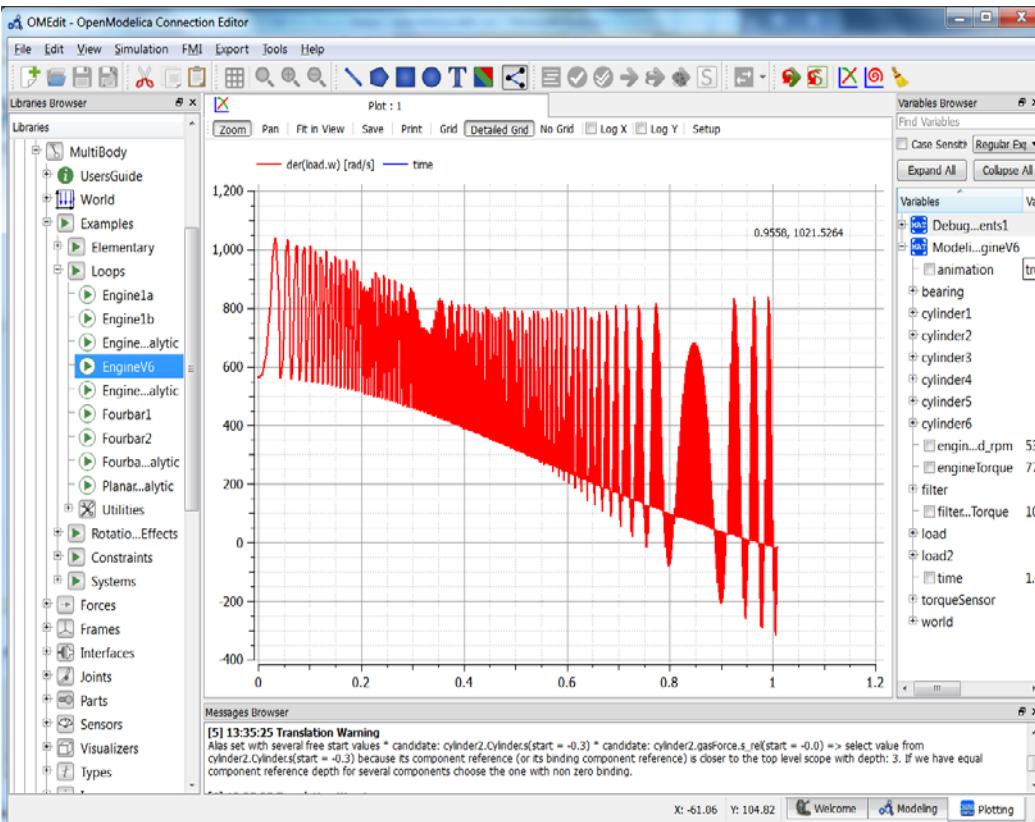


# Modeling, Simulation, and Development of Cyber-Physical Systems with OpenModelica and FMI



Presentation at the USA  
Modelica Conference

October 10, 2018

Adrian Pop  
[adrian.pop@liu.se](mailto:adrian.pop@liu.se)

Technical Coordinator at the Open Source  
Modelica Consortium

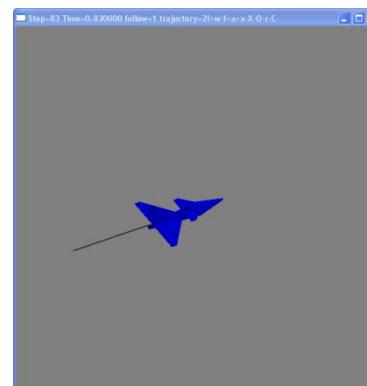
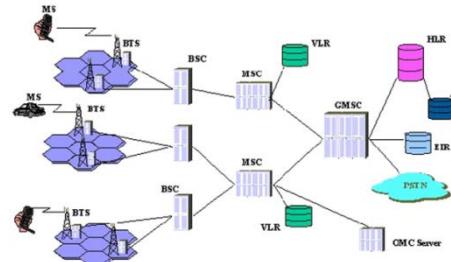
Adjunct Associate Professor at Linköping  
University

# Main Goals for the OpenModelica Effort

- A comprehensive Open Source Modelica and FMI modeling, compilation, simulation and optimization environment based on free software distributed in binary and source code form for research, teaching, and industrial usage
- Support model-based development of cyber-physical systems, from requirements, to models, to simulation and production code

# Industrial Challenges for Complex Cyber-Physical System Products of both Software and Hardware

- Increased **Software** Fraction
- **Shorter** Time-to-Market
- Higher demands on effective strategic **decision** making
- **Cyber-Physical** (CPS) – Cyber (software)  
Physical (hardware) products



# The OpenModelica Environment

## [www.openmodelica.org](http://www.openmodelica.org)

### Open Modelica

Login · Create an account

HOME DOWNLOAD TOOLS & APPS USERS DEVELOPERS FORUM EVENTS RESEARCH search ...

Top information

**Industrial Products**  
Commercial Applications using OpenModelica

**OMEdit**  
Enhanced OpenModelica Connection Editor.

**Library Coverage**  
Latest library coverage.

**Modelica/OpenModelica Videos**

**Overview of Modelica..**

Introduction

OPENMODELICA is an open-source Modelica-based modeling and simulation environment intended for industrial and academic usage. Its long-term development is supported by a non-profit organization – the [Open Source Modelica Consortium \(OSMC\)](#).

The goal with the OpenModelica effort is to create a comprehensive Open Source Modelica modeling, compilation and simulation environment based on free software distributed in binary and source code form for research, teaching, and industrial usage. We invite researchers and students, or any interested developer to participate in the project and cooperate around OpenModelica, tools, and applications.



Register yourself to get information about new releases.  
Participate in the [OpenModelicaInterest mailing list](#).  
Help us: get the latest [source code](#) or [nightly-build](#) and report [bugs](#).  
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

Latest news

January 17, 2017: OpenModelica 1.11 Beta3 released

December 20, 2016: OpenModelica 1.11 Beta2 released

November 22, 2016: OpenModelica 1.9.7 released

March 16, 2016: OpenModelica 1.9.6 released

March 9, 2016: OpenModelica 1.9.4 released

February 18, 2016: OpenModelica 1.9.4 beta2 released

Program OpenModelica Annual Workshop 2016

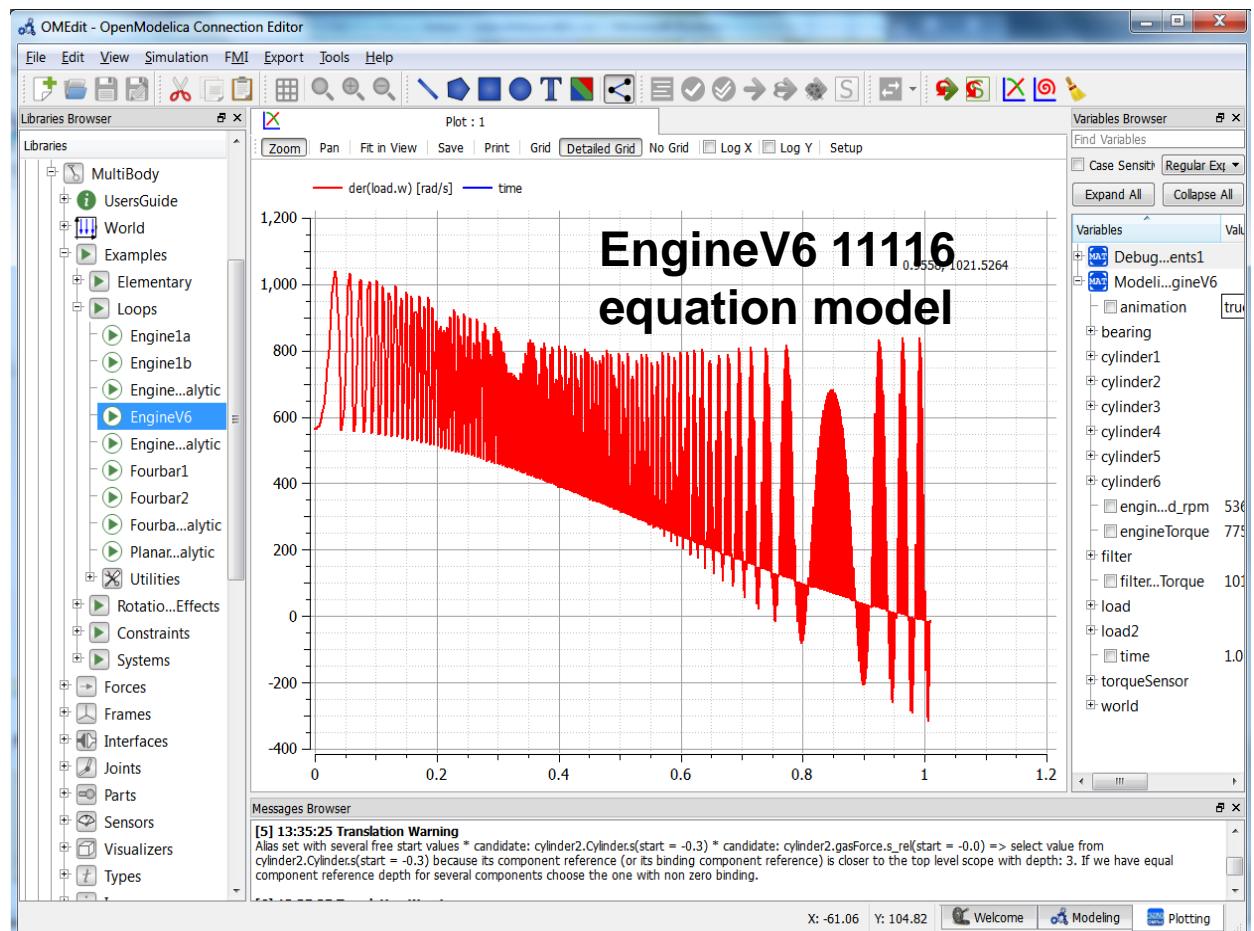
January 31, 2016: OpenModelica 1.9.4 beta1 released

September 8, 2015: OpenModelica 1.9.3 released

July 10, 2015: SIMS 2015 registration open

# 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



# OpenModelica Graphical Editor and Plotting

OMEdit - OpenModelica Connection Editor - [ChuaCircuit]

File Edit View Simulation FMI Export Debug Tools Help

Libraries Browser Filter Classes Documentation Browser

Modelica Electrical Analog Examples ChuaCircuit C:/OpenModelica/build/fb/om\_aalog/Examples/ChuaCircuit.mo

**Chua Circuit**

L=18  
R=12.5e-3  
C=100  
G=0.565  
C1  
Nr  
Gnd

Modelica.Electrical.Analog.Examples.ChuaCircuit

Chua's circuit, ns, V, A

Information

Chua's circuit is the most simple nonlinear circuit which shows chaotic behaviour. The circuit consists of linear basic elements (capacitors, resistor, conductor, inductor), and one nonlinear element, which is called Chua's diode. The chaotic behaviour is simulated.

The simulation end time should be set to 54. To get the chaotic behaviour please plot C1.v. Choose C2.v as the independent variable.

Reference:

Kennedy, M.P.: Three Steps to Chaos - Part I: Evolution. IEEE Transactions on CAS I 40 (1993)10, 640-656

Revisions

Variables Browser

Simulation Time Unit: s

Variables

Value	Display Unit	Description
0.14337	A	C1 Capacitance der(Voltage d... p.v - n.v) Current flow...n p to pin n
4.0	V	C2 Voltage drop... (p.v - n.v)
100.0	F	C1 Capacitance der(Voltage d... p.v - n.v) Current flow...n p to pin n
0.00673438	km2...1.g	C2 Capacitance der(Voltage d... p.v - n.v) Current flow...n p to pin n
0.673438	A	C2 Voltage drop... (p.v - n.v)

Plotting

X: 91.50 Y: -102.94

Variables Browser

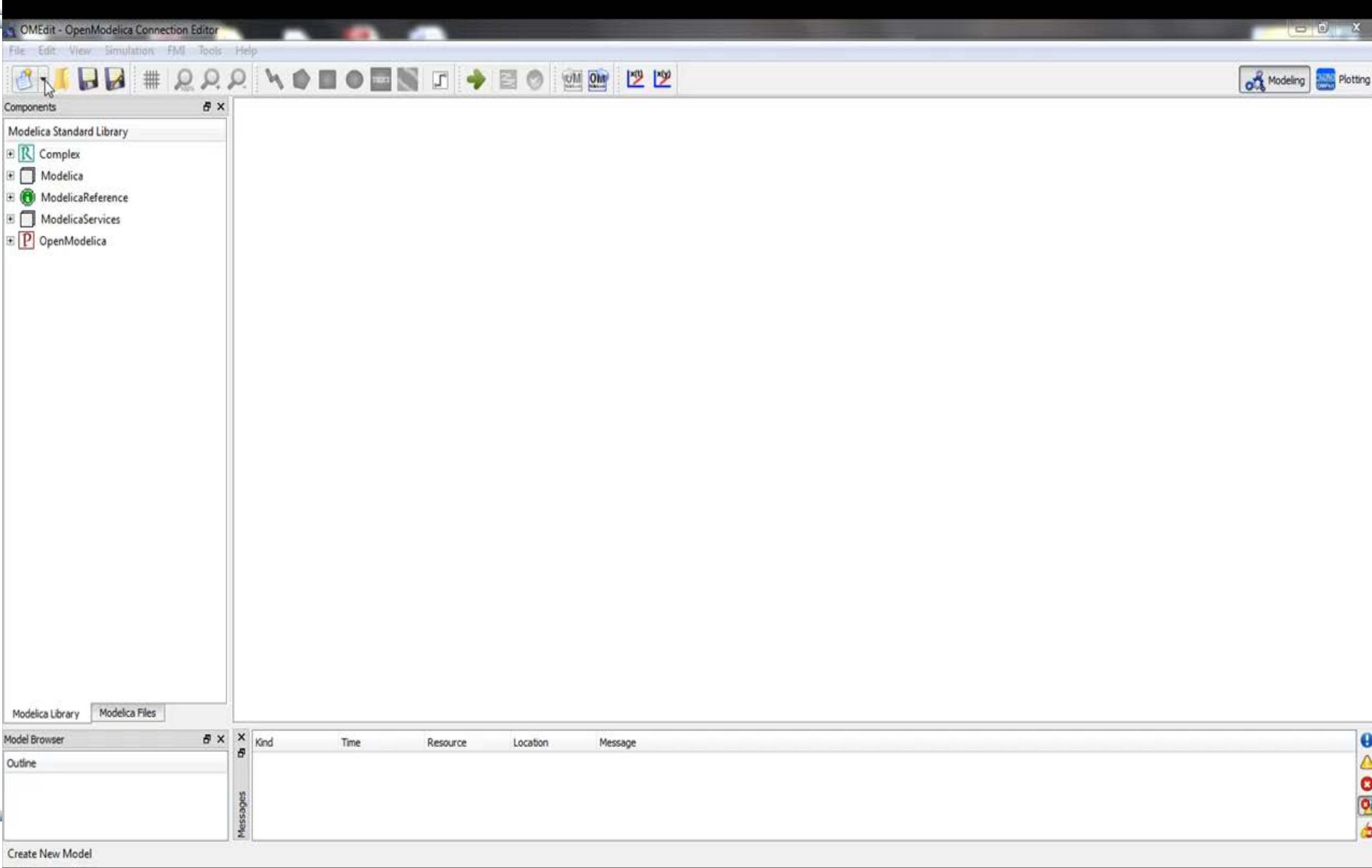
Variables

Modelica.Electrical.Analog.Examples.aCircuit

C1  
C  
der(v)  
i  
n  
p  
v

C2  
C  
der(v)  
i  
n  
p  
v

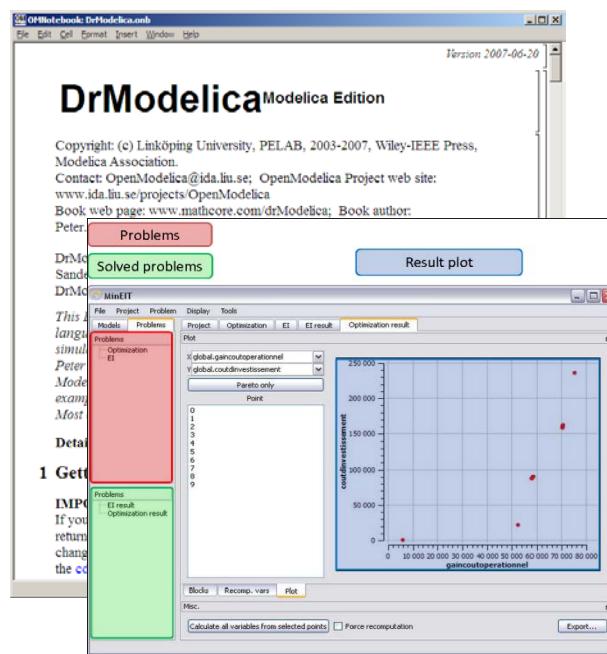
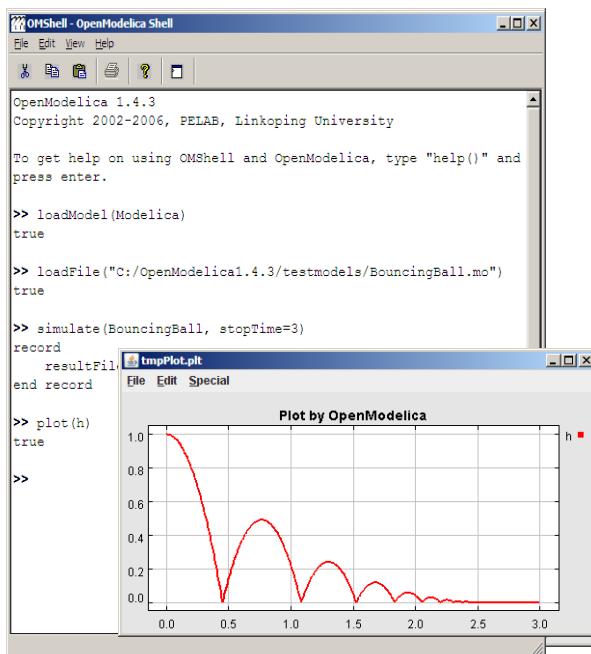
# Graphical Modeling with OpenModelica Environment



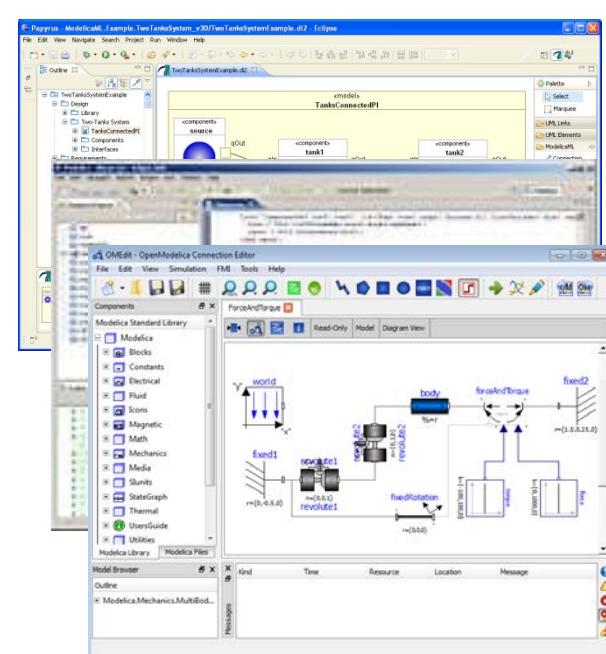
# The OpenModelica Open Source Environment

[www.openmodelica.org](http://www.openmodelica.org)

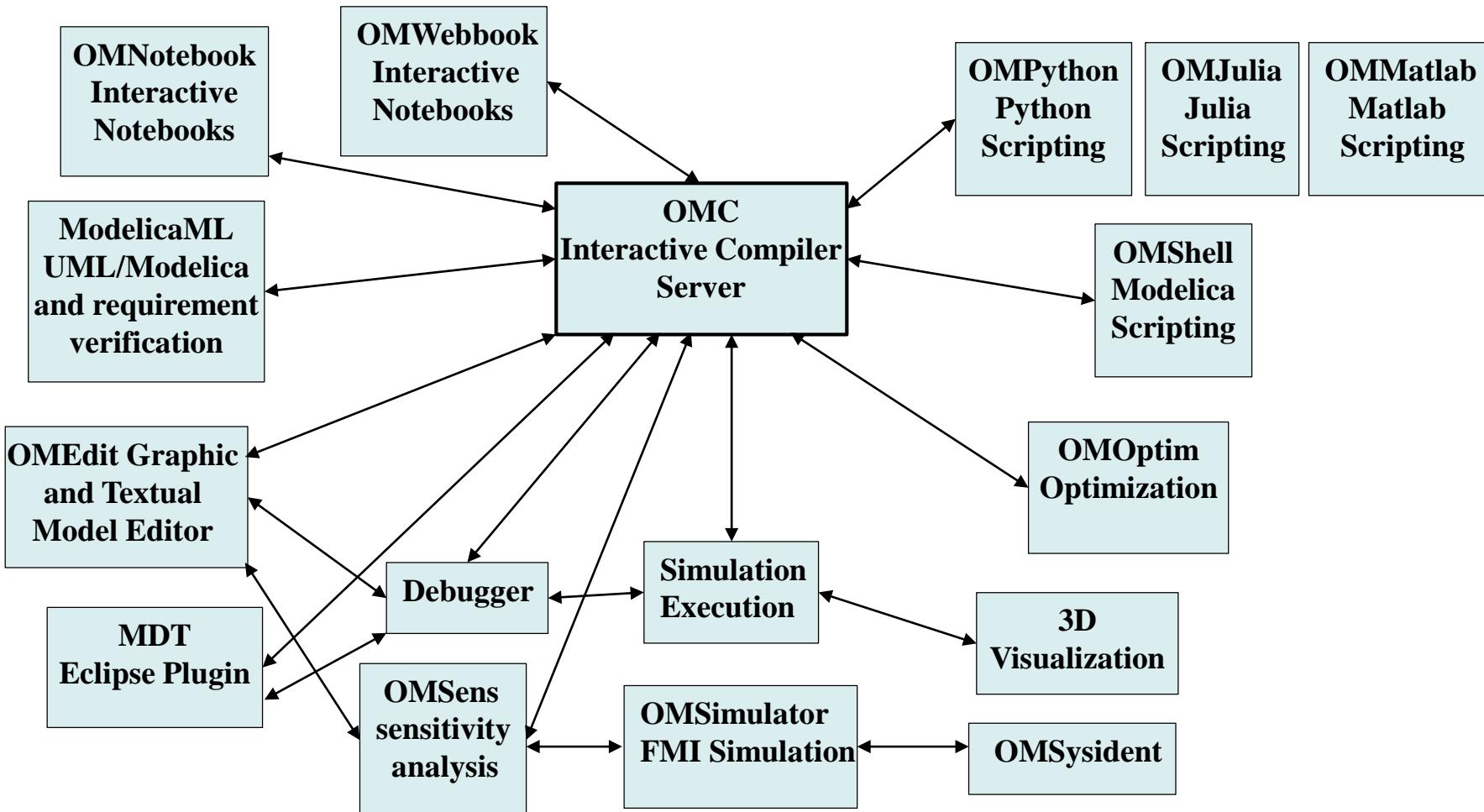
- Advanced Interactive Modelica compiler (OMC)
  - Supports most of the Modelica Language
  - **Modelica** and **Python** scripting
- Basic environment for creating models
  - **OMShell** – an interactive command handler
  - **OMNotebook** – a literate programming notebook
  - **MDT** – an advanced textual environment in Eclipse
- **OMEdit** graphic Editor
- **OMDebugger** for equations
- **OMOptim** optimization tool
- **OM Dynamic optimizer** collocation
- **ModelicaML UML Profile**
- **MetaModelica** extension
- **ParModelica** extension
- **OMSimulator** – FMI/TLM simulator



new



# The OpenModelica Tool Architecture



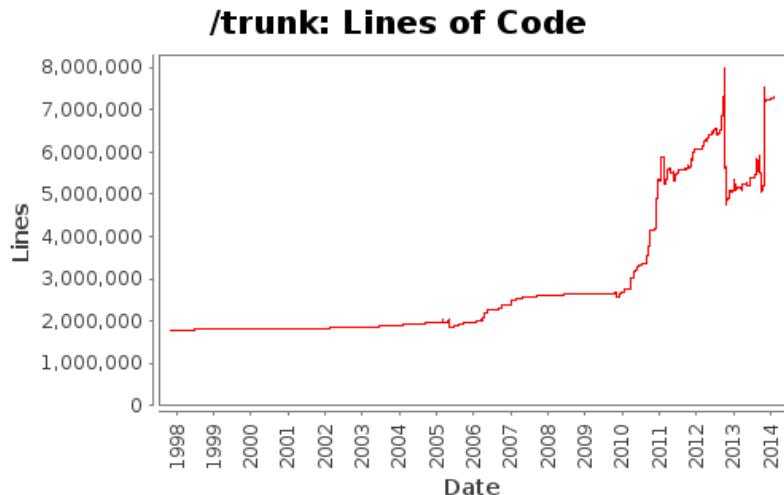
# OSMC – International Consortium for Open Source Model-based Development Tools, 53 members Febr 2018

Founded Dec 4, 2007

## Open-source community services

- Website and Support Forum
- Version-controlled source base
- Bug database
- Development courses
- [www.openmodelica.org](http://www.openmodelica.org)

## Code Statistics



## Industrial members

- ABB AB, Sweden
- Berger IT-Cosmos, Germany
- Bosch Rexroth AG, Germany
- Brainheart Energy AB, Sweden
- CDAC Centre, Kerala, India
- Creative Connections, Prague
- DHI, Aarhus, Denmark
- Dynamica s.r.l., Cremona, Italy
- EDF, Paris, France
- Equa Simulation AB, Sweden
- Fraunhofer IWES, Bremerhaven
- INRIA, Rennes, France
- ISID Dentsu, Tokyo, Japan

- Maplesoft, Canada
- RTE France, Paris, France
- Saab AB, Linköping, Sweden
- Scilab Enterprises, France
- SKF, Göteborg, Sweden
- TLK Thermo, Germany
- Siemens Turbo, Sweden
- Sozhou Tongyuan, China
- Talent Swarm, Spain
- VTI, Linköping, Sweden
- VTT, Finland
- Wolfram MathCore, Sweden

## University members

- FH Bielefeld, Bielefeld, Germany
- University of Bolivar, Colombia
- TU Braunschweig, Germany
- University of Calabria, Italy
- Univ California, Berkeley, USA
- Chalmers Univ, Control, Sweden
- Chalmers Univ, Machine, Sweden
- TU Darmstadt, Germany
- TU Delft, The Netherlands
- TU Dresden, Germany
- Université Laval, Canada
- Georgia Inst of Technology, USA
- Ghent University, Belgium
- Halmstad University, Sweden
- Heidelberg University, Germany
- TU Hamburg/Harburg Germany
- IIT Bombay, Mumbai, India
- KTH, Stockholm, Sweden
- Linköping University, Sweden
- Univ of Maryland, Syst Eng USA
- Univ of Maryland, CEEE, USA
- Politecnico di Milano, Italy
- Ecoles des Mines, CEP, France
- Mälardalen University, Sweden
- Univ Pisa, Italy
- Univ College SouthEast Norway
- Tsinghua Univ, Beijing, China
- Vanderbilt Univ, USA

# Spoken-Tutorial step-by-step OpenModelica and Modelica Tutorial Using OMEdit. Link from [www.openmodelica.org](http://www.openmodelica.org)

**OpenModelica** Login Create an account

Spoken Tutorial Software Training Creation News Forums About Statistics

HOME DOWNLOAD TOOLS & APPS USE Search Tutorials

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

English (804)

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 is an input or output variable. [Read more](#)

Number of students/teachers trained in their colleges/schools

Year	Number of students/teachers
2011	~50,000
2012	~70,000
2013	~100,000
2014	~400,000
2015	~500,000
2016	~1,500,000

About 12 results found.

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..

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 ..

3. [Developing an equation-based model](#)  
Foss : OpenModelica - English  
Outline: Introduction to OMEdit Declaration of variables and equations Simulation of a model in

# OMNotebook Interactive Electronic Notebook Here Used for Teaching Control Theory

OMNotebook: Kalman.onb

File Edit Cell Format Insert Window Help

1 Kalman Filter

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

Another difficulty is that the measured quantities often contain noise. We can model this noise as a disturbance  $e$ .

$$\dot{\hat{x}} = A\hat{x} + Bu + e$$

Here are  $e$  denoting a disturbance in the input signal. This disturbance can be evaluated by the difference  $y - \hat{x}$ .

$K(y(t))$

By using this quantity as feedback we obtain the observer equation:

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

Now form the error as

$$x - \hat{x}$$

The differential error is

Ready

OMNotebook: Kalman.onb\*

File Edit Cell Format Insert Window Help

```
model KalmanFeedback
    parameter Real A[:,size(A, 1)] = {{0,1},{1,0}} ;
    parameter Real B[size(A, 1),:] = {{0},{1}};
    parameter Real C[:,size(A, 1)] = {{1,0}};
    parameter Real[2,1] K = [2.4;3.4];
    parameter Real[1,2] L = [2.4,3.4];
    parameter Real[:,:] ABL = A-B*L;
    parameter Real[:,:] BL = B*L;
    parameter Real[:,:] Z = zeros(size(ABL,2),size(AKC,1));
    parameter Real[:,:] AKC = A-K*C;
    parameter Real[:,:] Anew = [0,1,0,0 ; -1.4, -3.4, 2.4,3.4; 0,0,-2.4,1;0,0,-2.4,0];
    parameter Real[:,:] Bnew = [0;1;0;0];
    parameter Real[:,:] Fnew = [1;0;0;0];
    stateSpaceNoise Kalman(stateSpace.A=Anew,stateSpace.B=Bnew, stateSpace.C=[1,0,0,0],
    stateSpace.F = Fnew);
    stateSpaceNoise noKalman;
end KalmanFeedback;
```

```
simulate(KalmanFeedback,stopTime=3)
plot({{Kalman.stateSpace.y[1],noKalman.stateSpace.y[1]}})
```

true

Plot by OpenModelica

Ready Ln 12, Col 39

# OpenModelica MDT Eclipse Plug-in: Code Outline and Hovering Info

The screenshot shows the Eclipse IDE interface for the OpenModelica MDT Plug-in. The top bar includes the title 'Modelica - OpenModelica/Compiler/Absyn.mo - Eclipse SDK' and standard menu items: File, Edit, Navigate, Search, Project, Run, Field Assist, Window, Help.

The left side features the 'Modelica Projects' view, which displays a tree structure of Modelica projects and files. A file named 'Absyn.mo' is currently selected.

The central workspace contains two open editors:

- The top editor shows the source code of 'Absyn.mo'. A tooltip is displayed over the line of code: 'function getCrefFromExp "function: getCrefFromExp Returns a flattened list of the component references in an expression"'. This illustrates the 'Hovering Info' feature.
- The bottom editor shows the 'Problems' view, indicating there are 113 errors, 0 warnings, and 0 infos. A list of errors is provided, all related to identifier differences.

A large blue callout box points from the text 'Identifier Info on Hovering' to the tooltip in the code editor.

On the right side, there is a vertical toolbar with icons for different views, and at the bottom right, a status bar with the text 'Ctrl Contrib (Bottom)'.

**Code Outline for easy navigation within Modelica files**

**Identifier Info on Hovering**

```
case (MATRIX(matrix = expl1))
  local list<list<list<ComponentRef>>> res1;
  equation
    res1 = Util.listListMap(expl1, getCrefFromExp);
    res2 = Util.listFlatten(res1);
    res = Util.listFlatten(res2);
  then
    res;
case (RANGE(start = e1, step = SOME(e3), stop = e2))
  equation
    11 = getCrefFromExp(e1);
    12 =
      function getCrefFromExp "function: getCrefFromExp
        Returns a flattened list of the
        component references in an expression"
        input Exp inExp;
        output list<ComponentRef> outComponentRefLst;
      end;
    res1 =
    13 =
    res =
  then
    res;
case (RAN)
equatio
  algorithm
    outComponentRefLst:=matchcontinue inExp
    local
      ComponentRef cr;
    11 =
    12 =
    res = listAppend(11, 12);
  then
```

113 errors, 0 warnings, 0 infos

Description

Errors (100 of 113 items)

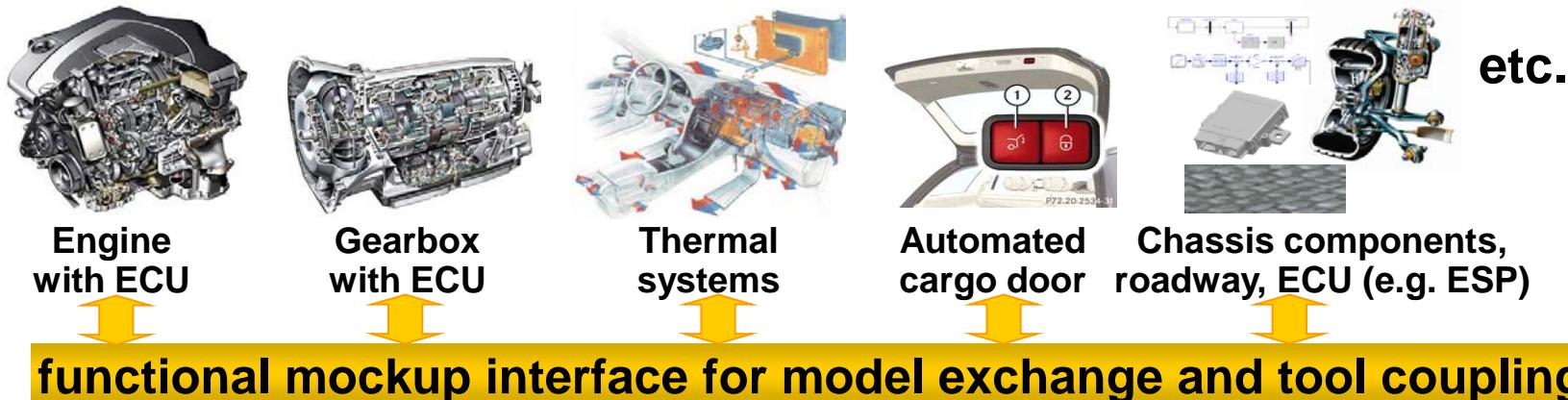
- The identifier at start and end are different
- The identifier at start and end are different
- The identifier at start and end are different, par

64M of 254M

Ctrl Contrib (Bottom)

# General Tool Interoperability & Model Exchange

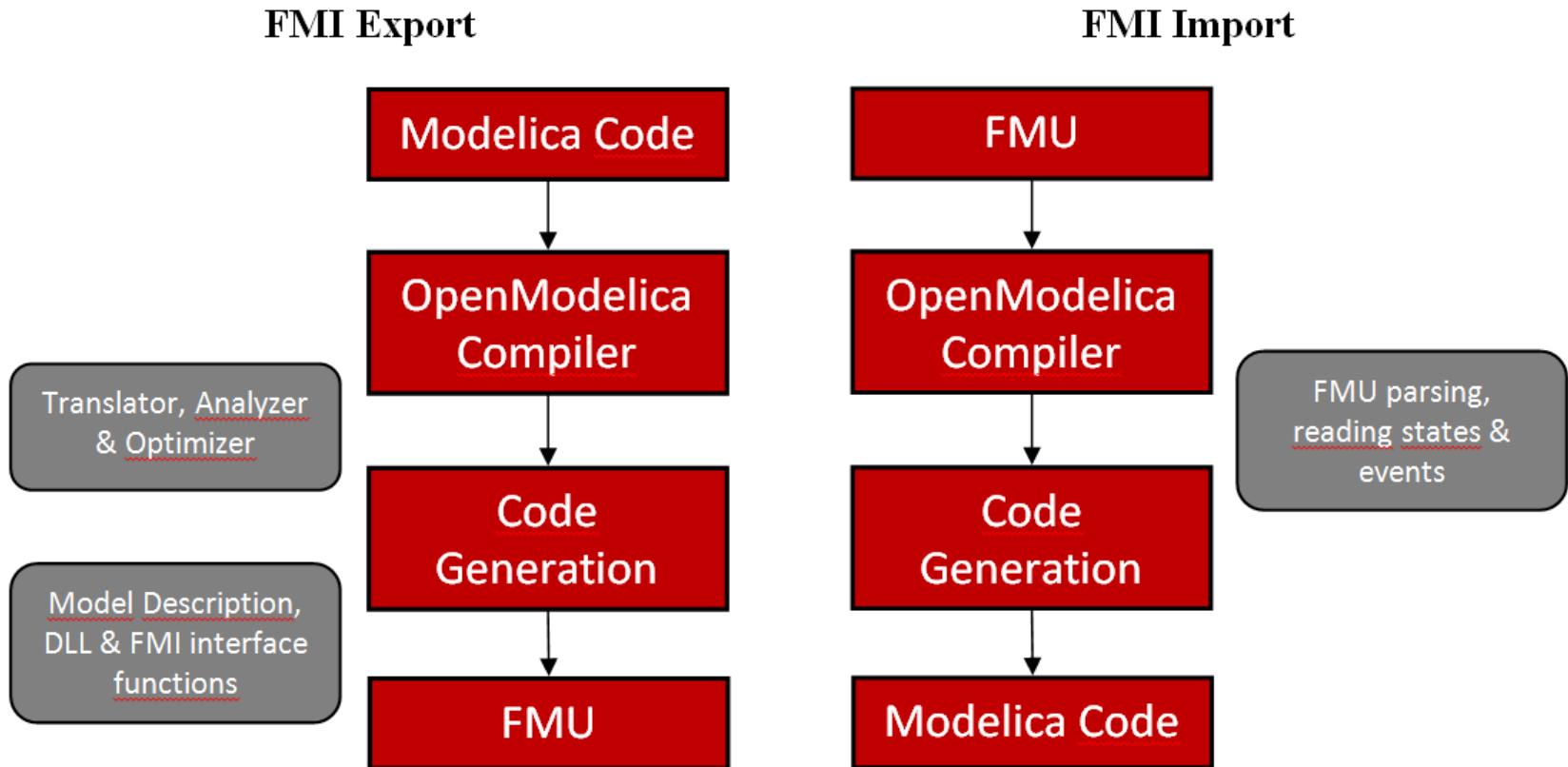
## Functional Mock-up Interface (FMI)



courtesy Daimler

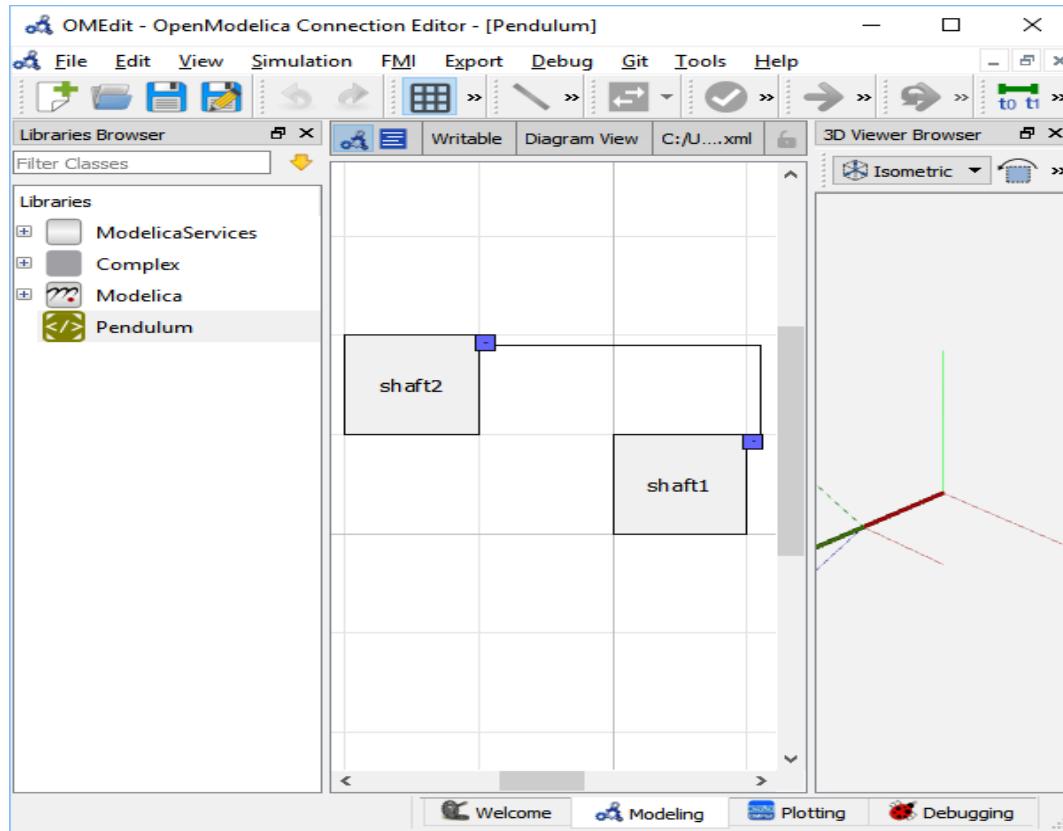
- FMI development was started by ITEA2 MODELISAR project. FMI is a Modelica Association Project now
- **Version 1.0**
- FMI for Model Exchange (released Jan 26,2010)
- FMI for Co-Simulation (released Oct 12,2010)
- **Version 2.0**
- FMI for Model Exchange and Co-Simulation (released July 25,2014)
- **> 100 tools** supporting it (<https://www.fmi-standard.org/tools>)

# OpenModelica Functional Mockup Interface (FMI)



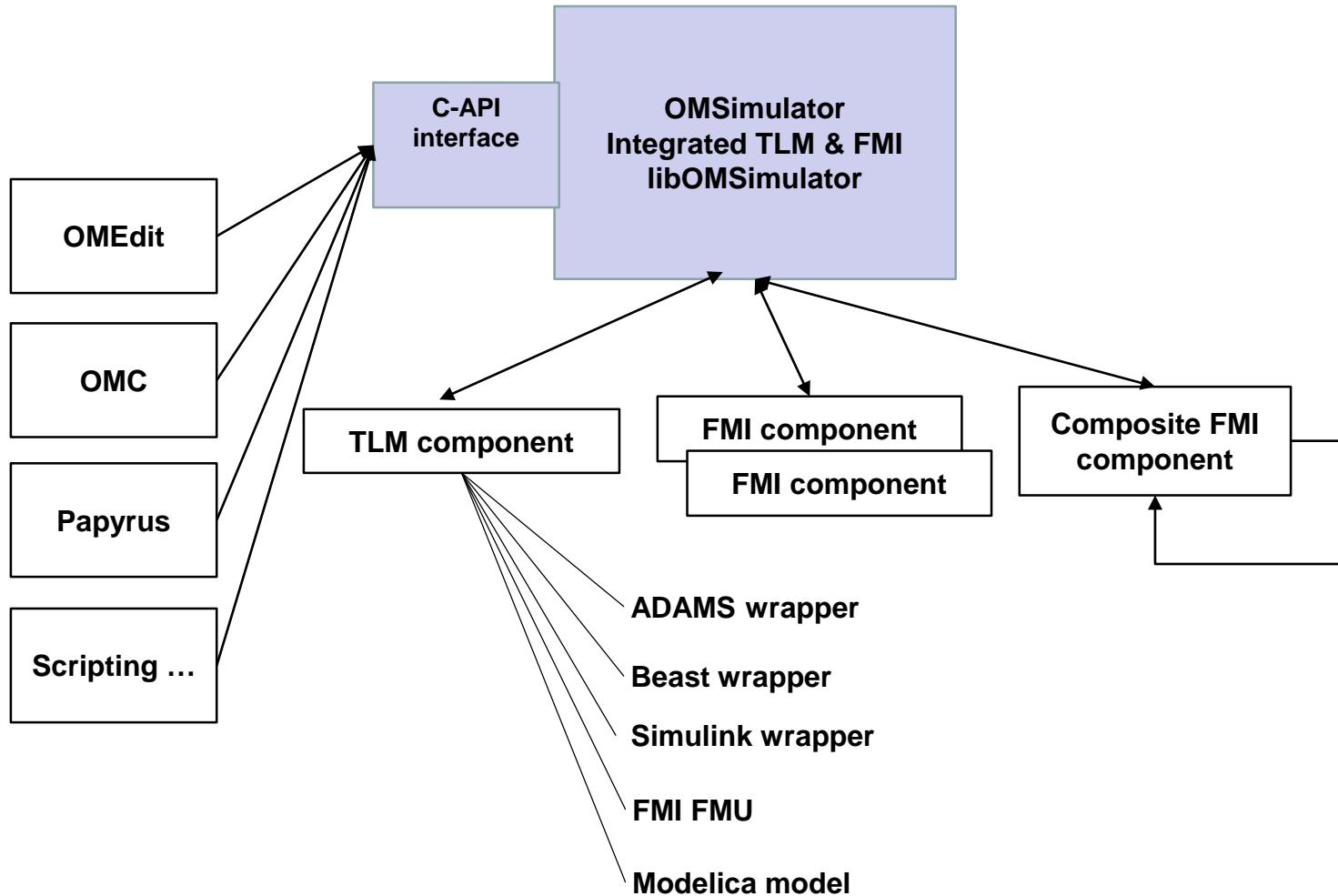
# OMSimulator Composite Model Editor with 3D Viewer

## Combine External (FMI) Models into New Models



- **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 **XML**-file

# OMSimulator – Integrated FMI and TLM-based Cosimulator/Simulator in OpenModelica



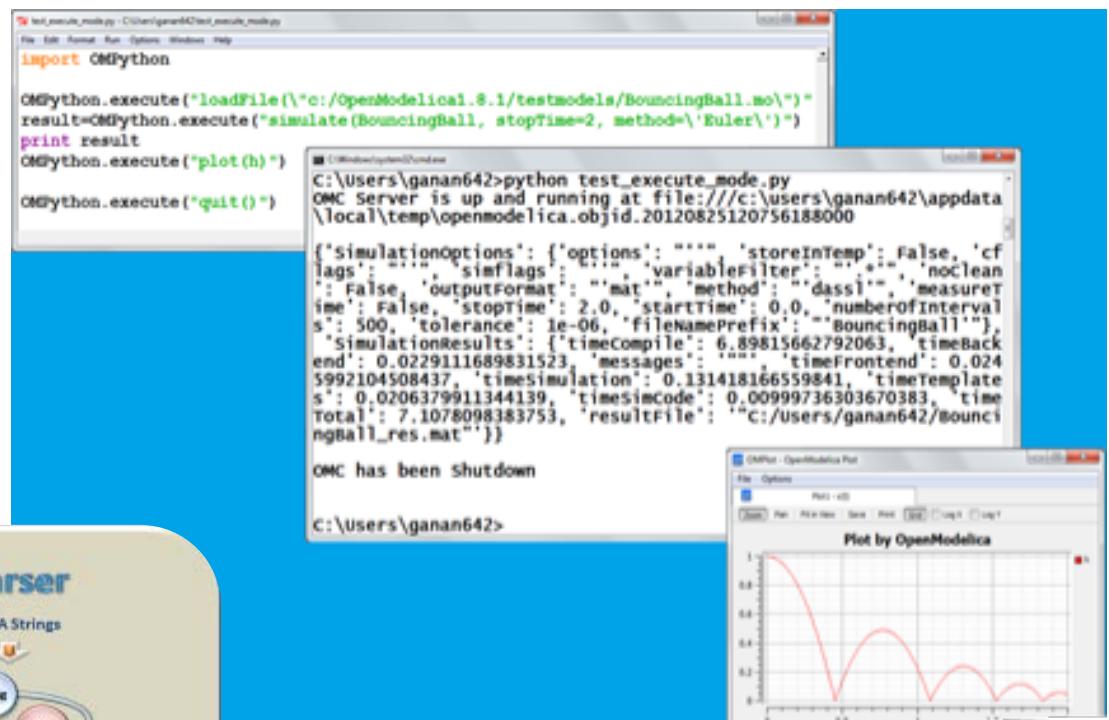
# OMSens – Sensitivity Analysis Subsystem

---

- ??? Fill in
- Under development, prototype available

# 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



The screenshot shows a Windows desktop environment with three windows open:

- A terminal window titled "test\_execute\_modapy" showing Python code execution:

```
import OMPython
OMPython.execute("loadFile(\"c:/OpenModelica1.8.1/testmodels/BouncingBall.mo\")")
result=OMPython.execute("simulate(BouncingBall, stopTime=2, method='Euler')")
print result
OMPython.execute("plot(h)")
OMPython.execute("quit()")
```

- An "OmcServer" window showing the command-line interface:

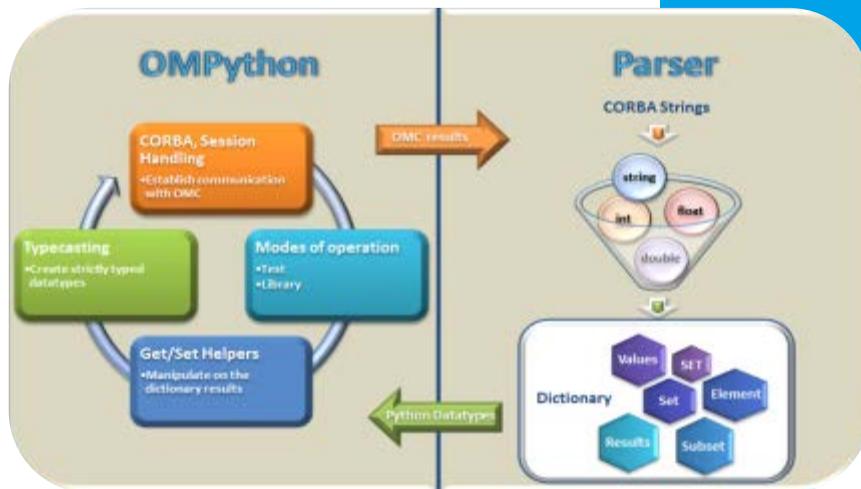
```
C:\Users\ganan642>python test_execute_modapy
OMC Server is up and running at file:///c:/users/ganan642/appdata\local\temp\openmodelica.objid.20120825120756188000
```

Output from the simulation:

```
{'simulationoptions': {'options': {'storeInTemp': False, 'cfTags': True, 'simflags': 'variableFilter', 'noclean': False, 'outputFormat': 'mat', 'method': 'dassl', 'measureTime': False, 'stopTime': 2.0, 'startTime': 0.0, 'numberOfIntervals': 500, 'tolerance': 1e-06, 'filenamePrefix': 'BouncingBall'}, 'SimulationResults': {'timeCompile': 6.89815662792063, 'timeBackend': 0.0229111689831523, 'messages': '', 'timeFrontend': 0.0245992104508437, 'timeSimulation': 0.131418166559841, 'timeTemplate': 0.0206379911344139, 'timesimcode': 0.00999736303670383, 'timeTotal': 7.1078098383753, 'resultFile': 'C:/users/ganan642/bouncingBall_res.mat'}}}
```

OMC has been shutdown

- A "OMPPlot - OpenModelica Plot" window showing a plot of a bouncing ball's height over time.



# 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 1.13.0 nightly build
- Works with Jupyter notebooks
- See separate presentation

## Control example with OMJulia in Jupyter notebooks

Use of Modelica + Julia in Process Systems Engineering Education

Complex models of "Seborg reactor"

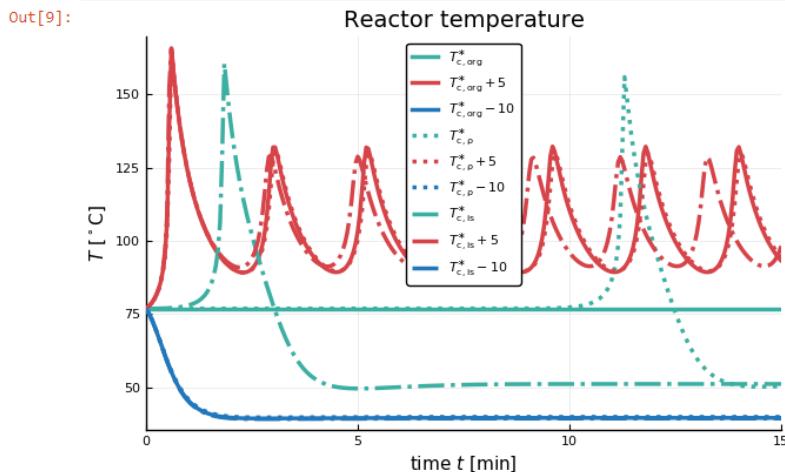
Bernt Lie\*, Arunkumar Palanisamy\*\*, Peter Fritzson\*\*

\*University of South-Eastern Norway, Norway

\*\*University of Linköping, Sweden

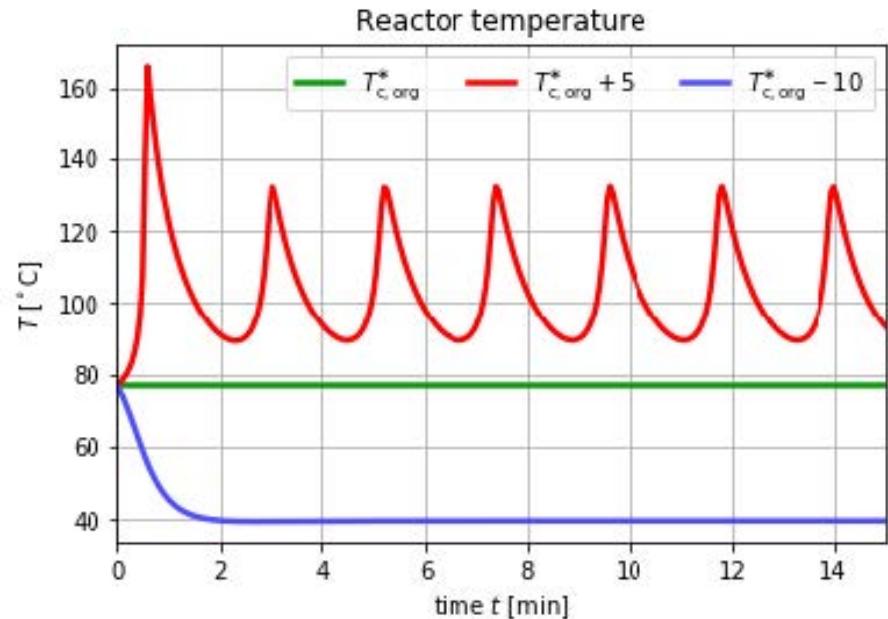
Introducing packages

```
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
```



# 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 nightly build
- Now (October 2018) basic version supporting basic simulation and plotting



# Embedded System Support in OpenModelica

- Code generation of real-time Controllers from Modelica models for small foot-print platforms



# 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!**

# Example – Code Generation to SHBS



# Code Generator Comparison, Full vs Simple

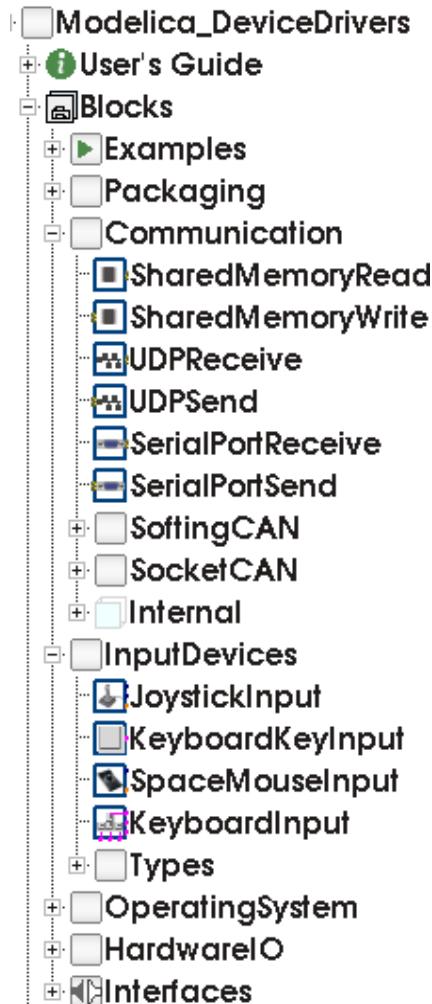
	<b>Full Source-code FMU targeting 8-bit AVR proc</b>	<b>Simple code generator targeting 8-bit AVR proc</b>
Hello World (0 equations)	43 kB flash memory 23 kB variables (RAM)	130 B flash memory 0 B variables (RAM)
SBHS Board (real-time PID controller, LCD, etc)	<b>68 kB</b> flash memory <b>25 kB</b> variables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

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

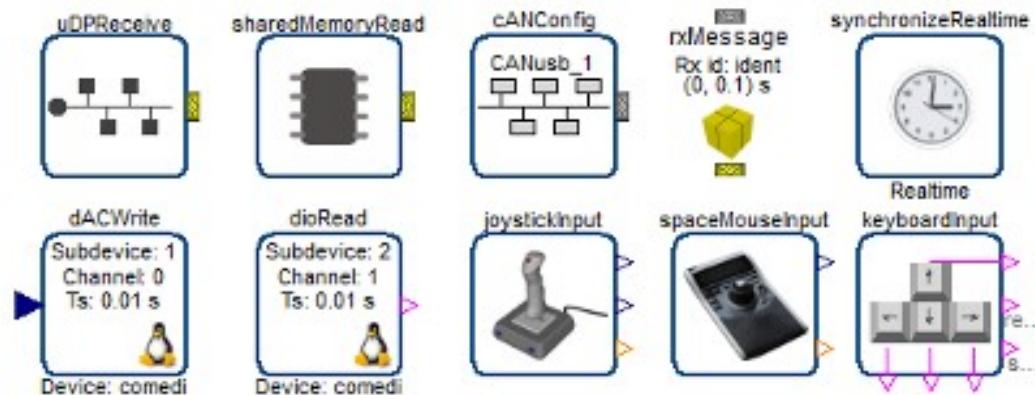
One of the more (ATmega328p; Arduino Uno) has 2 kB SRAM.

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

# Communication & I/O Devices: MODELICA\_DEVICEDRIVERS Library

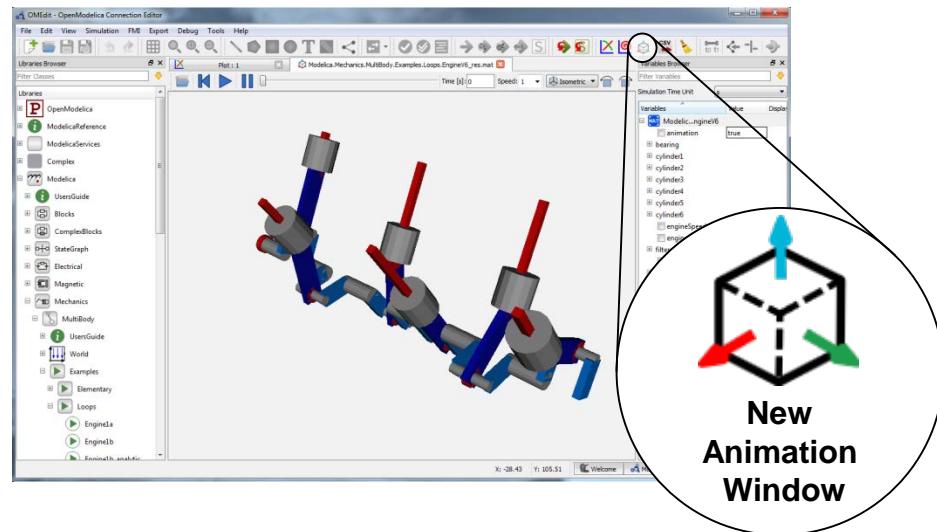
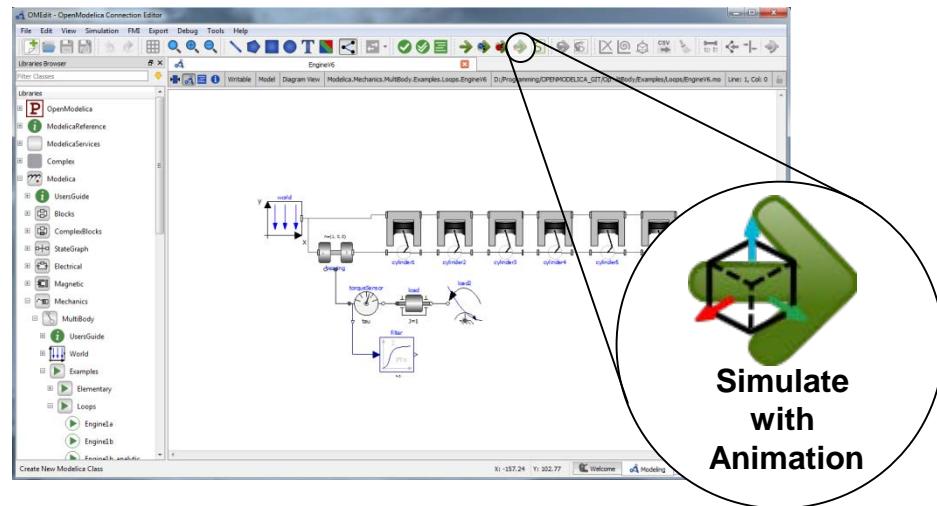
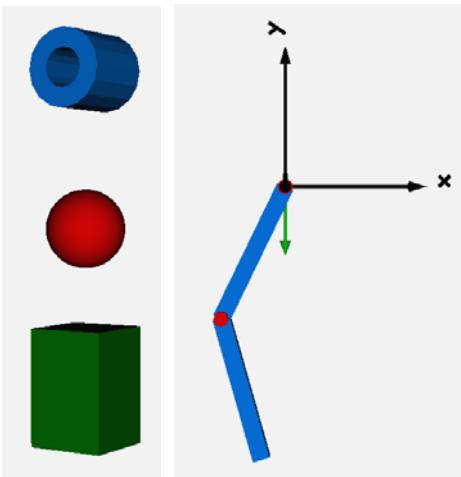
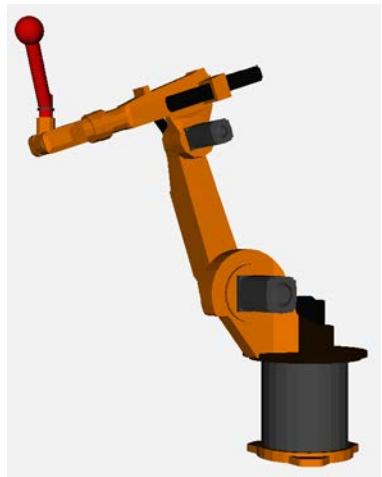


- Free library for interfacing hardware drivers
- Cross-platform (Windows and Linux)
- UDP, SharedMemory, CAN, Keyboard, Joystick/Gamepad
- DAQ cards for digital and analog IO (only Linux)
- Developed for **interactive real-time** simulations

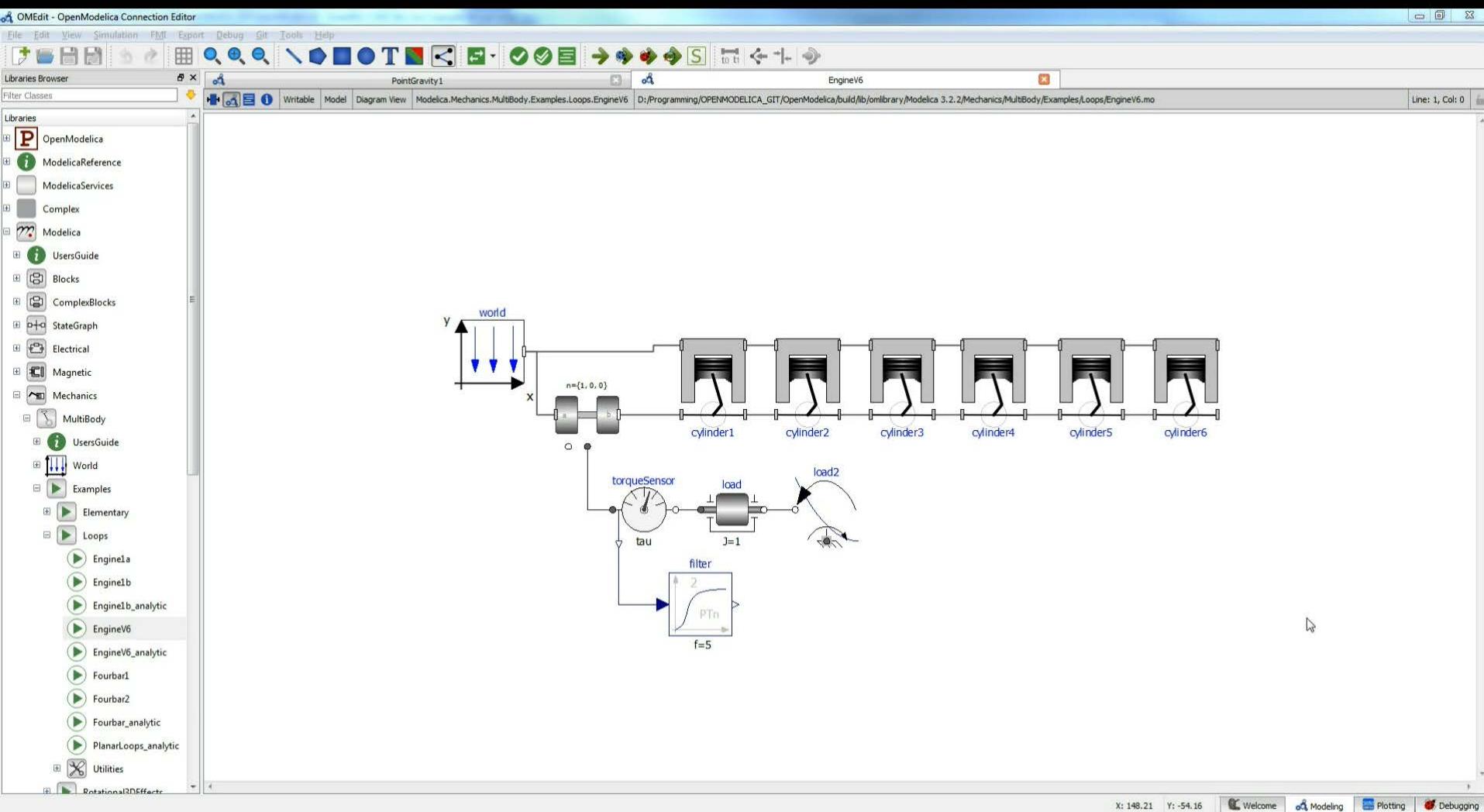


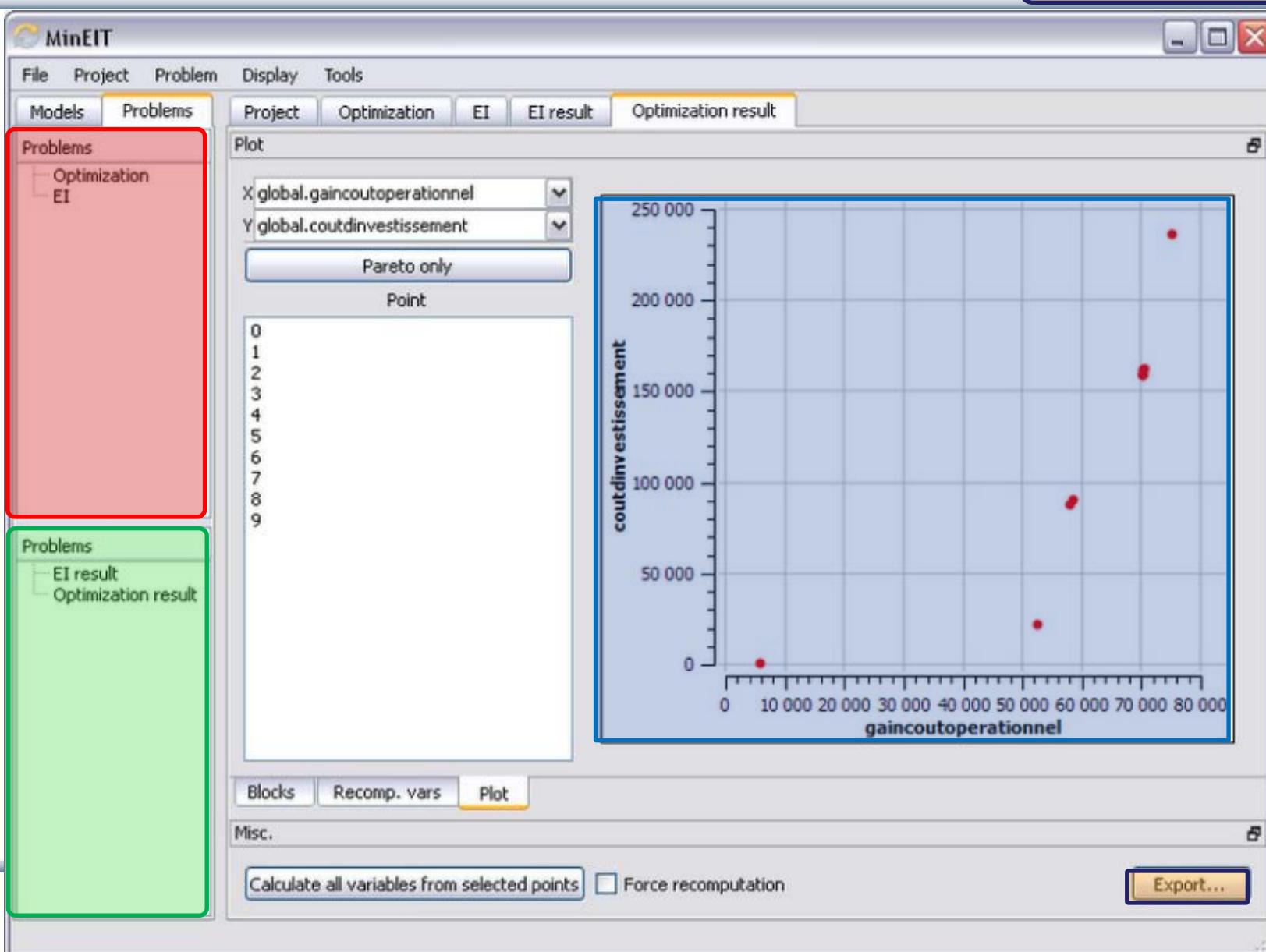
# OMEdit 3D Visualization of Multi-Body Systems

- 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



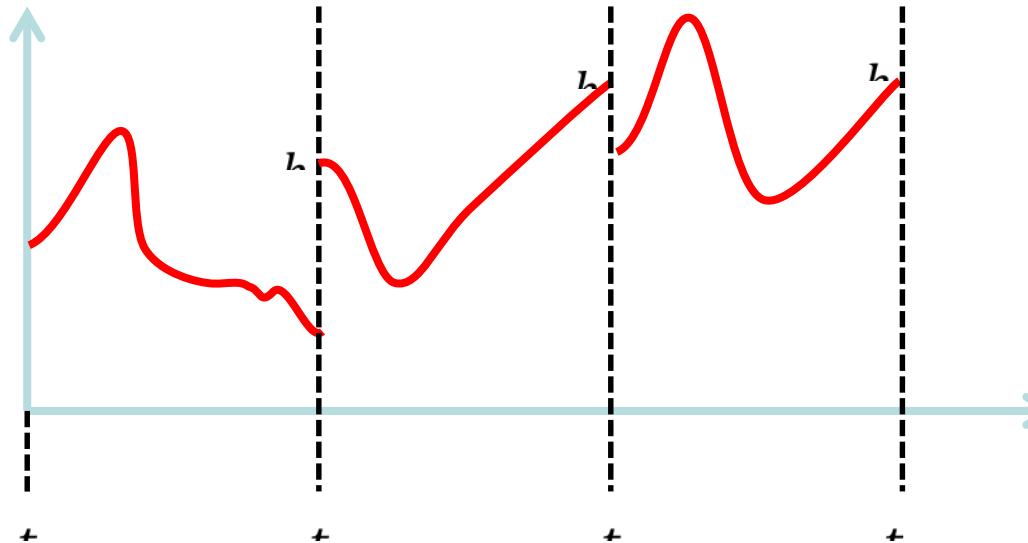


Here  
Pareto  
front  
optimiza-  
tion

# Optimization of Dynamic Trajectories Using Multiple-Shooting and Collocation

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

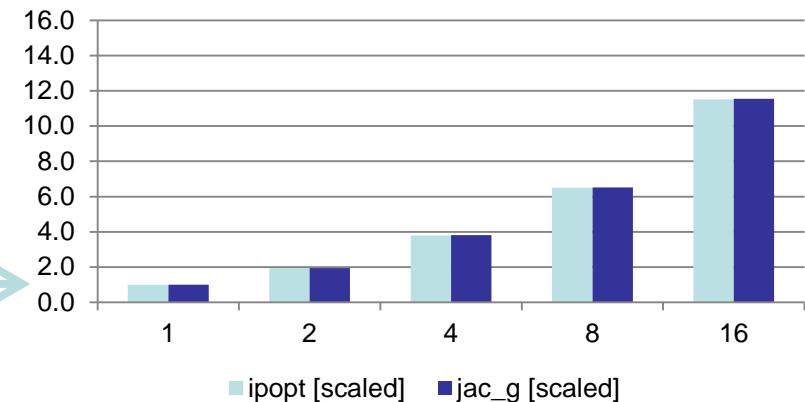
$$x_i(t_{i+1}) = h_i + \int_{t_i}^{t_{i+1}} f(x_i(t), u(t), t) dt \approx F(t_i, t_{i+1}, h_i, u_i), \quad x_i(t_i) = h_i$$



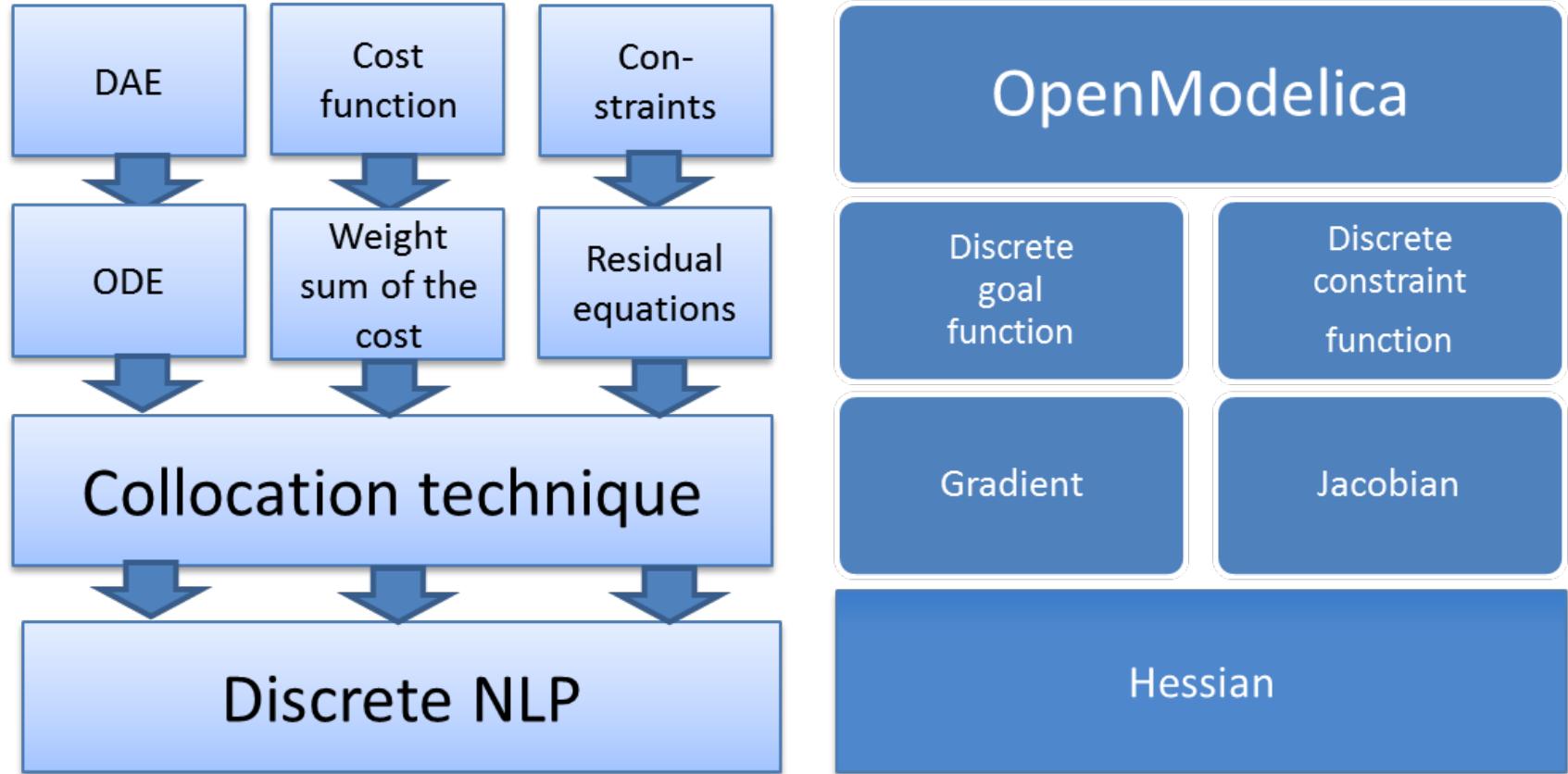
In OpenModelica 1.9.1  
beta release Jan 2014.

Example speedup, 16 cores:

MULTIPLE\_COLLOCATION



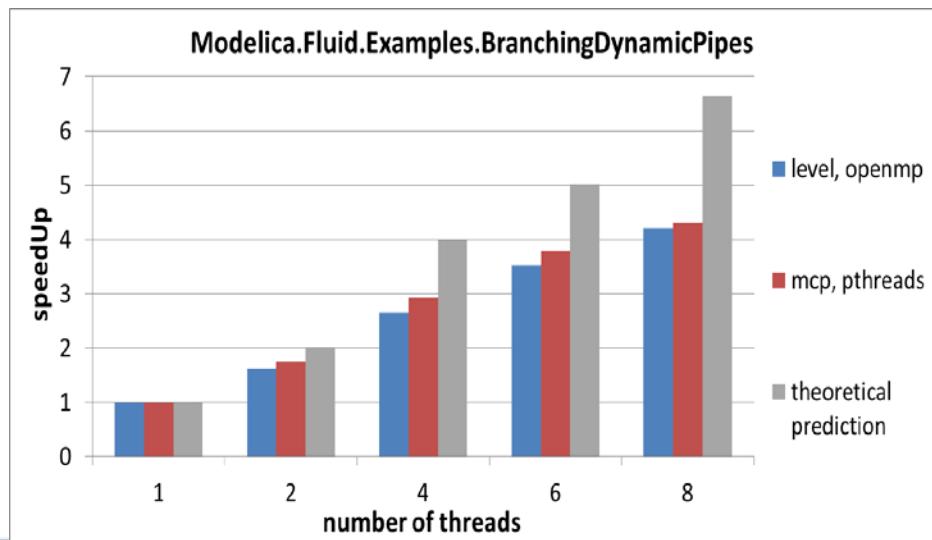
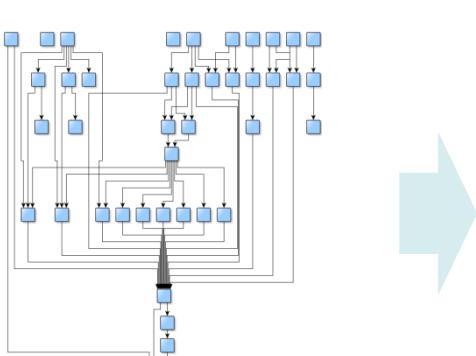
# OpenModelica Dynamic Optimization Collocation



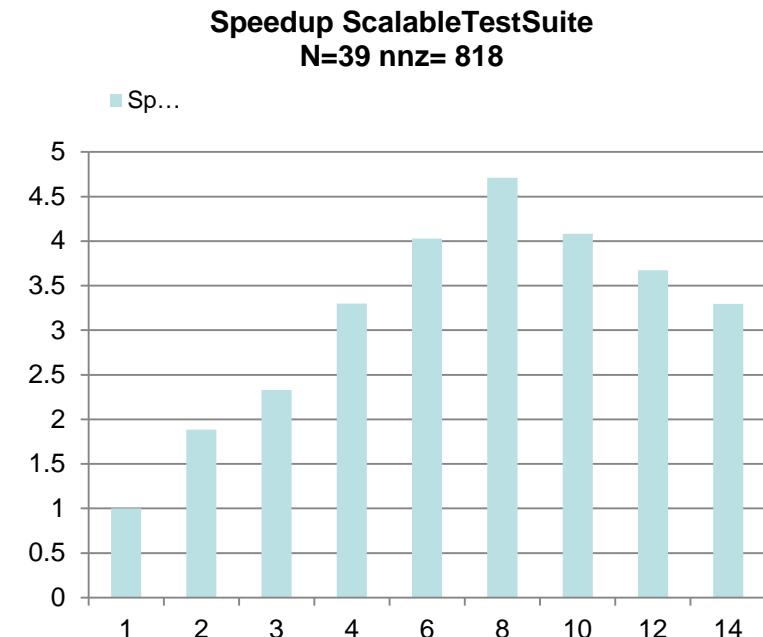
# OpenModelica Model Parallelization

## Faster Simulation on Multi-Core

### Automated parallelization of models



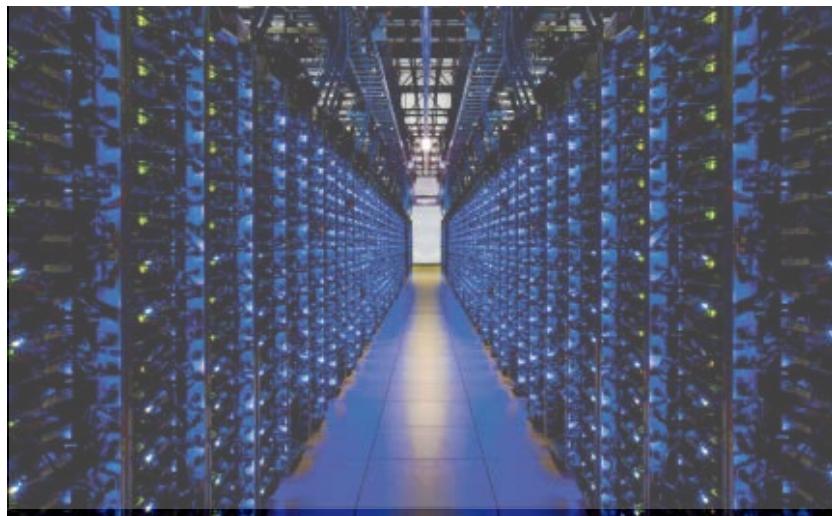
### Parallelizing numeric Jacobian computations in simulation



Speedup about 4  
using 8 threads

# 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**

# 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  
...

# Integrated Static-Dynamic OpenModelica Equation Model Debugger

Efficient  
handling  
of  
**Large**  
Equation  
Systems

Showing  
equation  
transfor  
mations  
of a  
model:

The screenshot shows the OMEdit Transformational Debugger interface with three main panels:

- Variables View:** Shows a tree view of variables under "Variables Browser" and two tables: "Defined In Equations" and "Used In Equations".
- Equations View:** Shows a table of equations with columns "Index", "Type", and "Equation". It also includes sections for "Defines" and "Depends".
- Source View:** Displays the source code of a Modelica model, specifically "Joints.mo". A red arrow points from the "Depends" section in the Equations View to the corresponding line in the source code where the variable is used.

Mapping dynamic run-time error to source model position

# Transformations Browser – EngineV6 Overview (11 116 equations in model)

The screenshot shows the OMEdit Transformational Debugger interface with the following panels:

- Variables**: Shows a tree view of variables under "Variables Browser". A red box highlights the "phi" node.
- Defined In Equations**: A table showing equations defined at index 587 and 5016. Index 587 is an initial assignment (nonlinear) for "phi". Index 5016 is a regular assignment (nonlinear) for "phi".
- Used In Equations**: A table showing equations where "phi" is used. It lists several assignments involving "cylinder3.B2.phi" and "cylinder3.Rod.body.w\_a[1]".
- Source Browser**: Displays the Modelica code for the "Connections.branch" function, which handles frame transformations and rotation calculations.
- Equations**: Shows a table of equations. A red box highlights the first equation: "regular (assignment) cylinder...ylinder3.Cylinder.s".
- Defines**: Shows the definitions for the selected equation: "der(cylinder3.B2.R\_relT[3,3])".
- Depends**: Shows dependencies for the selected equation: "cylinder3.B2.phi" and "cylinder3.Rod.body.w\_a[1]".
- Equation Operations**: Shows a list of operations performed on the equation, including solving, substituting, differentiating, scalarizing, simplifying, and inline expansion.

# Performance Profiling

(Here: Profiling all equations in MSL 3.2.1 DoublePendulum)

- ▶ Measuring performance of equation blocks to find bottlenecks
  - ▶ Useful as input before model simplification for real-time platforms
- ▶ Integrated with the debugger so it is possible to show what the slow equations compute
- ▶ Suitable for real-time profiling (less information), or a complete view of all equation blocks and function calls

Equations Browser							Defines
Index	Type	Equation	Execution count	Max time	Average time	Fraction	Variable
+ 876	regular	linear, size 2	4602	0.000501	0.0134	75.7%	damper.a_rel
- 836	regular	(assignment) ...evolute2.phi	1534	2.57e-05	0.000377	2.12%	revolute2.frame_b.f[2]
- 840	regular	(assignment) ...mper.phi_rel	1534	1.38e-05	0.000237	1.33%	
- 837	regular	(assignment) ...evolute2.phi	1534	8.38e-06	0.000235	1.32%	
- 841	regular	(assignment) ...mper.phi_rel	1534	8.48e-06	0.000192	1.08%	
- 849	regular	(assignment) ...mper.phi_rel	1534	8.04e-06	0.000146	0.824%	

# Equation Model Debugger on Siemens Model (Siemens Evaporator test model, 1100 equations)

Pointing out the buggy equation  
 $y = u1/u2;$   
that gives division by zero



The screenshot shows the OMEdit - Transformational Debugger interface. The Source Browser window displays a block named "Division" with the following code:

```
extents={{-100,-100},{100,100}}, graphics={Rectangle(extent={-100,-100},{100,100}), Line(points={(-100,-60), (0,0,255)}, color=(0,0,255)), Line(points={(-100,-60), (-30,-40)}, color=(0,0,255)), Line(points={(50,0),(100,0)}, color=(0,0,255)), Line(points={(-30,0),(30,0)}, color=(0,0,255)), Line(points={(-15,25.99),(15,25.99)}, color=(0,0,0)), Line(points={(-15,-25.99),(15,25.99)}, color=(0,0,0)), Ellipse(extent={(-50,50), (50,50)}), color=(0,0,255))}); product;
```

Below the code, the "equation" section contains:

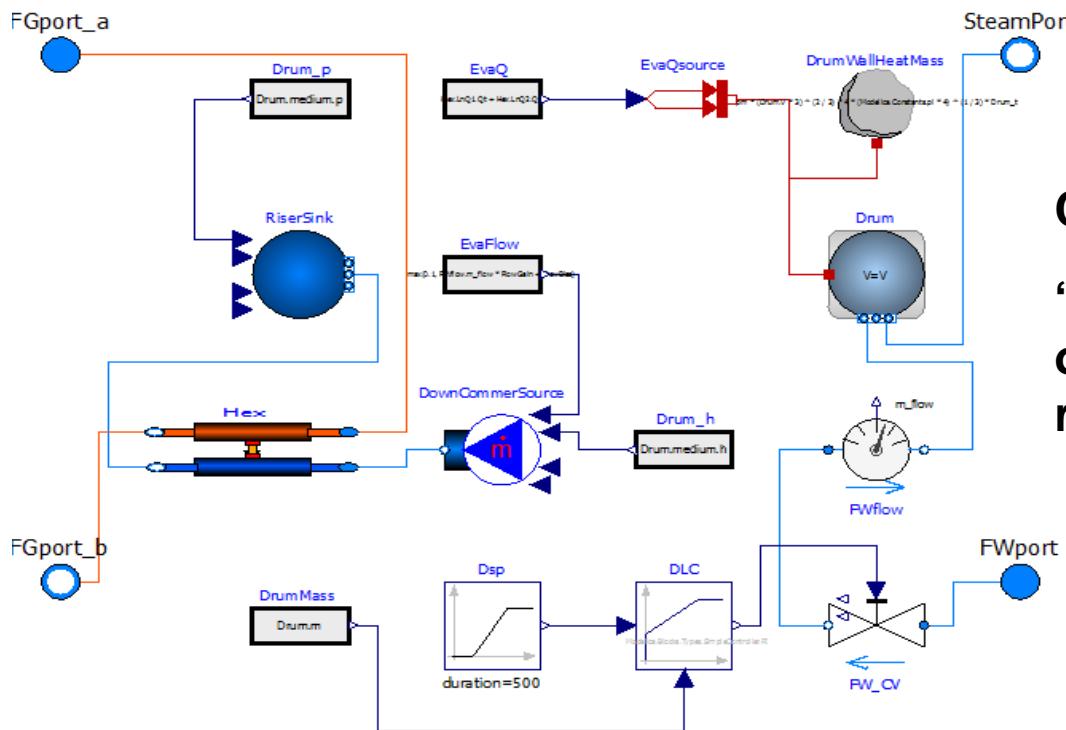
```
equation
y = u1/u2;
annotation (
Documentation(info=<html>
<pre>
This block computes the output <b>y</b> (element-wise)
by <i>dividing</i> the corresponding elements of
the two inputs <b>u1</b> and <b>u2</b>;
</pre>
<math>y = u1 / u2;</math>
</pre>
</html>),
Icon(coordinateSystem(
preserveAspectRatio=true,
extent={(-100,-100),(100,100)},
initialScale=0.1), graphics={
Line(points={(50,0),(100,0)}, color={0,0,127}),
Line(points={(-30,0),(30,0)}),
Ellipse(fillPattern=FillPattern.Solid, extent={(-5,20),
{5,30)}},
Ellipse(fillPattern=FillPattern.Solid,
extent={(-5,-30),(5,-20)}),
Ellipse(lineColor={0,0,127}, extent={(-50,-50),
{50,50)}},
Text(
lineColor={0,0,255},
extent={(-150,110),(150,150)},
textString="#name#")
}
```

The "Defined In Equations" and "Used In Equations" tables show various variables and their definitions. The "Equations" browser shows the equation  $y = u1/u2$  with its dependencies and operations.

# 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



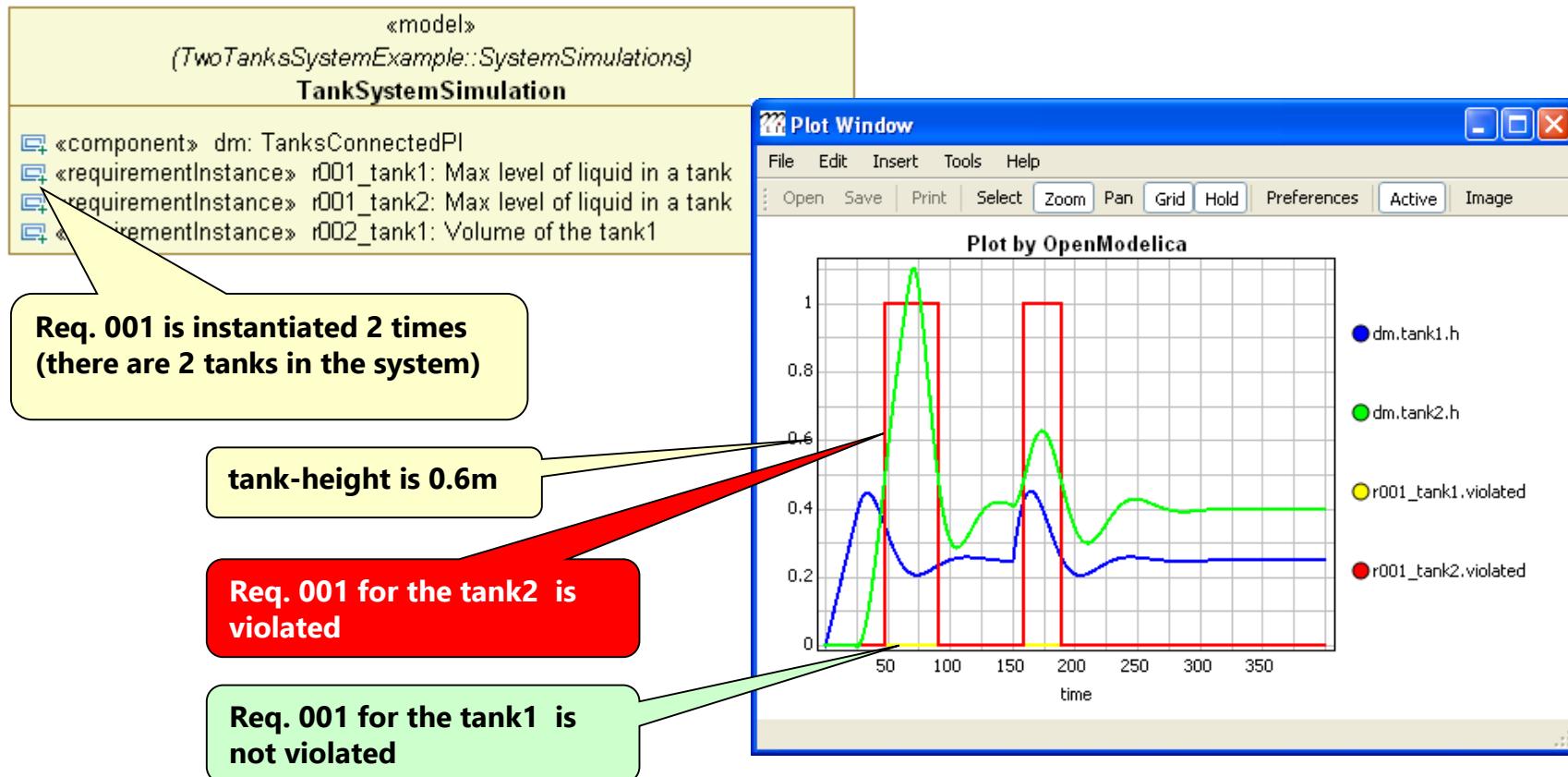
**Conclusion from the evaluation:**  
“...the profiler makes the process of performance optimization radically shorter.”

# OpenModelica – ModelicaML UML Profile

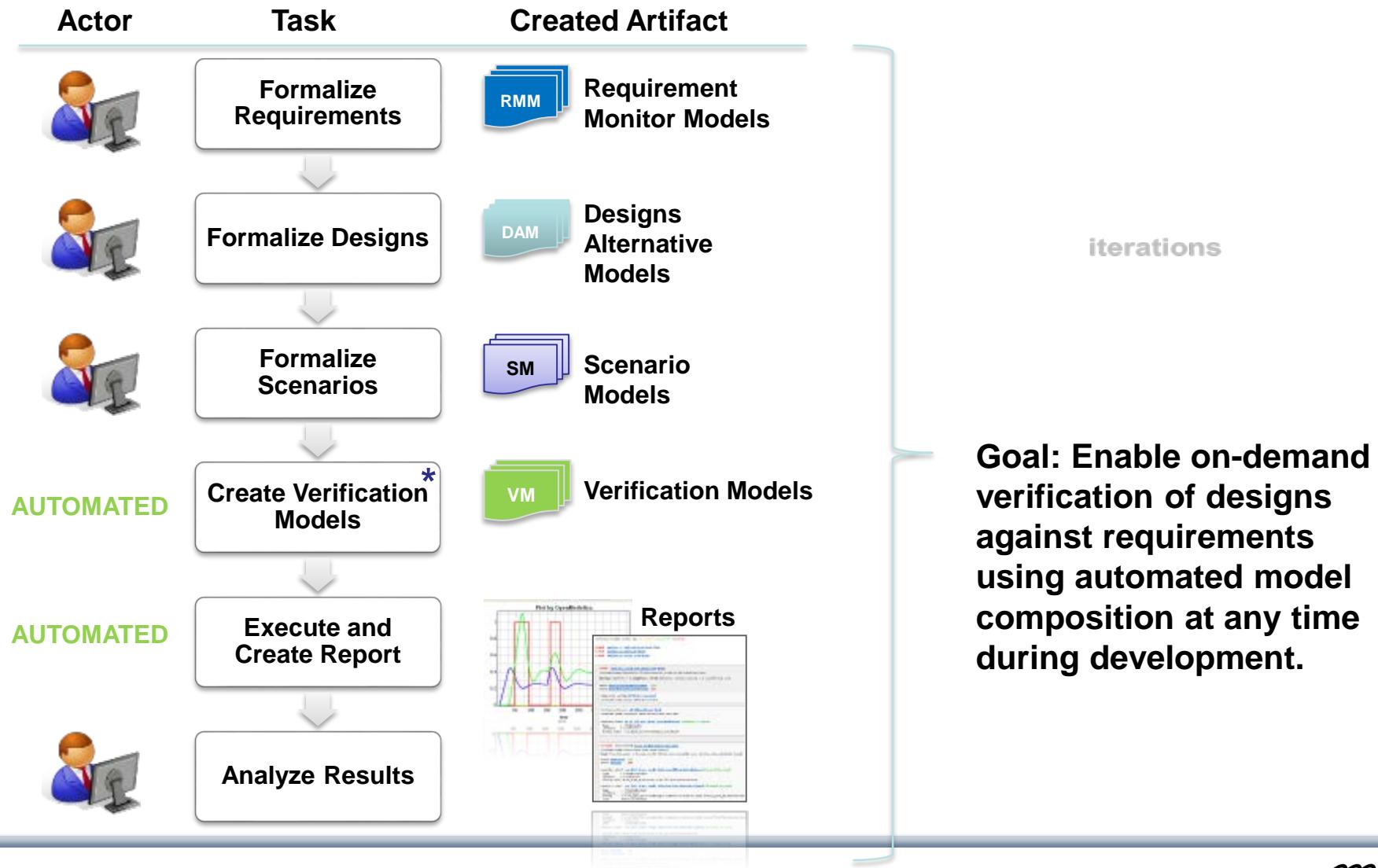
Based on Open-Source Papyrus UML and OpenModelica

- 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 yet exist in Modelica language (extension work ongoing)
    - 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

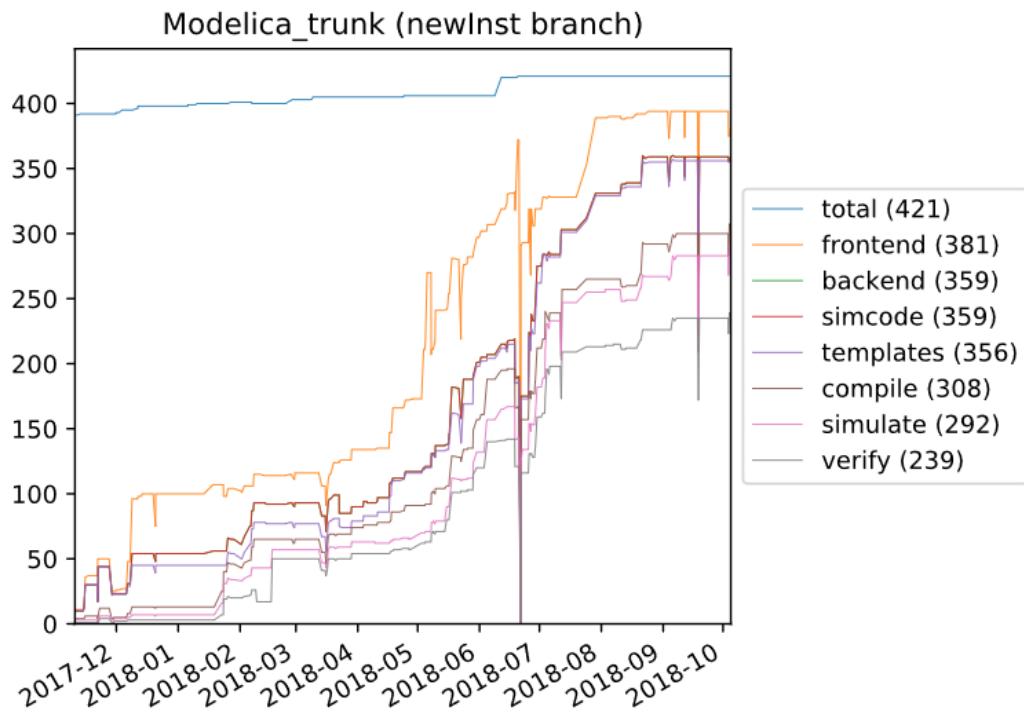


# vVDR Method – virtual Verification of Designs vs Requirements



# Outlook: New OpenModelica Frontend for Large-Scale models

- Soon: New OMC Compiler frontend for fast compilation and large-scale models
- Been under development the past 2-3 years
- Now (Oct 05) simulates 69% of MSL models, coverage increases about 6% per month
- About 10-200 times faster than the old frontend, depending on model



# OpenModelica DAEMode for Large-Scale models

- Goal – to handle hundreds of thousands to millions of equations
- Introduced sparse solvers in the solution chain:
  - KLU for linear algebraic equations,
  - Kinsol for nonlinear algebraic equations, and
  - IDA for sparse differential-algebraic equations.
- DAEmode: after index reduction, IDA solves the differential equations and the algebraic loops simultaneously
- Largest system so far: electro-mechanical power system model with about 600.000 differential-algebraic equations
- Under development for even larger systems

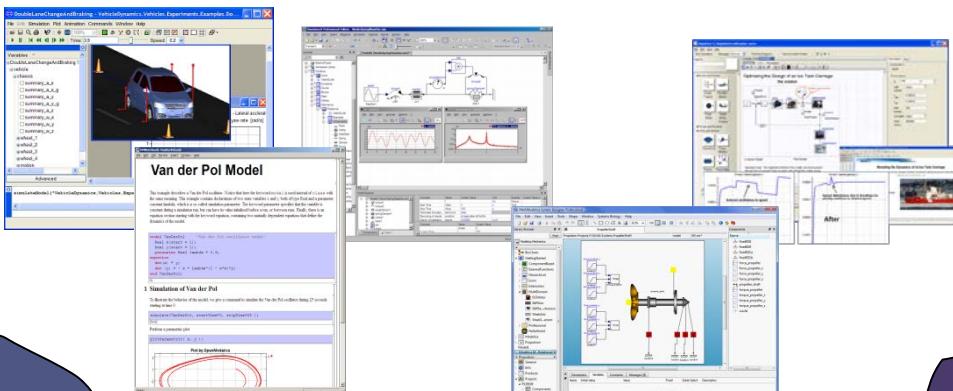
# Summary and Questions

Multi-Domain  
Modeling



Visual Acausal  
Component  
Modeling

[www.modelica.org](http://www.modelica.org) – Language, Standard Library  
[www.openmodelica.org](http://www.openmodelica.org) – Open Source Tool



Typed  
Declarative  
Textual Language

Thanks for listening!

Hybrid  
Modeling