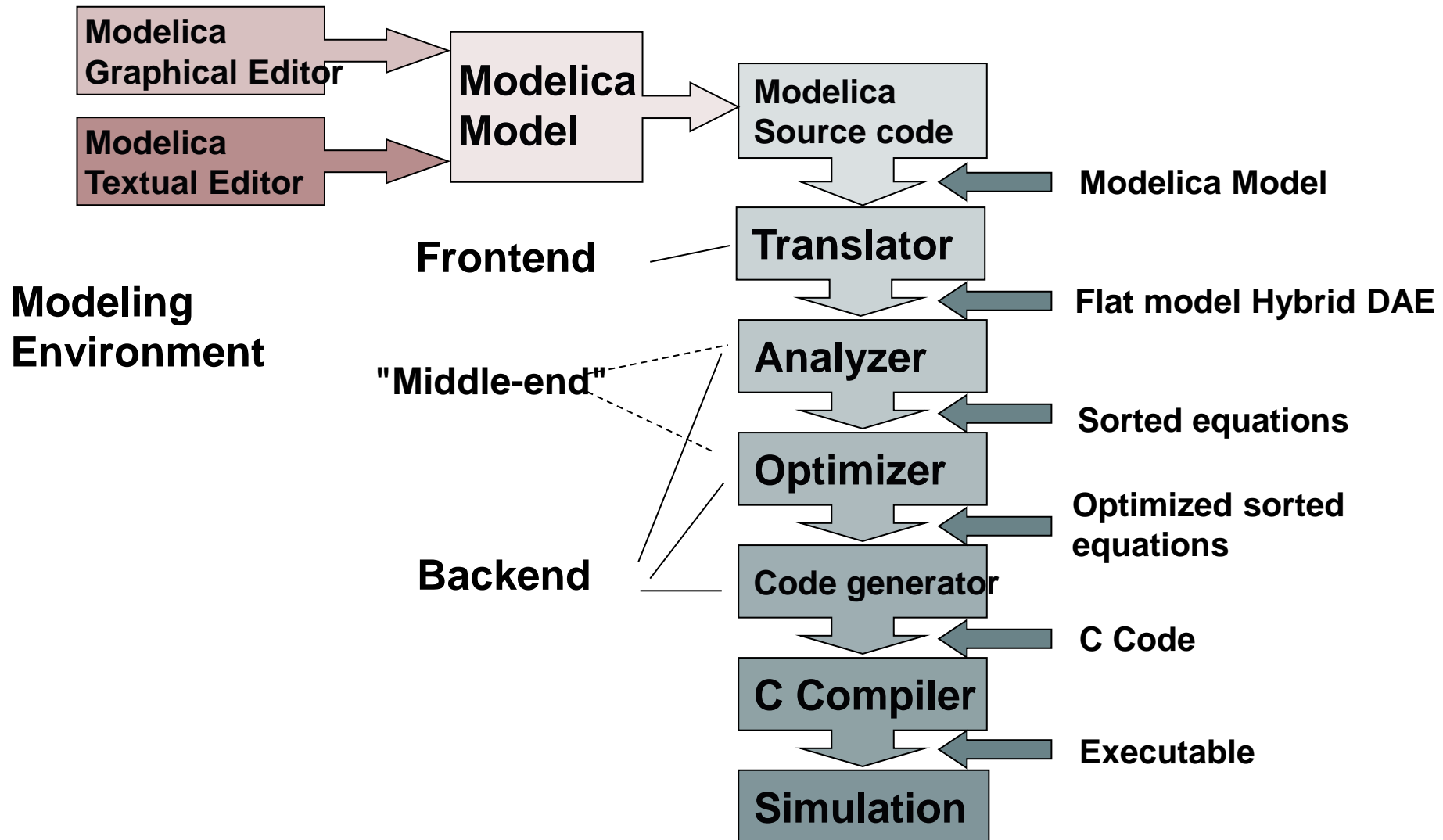
0 == DC.p.i + R.n.i	EM.u == EM.p.v - EM.n.v	R.u == R.p.v - R.n.v
DC.p.v == R.n.v	0 == EM.p.i + EM.n.i	0 == R.p.i + R.n.i
	EM.i == EM.p.i	R.i == R.p.i
0 == R.p.i + L.n.i	EM.u == EM.k * EM.ω	R.u == R.R * R.i
R.p.v == L.n.v	EM.i == EM.M / EM.k	
	EM.J * EM.ω == EM.M - EM.b * EM.ω	L.u == L.p.v - L.n.v
0 == L.p.i + EM.n.i		0 == L.p.i + L.n.i
L.p.v == EM.n.v	DC.u == DC.p.v - DC.n.v	L.i == L.p.i
	0 == DC.p.i + DC.n.i	L.u == L.L * L.i'
0 == EM.p.i + DC.n.i	DC.i == DC.p.i	
EM.p.v == DC.n.v	DC.u == DC.Amp * Sin[2 π DC.f * t]	
0 == DC.n.i + G.p.i		
DC.n.v == G.p.v		

(load component not included)

Automatic transformation to ODE or DAE for simulation:

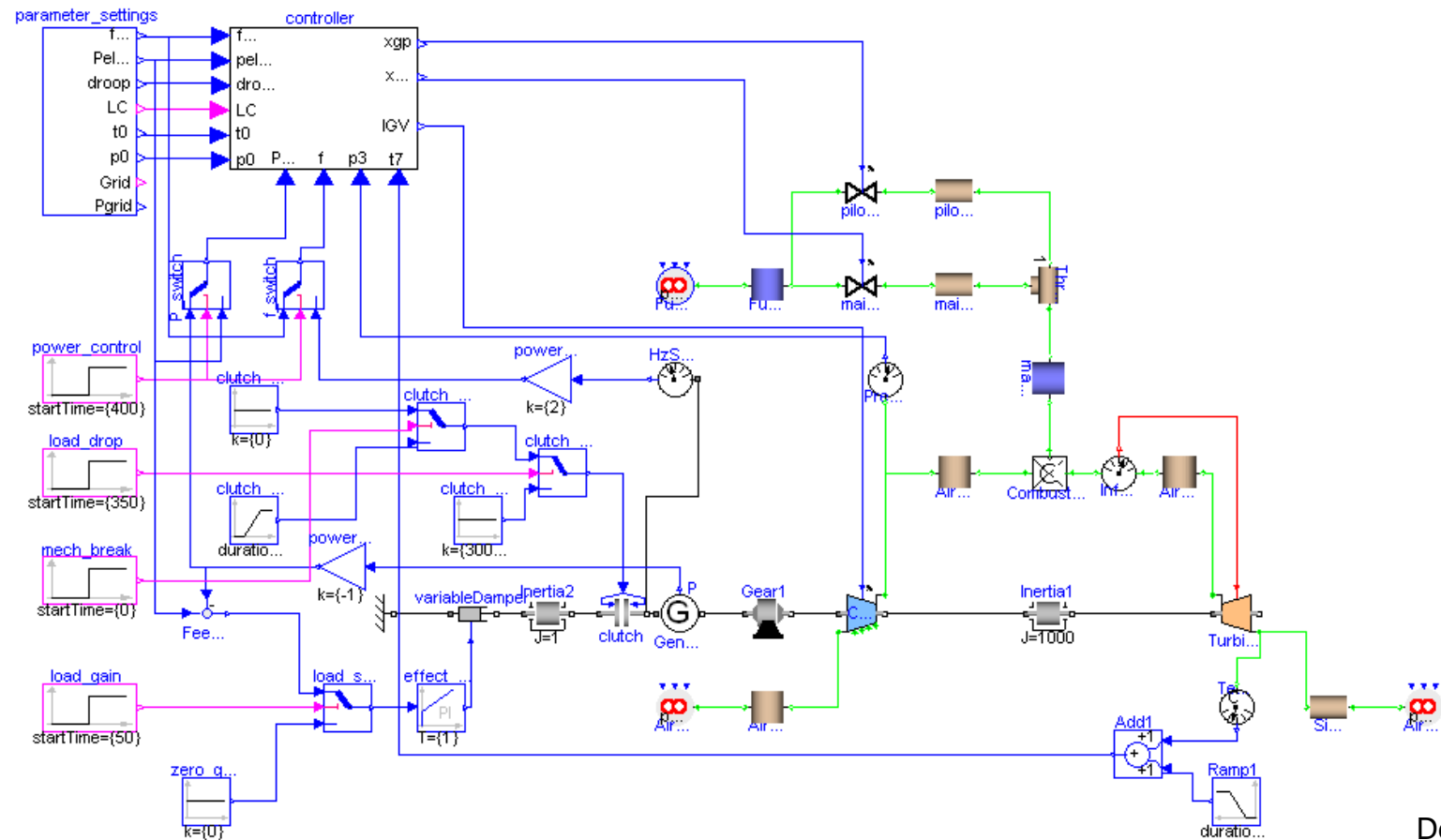
$$\frac{dx}{dt} == f[x, u, t] \quad g\left[\frac{dx}{dt}, x, u, t\right] == 0$$

Model Translation Process to Hybrid DAE to Code



Modelica in Power Generation

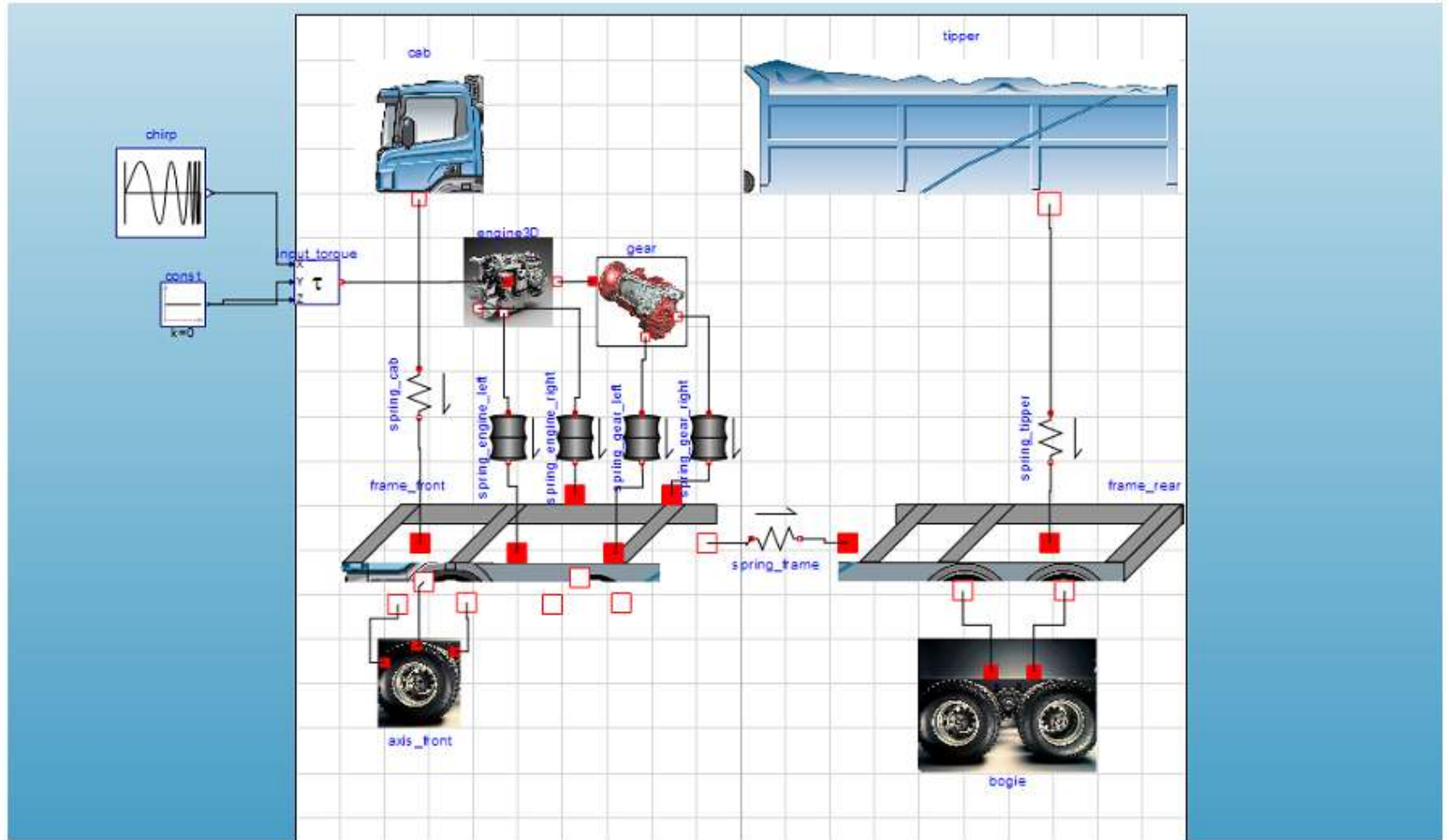
GTX Gas Turbine Power Cutoff Mechanism



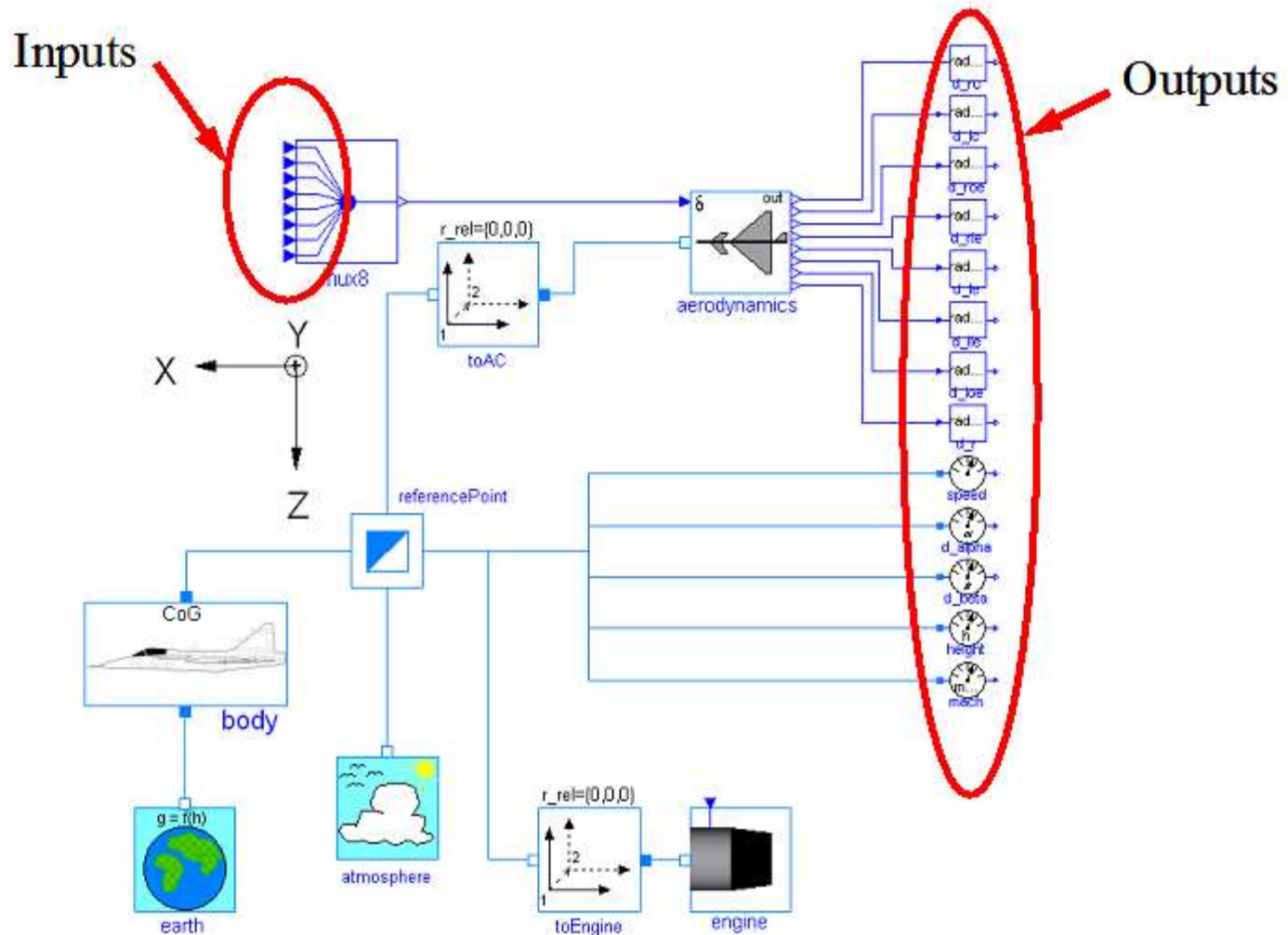
Developed
by MathCore
for Siemens

Courtesy of Siemens Energy AB

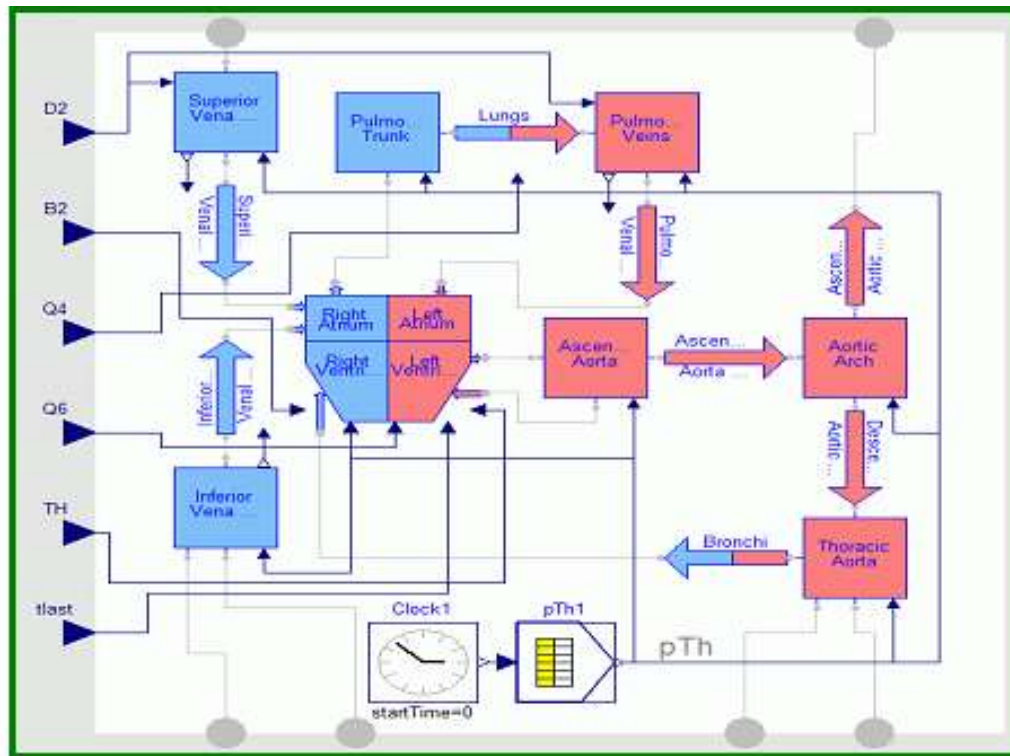
Modelica in Automotive Industry



Modelica in Avionics



Modelica in Biomechanics



Application of Modelica in Robotics Models

Real-time Training Simulator for Flight, Driving

- Using Modelica models generating real-time code
- Different simulation environments (e.g. Flight, Car Driving, Helicopter)
- Developed at DLR Munich, Germany
- Dymola Modelica tool

(Movie demo next page)



Courtesy of Tobias Bellmann, DLR,
Oberpfaffenhofen, Germany

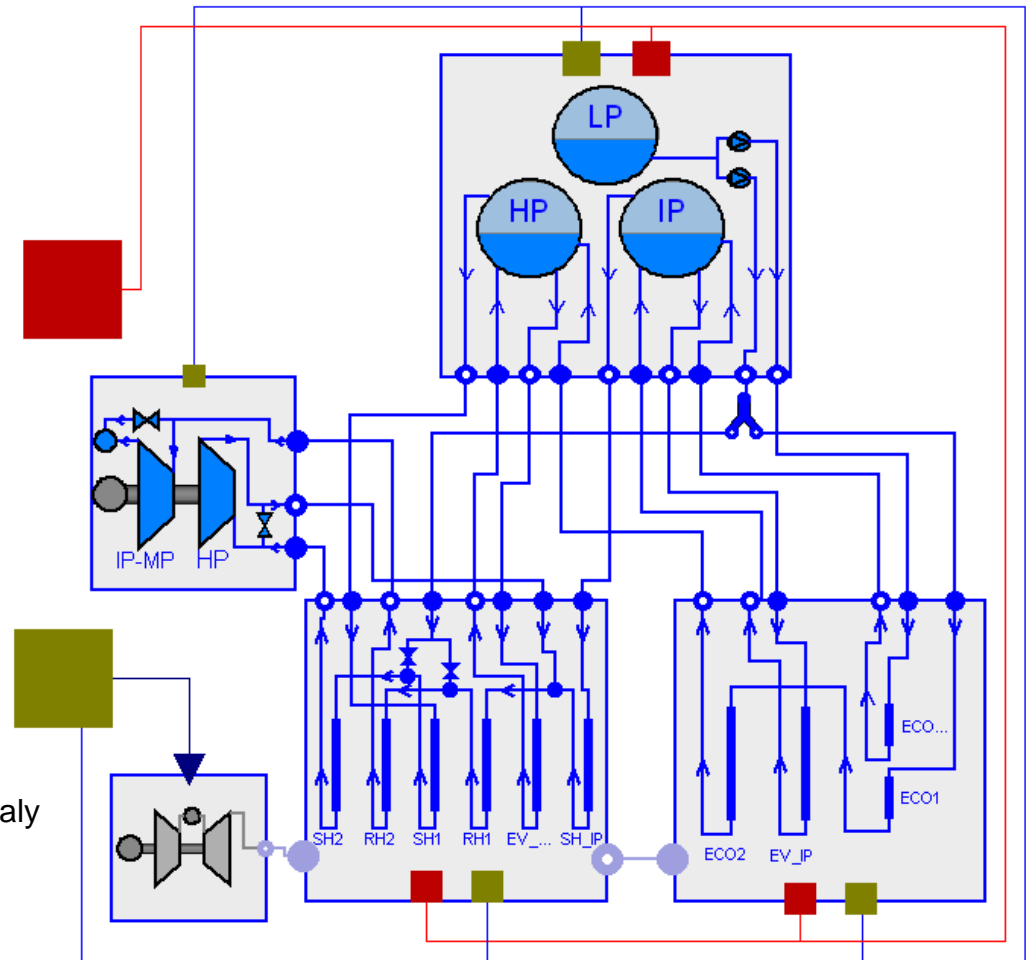
DLR Real-time Training Simulator Movie Demo



Combined-Cycle Power Plant

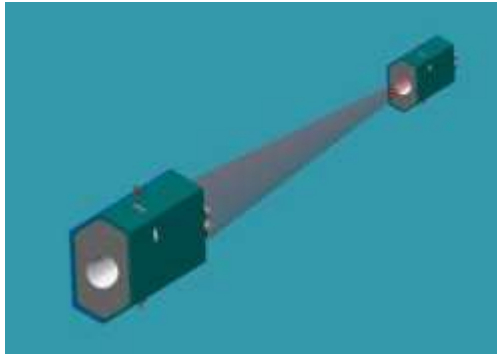
Plant model – system level

- GT unit, ST unit, Drum boilers unit and HRSG units, connected by thermo-fluid ports and by signal buses
- Low-temperature parts (condenser, feedwater system, LP circuits) are represented by trivial boundary conditions.
- GT model: simple law relating the electrical load request with the exhaust gas temperature and flow rate.



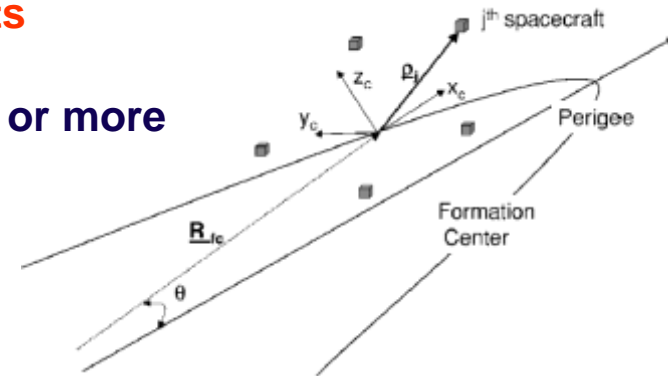
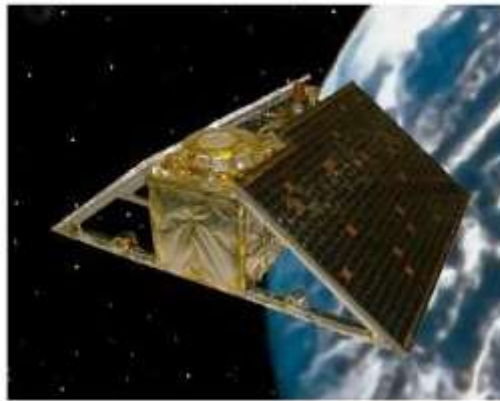
Courtesy Francesco Casella, Politecnico di Milano – Italy
and Francesco Pretolani, CESI SpA - Italy

Modelica Spacecraft Dynamics Library



Formation flying on elliptical orbits

Control the relative motion of two or more spacecraft



**Attitude control for satellites
using magnetic coils as actuators**

**Torque generation mechanism:
interaction between coils and
geomagnetic field**

Courtesy of Francesco Casella, Politecnico di Milano, Italy



Large-scale ABB OpenModelica Application

Generate code for controlling 7.5 to 10% of German Power Production



ABB OPTIMAX PowerFit

- Real-time optimizing control of large-scale virtual power plant for system integration
- **Software including OpenModelica** now used in managing more than 2500 renewable plants, total up to 1.5 GW

High scalability supporting growth

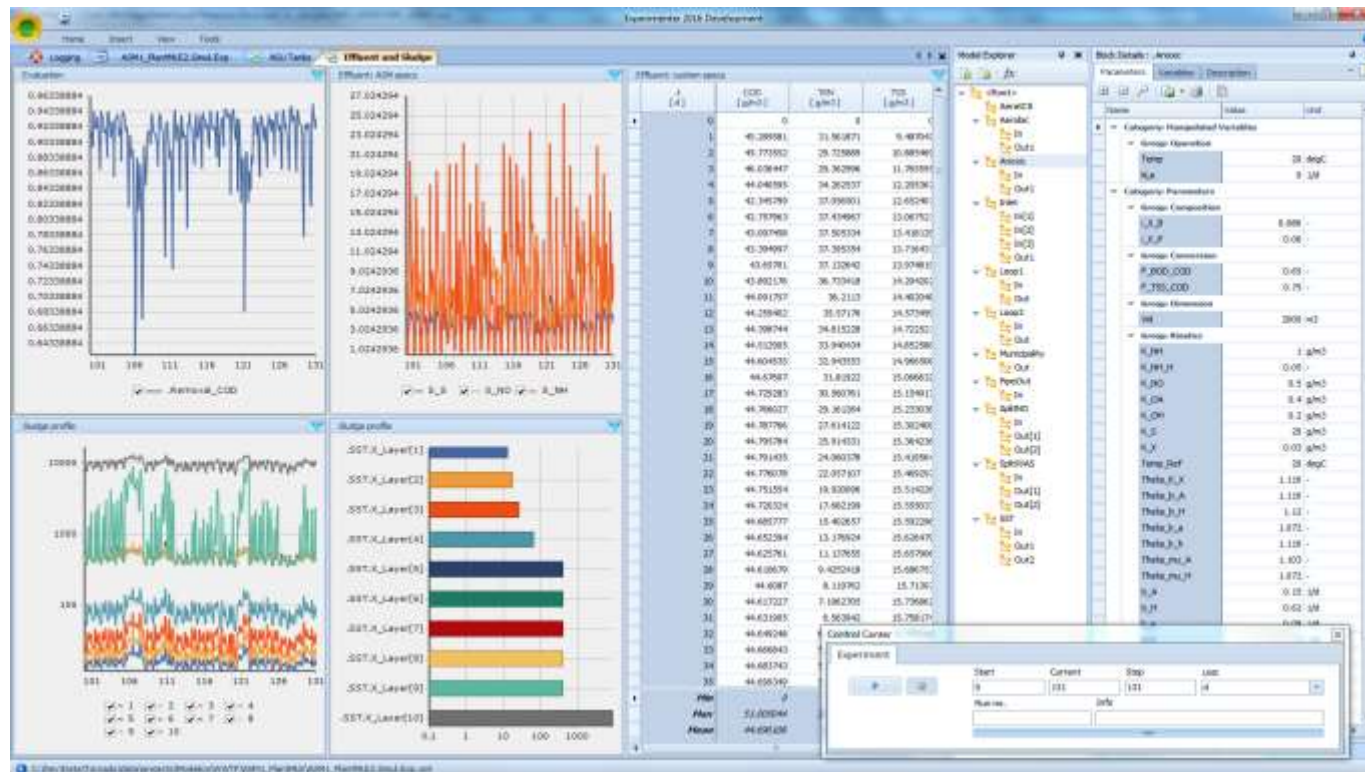
- 2012: initial delivery (for 50 plants)
- 2013: SW extension (500 plants)
- 2014: HW+SW extension (> 2000)
- 2015: HW+SW extension, incl. OpenModelica generating optimizing controller code in FMI 2.0 form

Manage 7.5% - 10% of German Power

- 2015, Aug: OpenModelica Exports FMUs for real-time optimizing control (seconds) of about **5.000 MW (7.5%) of power in Germany**

Industrial Product with OEM Usage of OpenModelica – MIKE by DHI, WEST Water Quality, Water Treatment and Sludge

- **MIKE by DHI**, www.mikebydhi.com, **WEST Water Quality** modeling and simulation environment
- Includes a large part of the OpenModelica compiler using the OEM license.
- Here a water treatment effluent and sludge simulation.

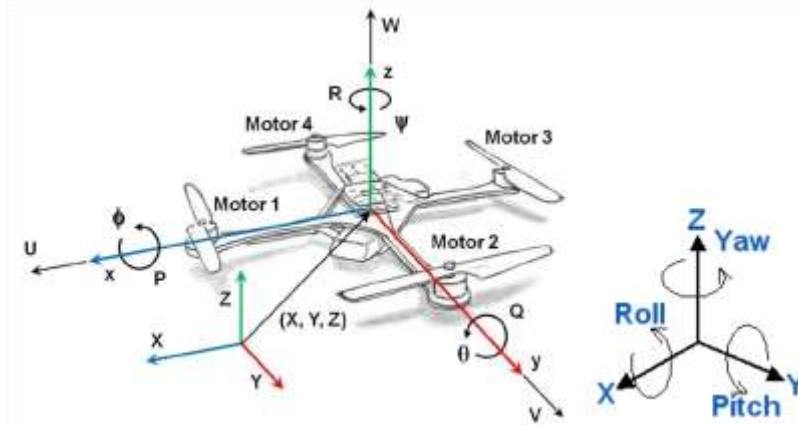


Digital Twin OpenModelica Applications by Modelicon

Model-based Control of UAVs and Walking Robots

- UAV control and simulation
- Walking 2-wheel robot

All models and control software done using OpenModelica!



UAV
Movie demo



Walking 2-wheel Robot,

Movie demo



More Sustainable Foetry – Digital Twin of Balloon-Assisted UAV – Collaboration with GI-LIFT AB and Modelicon

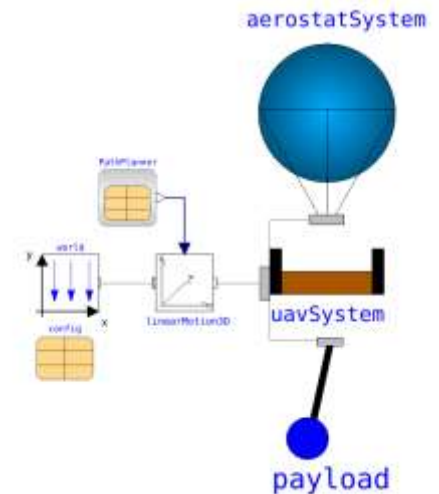
**Avoid
clear-cut
damage**



**Instead high-powered
Electric
Ballon-assisted
UAV lifting system**
(patent pending, GI-LIFT)



**Digital Twin
Using OpenModelica**

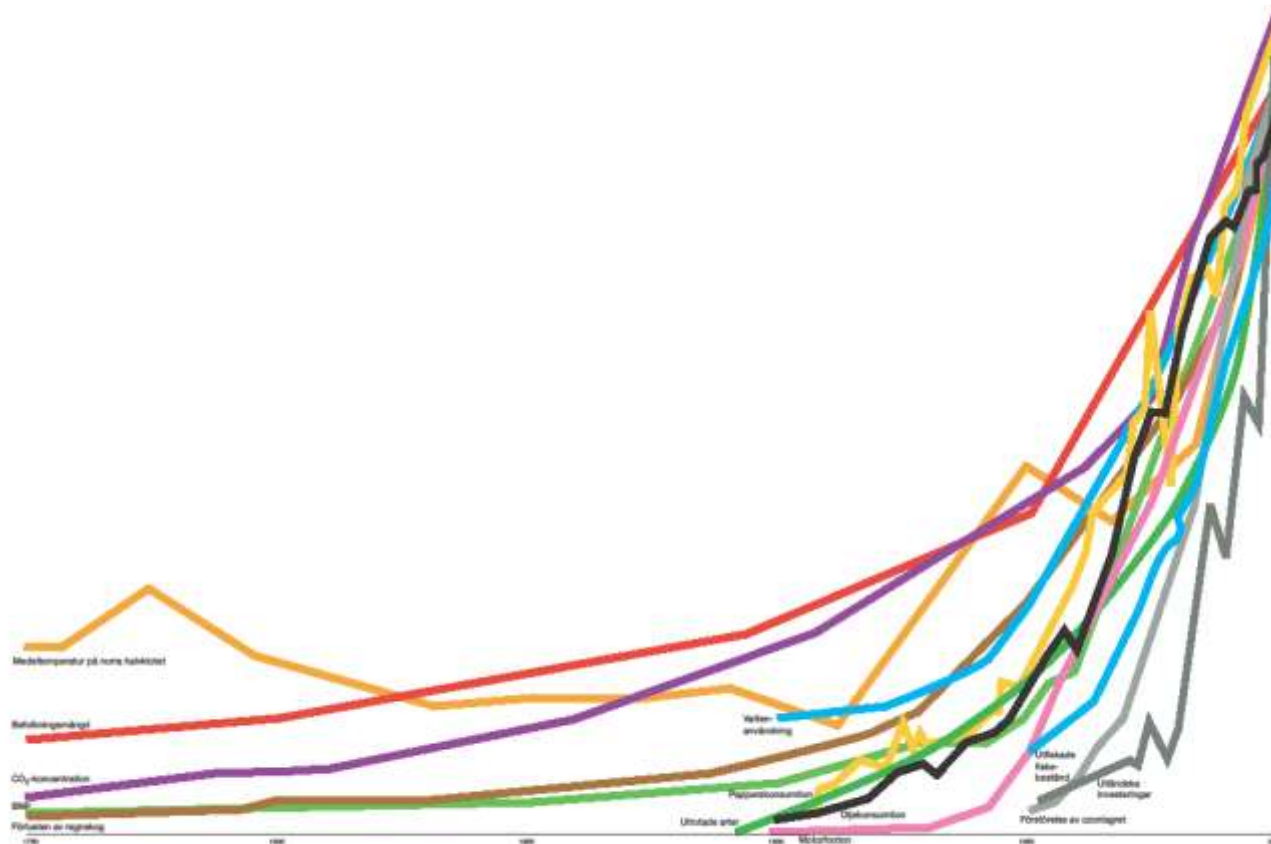


**Most important challenge
for humanity -
Develop a sustainable society!**

**Use Modelica in to model and optimize
sustainable technical innovations,
and a sustainable circular economy**

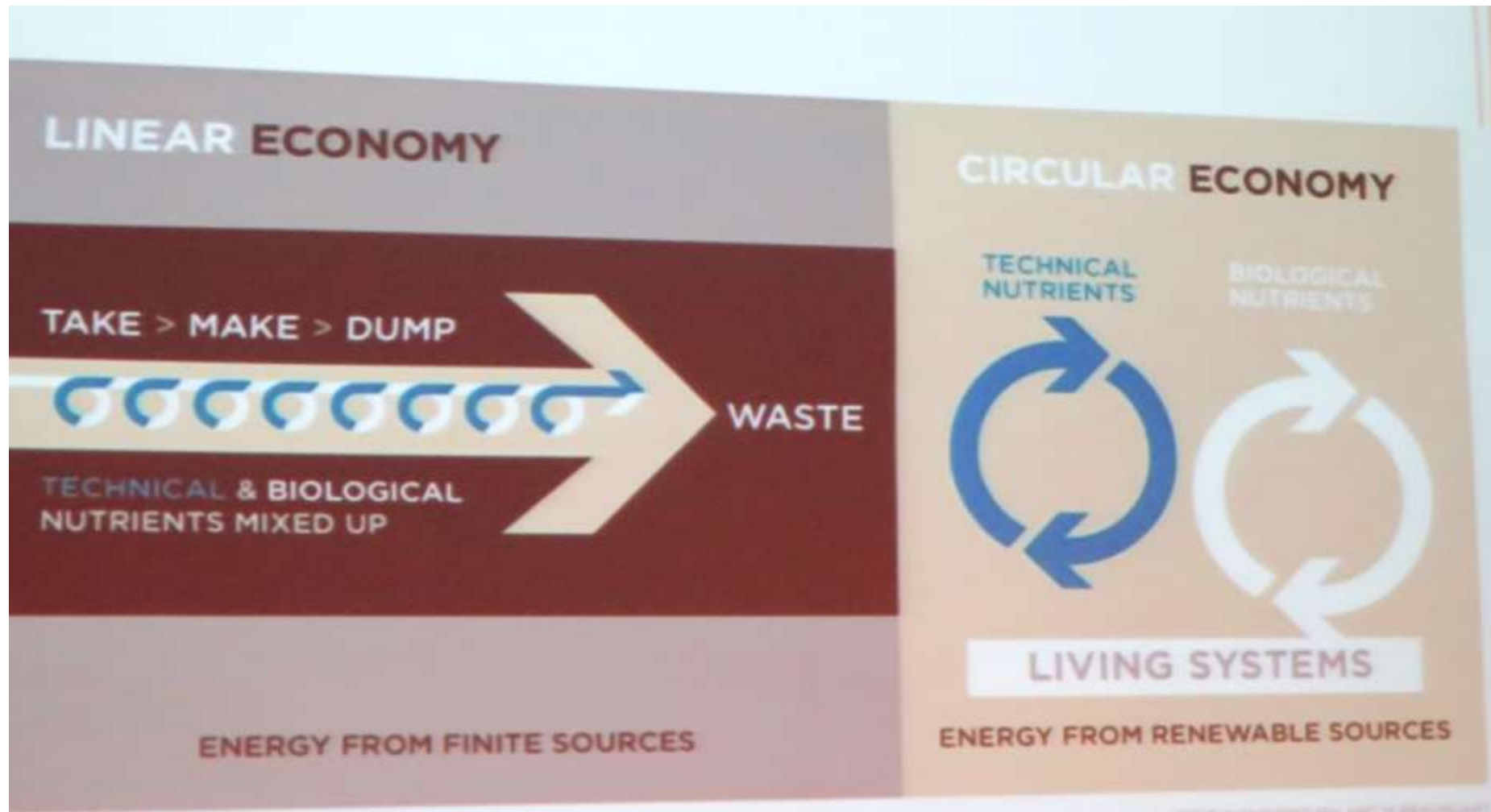
A Unique Point in History – Exponential Trends Approaches Planet Earth Boundaries

Year 1750-2000:



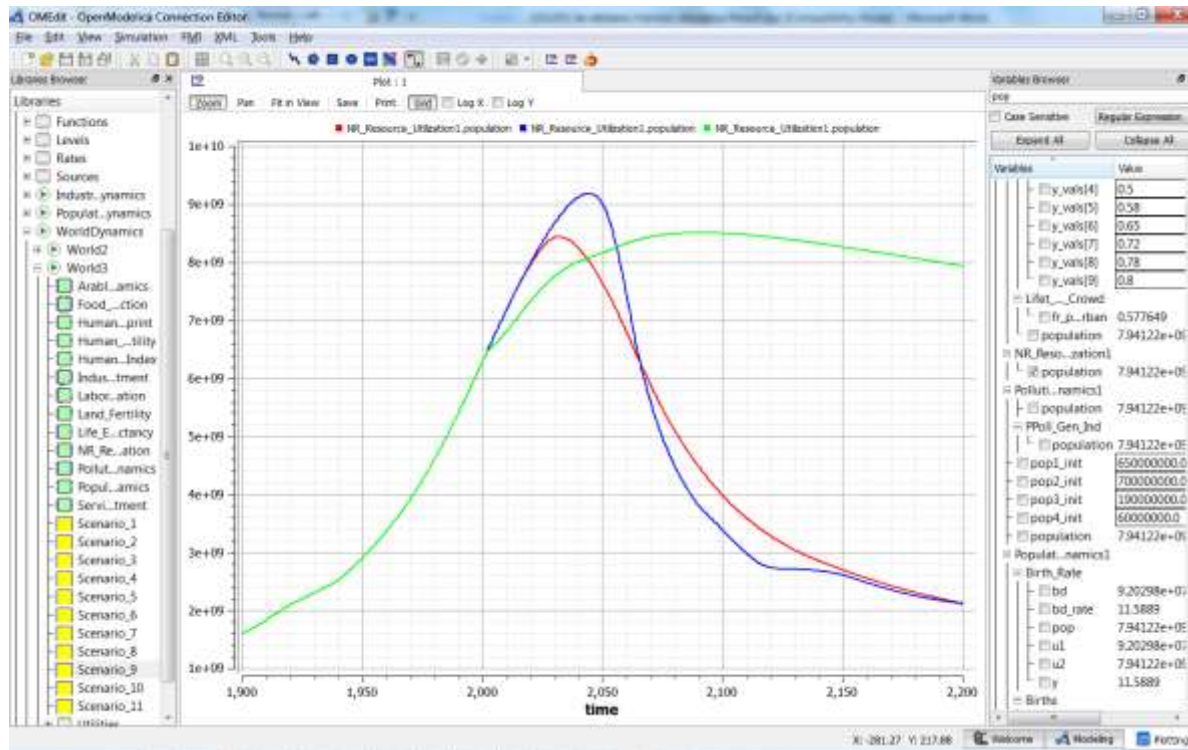
- Mean temperature north hemisphere,
- Population,
- CO₂-concentration,
- BNP,
- Loss av rain forest,
- Water usage
- Paper consumption,
- Exterminated species
- Oil consumption,
- Motor vehicles
- Destroyed fish populations
- Destruction of ozon layer
- Foreign investments

Challenge: Use Modeling and Simulation Technology to Support Circular Economy – for a Sustainable World



System Dynamics – World Society Simulation

Limits to Material Growth; Population, Energy and Material flows



Left. **World3 simulation** with OpenModelica

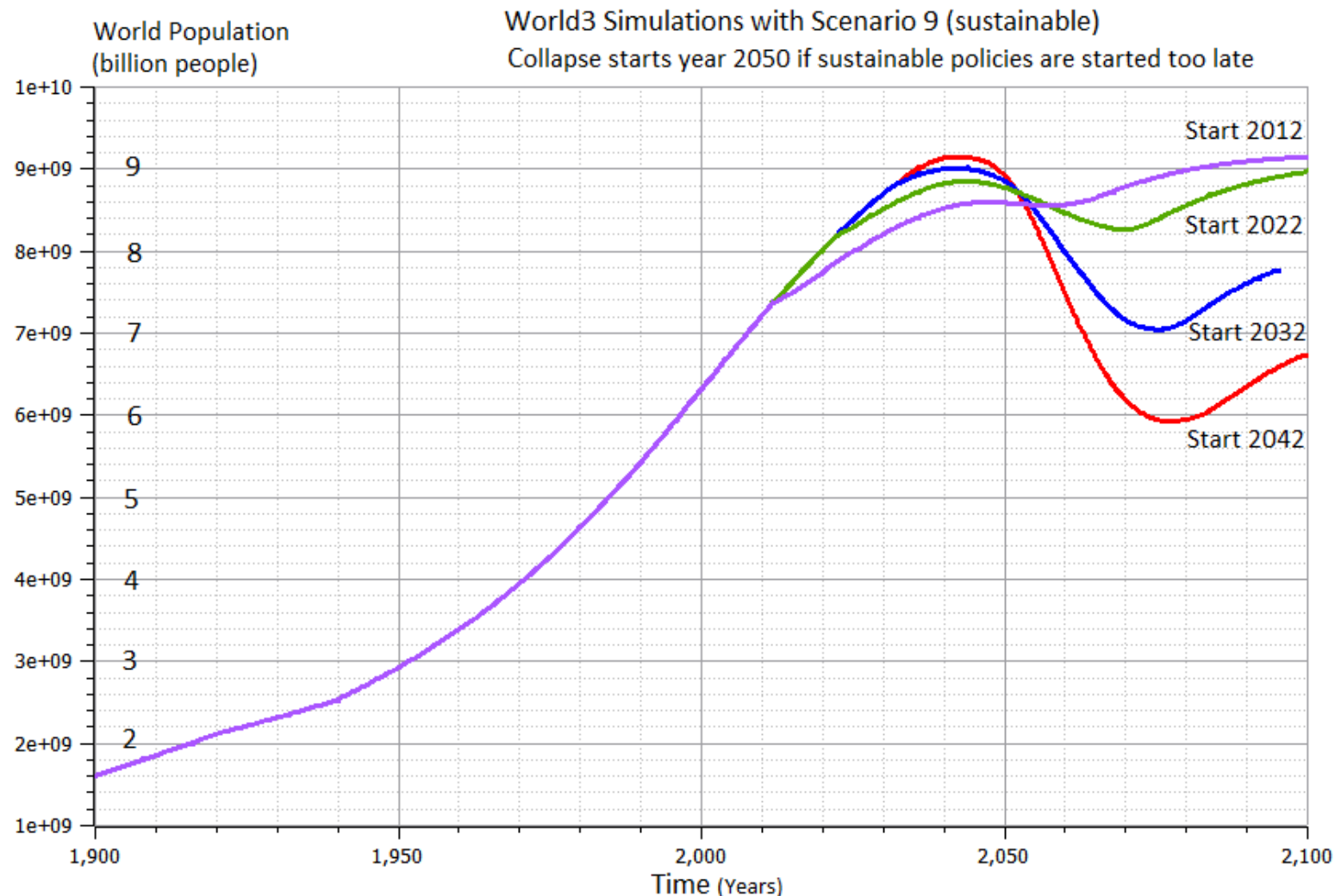
- 2 collapse scenarios (close to current developments)
- 1 sustainable scenario (green).

CO2 Emissions per person:

- USA 17 ton/yr
- Sweden 7 ton/yr
- India 1.4 ton/yr
- Bangladesh 0.3 ton/yr

- **System Dynamics Modelica library** by Francois Cellier (ETH), et al in OM distribution
- Warming converts many agriculture areas to deserts (USA, Europe, India, Amazonas)
- Ecological breakdown around 2080-2100, drastic reduction of world population
- To **avoid** this: Need for massive investments in sustainable technology and renewable energy sources

World3 Simulations with Different Start Years for Sustainable Policies – Collapse if starting too late



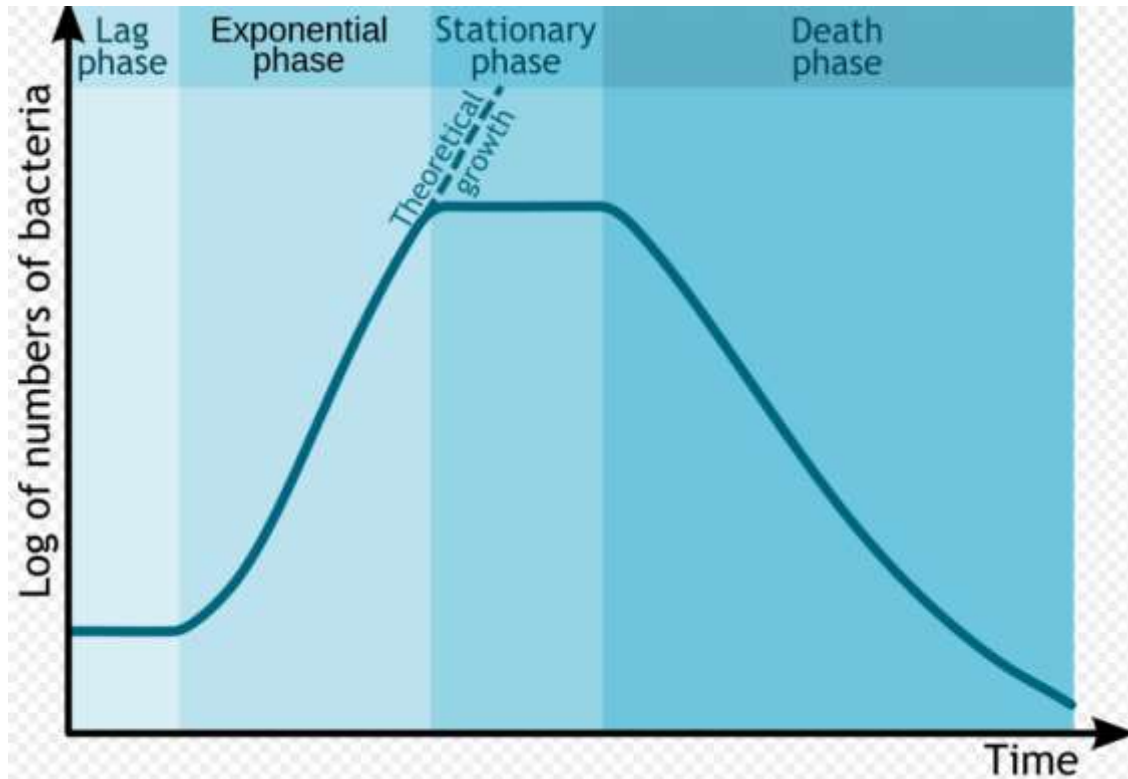
Are Humans More Intelligent than Bacteria?

Not yet evident!

Humans
on a
finite
Earth

vs

Bacteria
on a
finite
substrate



Bacterial growth curve /kinetic curve (Wikipedia)

LIMITS TO GROWTH



The 30-Year Update

DONELLA MEADOWS | JORGEN RANDERS | DENNIS MEADOWS

THE NEW YORK TIMES BESTSELLER

COLLAPSE

HOW SOCIETIES CHOOSE
TO FAIL OR SUCCEED

JARED DIAMOND

author of the Pulitzer Prize-winning

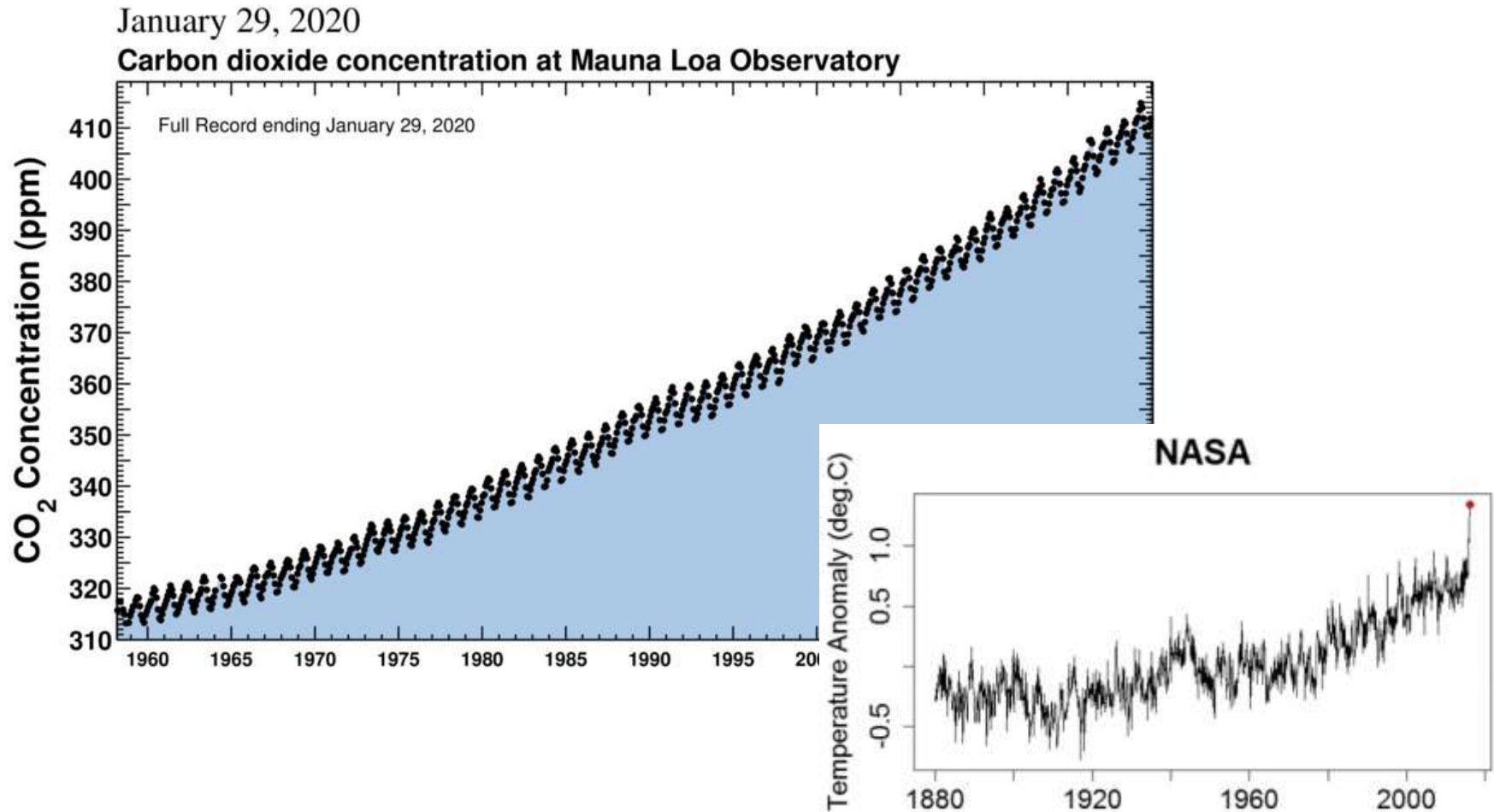
GUNS, GERMS, and STEEL

WITH A NEW AFTERWORD

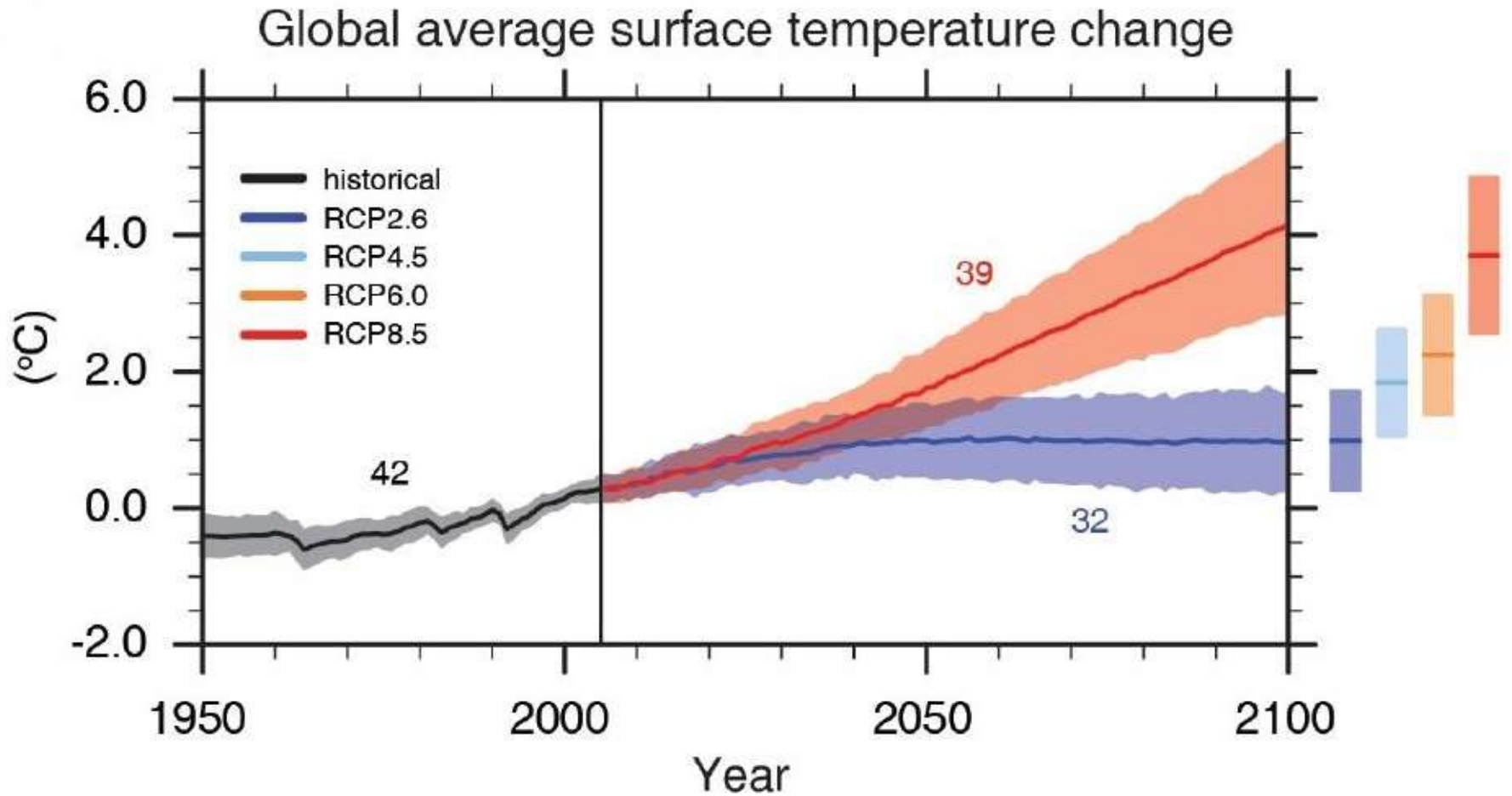


The Keeling Curve on CO₂ Concentrations until 2020

Temperatures 1880 - 2016



Global Temperature Rise Scenarios (IPCC 2013)

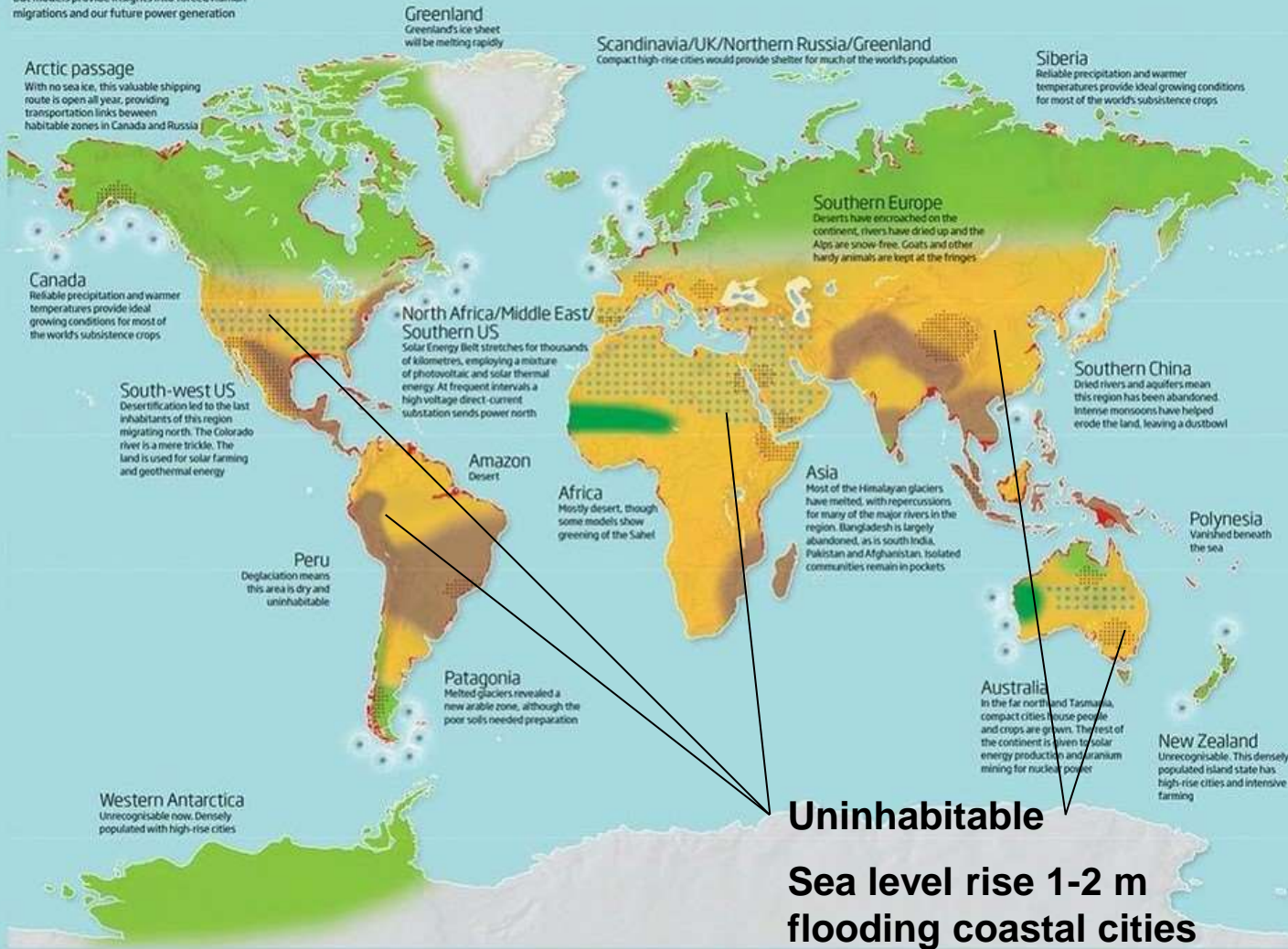


How the world could be in 80-100 years at a global warming of 4 degrees

Business-as-usual scenario, IPCC

The world: 4°C warmer

No one knows exactly what this world will look like, but models provide insights into forced human migrations and our future power generation



- Cities, agriculture
- Uninhabitable desert
- Uninhabitable due to extreme weather
- Flooded

Massive migration to northern Europe, Russia, and Canada

Example Emissions

CO₂e / person

- Earth can handle 2 ton/yr
- Flight Spain – 1 ton
- Flight Canarys – 2 ton
- Flight Thailand – 4 ton

References

New Scientist, 28 february 2009
IPCC, business as usual scenario
www.climate-lab-book.ac.uk
www.atmosfair.de

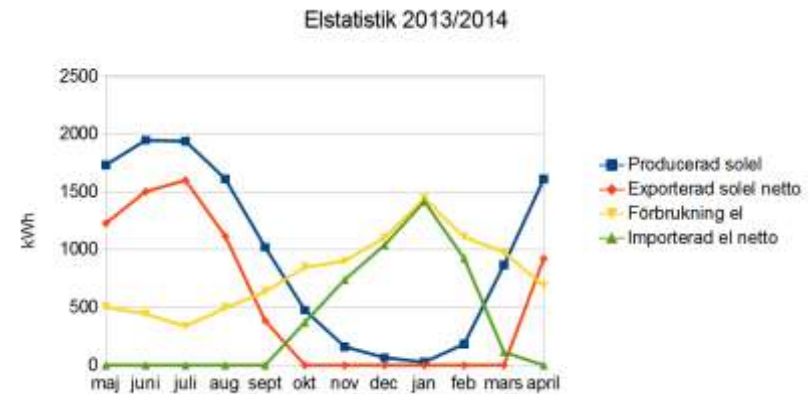
What Can You Do?

Need Global Sustainability Mass Movement

- Develop smart Cyber-Physical systems for reduced energy and material footprint
- Model-based circular economy for re-use of products and materials
- Promote sustainable lifestyle and technology
- Install electric solar PV panels
- Buy shares in cooperative wind power



20 sqm solar panels on garage roof, Nov 2012
Generated 2700 W at noon March 10, 2013



Expanded to 93 sqm, 12 kW, March 2013
House produced 11600 kwh, used 9500 kwh
Avoids 10 ton CO2 emission per year

Example Electric Cars

Can be charged by electricity from own solar panels



Renault ZOE; 5 seat; Range:
22kwh (2014) vs 51 kwh battery (2019)

- **Realistic Swedish drive cycle:**
- **Summer: 160 km, 2019 385 km**
- **Winter: 110 km, 2019 290 km**



2018, Tesla Model 3 LR, range 560 km



DLR ROboMObil

- **experimental electric car**
- **Modelica models**



2020, Volvo XC40 recharge, range 400 km

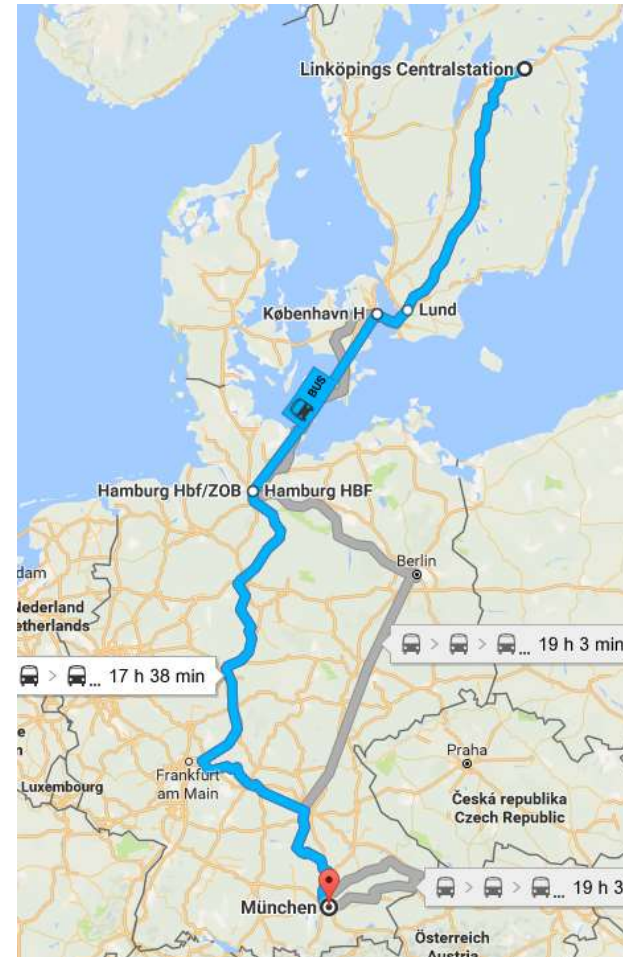
What Can You Do?

More Train Travel – Less Air Travel

- Air travel by Swedish Citizens – about the same emissions as all personal car traffic in Sweden!
- By train from Linköping to Munich and back – saves almost 1 ton of CO₂e emissions compared to flight
- Leave Linköping 07.00 in Munich 23.14

More Examples, PF travel 2016:

- Train Linköping-Paris, Dec 3-6, EU project meeting
- Train Linköping-Dresden, Dec 10-16, 1 week workshop



Train
travel
Linköping
- Munich

A world map showing the distribution of solar energy potential. The map is color-coded by latitude, with the warmest colors (red/orange) in the tropics and the coldest colors (blue/purple) at the poles. Small yellow rectangles are placed on the map to indicate the surface area needed for 100% solar energy for humanity in the year 2012. These rectangles are located in Mexico, Brazil, Africa, the Middle East, India, China, and Australia.

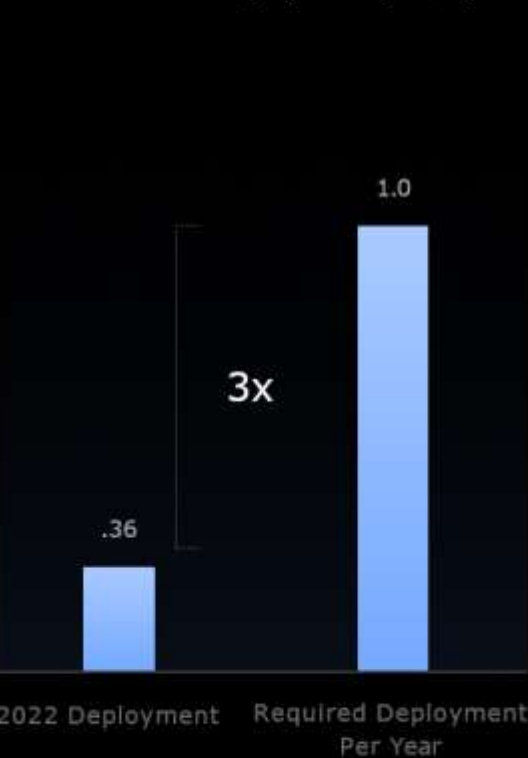
**Small rectangles – surface needed for
100% solar energy for humanity
Year 2012 version**

More Than Enough Renewable Resources Available (Tesla Master Plan, Investor days, March 1, 2023)

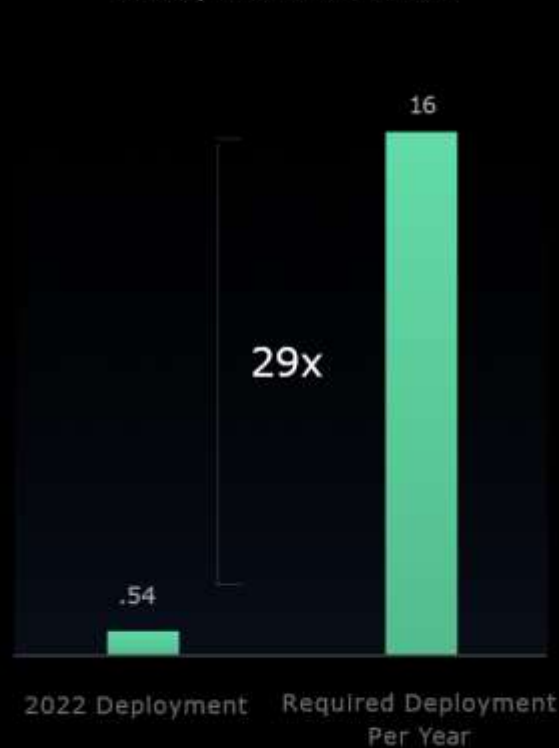


If We Grow our Production Capacity as Shown by 2030 We Can Be 100% Sustainable by 2050

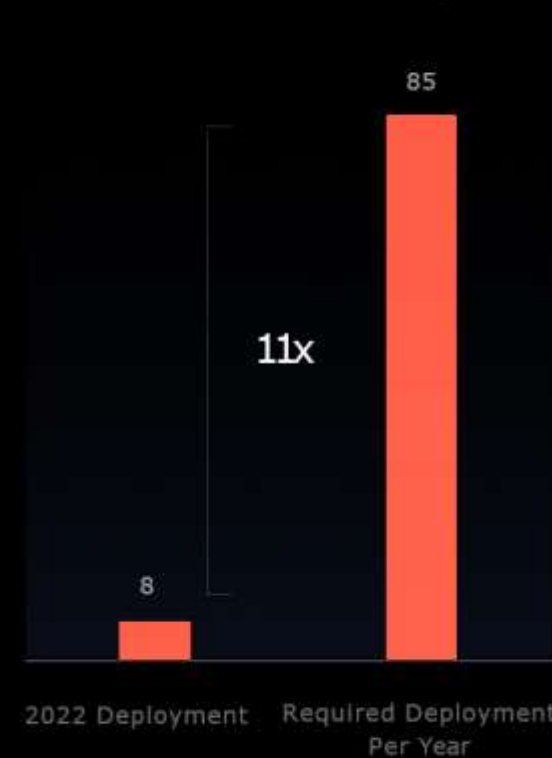
Solar & Wind Deployment (TW/Yr)



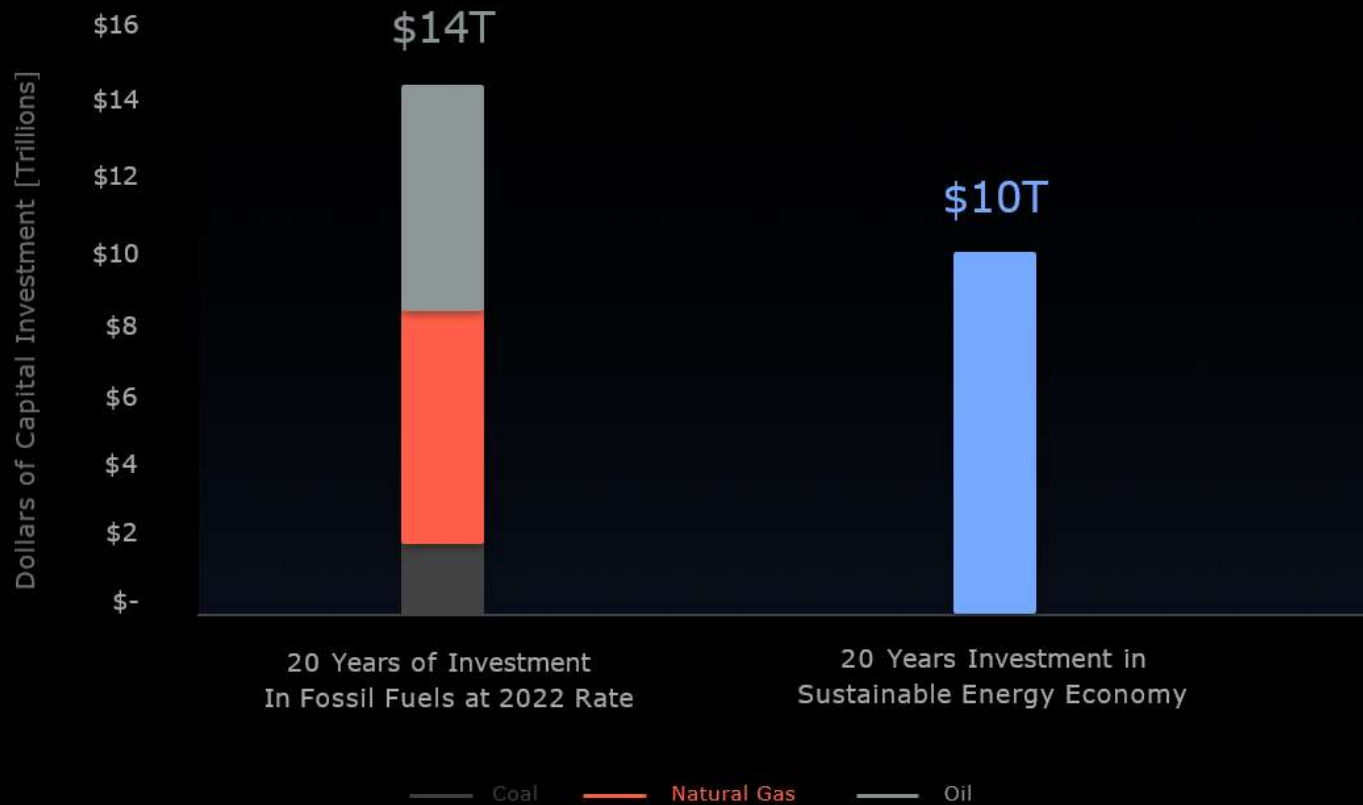
Vehicle, Stationary, & Thermal
Battery Production TWh/Yr



Electric Vehicle Production Millions/Yr



A Sustainable Energy Economy Is 60% The Cost of Continuing Fossil Fuel Investments



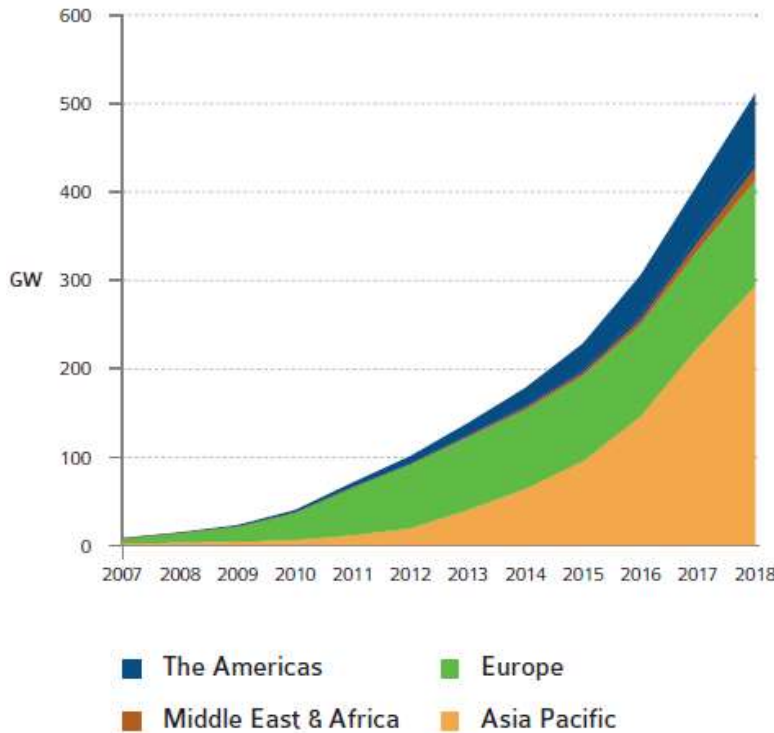
A Sustainable Energy Economy Is Within Reach & We Should Accelerate It

HOW THE MASTER PLAN WORKS

240TWh	30TW	\$10T	1/2	<0.2%	10%	ZERO
Storage	Renewable Power	Manufacturing Investment	The Energy Required	Land Area Required	2022 World GDP	Insurmountable Resource Challenges

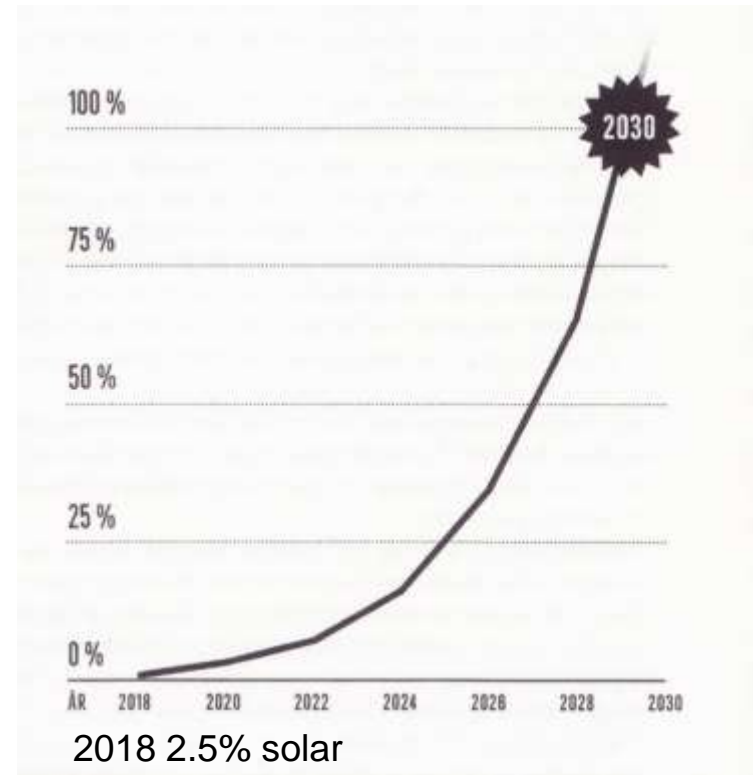
Solar Energy PhotoVoltaics Growth Trends

FIGURE 2.5: EVOLUTION OF REGIONAL PV INSTALLATIONS (GW)



Almost
Exponential
worldwide
Growth of
Photovoltaics
2006 – 2018

IEA PVPS
TRENDS IN
PHOTOVOLTAIC
APPLICATIONS
2019



100% of global electricity
production year 2030 if
strong exponential growth
continues

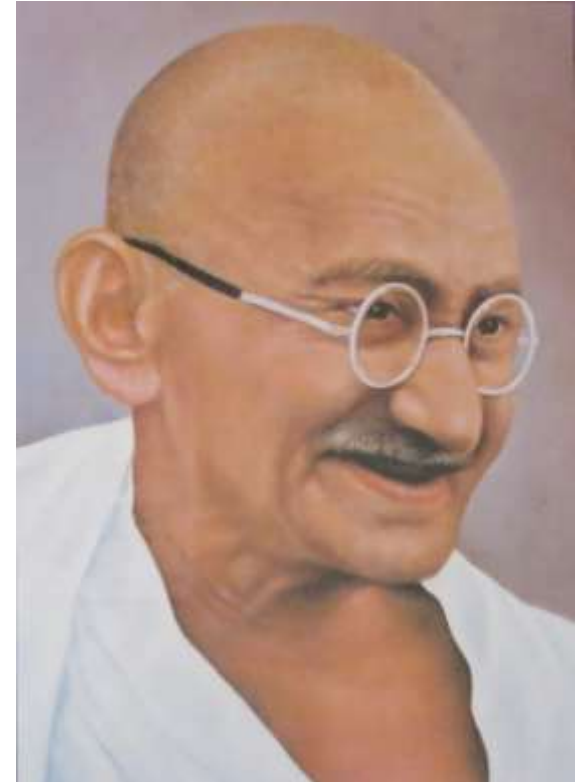
Sustainable Society Necessary for Human Survival

Almost Sustainable

- India, recently 1.4 ton CO₂/person/year
- Healthy vegetarian food
- Small-scale agriculture
- Small-scale shops
- Simpler life-style (Mahatma Gandhi)

Non-sustainable

- USA 17 ton CO₂, Sweden 7 ton CO₂/yr
- High meat consumption (1 kg beef uses ca 4000 L water for production)
- Hamburgers, unhealthy , includes beef
- Energy-consuming mechanized agriculture
- Transport dependent shopping centres
- Stressful materialistic lifestyle



Gandhi – role model for future less materialistic life style

Brief Modelica History

- First Modelica design group meeting in fall 1996
 - International group of people with expert knowledge in both language design and physical modeling
 - Industry and academia
- Modelica Versions
 - 1.0 released September 1997
 - 2.0 released March 2002
 - 2.2 released March 2005
 - 3.0 released September 2007
 - 3.1 released May 2009
 - 3.2 released March 2010
 - 3.3 released May 2012
 - 3.2 rev 2 released November 2013
 - 3.3 rev 1 released July 2014
 - 3.4 released April 2017
 - 3.5 released February 2021
- Modelica Association was established in 2000 in Linköping
 - Open, non-profit organization

Modelica Conferences

- The 1st International Modelica conference October, 2000
- The 2nd International Modelica conference March 18-19, 2002
- The 3rd International Modelica conference November 5-6, 2003 in Linköping, Sweden
- The 4th International Modelica conference March 6-7, 2005 in Hamburg, Germany
- The 5th International Modelica conference September 4-5, 2006 in Vienna, Austria
- The 6th International Modelica conference March 3-4, 2008 in Bielefeld, Germany
- The 7th International Modelica conference Sept 21-22, 2009 in Como, Italy
- The 8th International Modelica conference March 20-22, 2011 in Dresden, Germany
- The 9th International Modelica conference Sept 3-5, 2012 in Munich, Germany
- The 10th International Modelica conference March 10-12, 2014 in Lund, Sweden
- The 11th International Modelica conference Sept 21-23, 2015 in Versailles, Paris
- The 12th International Modelica conference May 15-17, 2017 in Prague, Czech Rep
- The 13th International Modelica conference March 4-6, 2019, Regensburg, Germany
- The 14th International Modelica conference Sept 20-24, 2021, Linköping, Sweden
- Also: Asian Modelica conferences 2016, 2017, 2018, 2020, 2022
- Also: US Modelica conference 2018, 2020, 2022

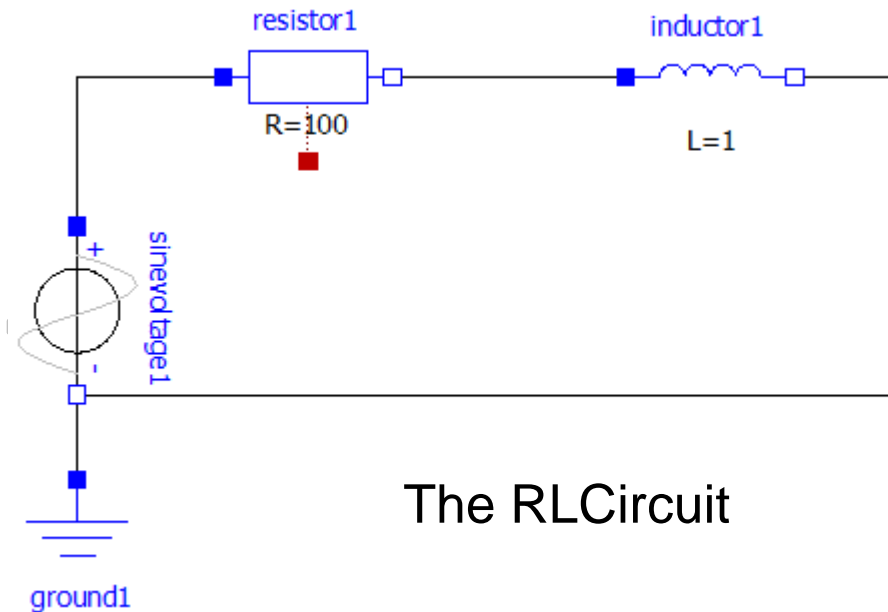
Exercises Part I

Hands-on graphical modeling

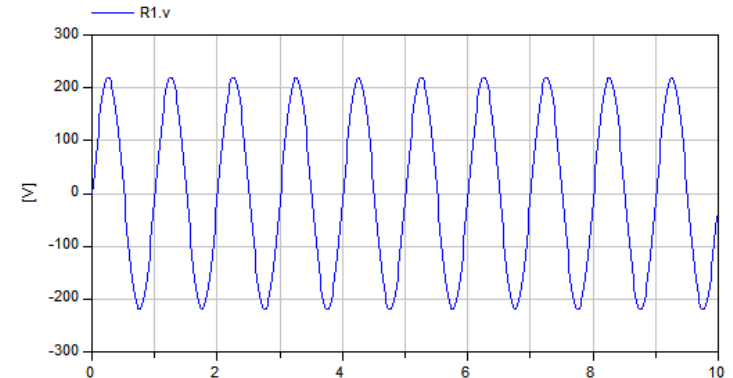
(15 minutes)

Exercises Part I – Basic Graphical Modeling

- (See instructions on next two pages)
- Start the OMEdit editor (part of OpenModelica)
- Draw the RLCircuit
- Simulate




The RLCircuit




Simulation

Exercises Part I – OMEdit Instructions (Part I)

- Start OMEdit from the Program menu under OpenModelica
- Go to **File** menu and choose **New**, and then select **Model**.
- E.g. write *RLCircuit* as the model name.
- For more information on how to use OMEdit, go to **Help** and choose **User Manual** or press **F1**.

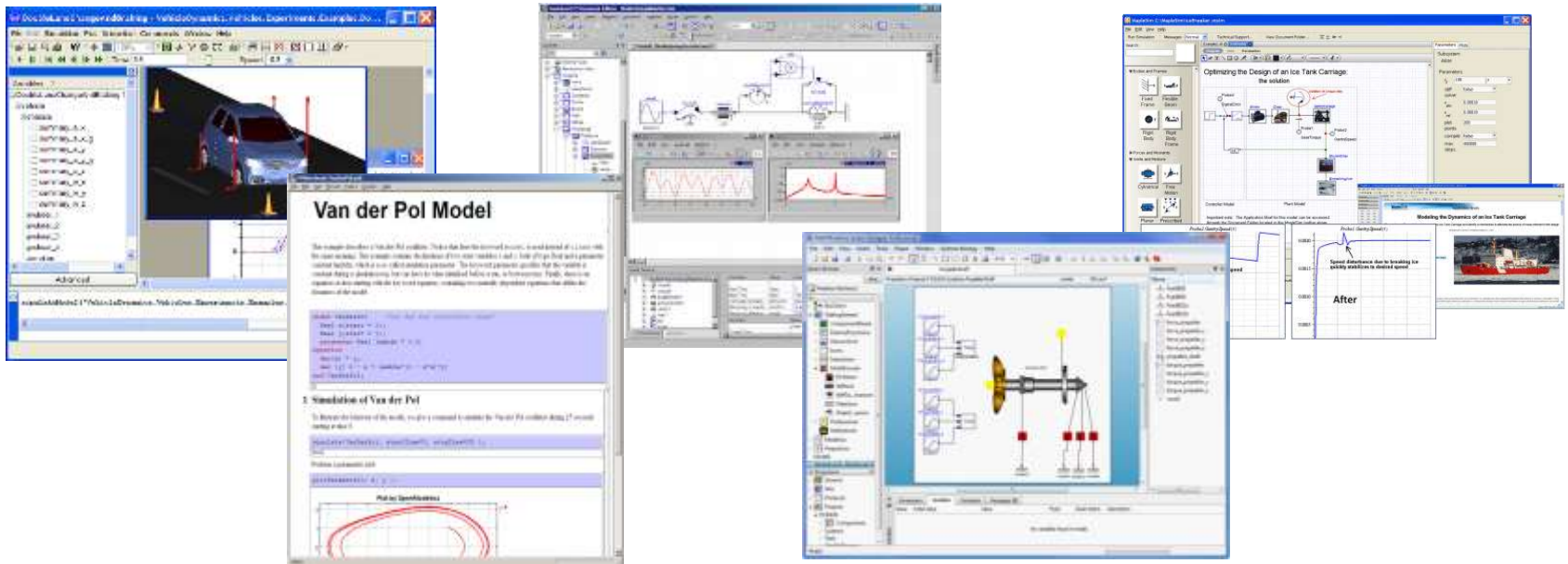
- 
- Under the **Modelica Library**:
 - Contains The standard Modelica library components
 - The **Modelica files** contains the list of models you have created.

Exercises Part I – OMEdit Instructions (Part II)

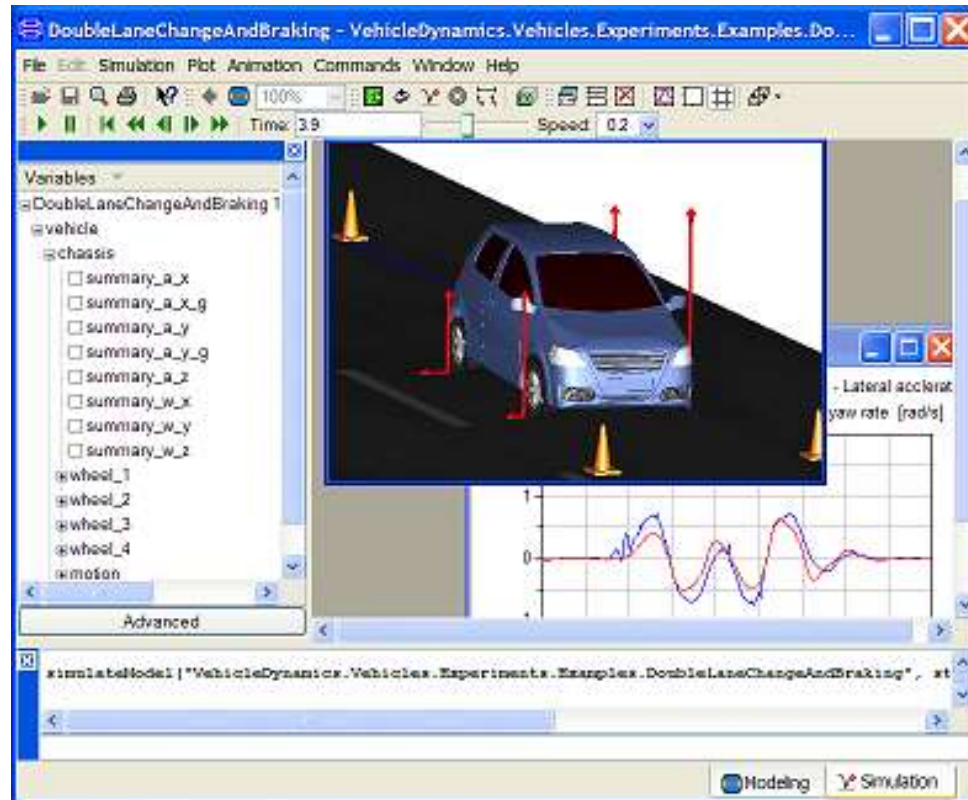
- For the RLCircuit model, **browse** the Modelica standard library and **add** the following component models:
 - Add `Ground`, `Inductor` and `Resistor` component models from `Modelica.Electrical.Analog.Basic` package.
 - Add `SineVoltage` component model from `Modelica.Electrical.Analog.Sources` package.
- Make the corresponding **connections** between the component models as shown in the previous slide.
- To **draw a connection line**: first single-click on a connector box; then start drawing while keeping the mouse button down; after drawing a little you can release the mouse button and continue drawing.
- **Simulate** the model
 - Go to the Simulation menu and choose simulate or click on the simulate button  in the toolbar.
- **Plot** the instance variables
 - Once the simulation is completed, a plot variables list will appear on the right side. Select the variable that you want to plot.

Part II

Modelica environments and OpenModelica

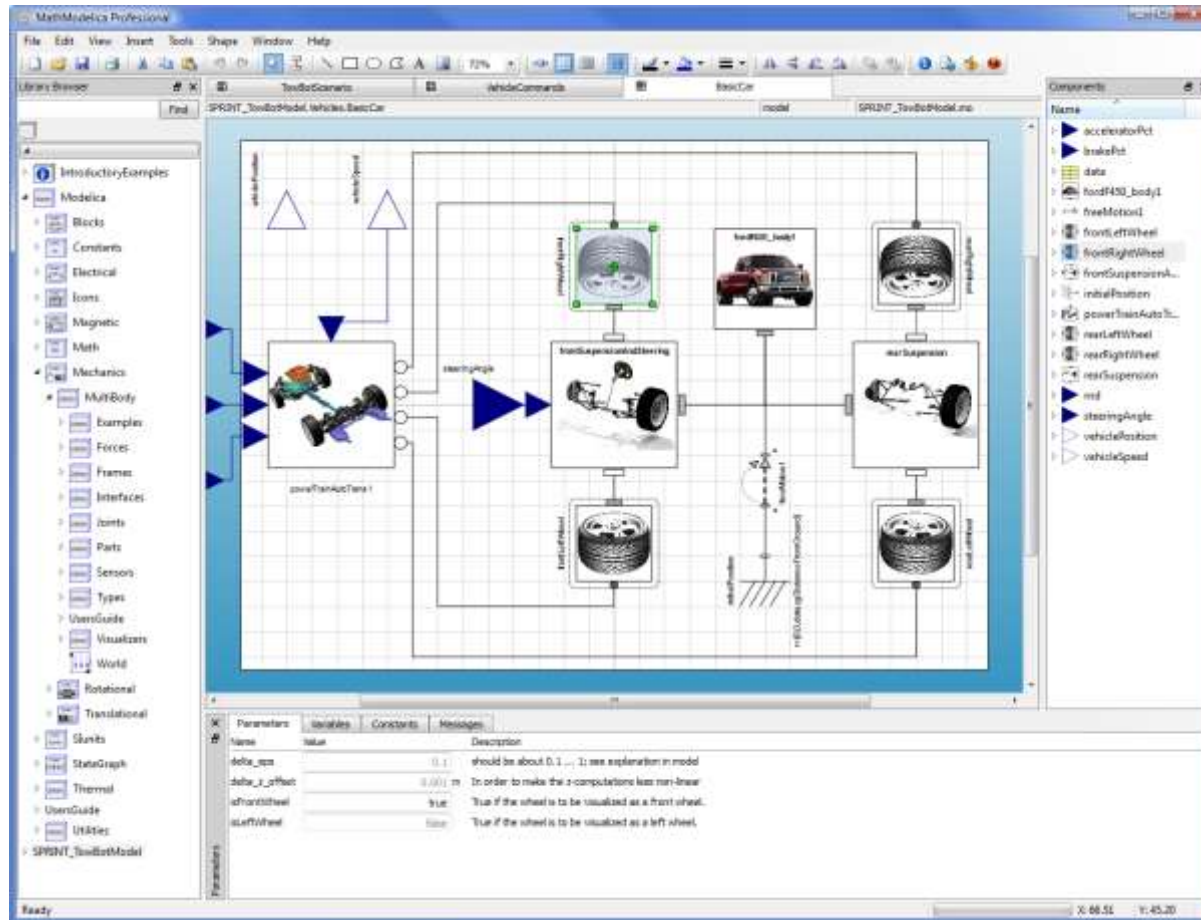


Dymola

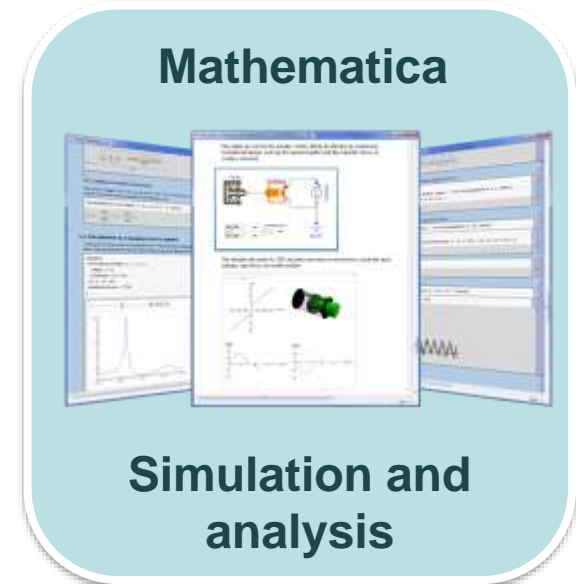


- Dassault Systemes Sweden
- Sweden
- First Modelica tool on the market
- Initial main focus on automotive industry
- www.dymola.com

Wolfram System Modeler – Wolfram MathCore



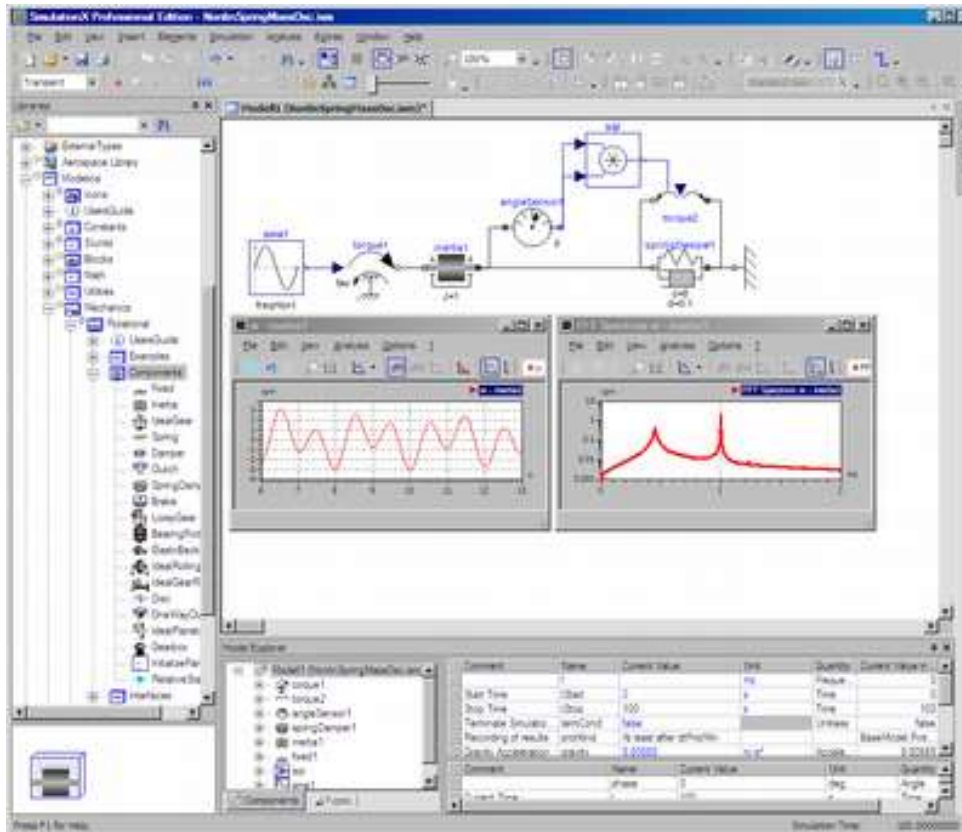
- Wolfram Research
- USA, Sweden
- General purpose
- Mathematica integration
- www.wolfram.com
- www.mathcore.com



Car model graphical view

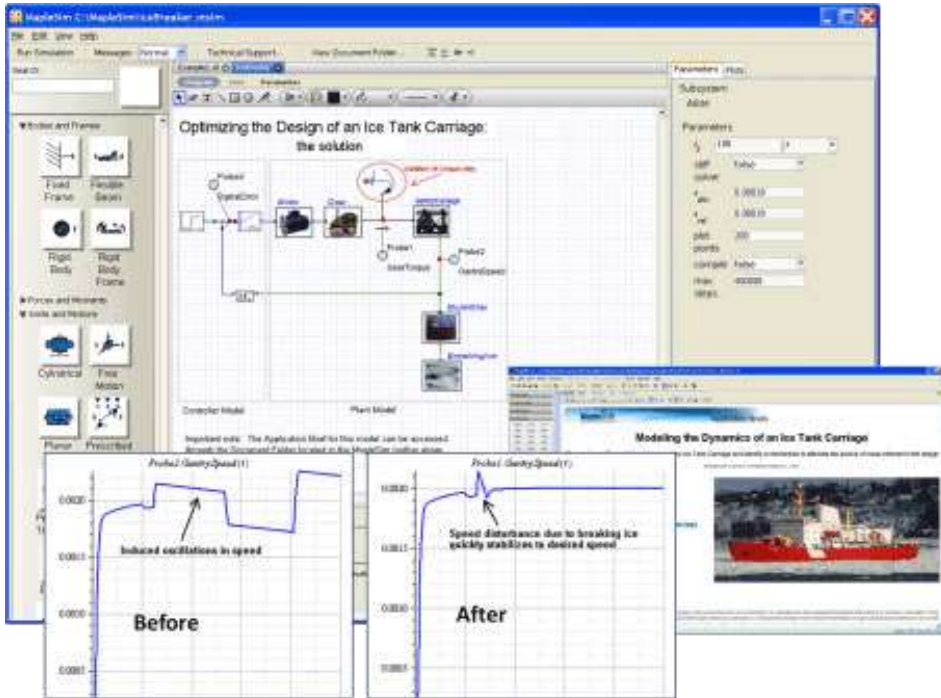
Courtesy
Wolfram
Research

Simulation X



- ITI Gmbh (Part of ESI Group)
- Germany
- Mechatronic systems
- www.simulationx.com

MapleSim



- Maplesoft
- Canada
- Recent Modelica tool on the market
- Integrated with Maple
- www.maplesoft.com

Modelon



Modelon Library Suite

Powered by Modelica

Our suite of libraries, built on the Modelica open standard, delivers state-of-the-art system models for a wide range of industrial applications.



Modelon Creator Suite

Our creator suite is a powerful platform for model creation, automation, simulation and optimization.



Modelon Deployment Suite

Powered by FMI

Our comprehensive suite of deployment products, built on the FMI open standard, enables collaboration and rapid deployment of system models across multiple platforms, varying tools, and organizations.

- Modelon
- Sweden and International
- Library Suite
- Creator Suite with Impact product and Optimica Compiler Toolbox and WAMS model editor
- www.modelon.com

The OpenModelica Environment

www.OpenModelica.org

The screenshot shows the OpenModelica website homepage. At the top is a blue header with the 'OpenModelica' logo on the left and 'Login' and 'Create an account' links on the right. Below the header is a dark navigation bar with links: HOME, DOWNLOAD, TOOLS & APPS, USERS, DEVELOPERS, FORUM, EVENTS, and RESEARCH. A search bar is located on the right side of this bar.

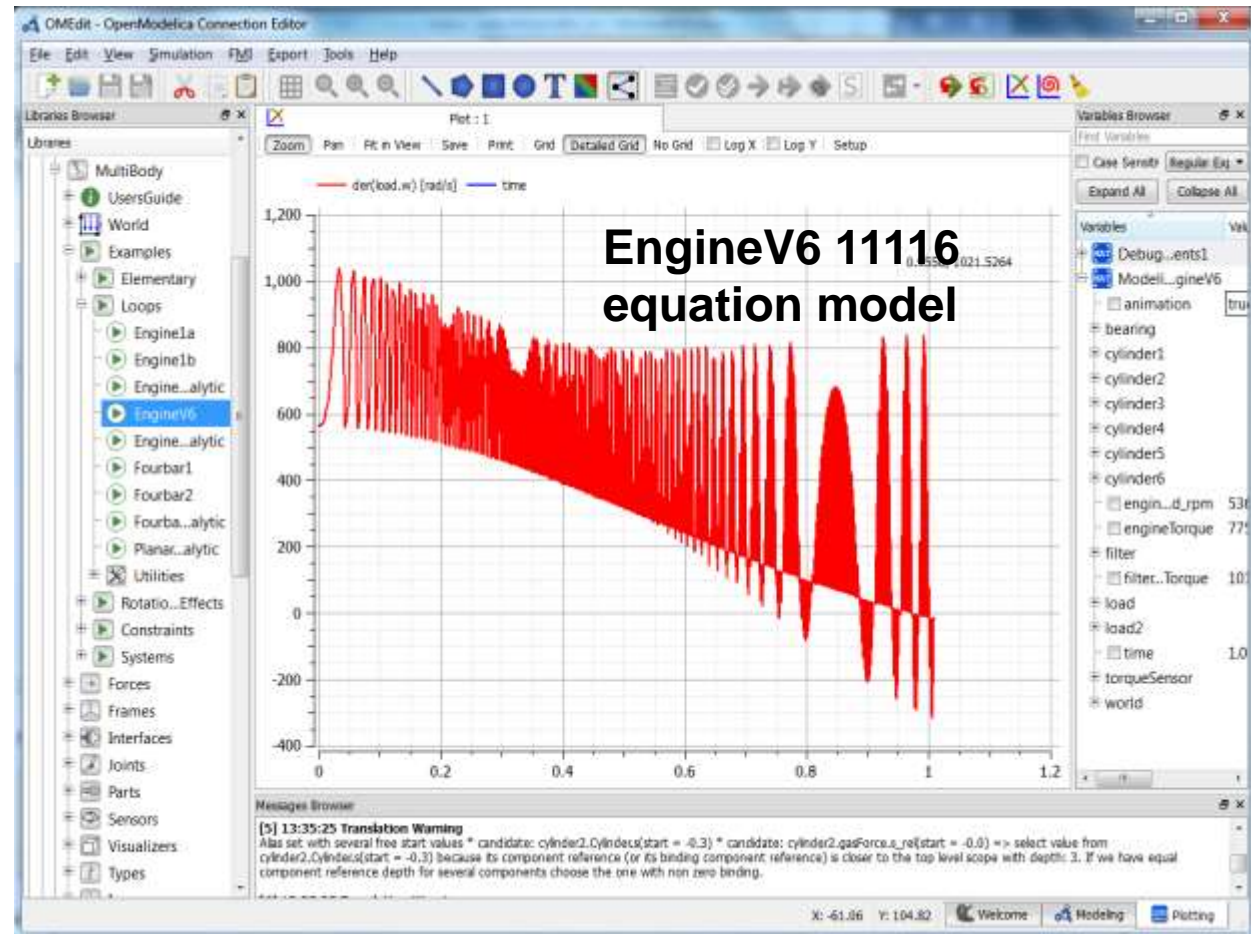
The main content area is divided into three columns:

- Top information:** Contains three sections: 'Industrial Products' (Commercial Applications using OpenModelica), 'OMEdit' (Enhanced OpenModelica Connection Editor), and 'Library Coverage' (Latest library coverage).
- Introduction:** A central text block stating that OPENMODELICA is an open-source Modelica-based modeling and simulation environment. It mentions support from the Open Source Modelica Consortium (OSMC) and provides links to a journal paper and slides. Below the text is a screenshot of the OMEdit software interface showing various windows and plots. At the bottom of this section are links to register for new releases, join a mailing list, get source code, and learn about Modelica.
- Latest news:** A list of recent updates and events, including the release of OpenModelica 1.18.0, 1.18.0-dev.beta1, 1.17.0, 1.16.5, 1.16.4, and 1.16.1, as well as the HUBCAP Open Calls and an overview article in the MIC Journal.

At the bottom left, there is a section for 'Modelica/OpenModelica Videos' featuring a video player with the title 'Overview of M...'.

OpenModelica – Free Open Source Tool developed by the Open Source Modelica Consortium (OSMC)

- Graphical editor
- Model compiler and simulator
- Debugger
- Performance analyzer
- Dynamic optimizer
- Symbolic modeling
- Parallelization
- Electronic Notebook and OMWebbook for teaching
- Spokentutorial for teaching

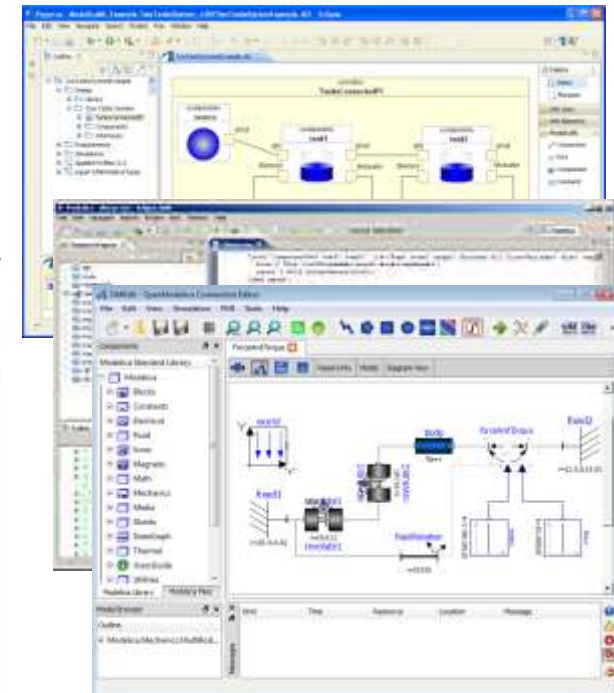
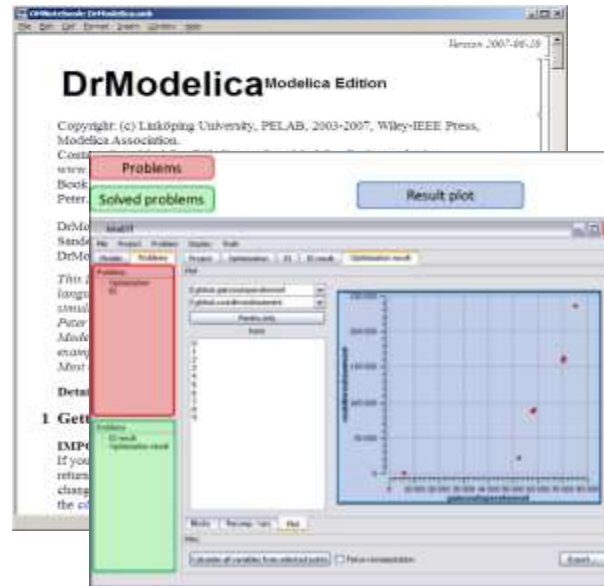
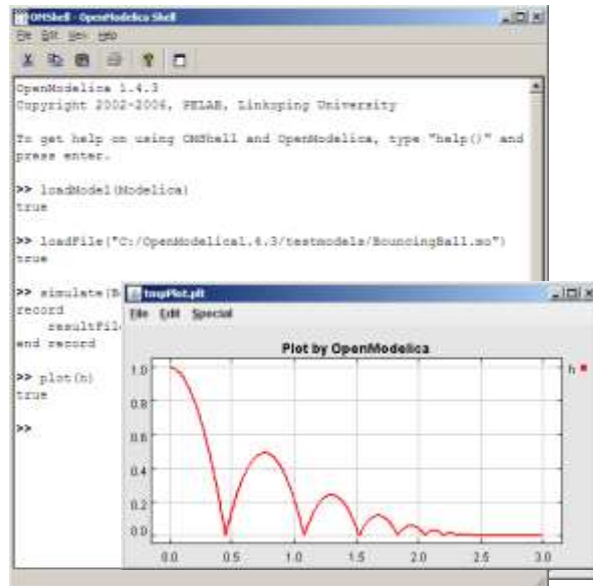


The OpenModelica Open Source Environment

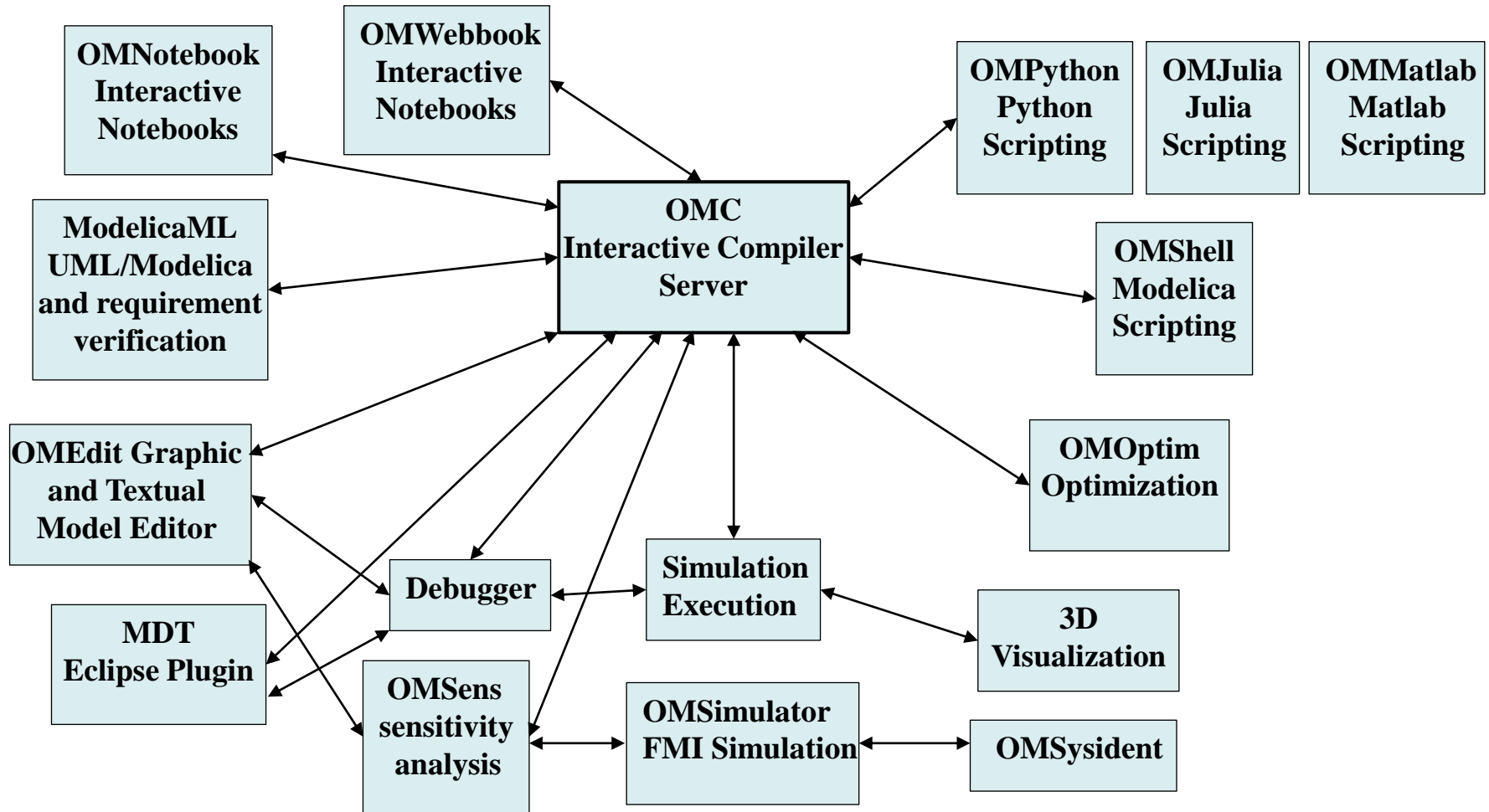
www.openmodelica.org

- Advanced Interactive Modelica compiler (OMC)
 - Supports most of the Modelica Language
 - **Modelica, Python, Julia, Matlab scripting**
- OMSimulator – FMI Simulation/Co-simulation
- Basic environment for creating models
 - **OMShell** – an interactive command handler
 - **OMNotebook** – a literate programming notebook
 - **MDT** – an advanced textual environment in Eclipse

- **OMEdit** graphic Editor
- **OMDebugger** for equations
- **OMOptim** optimization tool
- **OM Dynamic optimizer** collocation
- **ModelicaML** UML Profile
- **MetaModelica** extension
- **ParModelica** extension



The OpenModelica Tool Architecture



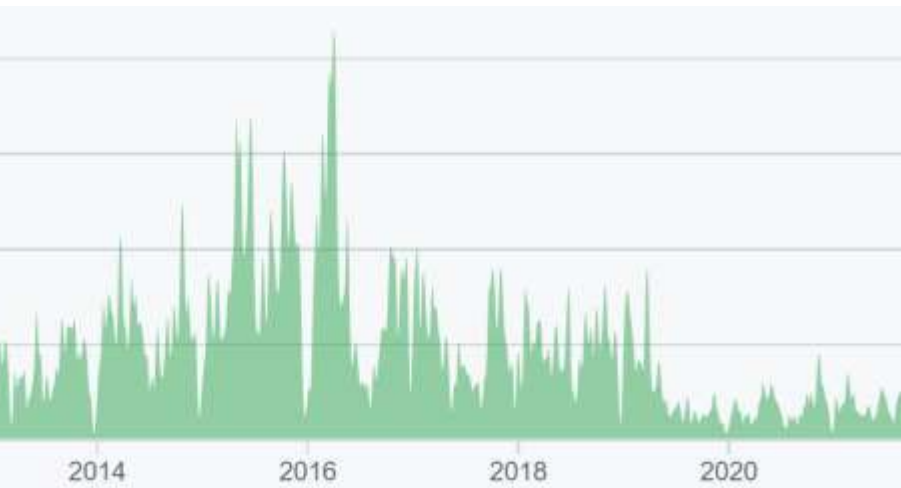
OSMC – International Consortium for Open Source Model-based Development Tools, 53 members Feb 2023

Founded Dec 4, 2007

Open-source community services

- Website and Support Forum
- Version-controlled source base
- Bug database
- Development courses
- www.openmodelica.org

Commits Statistics




Industrial members

- ABB AB, Sweden
- Bosch Rexroth AG, Germany
- Creative Connections, Prague
- DHI, Aarhus, Denmark
- Dynamica s.r.l., Cremona, Italy
- EDF, Paris, France
- Equa Simulation AB, Sweden
- Fraunhofer IWES, Bremerhaven
- Fraunhofer FCC, Gthenburg
- INRIA, Rennes, France
- ISID Dentsu, Tokyo, Japan
- Modelicon LLP, Bangalore, India
- JSOL Company, Japan
- Shanghai Duanyan Inf Tech China
- Perpetual Labs, London, UK
- Juelich, FZI, Germany
- Maplesoft, Canada
- Metroscope, France
- REUSE company, Spain
- RTE France, Paris, France
- Saab AB, Linköping, Sweden
- SmartFluidPower, Italy,
- TLK Thermo, Germany
- Sozhou Tongyuan, China
- SRON Space Ins Netherlands
- Talent Swarm, Spain
- Volvo Cars, Sweden
- VTI, Linköping, Sweden
- XRG Simulation, Germany

University members

- Augsburg University, Germany
- Australian Nation Univ., Australia
- FH Bielefeld, Bielefeld, Germany
- University of Bolivar, Colombia
- TU Braunschweig, Germany
- Univ of Buenos Aires, Argentina
- Univ Catalunya, Spain
- Chalmers Univ, Control, Sweden
- Chalmers Univ, Machine, Sweden
- TU Darmstadt, Germany
- TU Delft, The Netherlands
- TU Dresden, Germany
- Université Laval, Canada
- TU Hamburg/Harburg Germany
- KU Leuven, Leuven, Belgium
- Univ Linnaeus, Sweden
- IIT Bombay, Mumbai, India
- Linnaeus University, Sweden
- Linköping University, Sweden
- Univ of Maryland, Syst Eng USA
- Univ of Maryland, CEEE, USA
- Politecnico di Milano, Italy
- Politecnica Catalunya Spain
- Mälardalen University, Sweden
- RPI, Troy, USA
- Univ Pisa, Italy
- Univ College SouthEast Norway
- Vanderbilt Univ, USA

Spoken-Tutorial step-by-step OpenModelica and Modelica Tutorial Using OMEdit. Link from www.openmodelica.org



Login Create an account

Spoken Tutorial Software Training Creation News Forums About Statistics

HOME DOWNLOAD TOOLS & APPS

Search Tutorials

https://spoken-tutorial.org/tutorial-search/?search_foss=OpenModelica&search_language=English

To learn about Modelica, read a [book](#) or a [tutorial](#) about [Modelica®](#).
Interactive step-by-step beginners Modelica [on-line spoken tutorials](#)
Interactive [OMWebbook](#) with examples of Modelica textual modeling

OpenModelica is an open source modelling and simulation environment intended for industrial and academic usage. It is an object oriented declarative multi domain modelling language for complex systems. This environment can be used to work for both steady state as well as dynamic systems. Attractive strategy when dealing with design and optimization problems. As all the equations are solved simultaneously it doesn't matter whether the unknown variable in an input or output variable. [Read more](#)

About 12 results found.

 [Instruction Sheet](#)



1. Introduction to OMEdit

Foss : OpenModelica - English

Outline: Introduction to OpenModelica Introduction to OMEdit Perspectives in OMEdit Browsers in OMEdit View icons in OMEdit Open a Class from Libraries Browser Checking for correctness..

Basic

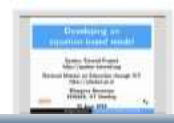



2. Examples through OMEdit

Foss : OpenModelica - English

Outline: Expand Modelica library Expand Electrical library Expand Analog library Open Rectifier Class Compare the values of IDC & Losses time vs Losses plot Expand Mechanics library ..

Basic

3. Developing an equation-based model

Foss : OpenModelica - English

Outline: Introduction to OMEdit Declaration of variables and equations Simulation of a model in

Basic


OMNotebook Electronic Notebook with DrModelica

- Primarily for teaching
- Interactive electronic book
- Platform independent

Commands:

- *Shift-return (evaluates a cell)*
- File Menu (open, close, etc.)
- Text Cursor (vertical), Cell cursor (horizontal)
- Cell types: text cells & executable code cells
- Copy, paste, group cells
- Copy, paste, group text
- Command Completion (shift-tab)



OMnotebook Interactive Electronic Notebook

Here Used for Teaching Control Theory

1 Kalman Filter

Often we don't have access to the internal states of a system. We have to reconstruct the state of the system based on measurements. The idea with an observer is that we feedback the difference between the measurement and the estimation. If the estimation is correct then the difference should be zero.

Another difficulty is that the measured quantities often contain noise.

$$\begin{Bmatrix} \hat{x} \\ \hat{y} \end{Bmatrix}$$

Here e denotes a disturbance in the input signal that can be evaluated by the difference

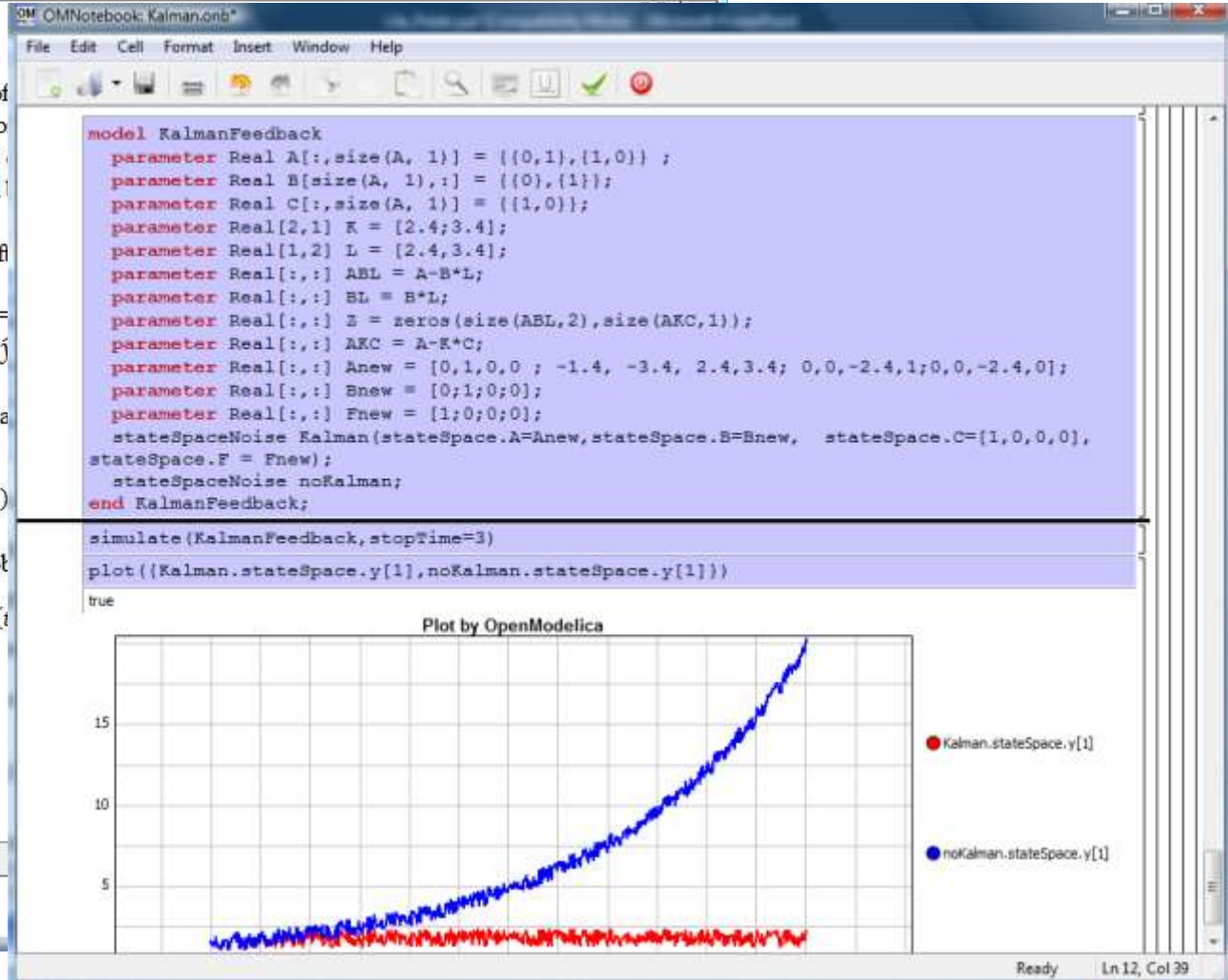
$$K(y(t) - \hat{y}(t))$$

By using this quantity as feedback we obtain the observer

$$\dot{\hat{x}} = A\hat{x}(t) + Bu(t) + K(y(t) - \hat{y}(t))$$

Now form the error as

The differential error is



Mathematical Typesetting in OMNotebook and OMWebbook

OMNotebook supports Latex formatting for mathematics

The screenshot shows the OMNotebook application window titled "OMNotebook: (untitled)*". The menu bar includes File, Edit, Cell, Format, Insert, Window, and Help. The toolbar contains various icons for file operations, editing, and formatting. The main content area displays the following text:

1 Chemical Reaction Kinetics of Hydrogen Iodine

A chemical reaction represented by a reaction formula transforms the chemical species on the left-hand side of the arrow, called reactants, to the species on the right-hand side of the arrow, called products:
reactants -> products

Consider a chemical reaction between hydrogen gas and iodine gas to form hydrogen iodine:
$$\text{H}_2 + \text{I}_2 \rightleftharpoons 2\text{HI}$$

We can formulate the differential equations for the whole reaction system as below.

$$\frac{d}{dt} [\text{H}_2] = k_2 \cdot [\text{HI}^2] - k_1 \cdot [\text{H}_2] \cdot [\text{I}_2]$$
$$\frac{d}{dt} [\text{I}_2] = k_2 \cdot [\text{HI}^2] - k_1 \cdot [\text{H}_2] \cdot [\text{I}_2]$$
$$\frac{d}{dt} [\text{HI}] = 2k_2 \cdot [\text{H}_2] \cdot [\text{I}_2] - 2k_1 \cdot [\text{HI}^2]$$

Below the equations, the LaTeX source code is visible:

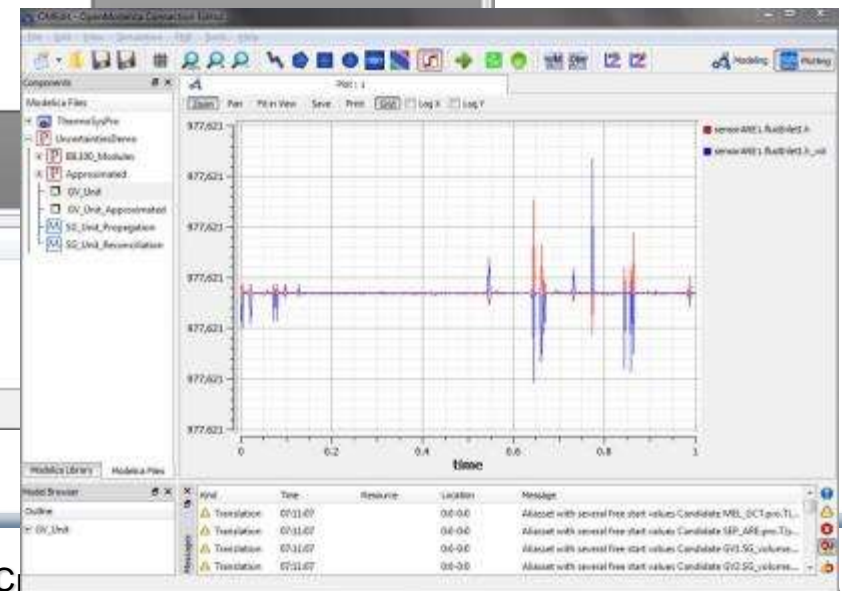
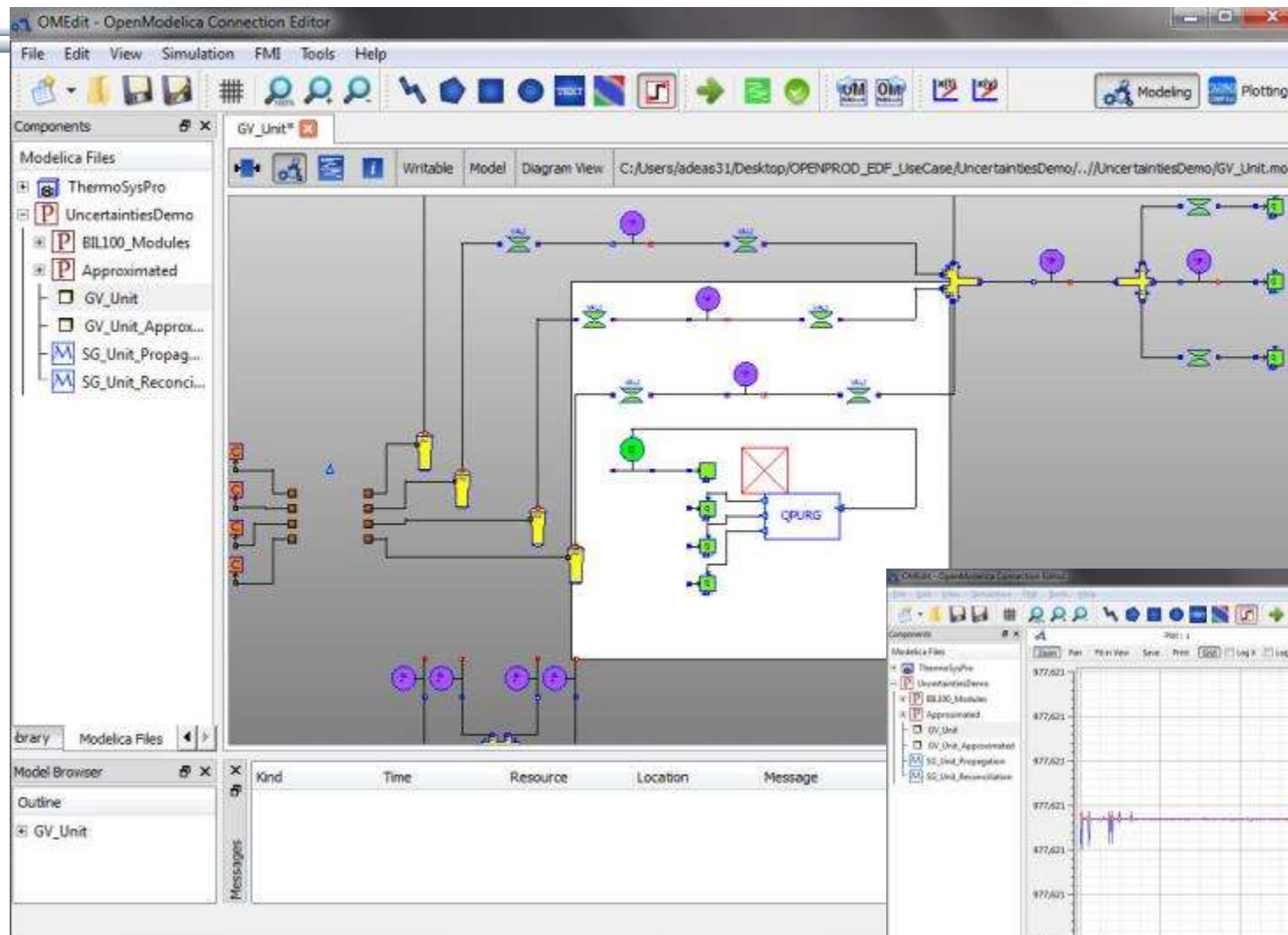
```
\documentclass[12pt]{article}
\begin{document}
\thispagestyle{empty}
\[\frac{\mathrm{d}}{\mathrm{d}t}[\mathrm{H}_2] = k_2 \cdot [\mathrm{HI}^2] - k_1 \cdot [\mathrm{H}_2] \cdot [\mathrm{I}_2]\]
\[\frac{\mathrm{d}}{\mathrm{d}t}[\mathrm{I}_2] = k_2 \cdot [\mathrm{HI}^2] - k_1 \cdot [\mathrm{H}_2] \cdot [\mathrm{I}_2]\]
\[\frac{\mathrm{d}}{\mathrm{d}t}[\mathrm{HI}] = 2k_2 \cdot [\mathrm{H}_2] \cdot [\mathrm{I}_2] - 2k_1 \cdot [\mathrm{HI}^2]\]
\end{document}
```

The status bar at the bottom right indicates "Ready".

Contents in
OMWebbook
Generated from
OMNotebook

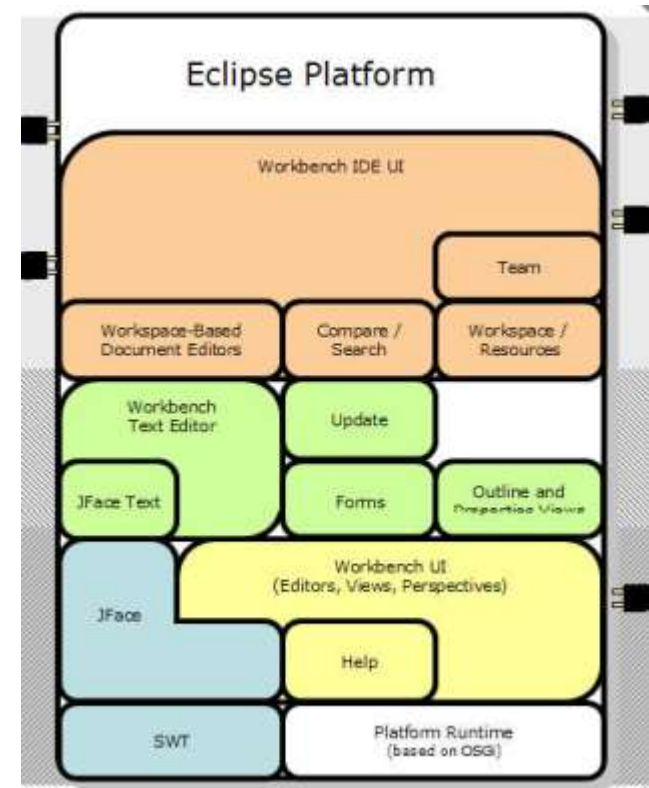
Latex instructions
can be hidden by
double clicking the
Cell in tree view

OpenModelica Environment Demo



OpenModelica MDT – Eclipse Plugin

- Browsing of packages, classes, functions
- Automatic building of executables;
separate compilation
- Syntax highlighting
- Code completion,
Code query support for developers
- Automatic Indentation
- Debugger
(Prel. version for algorithmic subset)



OpenModelica MDT: Code Outline and Hovering Info

The screenshot displays the OpenModelica MDT Eclipse IDE interface. The top toolbar includes standard Eclipse actions like File, Edit, Navigate, Search, Project, Run, Field Assist, Window, and Help. The left sidebar shows the 'Modelica Projects' tree with a list of files including rml2sig, runtime, scripts, test_codegen, tools, VC7, and Absyn.mo. The main editor window shows the 'Absyn.mo' file with a code snippet. A yellow tooltip is visible over the 'getCrefFromExp' function, providing a description: 'Returns a flattened list of the component references in an expression'. The bottom-left pane shows the 'Outline' view with a tree structure of the code. The bottom-right pane shows the 'Problems' view with a list of errors. A large blue callout box with white text is overlaid on the right side of the image, pointing to the tooltip and the Problems view.

Modelica - OpenModelica/Compiler/Absyn.mo - Eclipse SDK

File Edit Navigate Search Project Run Field Assist Window Help

Modelica Projects

- rml2sig
- runtime
- scripts
- test_codegen
- tools
- VC7
- Absyn.mo 3116 2008-02-04 14:44 krsta
- Absyn 3116 2008-02-04 14:44 krsta
- Algorithm.mo 2992 2007-12-22 22:17 adrho
- Builtin.mo 3585 2008-05-22 07:03 adrho
- Ceval.mo 3605 2008-05-27 02:48 adrho
- ClassInf.mo 3496 2008-04-23 11:59 krsta
- ClassLoader.mo 3193 2008-02-15 05:17 adrho
- Codegen.mo 3585 2008-05-22 07:03 adrho
- Connect.mo 3584 2008-05-22 06:45 adrho
- Constants.mo 3011 2007-12-22 22:36 adrho
- Convert.mo 3496 2008-04-23 11:59 krsta

Absyn.mo

```
case (MATRIX(matrix = exp11))
  local list<list<list<ComponentRef>>> res1;
  equation
    res1 = Util.listListMap(exp11, getCrefFromExp);
    res2 = Util.listFlatten(res1);
    res = Util.listFlatten(res2);
  then
    res;
case (RANGE(start = e1, step = SOME(e3), stop = e2))
  equation
    l1 = getCrefFromExp(e1);
    l2 =
      function getCrefFromExp "function: getCrefFromExp
        Returns a flattened list of the
        component references in an expression"
        input Exp inExp;
        then
          output list<ComponentRef> outComponentRefList;
        algorithm
          outComponentRefList:=matchcontinue inExp
          local
            l1 =
              ComponentRef cr;
            l2 =
              listAppend(l1, l2);
          then
```

function getCrefFromExp "function: getCrefFromExp
Returns a flattened list of the
component references in an expression"
input Exp inExp;
then
output list<ComponentRef> outComponentRefList;
algorithm
outComponentRefList:=matchcontinue inExp
local
l1 =
ComponentRef cr;
l2 =
listAppend(l1, l2);
then

Outline

- Absyn
 - ADD
 - ALG_ASSIGN(Exp assignComponent, Exp value)
 - ALG_BREAK
 - ALG_CATCH(list<AlgorithmItem> catchBody)
 - ALG_EQUALITY(Algorithm equ)
 - ALG_FAILURE(Algorithm equ)
 - ALG_FOR(ForIterators iterators, list<AlgorithmItem> forBo
 - ALG_GOTO(String labelName)
 - ALG_IF(Exp ifExp, list<AlgorithmItem>
 - ALG_LABEL(String labelName)
 - ALG_NORETCALL(ComponentRef
 - ALG_RETURN
 - ALG_THROW
 - ALG_TRY(list<AlgorithmItem> tryBody
 - ALG_WHEN_A(Exp whenStmt, list<Alg

Problems

113 errors, 0 warnings, 0 infos

Description

Errors (100 of 113 items)

- The identifier at start and end are different
- The identifier at start and end are different
- The identifier at start and end are different, par

Identifier Info on Hovering

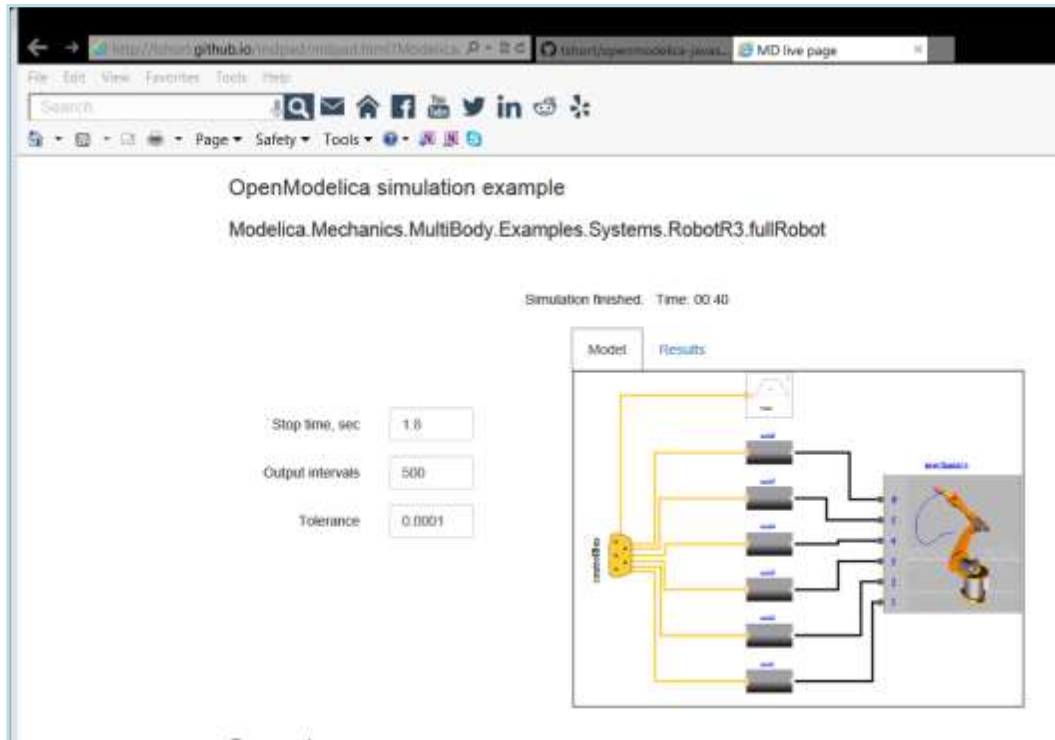
Code Outline for easy navigation within Modelica files

ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig
ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig
ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig
ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig
ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig
ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig
ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig
ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig
ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig
ken'; on line	rml2sig.mo	OpenModelica/tools/rml2sig

64M of 254M

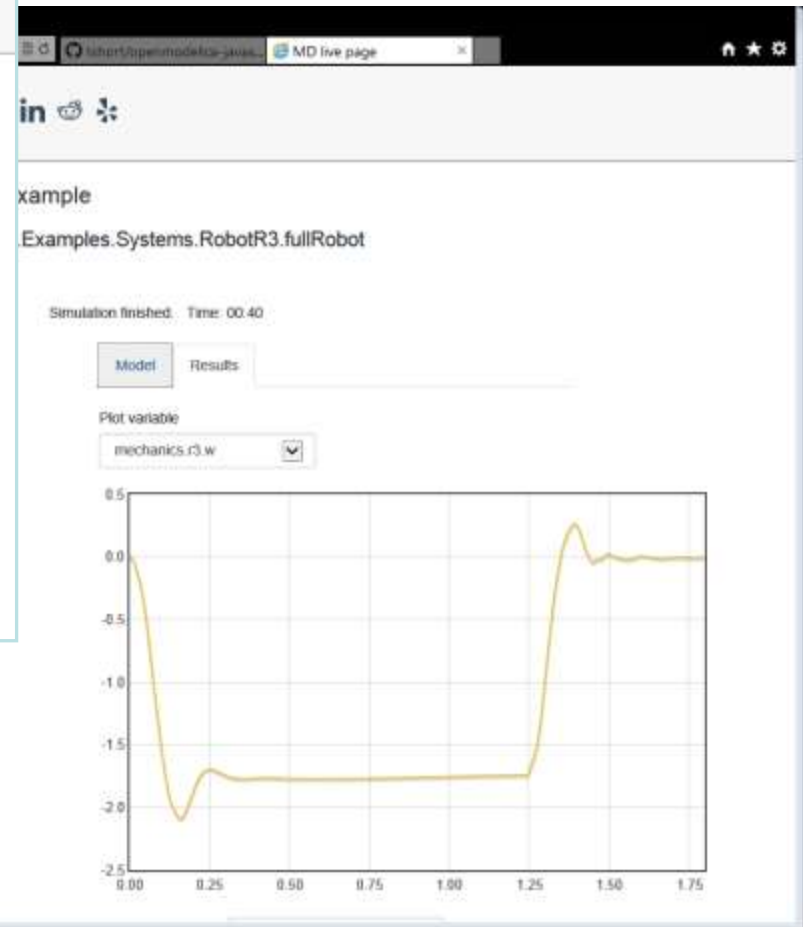
Ctrl Contrib (Bottom)

OpenModelica Simulation in Web Browser Client



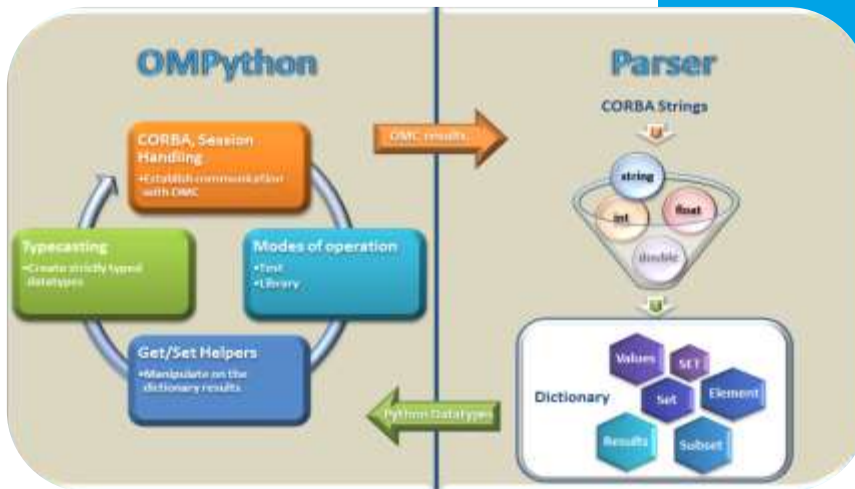
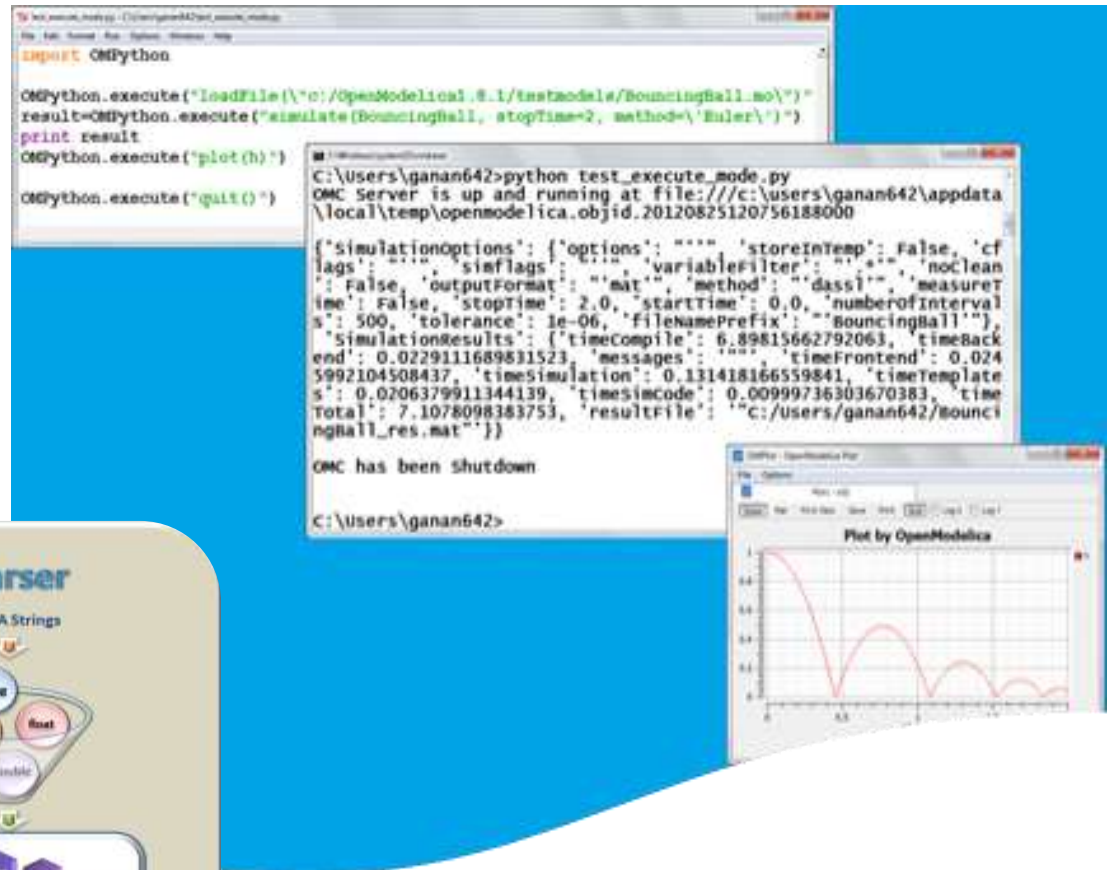
OpenModelica compiles to efficient Java Script code which is executed in web browser

MultiBody RobotR3.FullRobot



OMPython – Python Scripting with OpenModelica

- Interpretation of Modelica commands and expressions
- Interactive Session handling
- Library / Tool
- Optimized Parser results
- Helper functions
- Deployable, Extensible and Distributable



OMJulia – Julia Scripting with OpenModelica

- Interpretation of Modelica commands and expressions from Julia, transfer of data
- Control design using Julia control package together with OpenModelica
- Interactive Session handling
- Library / Tool
- Separately downloadable. be run with OpenModelica 1.13.2 or later
- Works with Jupyter notebooks

Control example with OMJulia in Jupyter notebooks

Use of Modelica + Julia in Process Systems Engineering Education

Complex models of "Seborg reactor"

Bernt Lie*, Arunkumar Palanisamy**, Peter Fritzson**

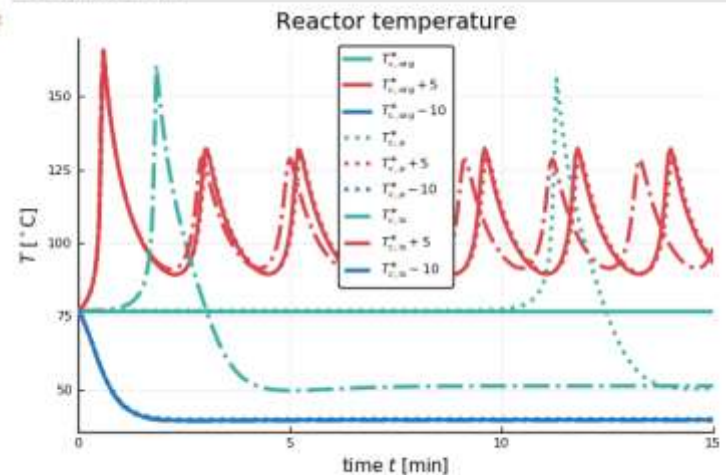
*University of South-Eastern Norway, Norway

**University of Linköping, Sweden

Introducing packages

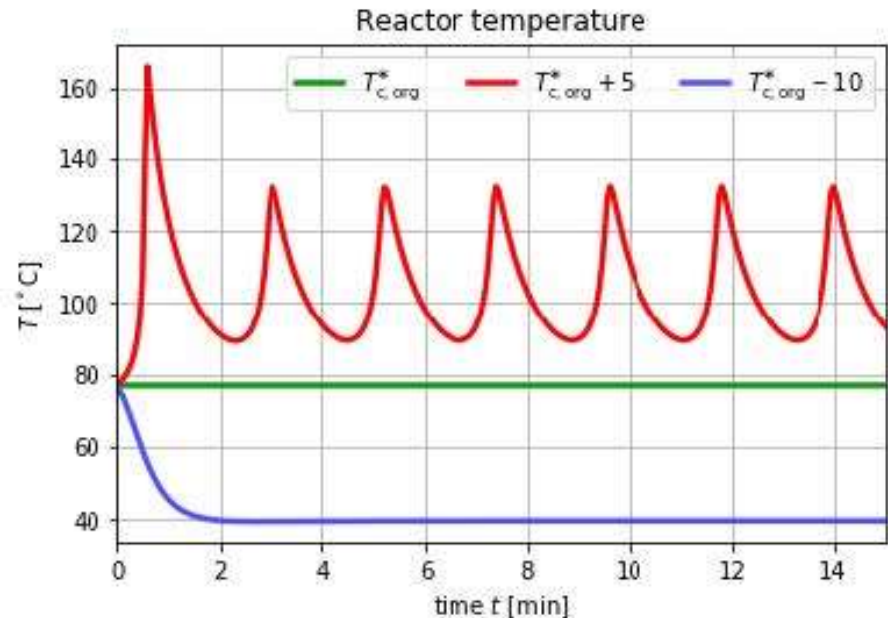
```
In [1]: # Pkg.add("Plots") -- we assume that this step already has been carried out
using Plots; pyplot()
using LaTeXStrings
using DataFrames
using OMJulia
using DifferentialEquations
```

Out[9]:



OMMatlab – Matlab Scripting with OpenModelica

- Interpretation of Modelica commands and expressions from Matlab, transfer of data
- Interactive Session handling
- Library / Tool
- Separately downloadable. be run with OpenModelica 1.13.0 or later
- Similar API functions as in OMJulia and OMPython
- Complete API e.g. useful for control system design



Experimental OpenModelica Compiler in Julia

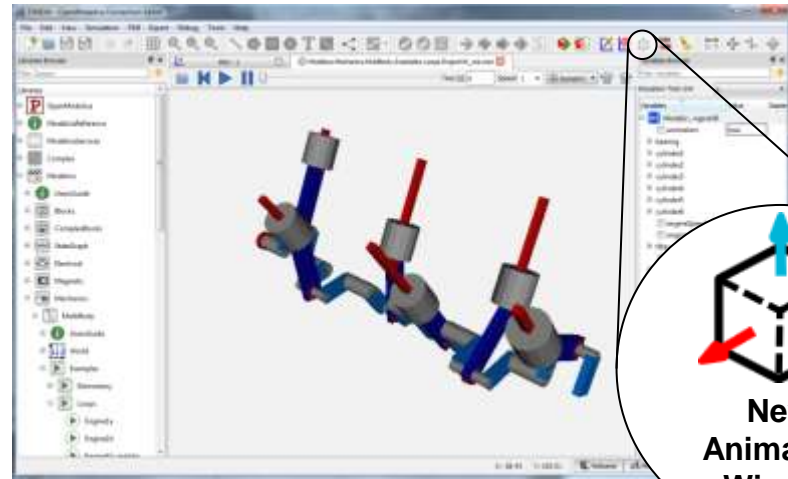
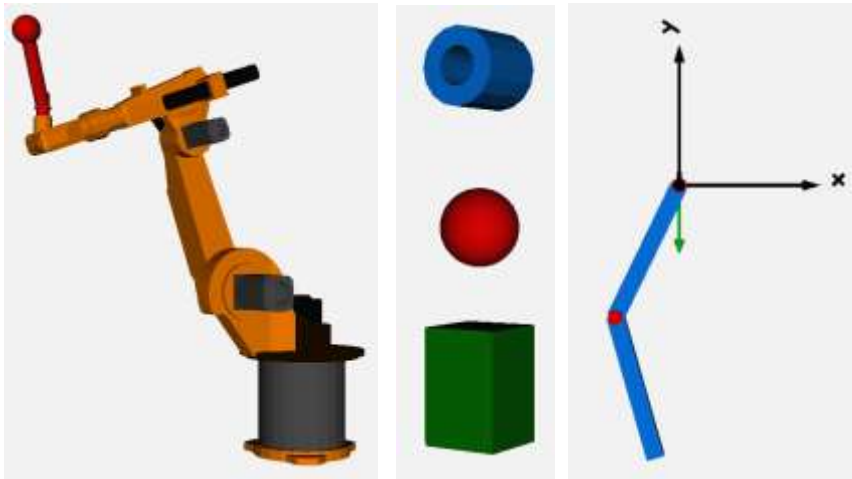
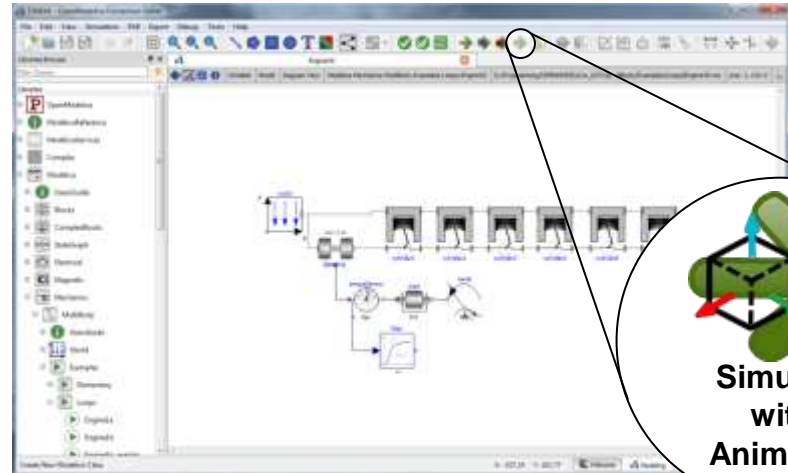
John Tinnerholm and Adrian Pop)

- OpenModelica.jl: modular and extensible Modelica compiler framework in Julia
- Developed a preliminary MetaModelica to Julia translator
- **Translated the high-performance frontend.**
- **Able to execute and translate MetaModelica functions**
- **Able to simulate discrete-hybrid systems + regular continuous systems**
- **Experimental backends developed**
 - Targeting DifferentialEquations.jl and ModelingToolkit.jl (MTK)
 - Completed causalization sorting, matching.
 - Integrated LightGraphs.jl package, DAG representation of the hybrid DAE
 - Integrated Plots.jl for interactive plotting and animation
 - Integrated the Reduce Computer Algebra system for automatic symbolic manipulation and symbolic derivation.
 - Integration with Sundials. IDAS used for numerical integration
- Further performance **tuning needed**
- Currently experimenting with **variable-structure systems**

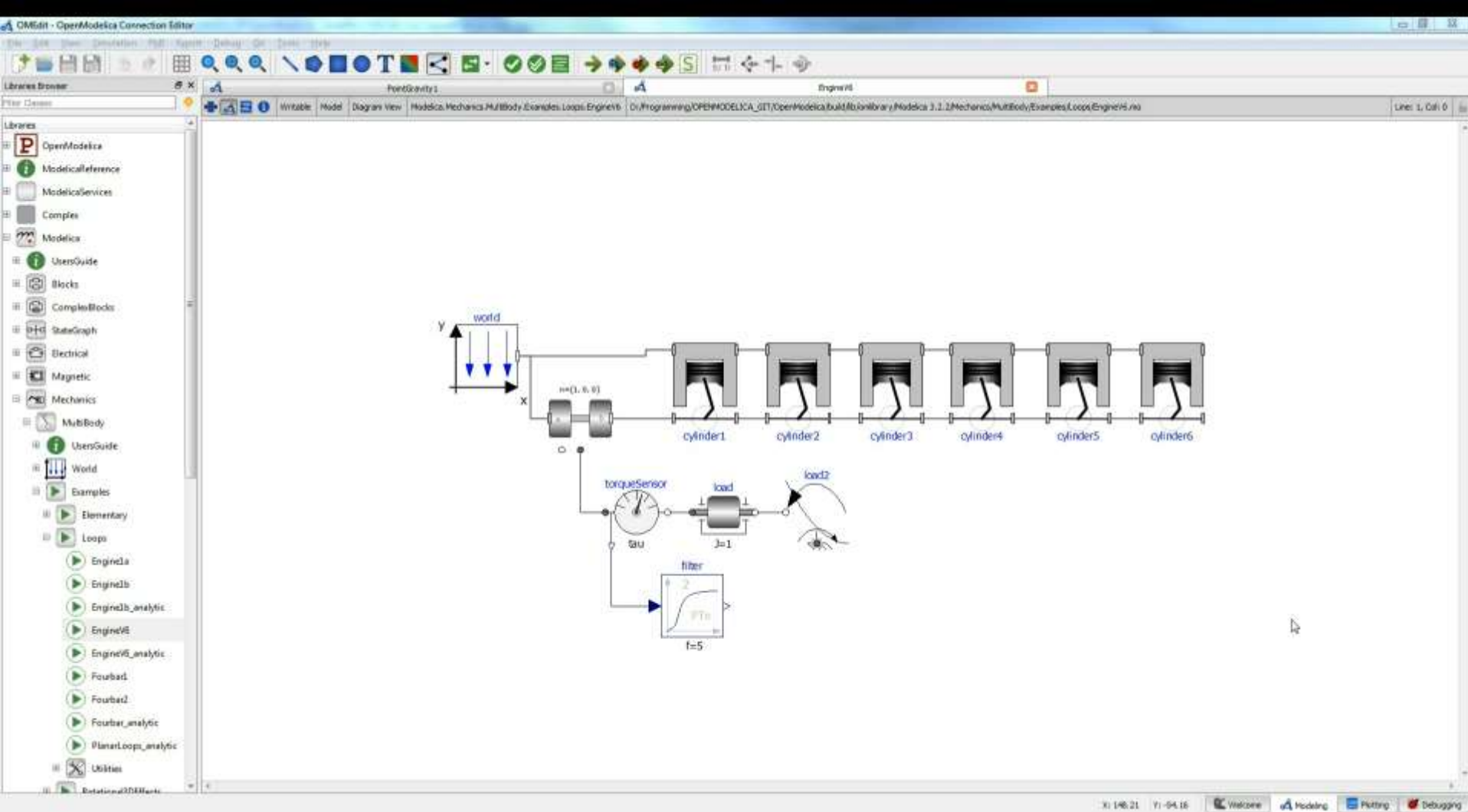


OMEdit 3D Visualization of Multi-Body Systems

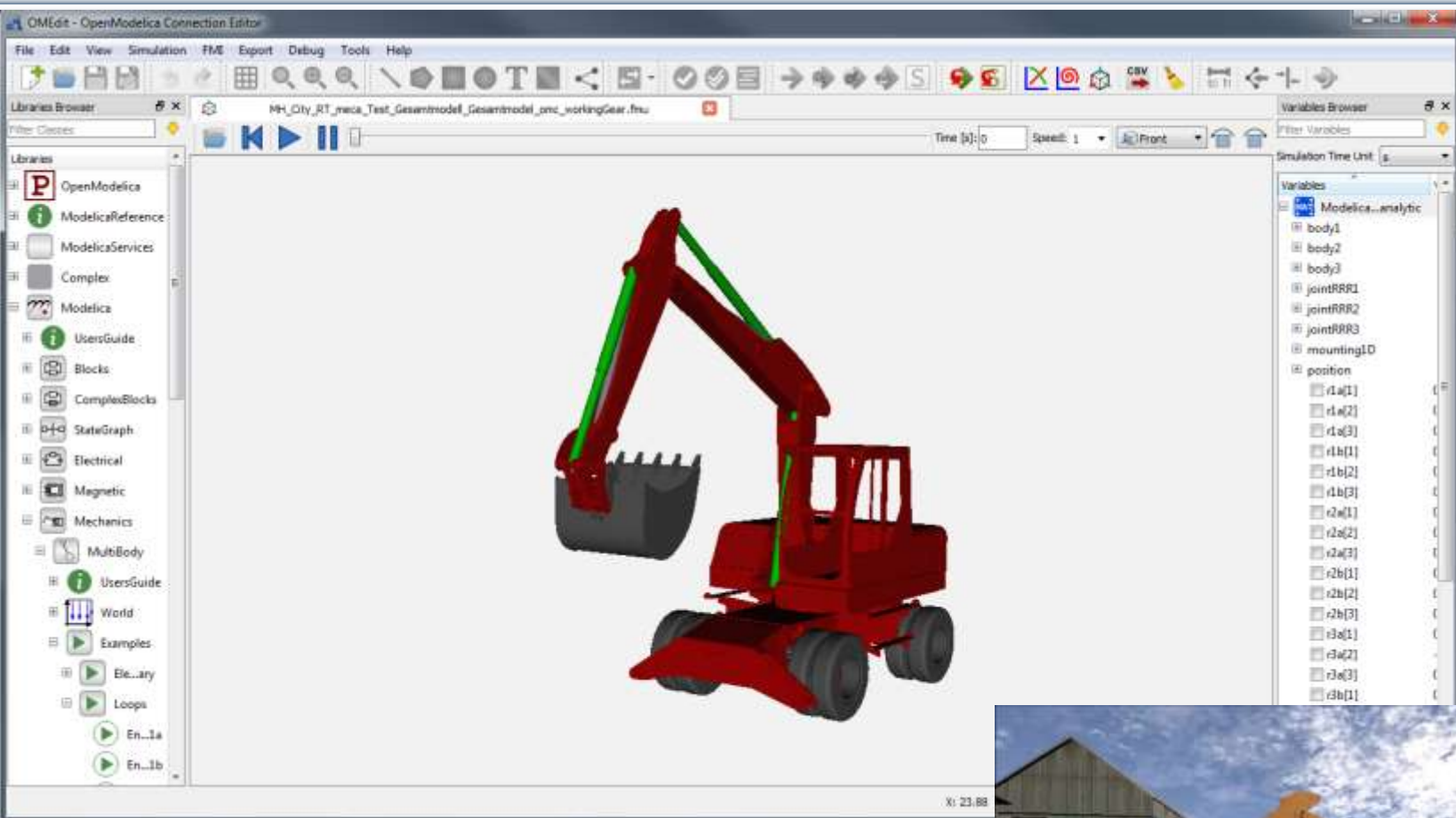
- Built-in feature of OMEdit to animate MSL-Multi-Body shapes
- Visualization of simulation results
- Animation of geometric primitives and CAD-Files



OpenModelica 3D Animation Demo (V6Engine and Excavator)



OpenModelica 3D Animation – Excavator

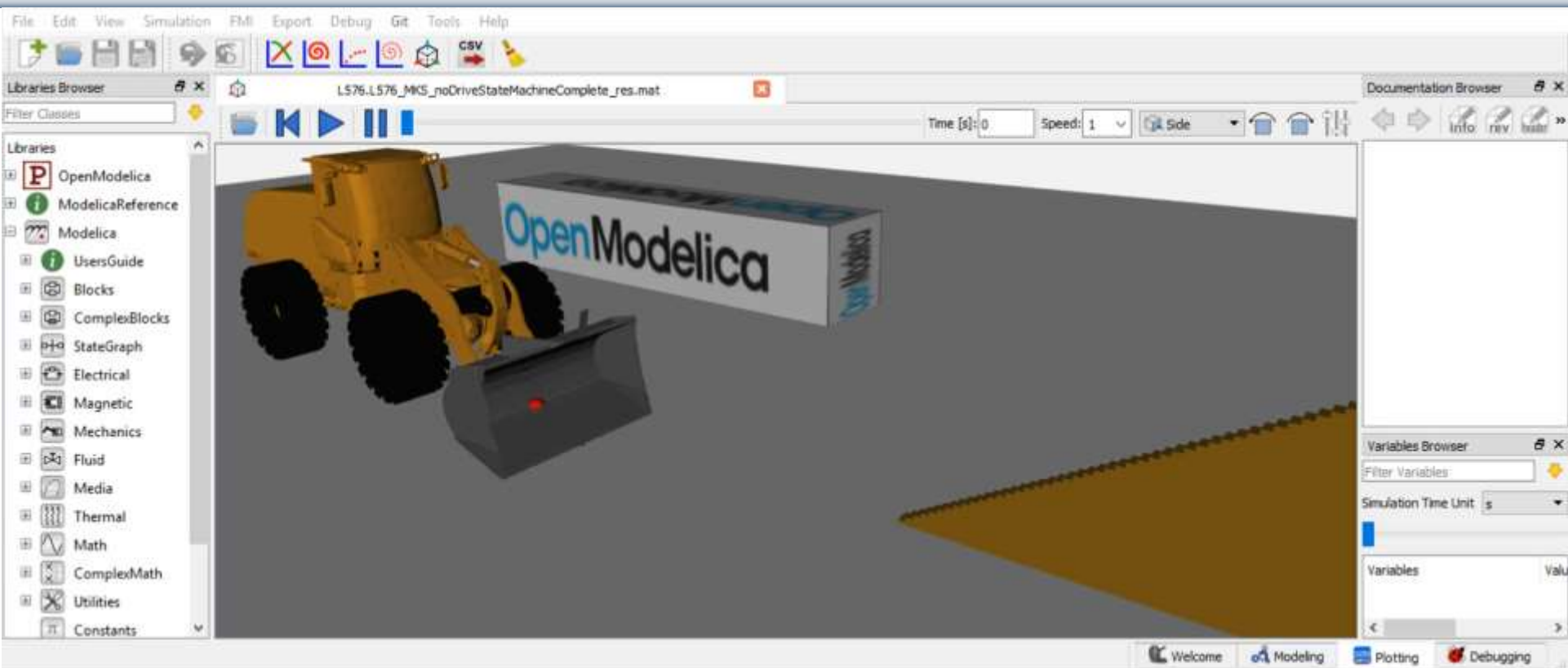


Connection with Unity

Courtesy of Volker Waurich - TU Dresden

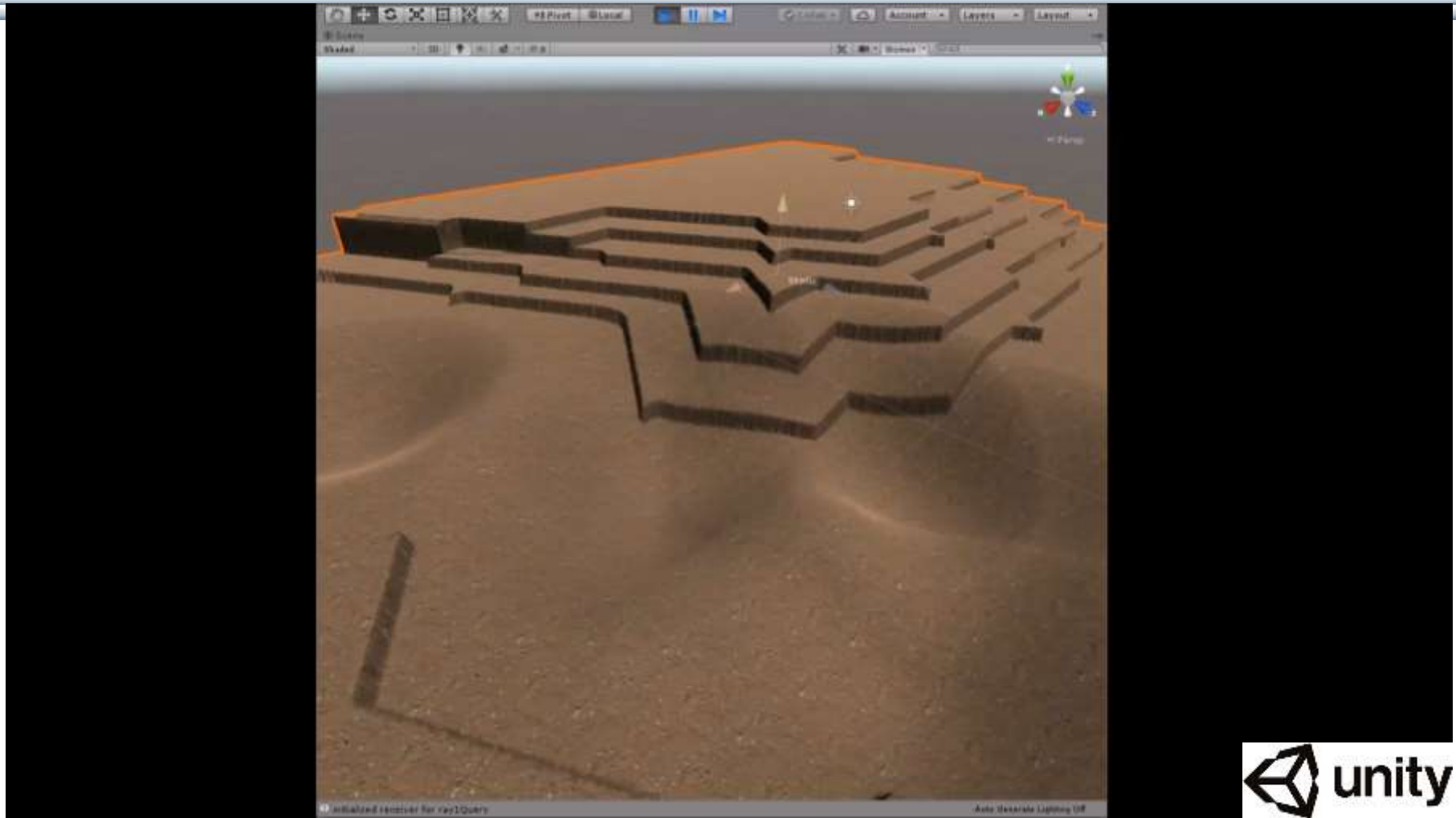


OpenModelica 3D Animation – WheelLoader



Courtesy of Volker Waurich - TU Dresden

OpenModelica 3D Animation – BouncingBall

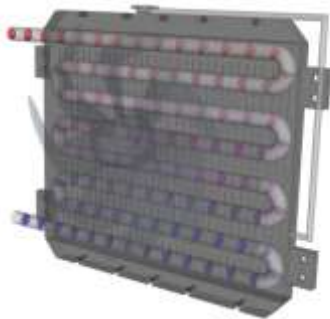


Collision detection in Unity

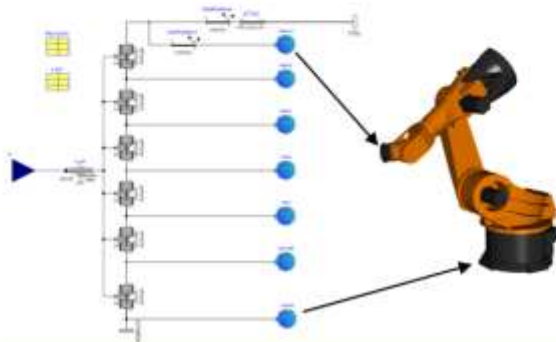
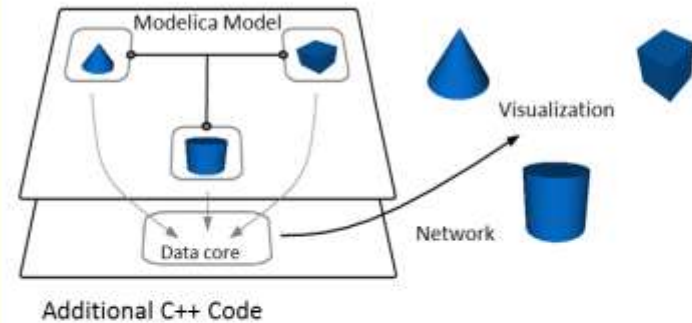
Courtesy of Volker Waurich - TU Dresden

Visualization using Third-Party Libraries: DLR Visualization Library

- Advanced, model-integrated and vendor-unspecific visualization tool for Modelica models
- Offline, online and real-time animation
- Video-export function
- Commercial library, feature reduced free Community Edition exists



Integration of visualizer blocks into the model and Communication to an external viewer (SimVis)

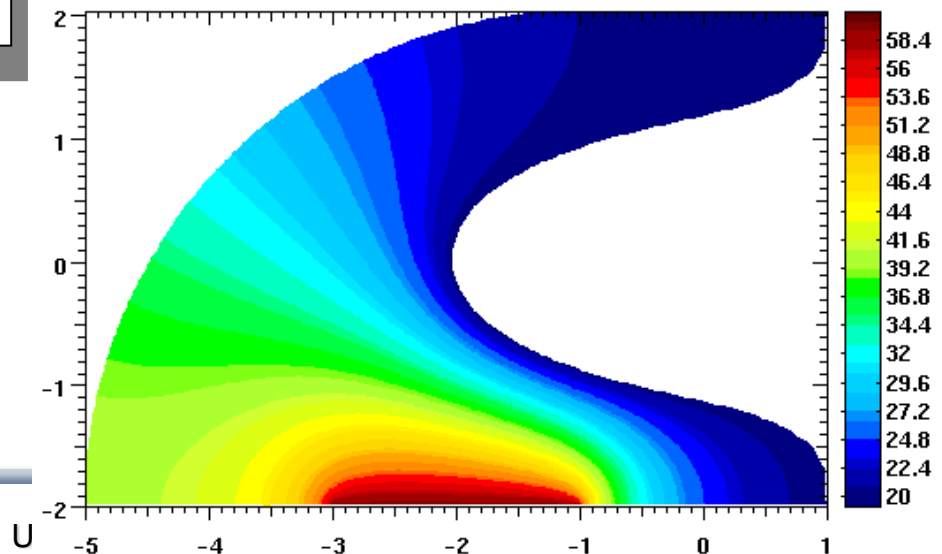
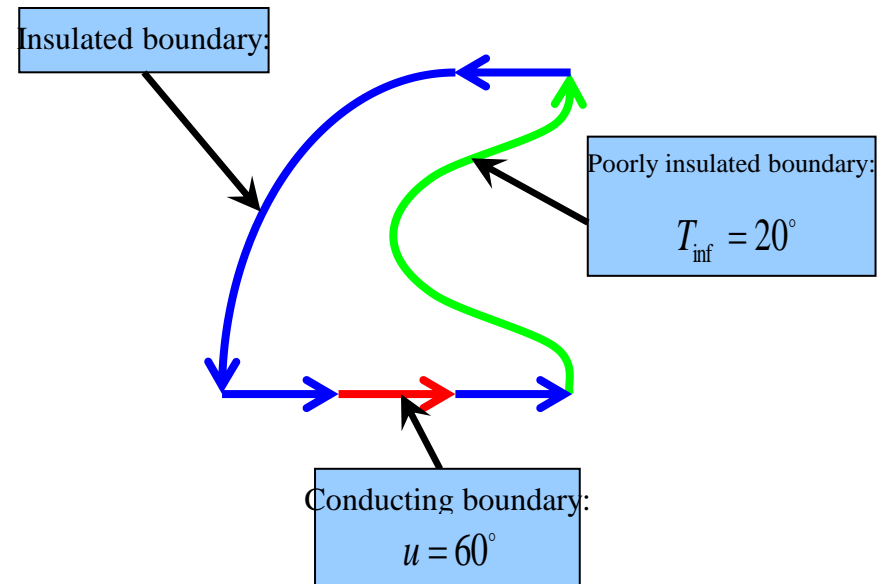


Courtesy of Dr. Tobias Bellmann (DLR)

Extending Modelica with PDEs for 2D, 3D flow problems – Research

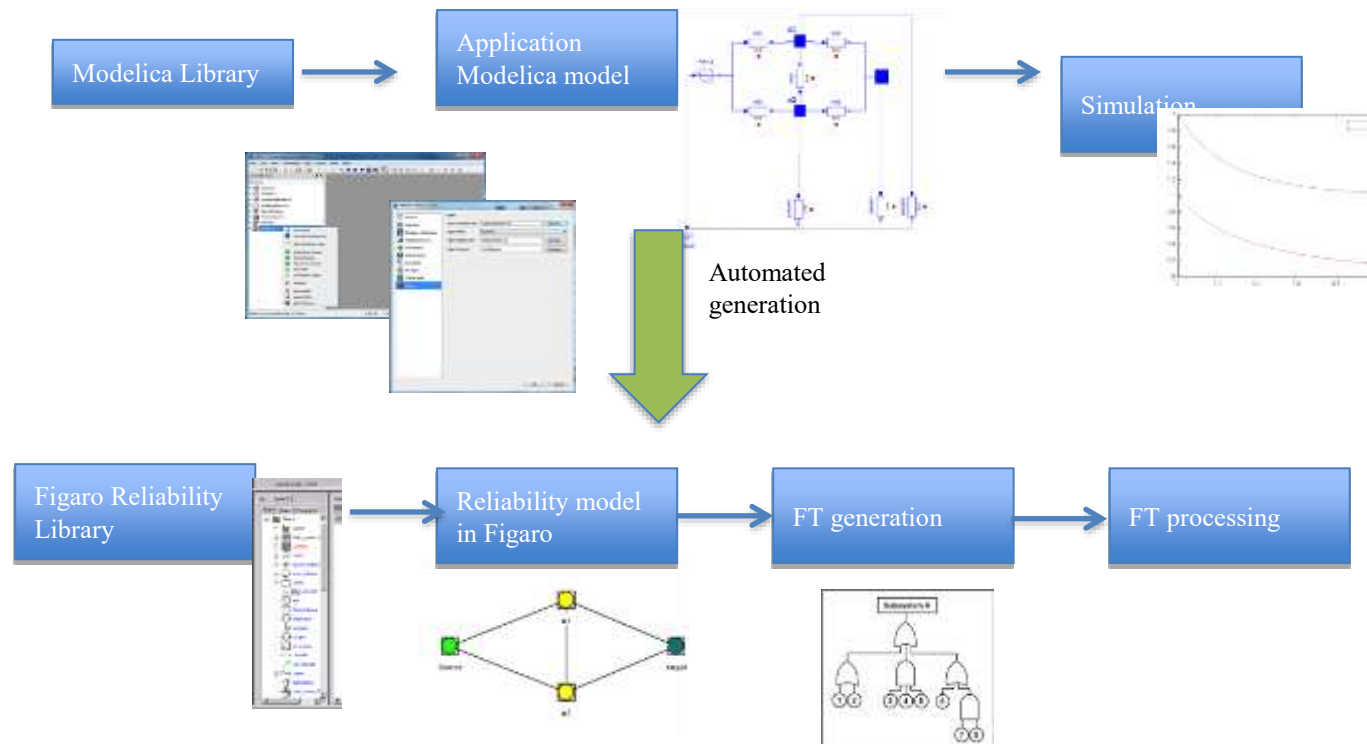
```
class PDEModel
  HeatNeumann h_iso;
  Dirichlet h_heated(g=50);
  HeatRobin h_glass(h_heat=30000);
  HeatTransfer ht;
  Rectangle2D dom;
equation
  dom.eq=ht;
  dom.left.bc=h_glass;
  dom.top.bc=h_iso;
  dom.right.bc=h_iso;
  dom.bottom.bc=h_heated;
end PDEModel;
```

Prototype in OpenModelica 2005
PhD Thesis by Levon Saldamli
www.openmodelica.org
Currently not operational



Failure Mode and Effects Analysis (FMEA) in OM

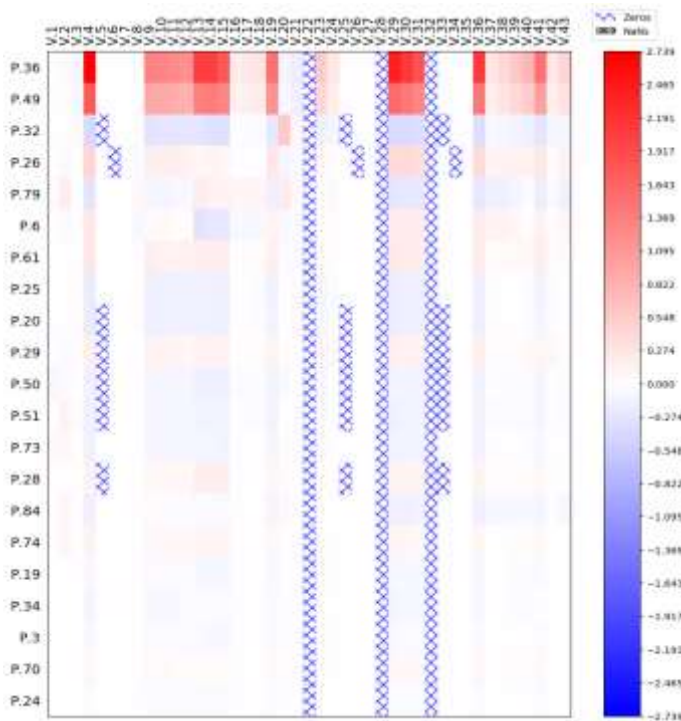
- Modelica models augmented with reliability properties can be used to generate reliability models in Figaro, which in turn can be used for static reliability analysis
- Prototype in OpenModelica integrated with the Figaro tool.



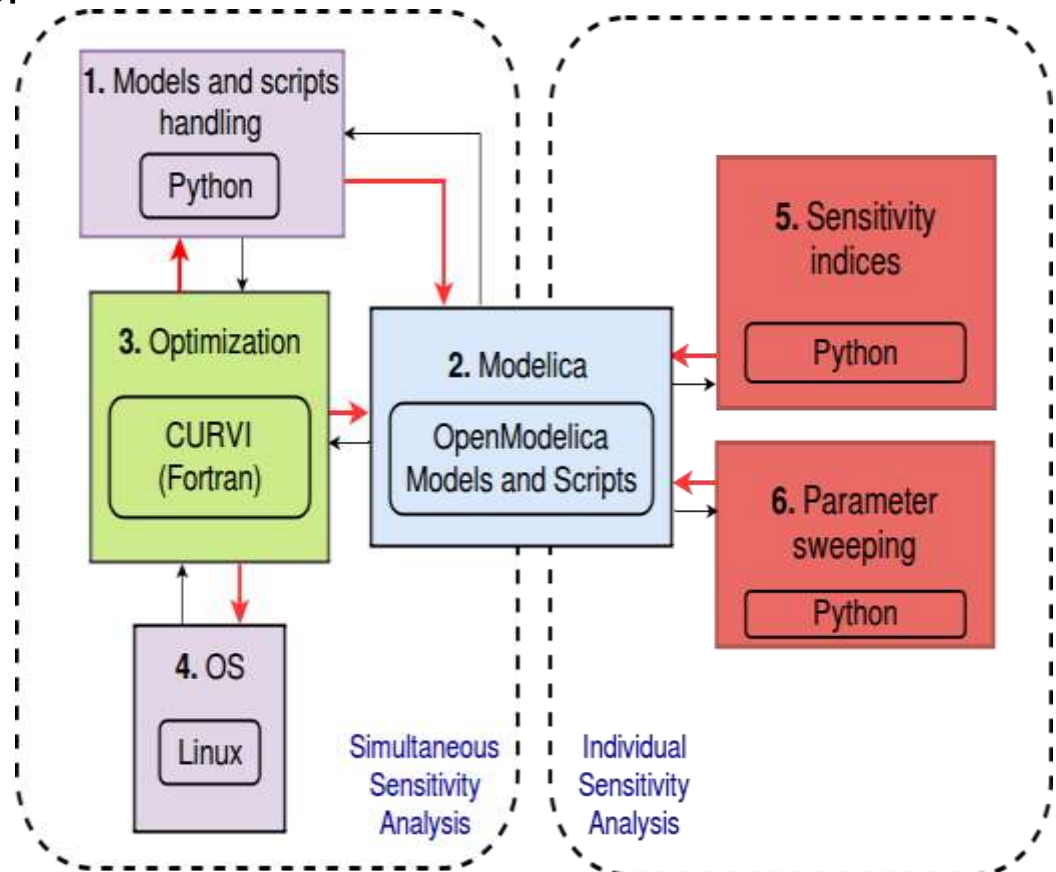
OMSens – Multi-Parameter Sensitivity Analysis and Robust Optimization

- Individual and simultaneous multi-parameter analysis
- Optimization-based simultaneous analysis
- Robust derivative free optimizer

Heatmap visualization



Tool architecture



OMSysIdent – System Parameter Identification

- OMSysIdent is a module for parameter estimation of behavioral models (wrapped as FMUs) on top of the OMSimulator API.
- Identification of the parameter values is typically based on measurement data
- It uses the Ceres solver (<http://ceres-solver.org/>) for the optimization task.

OMOptim – Optimization (1)

Model structure

Model Variables

Optimized
parameters

Optimized
Objectives

MinEIT

File Project Problem Display Tools

Models Problems

Name

- Pc
- Va
- Vb
- Ia
- Ib
- Ic
- Ea
- Eb
- Ec
- coutdinvestissement
- gaincoperationnel
- EmCO2PAC1
- Ca
- Cb
- Cc
- Puissae
- Puissbe
- Puissce
- n
- na
- nb
- nc
- OCb
- OChp
- coutdefonctavecPAC
- T0SygmaA
- T0SygmaB
- T0SygmaECS
- COPECSsystem
- PElecECSMax
- ⊕ EchIAOutCold
- ⊕ Sortieeffluents
- ⊕ echA
- ⊕ Sourcemod
- ⊕ scenarioEchA
- ⊕ scenarioPACA
- ⊕ echB

Project Optimization EI EI result

Variables

Filter :

Name	Value	Description
global.sourceaudeville.h	1,18294e+06	[J/kg]
global.sourceaudeville.flowPort.p	100000	
global.sourceInEchColdB.h	1,41347e+06	[J/kg]
global.sourceInEchColdB.flowPort.p	100000	
global.sourceInEchColdB.debit	12,78	[kg/s]
global.sourceEffluentsECS.h	1,35495e+06	[J/kg]
global.sourceEffluentsECS.flowPort.p	100000	
global.sourceEffluentsECS.etat	1	
global.sourceEffluentsECS.debit1	0	
global.sourceEffluentsECS.debit	1	[kg/s]
global.sourceEffluentsB.h	1,35495e+06	[J/kg]
global.sourceEffluentsB.flowPort.p	100000	
global.sourceEffluentsB.etat	1	
global.sourceEffluentsB.debit	1,22612	[kg/s]
global.sourceEffluentsA.h	1,35495e+06	[J/kg]
global.sourceEffluentsA.flowPort.p	100000	
global.sourceEffluentsA.etat	1	
global.sourceEffluentsA.debit	0,601234	[kg/s]
global.scenariosourceEaudeville.debit	0,940001	[kg/s]
global.scenariodepartB.z	0	

Optimized variables

Name	Description	Opt Minimu
global.sourceEffluentsB.debit	[kg/s]	0
global.sourceEffluentsA.debit	[kg/s]	0
global.scenarioPACB.MySpecPcomp		0
global.scenarioPACA.MvSpecPcomp		0

Scanned variables

Name	Description	Scan Minimum	Scan Maximum	n
------	-------------	--------------	--------------	---

Optimization objectives

Name	Description	Direction	M
global.gaincoperationnel		Maximize	0
global.coutdinvestissement		Minimize	0

Variables Components Launch

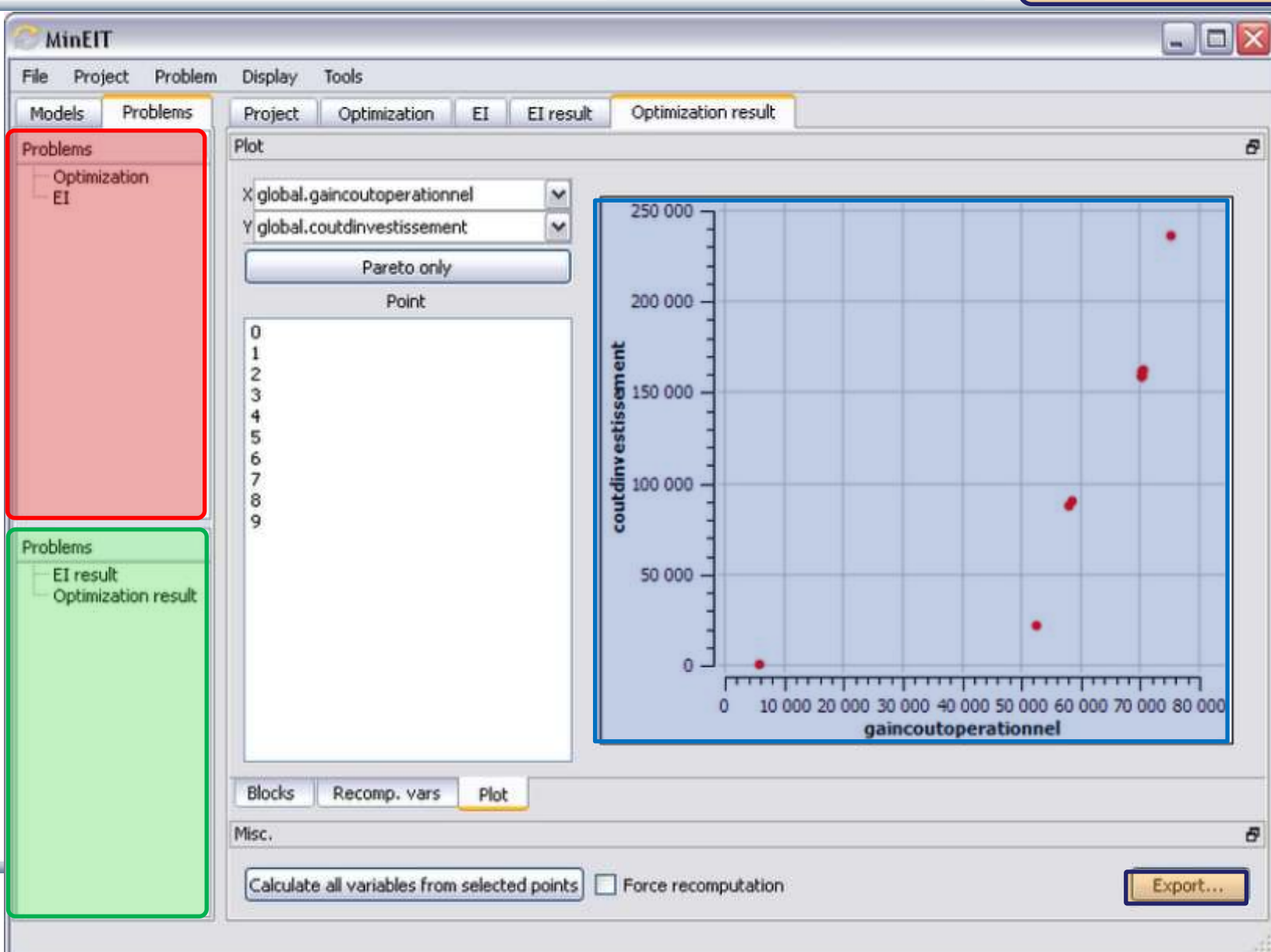
Problems

OMOptim – Optimization (2)

Solved problems

Result plot

Export result data .csv

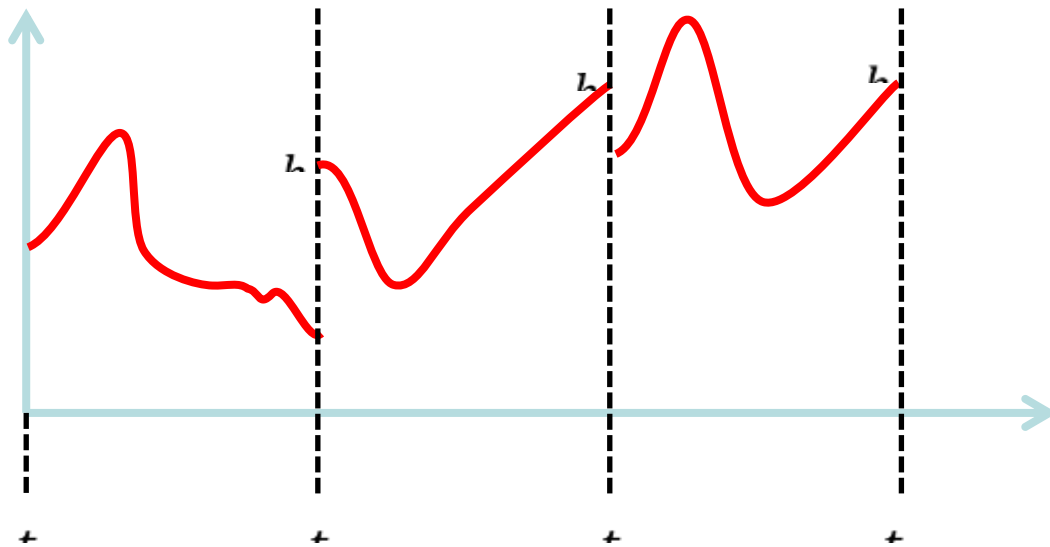


Multiple-Shooting and Collocation

Dynamic Trajectory Optimization

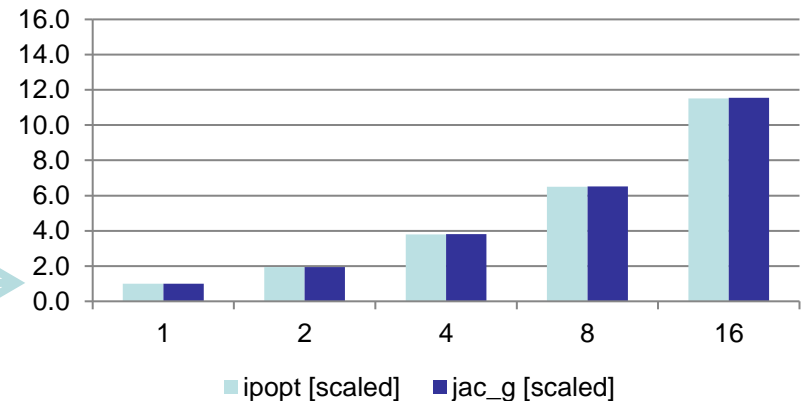
- Minimize a goal function subject to model equation constraints, useful e.g. for NMPC
- Multiple Shooting/Collocation
 - Solve sub-problem in each sub-interval

$$x_i(t_{i+1}) = h_i + \int_{t_i}^{t_{i+1}} f(x_i(t), u(t), t) dt \approx F(t_i, t_{i+1}, h_i, u_i), \quad x_i(t_i) = h_i$$

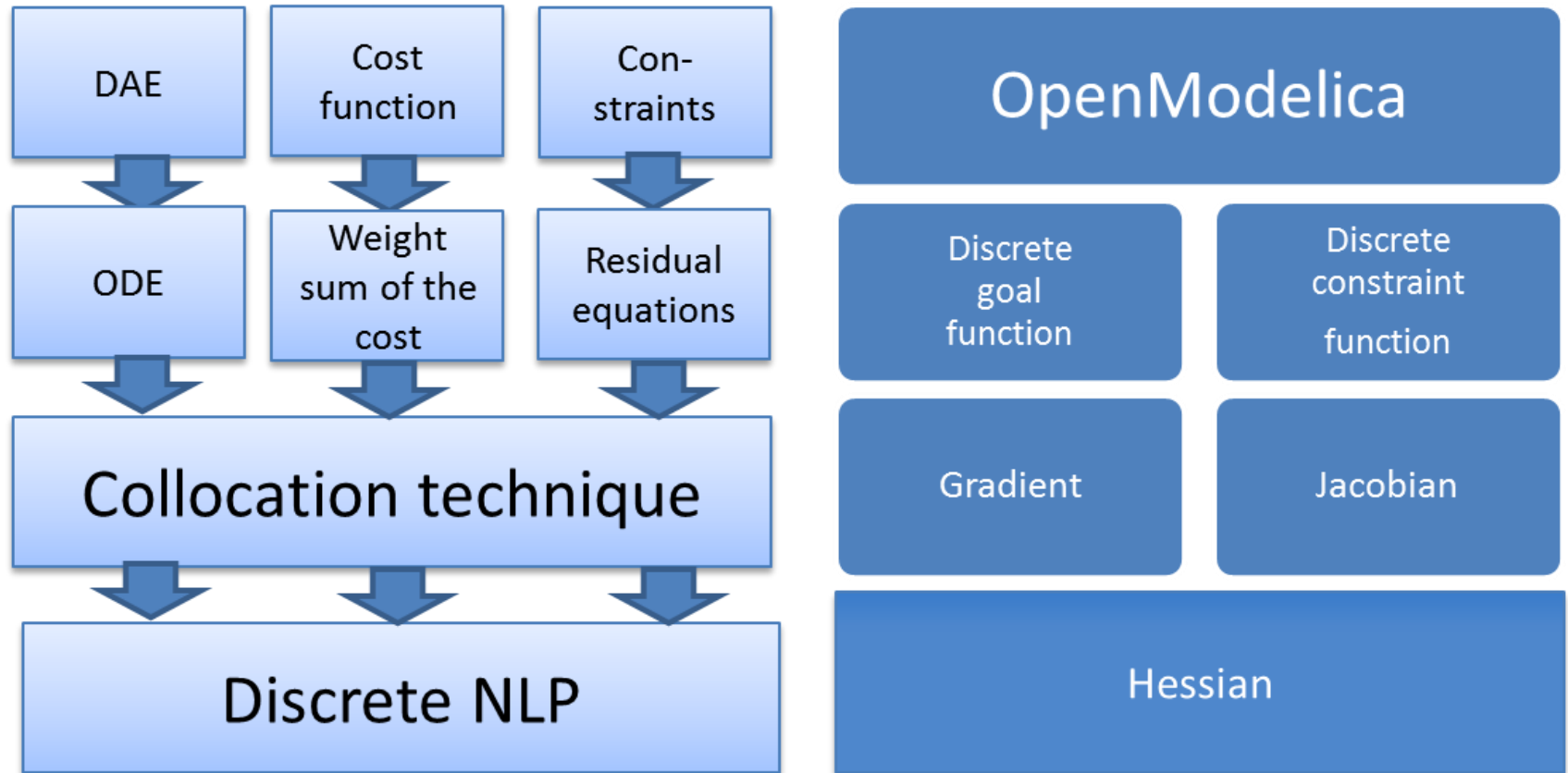


Example speedup, 16 cores:

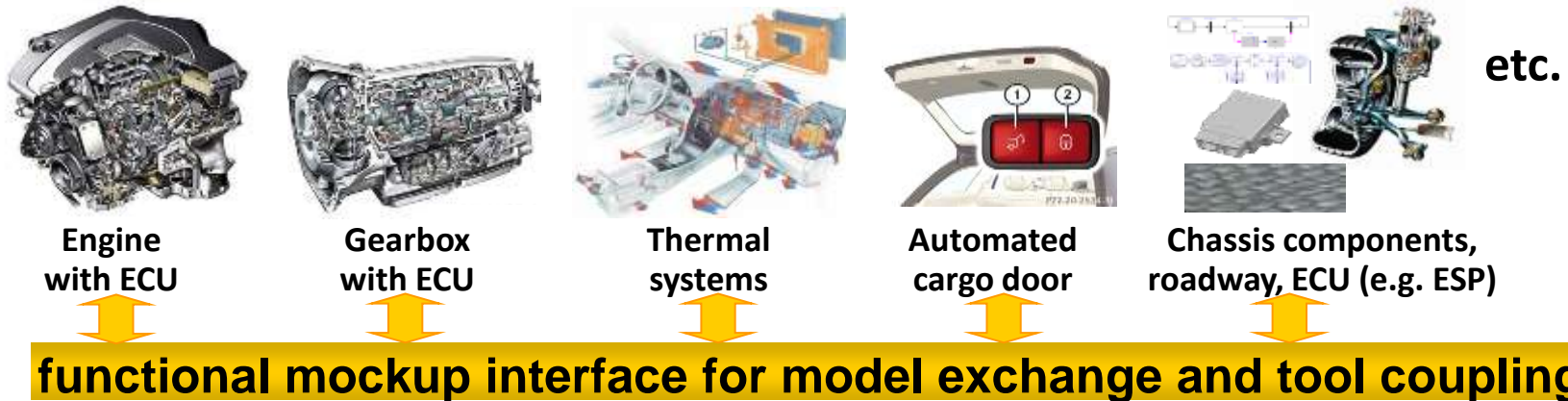
MULTIPLE_COLLOCATION



OpenModelica Dynamic Optimization Collocation



General Tool Interoperability & Model Exchange Functional Mock-up Interface (FMI)

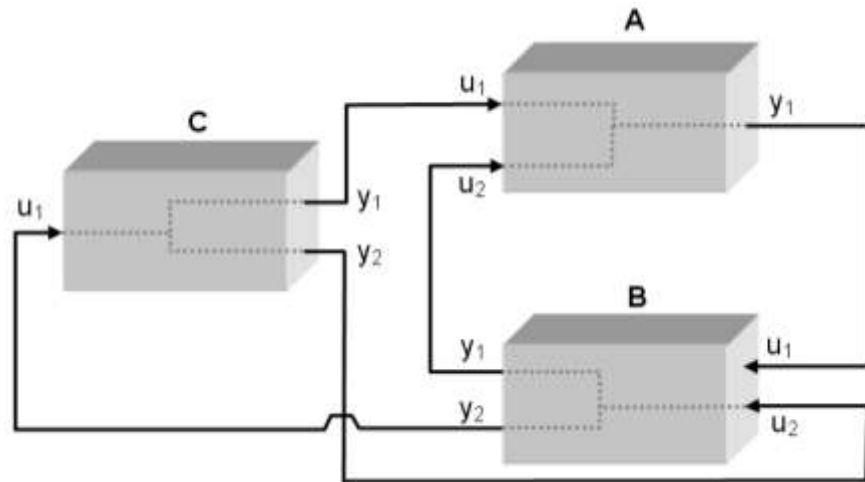


courtesy Daimler

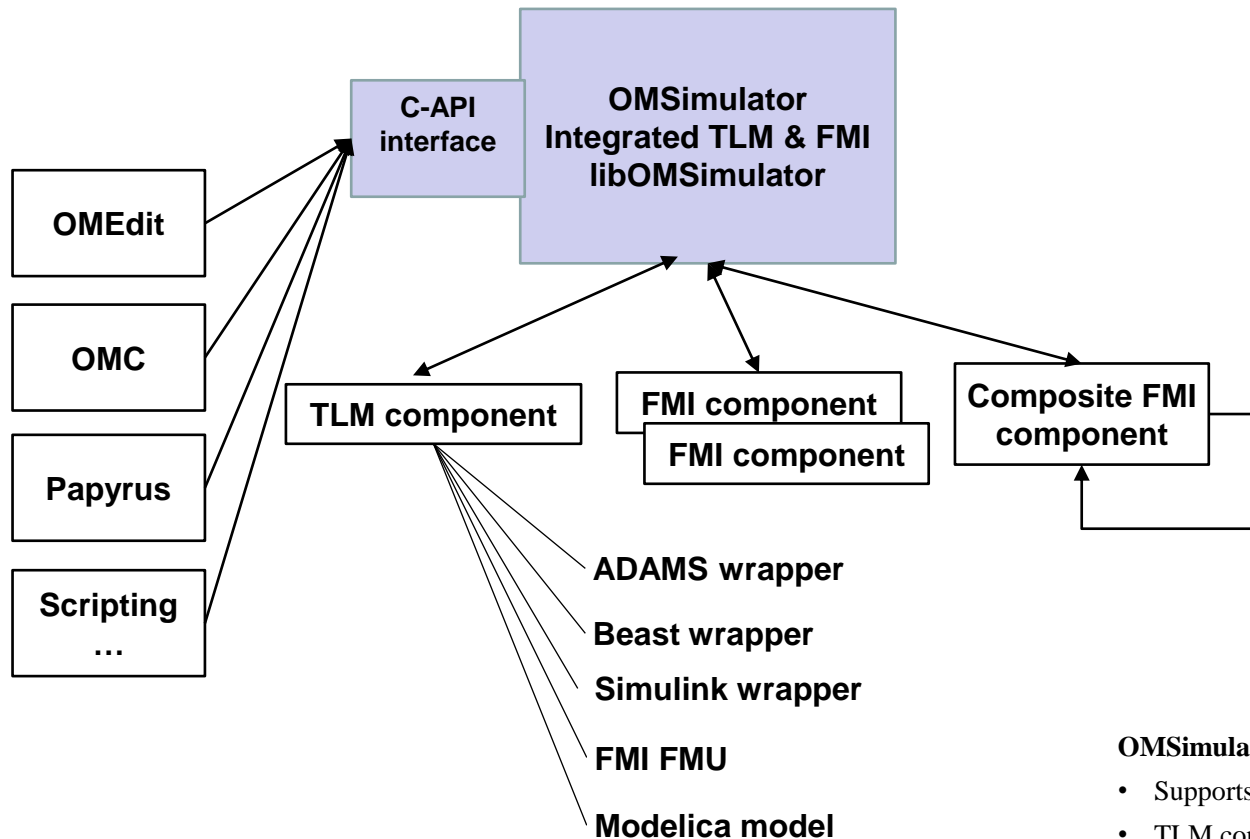
- FMI development was started by ITEA2 MODELISAR project. FMI is now a Modelica Association Project
- **Version 1.0**
- FMI for Model Exchange (released Jan 26, 2010)
- FMI for Co-Simulation (released Oct 12, 2010)
- **Version 2.0** (released July 25 2014) **2.0.4** (released Dec 1, 2022)
- **Version 3.0** (release May 10 2022)
- FMI for Model Exchange and Co-Simulation
- ~ **180 tools** supporting it (<https://www.fmi-standard.org/tools>)

Functional Mockup Units

- Import and export of input/output blocks –
Functional Mock-Up Units – FMUs, described by
 - differential-, algebraic-, discrete equations,
 - with time-, state, and step-events
- An FMU can be large (e.g. 100 000 variables)
- An FMU can be used in an embedded system (small overhead)
- FMUs can be connected together



OMSimulator – Integrated FMI and TLM-based Cosimulator/Simulator



Main Framework Aspects

Unified co-simulation/simulation tool

- FMI 2.0 (model exchange and co-simulation)
- TLM (transition line modelling)
- Real-time and offline simulation

Standalone open source simulation tool with rich interfaces

- C/Java
- Scripting languages

Co-simulation framework as a solid base for engineering tools

- Integration into OpenModelica/Papyrus
- Open for integration into third-party tools and specialized applications (e.g. flight simulators, optimization)

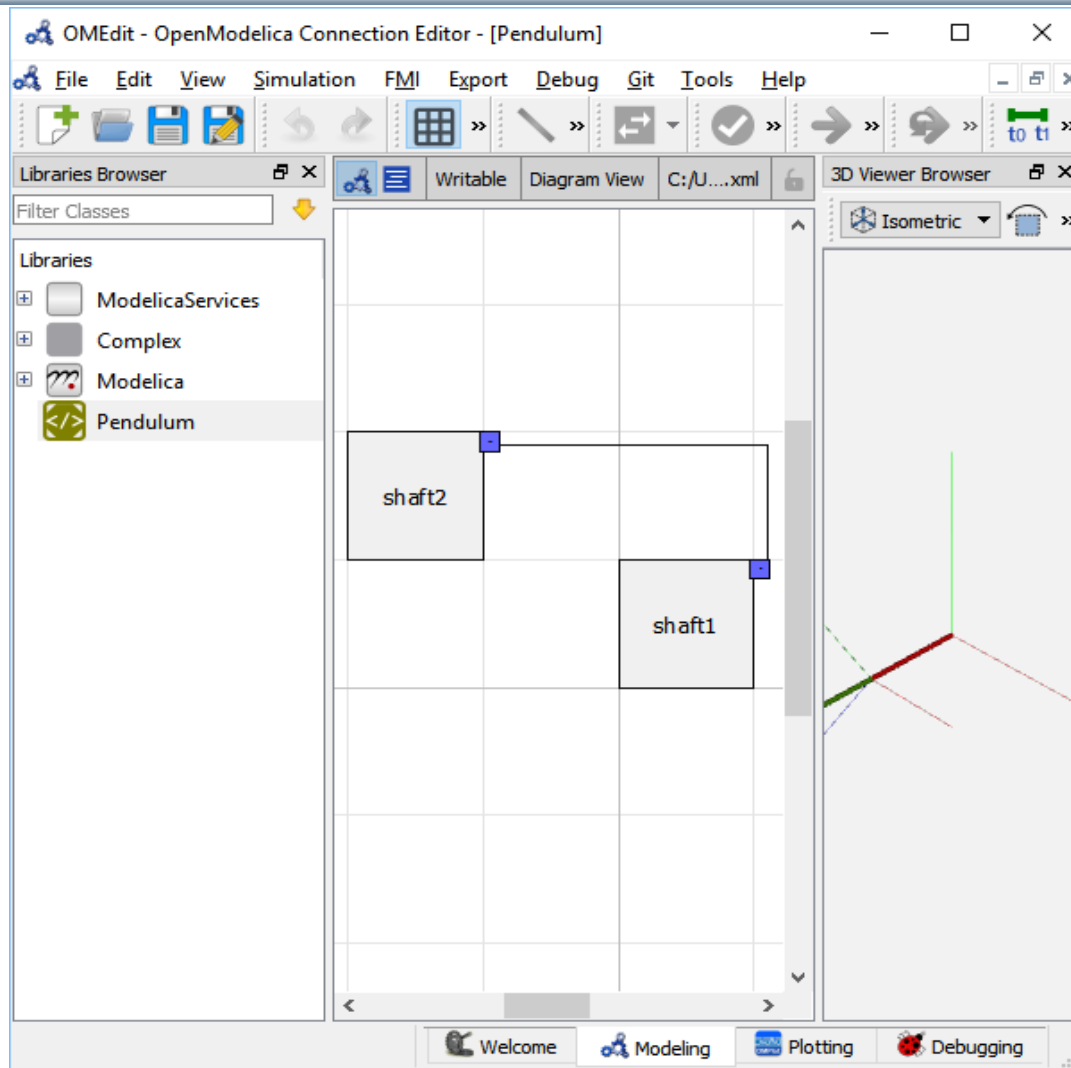
OMSimulator 2.0 in OpenModelica 1.13.0

- Supports both FMI and TLM
- TLM connections are optional
- Co-simulation to multiple tools
- Composite model editor operational
- External API interface and scripting not yet finalized

Received ITEA Award of Excellence, Sept, 2019

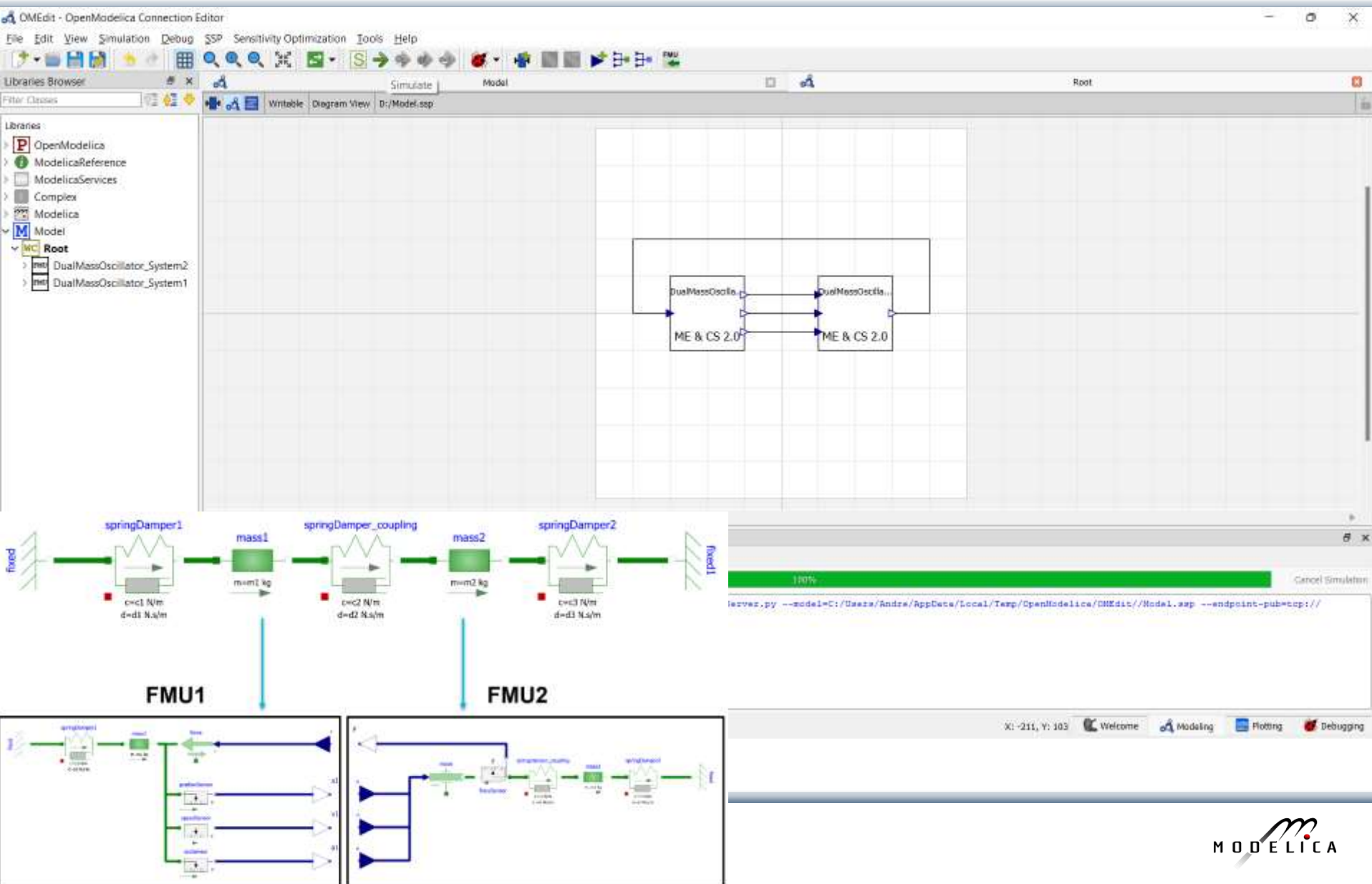
- The **OPENCPS** project for which **OMSimulator** was the major result, received the ITEA award of excellence, September 2019
- ITEA Vice-chairman, Philippe Letellier referred to the major results delivered by the project, calling it "**a milestone on the path of open and standardised co-design and simulation of complex systems, that delivers major results**".

OMSimulator Composite Model Editor with 3D Viewer



- **Composite model editor** with 3D visualization of connected mechanical model components which can be FMUs, Modelica models, etc., or co-simulated components
- **3D animation** possible
- Composite model saved as SSP XML-file
- **Support for SSP** – System Structure and Parameterization standard
- **Numerically stable** co-simulation with **TLM**

OMSimulator – GUI and SSP support



OMSimulator Simulation, SSP, and Tool Comparison

Adding SSP bus connections

OMEdit - Add Bus Connection

Add Bus Connection

Connect **bus2** input connectors to **bus1** output connectors

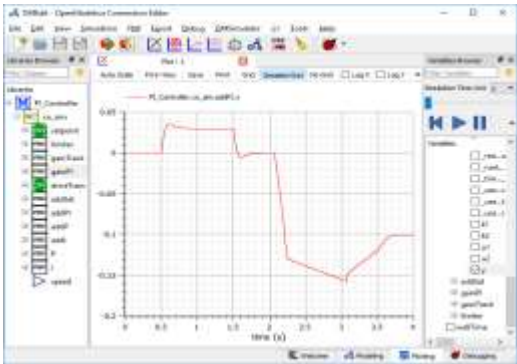
	bus2 inputs	bus1 outputs	ssid:Connection
1	<input checked="" type="checkbox"/> u1	y	<ssid:Con...t="sc2"
2	<input type="checkbox"/> u2		

Connect **bus2** output connectors to **bus1** input connectors

	bus2 outputs	bus1 inputs	ssid:Connection
1	<input checked="" type="checkbox"/> y1	u1	<ssid:Con...t="sc2"
2	<input checked="" type="checkbox"/> y2	u2	<ssid:Con...t="sc2"
3	<input type="checkbox"/> y3		

OKCancel

FMI Simulation
results
in OMEdit

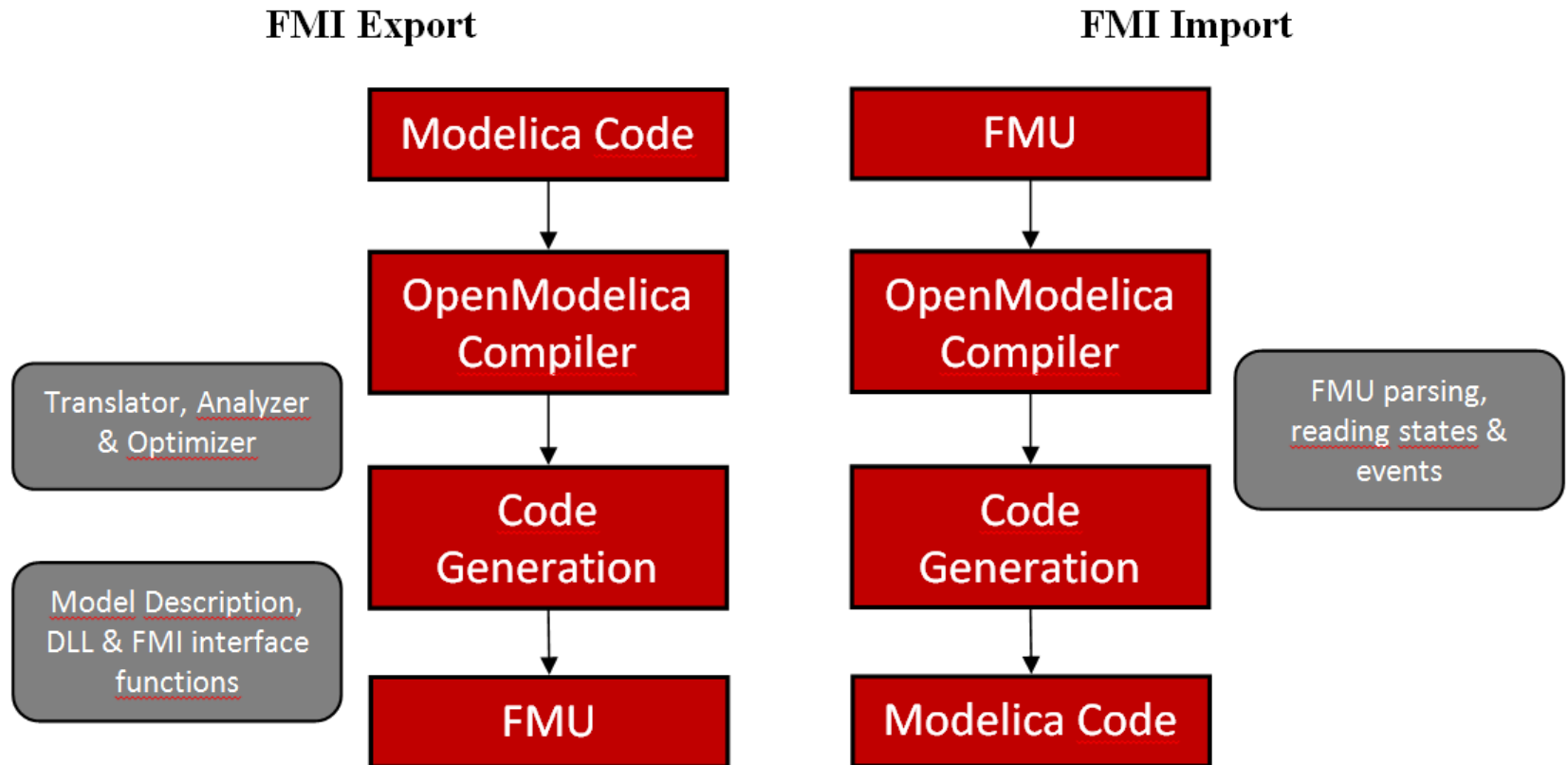


FMI Simulation Tool Comparison

	OMSimulator	DACCOSIM	Simulink	PyFMI
Commercial	No	No	Yes	No
Open-source	OSMC-PL, GPL	AGPL2	No	LGPL
Lookup Table	Yes	Yes	Yes	No
Alg. Loops	Yes	Yes	No	Yes
Scripting	Python, Lua	proprietary	proprietary	Python
GUI	Yes	Yes	Yes	No
SSP	Yes	No	No	No
platform	Linux/Win/macOS	Linux/Win	Linux/Win/macOS	Linux/Win/macOS

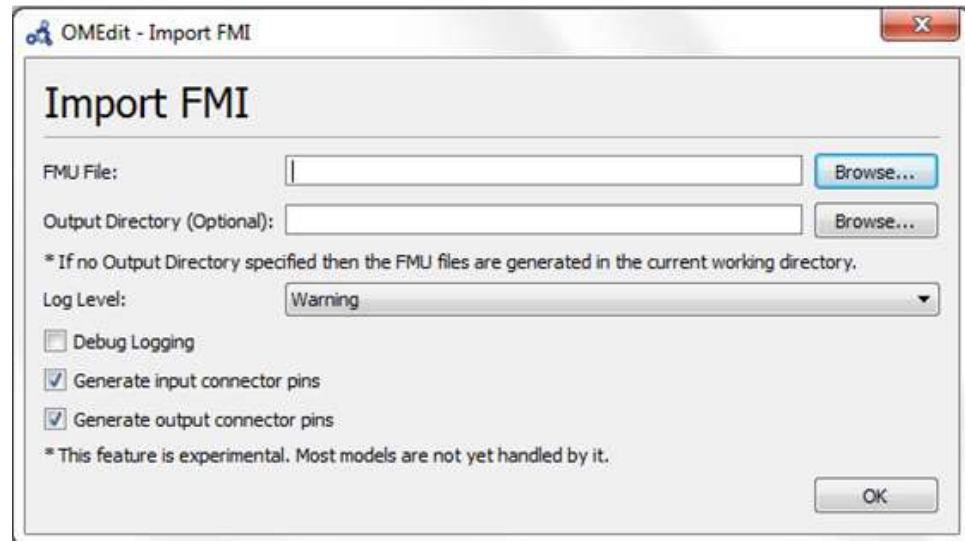
	Dymola	PySimulator	FMI Go!	FMI Composer
Commercial	Yes	No	No	Yes
Open-source	No	BSD	MIT	No
Lookup Table	Yes	Yes	Yes	Yes
Alg. Loops	Yes	Yes	Yes	Yes
Scripting	proprietary	Python	Go	No
GUI	Yes	Yes	No	Yes
SSP	No	No	Yes	Yes
platform	Linux/Win	Linux/Win	Linux/Win/macOS	Linux/Win/macOS

OpenModelica Functional Mockup Interface (FMI)



FMI in OpenModelica

- Model Exchange implemented (FMI 2.0)
- FMI 2.0 Co-simulation implemented
- The FMI interface is accessible via the **OpenModelica scripting environment**, the **OpenModelica Connection Editor** and the **OMSimulator** tool in OpenModelica



OpenModelica Code Generators for Embedded Real-time Code

- A **full-fledged** OpenModelica-generated source-code FMU (Functional Mockup Unit) code generator
 - Can be used to **cross-compile FMUs** for platforms with more available memory.
 - These platforms can **map** FMI inputs/outputs to analog/digital I/O in the importing FMI master.
- A very **simple code generator** generating a **small footprint** statically linked executable.
 - Not an FMU because there is no OS, filesystem, or shared objects in microcontrollers.

Code Generator Comparison, Full vs Simple

	Full Source-code FMU targeting 8-bit AVR proc	Simple code generator targeting 8-bit AVR proc
Hello World (0 equations)	43 kB flash memory 23 kB variables (RAM)	130 B flash memory 0 B variables (RAM)
SBHS Board (real-time PID controller, LCD, etc)	68 kB flash memory 25 kB variables (RAM)	4090 B flash memory 151 B variables (RAM)

The largest 8-bit AVR processor MCUs (Micro Controller Units) have 16 kB SRAM.

One of the more (ATmega328p; Arduino Uno) has 2 kB SRAM.

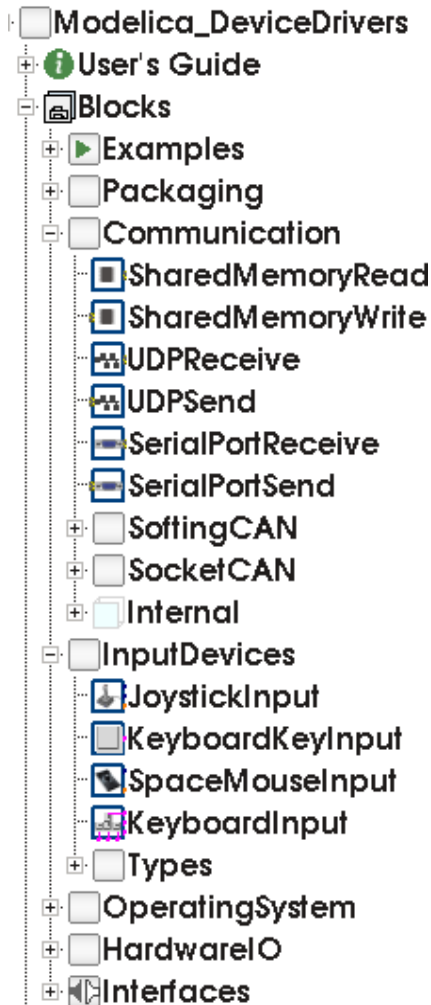
The ATmega16 we target has **1 kB SRAM available** (stack, heap, and global variable

The Simple Code Generator

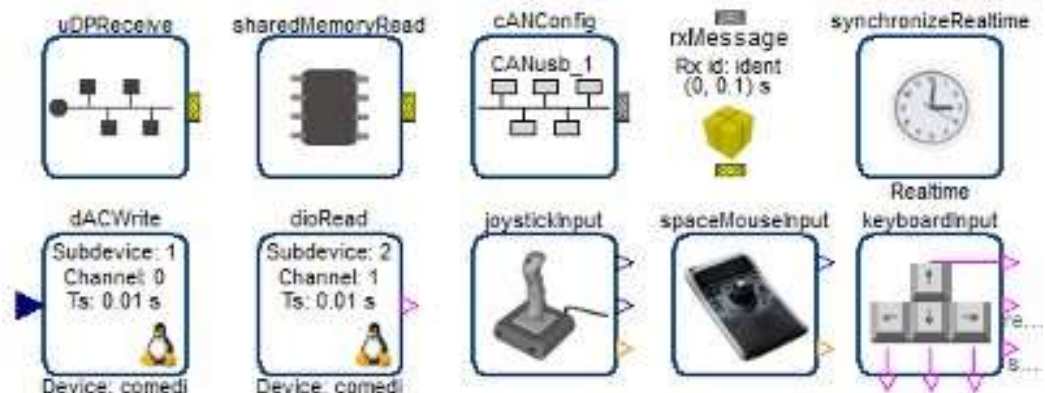
Supports only a limited Modelica subset

- No initialization (yet)
- No strongly connected components
- No events
- No functions (except external C and built-in)
- Only parts that OpenModelica can generate good and efficient code for right now (extensions might need changes in the intermediate code)
 - Unused variables are not accepted (OM usually duplicates all variables for pre() operators, non-linear system guesses, etc... but only a few of them are actually used)
- FMU-like interface (but statically linked)

Communication & I/O Devices: MODELICA_DEVICEDRIVERS Library



- **Free library** for interfacing hardware drivers
- **Cross-platform** (Windows and Linux)
- UDP, SharedMemory, CAN, Keyboard, Joystick/Gamepad
- DAQ cards for digital and analog IO (only Linux)
- Developed for **interactive real-time** simulations

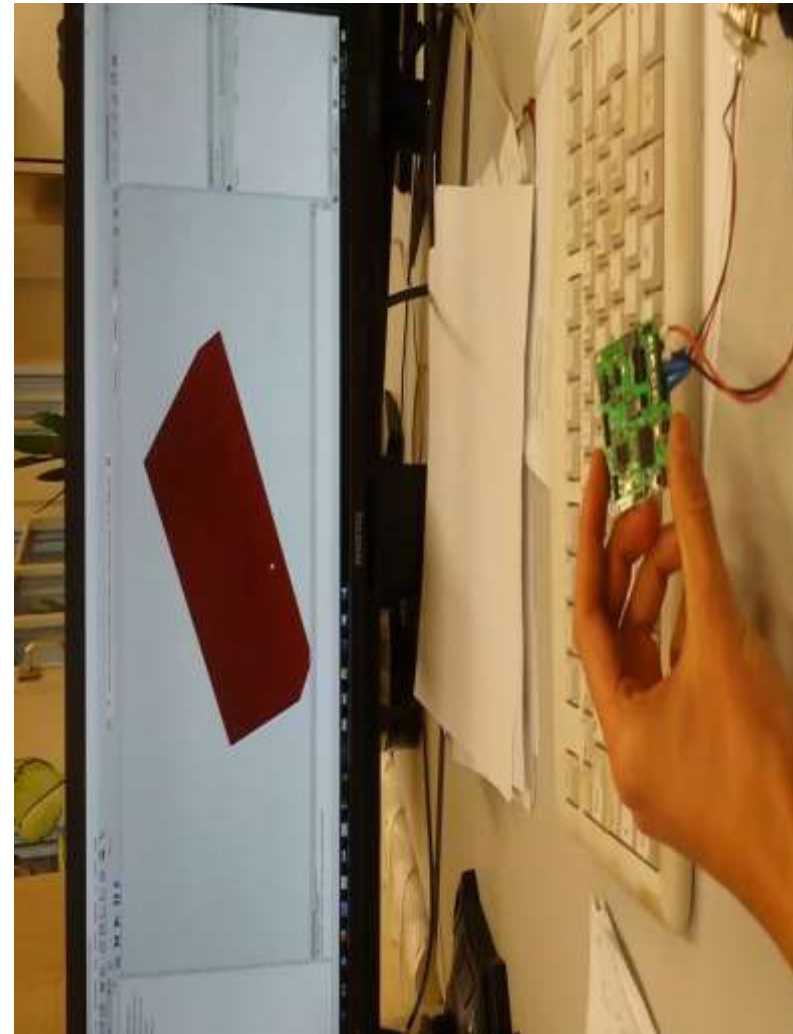


https://github.com/modelica/Modelica_DeviceDrivers/

Modelica connected to external hardware

- IMU (Inertial Measurement Unit)
- Interfaced with a CAN-bus (Controller Area Network bus) - uses Modelica_DeviceDrivers Library
- Visualized in OMEdit

Courtesy of Volker Waurich - TU Dresden



OpenModelica and Device Drivers Library

AVR Processor Support

- No direct Atmel AVR or Arduino support in the OpenModelica compiler
- **Everything is done by the Modelica DeviceDrivers library**
- All **I/O is modeled explicitly in Modelica**, which makes code generation very simple

Modelica Device Drivers Library - AVR processor sub-packages:

- IO.AVR.Analog (ADC – Analog Input)
- IO.AVR.PWM (PWM output)
- IO.AVR.Digital.LCD (HD44780 LCD driver on a single 8-pin digital port)
- OS.AVR.Timers (Hardware timer setup, used by real-time and PWM packages)
- OS.AVR.RealTime (very simple real-time synchronization; one interrupt per clock cycle; works for single-step solvers)

Use Case: SBHS (Single Board Heating System)

Single board heating system (IIT Bombay)

- Use for teaching basic control theory
- Usually controlled by serial port (set fan value, read temperature, etc)
- OpenModelica can generate code targeting the ATmega16 on the board (AVR-ISP programmer in the lower left).
Program size is 4090 bytes including LCD driver and PID-controller (out of 16 kB flash memory available).



Movie Demo, see next page!

Example – Code Generation to SHBS

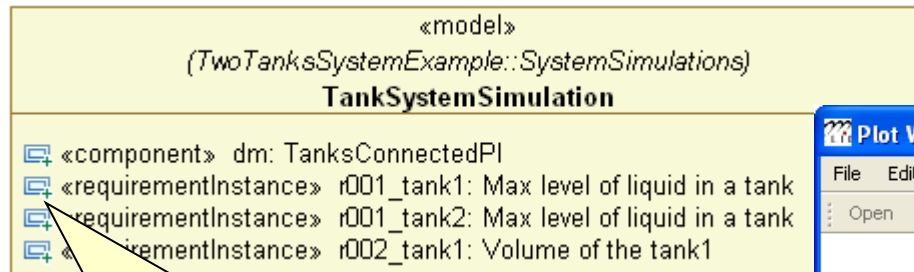


OpenModelica – ModelicaML UML Profile

SysML/UML to Modelica OMG Standardization

- ModelicaML is a UML Profile for SW/HW modeling
 - Applicable to “pure” UML or to other UML profiles, e.g. SysML
- Standardized Mapping UML/SysML to Modelica
 - Defines transformation/mapping for **executable** models
 - Being **standardized** by OMG
- ModelicaML
 - Defines graphical concrete syntax (graphical notation for diagram) for representing Modelica constructs integrated with UML
 - Includes graphical formalisms (e.g. State Machines, Activities, Requirements)
 - Which do not exist in Modelica language
 - Which are translated into executable Modelica code
 - Is defined towards generation of executable Modelica code
 - Current implementation based on the Papyrus UML tool + OpenModelica

Example: Simulation and Requirements Evaluation

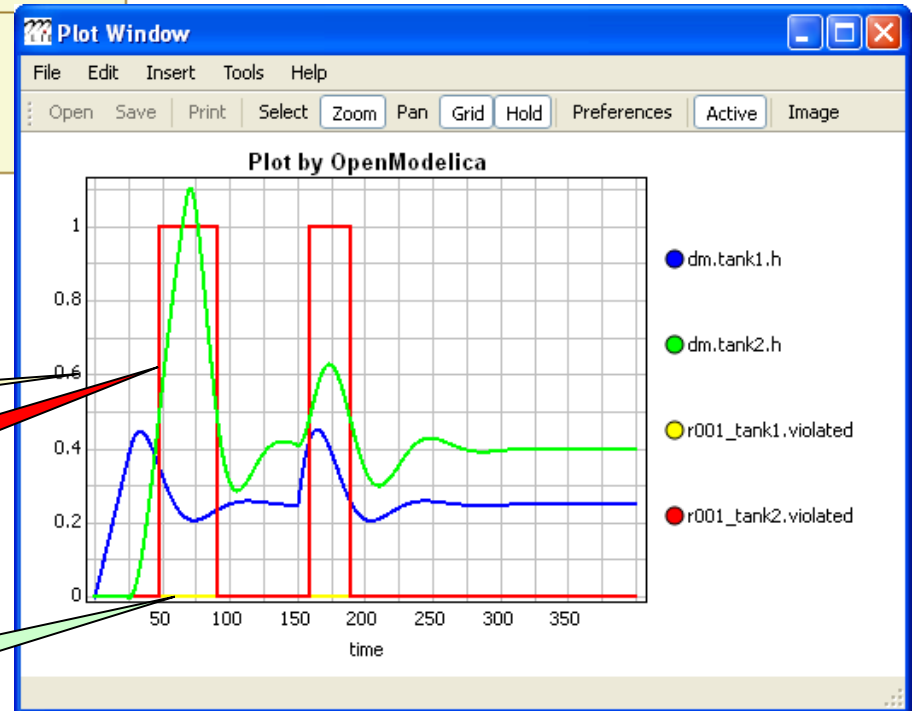


Req. 001 is instantiated 2 times (there are 2 tanks in the system)

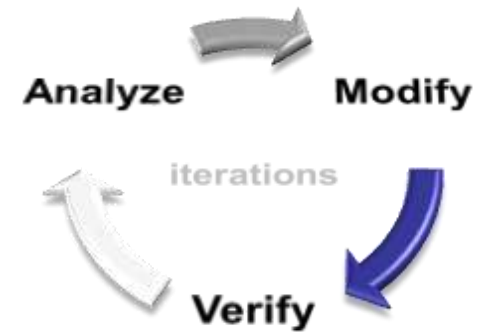
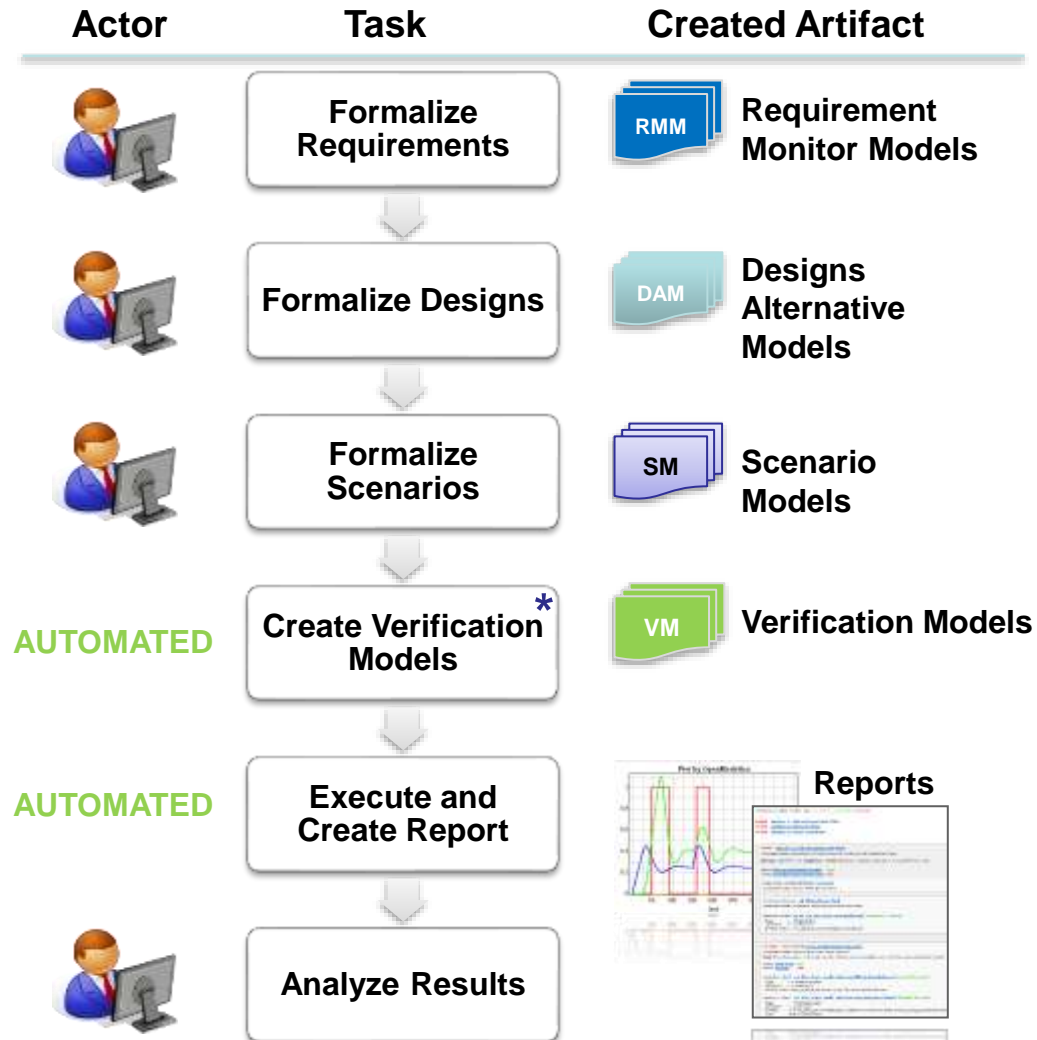
tank-height is 0.6m

Req. 001 for the tank2 is violated

Req. 001 for the tank1 is not violated



vVDR Method – virtual Verification of Designs vs Requirements



Goal: Enable on-demand verification of designs against requirements using automated model composition at any time during development.

Need for Debugging Tools

Map Low vs High Abstraction Level

- A **major part** of the total **cost** of software projects is due to testing and debugging
- US-Study 2002:
Software errors cost the US economy **annually~ 60 Billion \$**
- **Problem: Large Gap in Abstraction Level**
from **Equations** to **Executable Code**
- Example error message (hard to understand)
Error solving nonlinear system 132
time = 0.002
residual[0] = 0.288956
x[0] = 1.105149
residual[1] = 17.000400
x[1] = 1.248448
...

OpenModelica MDT Algorithmic Code Debugger

The screenshot displays the Eclipse IDE interface for the OpenModelica MDT Algorithmic Code Debugger. The main window is titled "Debug - HelloWorld/SimulationModel.mo - Eclipse SDK". The interface is divided into several panes:

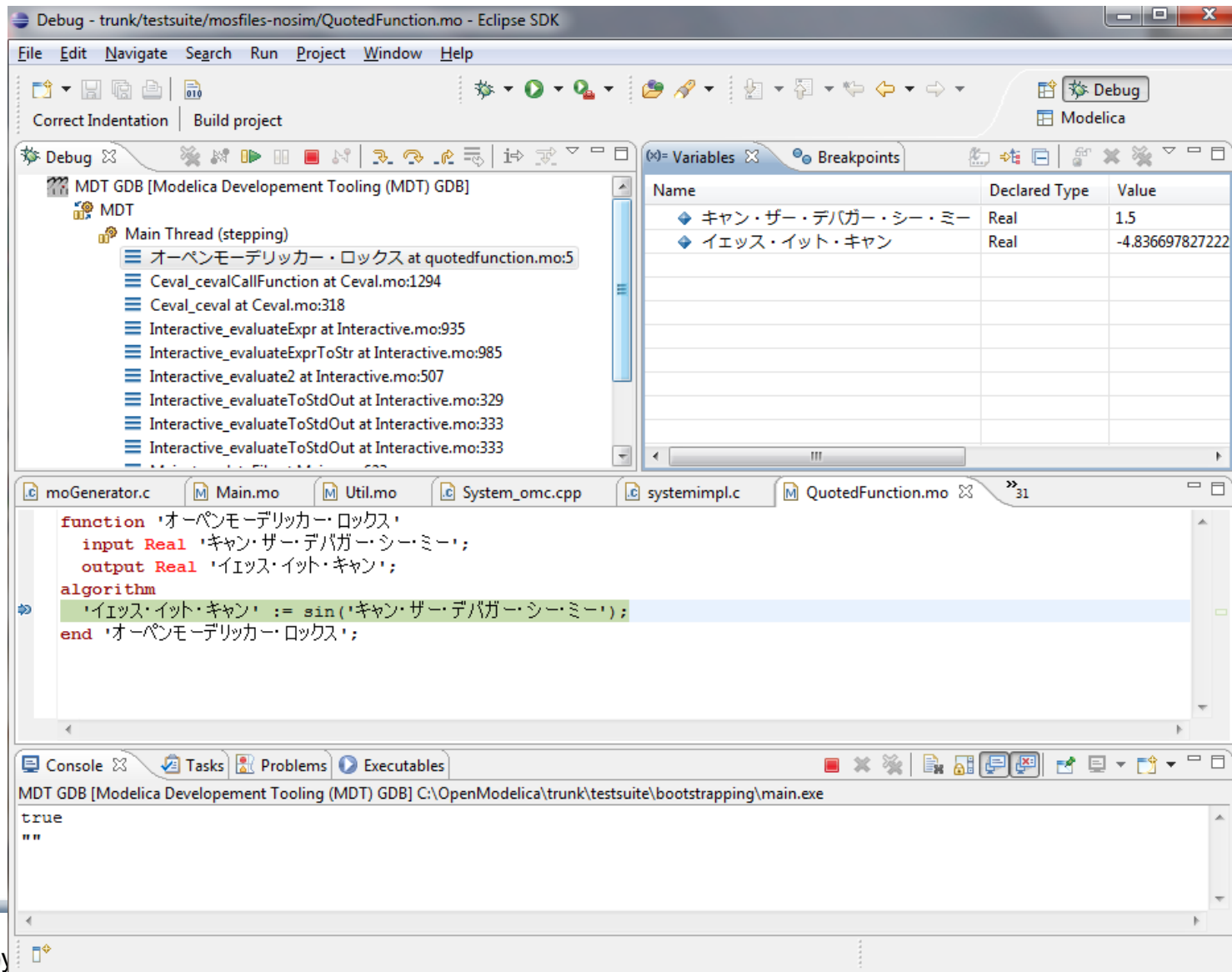
- List of Stack Frames:** Located in the top-left pane, it shows the current execution stack. The top frame is "Simulation Model [Modelica Development Tooling (MDT) GDB]" with a sub-frame "MDT". Below this, the "Main Thread (stepping)" is shown, with the current function being "getValueMultipliedByTwo at simulationmodel.mo:13". The file path is "C:\Users\adeas31\workspaceMDT\HelloWorld\SimulationModel.exe".
- Variables View:** Located in the top-right pane, it displays the current state of variables. The table below shows the variables:

Name	Declared Type	Value	Actual Type
inValue	Real	1	double
outValue	Real	6.9453280720608359e-308	double

- Outline:** Located in the middle-right pane, it shows the project structure. The "getValueMultipliedByTwo" function is selected, showing its sub-variables: "inValue (Real - IN)" and "outValue (Real - OUT)". The "SimulationModel" package is also visible, containing variables "x" and "y".
- Code Editor:** The main editor shows the source code of "SimulationModel.mo". The current function being debugged is "getValueMultipliedByTwo", which takes an input "inValue" and outputs "outValue". The algorithmic code is highlighted in green.
- Output View:** Located in the bottom-left pane, it shows the console output. The text "Simulation Model [Modelica Development Tooling (MDT) GDB] C:\Users\adeas31\workspaceMDT\HelloWorld\SimulationModel.exe" is visible.

The OpenModelica MDT Debugger (Eclipse-based)

Using Japanese Characters



Debug - trunk/testsuite/mosfiles-nosim/QuotedFunction.mo - Eclipse SDK

File Edit Navigate Search Run Project Window Help

Correct Indentation Build project

Debug MDT GDB [Modelica Development Tooling (MDT) GDB]

MDT

Main Thread (stepping)

- オープンモデリッカー・ロック at quotedfunction.mo:5
- Ceval_cevalCallFunction at Ceval.mo:1294
- Ceval_ceval at Ceval.mo:318
- Interactive_evaluateExpr at Interactive.mo:935
- Interactive_evaluateExprToStr at Interactive.mo:985
- Interactive_evaluate2 at Interactive.mo:507
- Interactive_evaluateToStdOut at Interactive.mo:329
- Interactive_evaluateToStdOut at Interactive.mo:333
- Interactive_evaluateToStdOut at Interactive.mo:333

Variables Breakpoints

Name	Declared Type	Value
キャン・ザー・デバガー・シー・ミー	Real	1.5
イエス・イット・キャン	Real	-4.836697827222

moGenerator.c Main.mo Util.mo System_omc.cpp systemimpl.c QuotedFunction.mo

```
function 'オープンモデリッカー・ロック'  
  input Real 'キャン・ザー・デバガー・シー・ミー';  
  output Real 'イエス・イット・キャン';  
algorithm  
  'イエス・イット・キャン' := sin('キャン・ザー・デバガー・シー・ミー');  
end 'オープンモデリッカー・ロック';
```

Console Tasks Problems Executables

MDT GDB [Modelica Development Tooling (MDT) GDB] C:\OpenModelica\trunk\testsuite\bootstrapping\main.exe

```
true  
""
```

OpenModelica Equation Model Debugger

The screenshot displays the OMEdit - Transformational Debugger interface with three main panels highlighted by red boxes:

- Variables View:** Shows a tree structure of variables (frame, boxBody1, body, frame_a, R, T) and their definitions in equations. It includes filters for Case Sensitive, Regular Expression, and buttons for Expand All and Collapse All.
- Equations View:** Displays a list of equations (Index, Type, Equation) and a detailed view of equation operations (solve, scalarize, simplify, inline, substitute) for a specific equation.
- Source View:** Shows the source code of the model, with line numbers and comments. The code includes relationships between quantities of frame_a and frame_b, and transformations.

Showing equation transformations of a model:

$0 = y + \text{der}(x * \text{time} * z); z = 1.0;$

(1) substitution:

$y + \text{der}(x * (\text{time} * z))$
 $\Rightarrow y + \text{der}(x * (\text{time} * 1.0))$

(2) simplify:

$y + \text{der}(x * (\text{time} * 1.0))$
 $\Rightarrow y + \text{der}(x * \text{time})$

(3) expand derivative (symbolic diff):

$y + \text{der}(x * \text{time})$
 $\Rightarrow y + (x + \text{der}(x) * \text{time})$

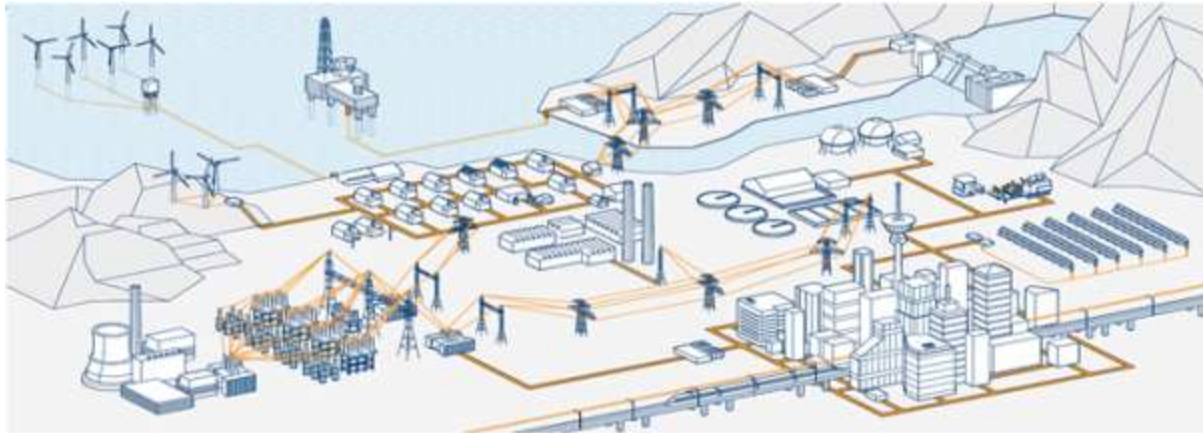
(4) solve:

$0.0 = y + (x + \text{der}(x) * \text{time})$
 \Rightarrow
 $\text{der}(x) = ((-y) - x) / \text{time}$
 $\text{time} < 0$

Mapping run-time error to source model position

ABB Industry Use of OpenModelica FMI 2.0 and Debugger

- ABB OPTIMAX® provides advanced model based control products for power generation and water utilities

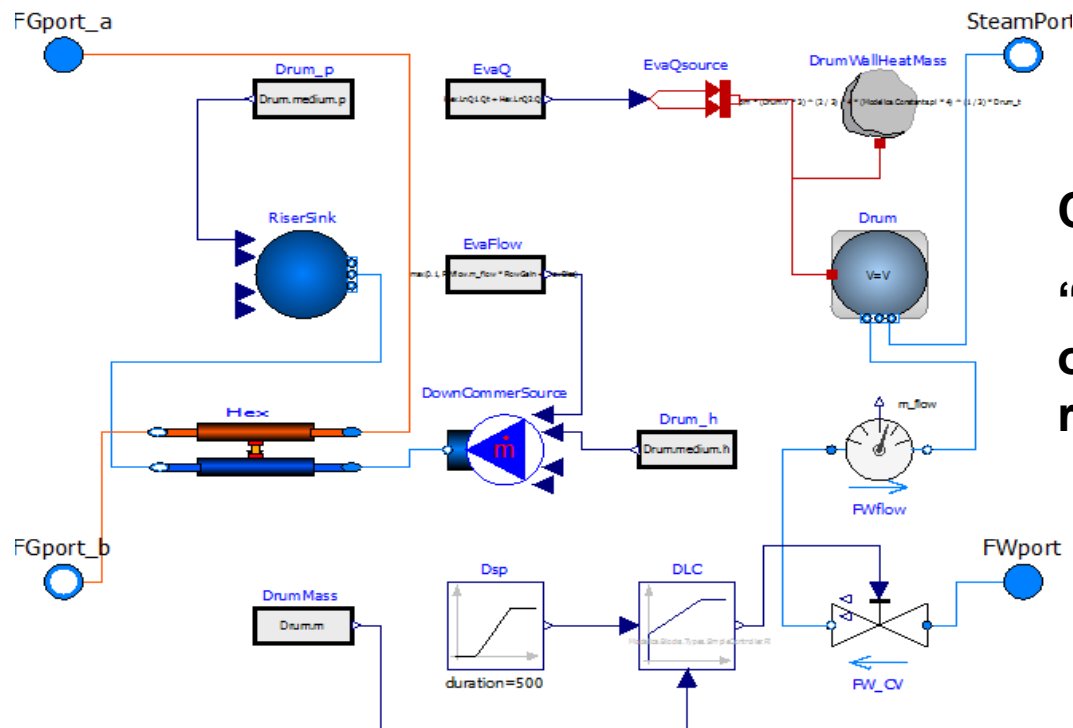


- ABB: *“ABB uses several compatible Modelica tools, including OpenModelica, depending on specific application needs.”*
- ABB: *“OpenModelica provides outstanding debugging features that help to save a lot of time during model development.”*

Performance Profiling for faster Simulation

(Here: Profiling equations of Siemens Drum boiler model with evaporator)

- Measuring **performance** of equation blocks to find bottlenecks
 - Useful as input before model simplification for real-time applications
- Integrated with the debugger to **point out the slow equations**
- Suitable for **real-time profiling** (collect less information), or a complete view of all equation blocks and function calls



Conclusion from the evaluation:

“...the profiler makes the process of performance optimization radically shorter.”

Exercise: Try Some Spoken-Tutorial Exercise on OpenModelica. Link from www.openmodelica.org



To learn about Modelica, read a [book](#) or a [tutorial](#) about [Modelica®](#).
Interactive step-by-step beginners Modelica [on-line spoken tutorials](#)
Interactive [OMWebbook](#) with examples of Modelica textual modeling

[Reset dropdowns](#)

OpenModelica is an open source modelling and simulation environment intended for industrial and academic usage. It is an object oriented declarative multi domain modelling language for complex systems. This environment can be used to work for both steady state as well as dynamic systems. Attractive strategy when dealing with design and optimization problems. As all the equations are solved simultaneously it doesn't matter whether the unknown variable in an input or output variable. [Read more](#)

https://spoken-tutorial.org/tutorial-search/?search_foss=OpenModelica&search_language=English

About 12 results found.

[Instruction Sheet](#)

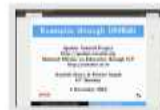


1. Introduction to OMEdit

Foss : OpenModelica - English

Outline: Introduction to OpenModelica Introduction to OMEdit Perspectives in OMEdit Browsers in OMEdit View icons in OMEdit Open a Class from Libraries Browser Checking for correctness..

Basic
Progress bar

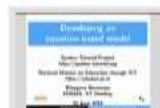


2. Examples through OMEdit

Foss : OpenModelica - English

Outline: Expand Modelica library Expand Electrical library Expand Analog library Open Rectifier Class Compare the values of IDC & Losses time vs Losses plot Expand Mechanics library ..

Basic
Progress bar



3. Developing an equation-based model

Foss : OpenModelica - English

Outline: Introduction to OMEdit Declaration of variables and equations Simulation of a model in

Basic
Progress bar

Summary

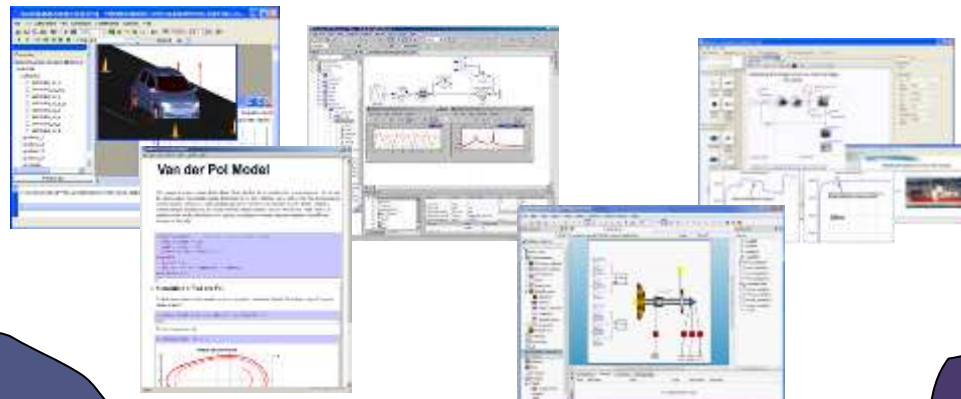
Multi-Domain
Modeling

Visual Acausal
Component
Modeling



Typed
Declarative
Textual Language

Hybrid
Modeling



Thanks for listening!