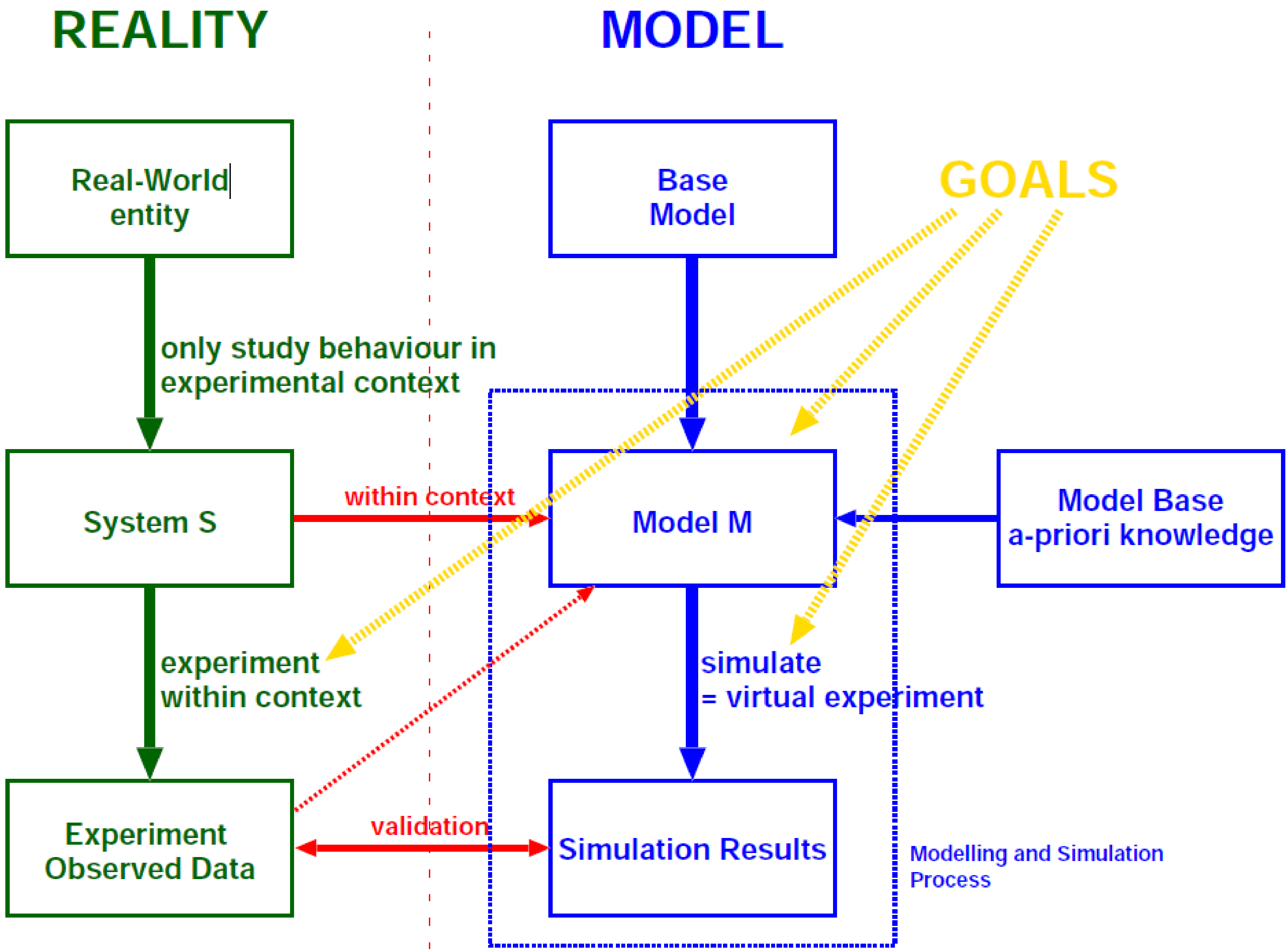
Modelling of Physical Systems (for Computer Scientists)

Hans Vangheluwe

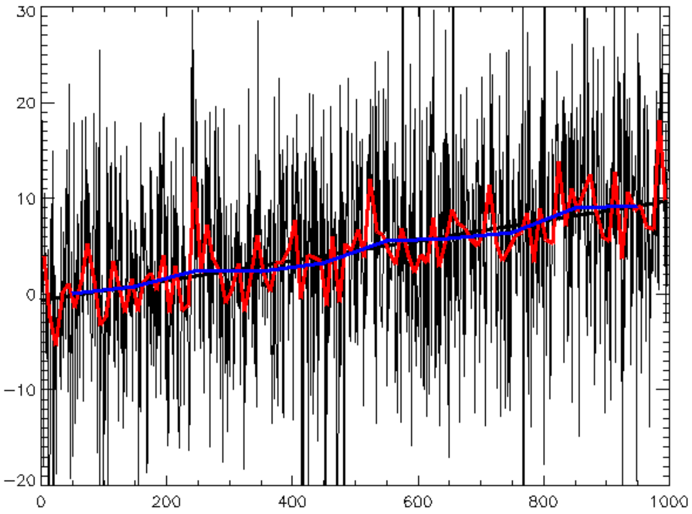
Universiteit Antwerpen
Ansymo
Antwerp Systems & Software Modelling
University of Antwerp
Nexor
Cyber-Physical Systems
University of Antwerp

FLANDERS MAKE
MANUFACTURING INNOVATION NETWORK
MPM4CPS
cost



Bernard P. Zeigler. *Multi-faceted Modelling and Discrete-Event Simulation*. Academic Press, 1984.

models based on measurements



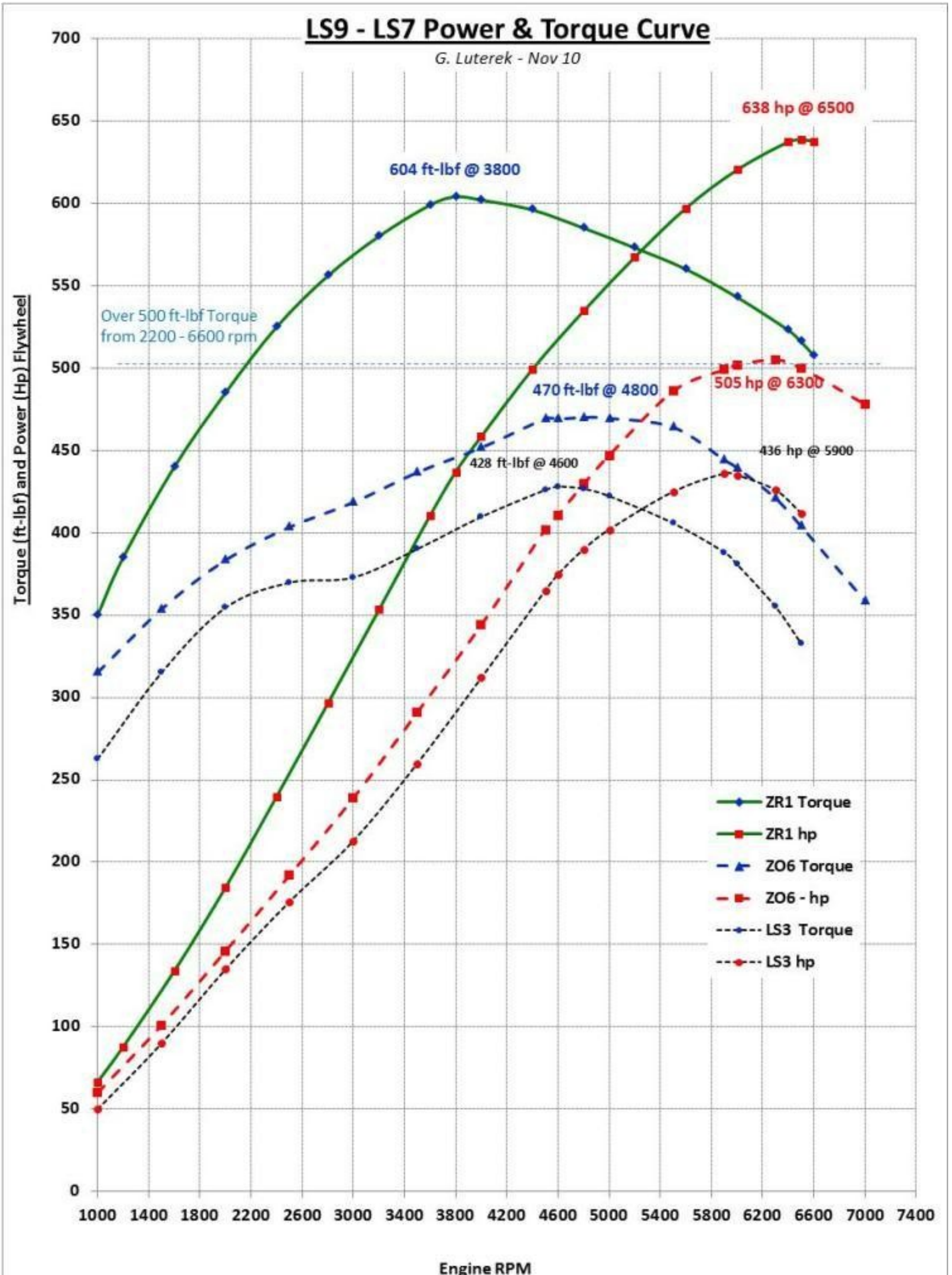
- instance (technology) – specific

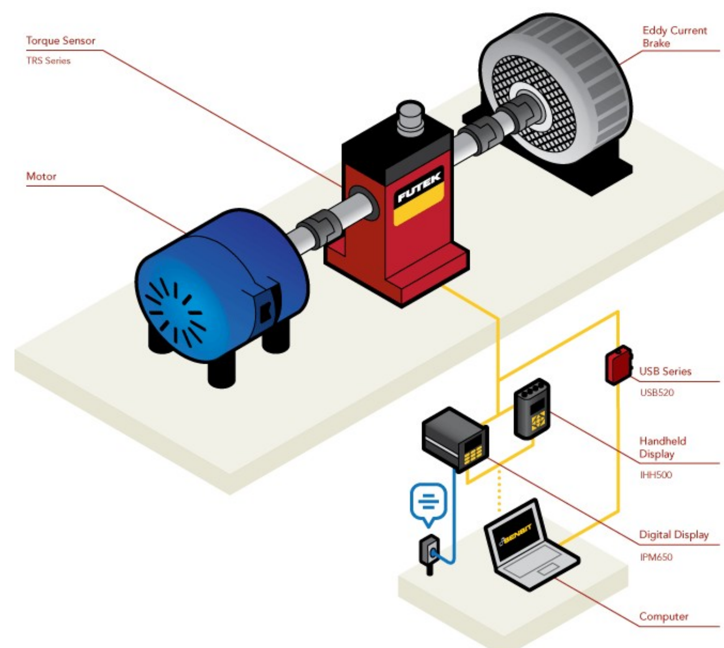
- high (experimentation) cost
- may not even be possible to measure

- allows reproducing data, no extrapolation;
  no insight/explanation

- inductive vs. deductive modelling workflow
  science vs. engineering, usually combination

Torque Curve "model" (measured)





mathematical model + data

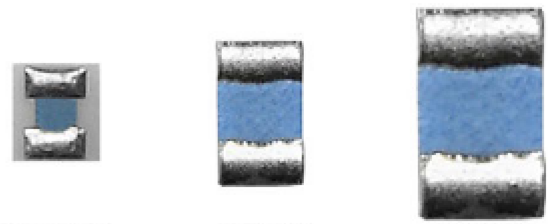
VISHAY.

www.vishay.com

CH

Vishay Sfernice

High Frequency 50 GHz Thin Film Chip Resistor



CH02016 (flip chip) | CH0402 (flip chip) | CH0603 (flip chip)

FEATURES

- Operating frequency 50 GHz
- Thin film microwave resistors
- Flip chip, wraparound or one face termination
- Small size, down to 20 mils by 16 mils
- Edged trimmed block resistors
- Pure alumina substrate (99.5 %)
- Ohmic range: 10R to 500R
- Design kits available
- Small internal reactance (LC down to 1 x 10^{-24})
- Tolerance 1 %, 2 %, 5 %
- TCR: 100 ppm/°C in (-55 °C, +155 °C) temperature range
- TCR: 50 ppm/°C available upon request for 10 Ω to150 Ω ohmic range
- Material categorization: for definitions of compliance please see www.vishay.com/doc?99912

Pb-free

RoHS COMPLIANT

HALOGEN FREE

GREEN (5-2008)

ADDITIONAL RESOURCES

3D Models

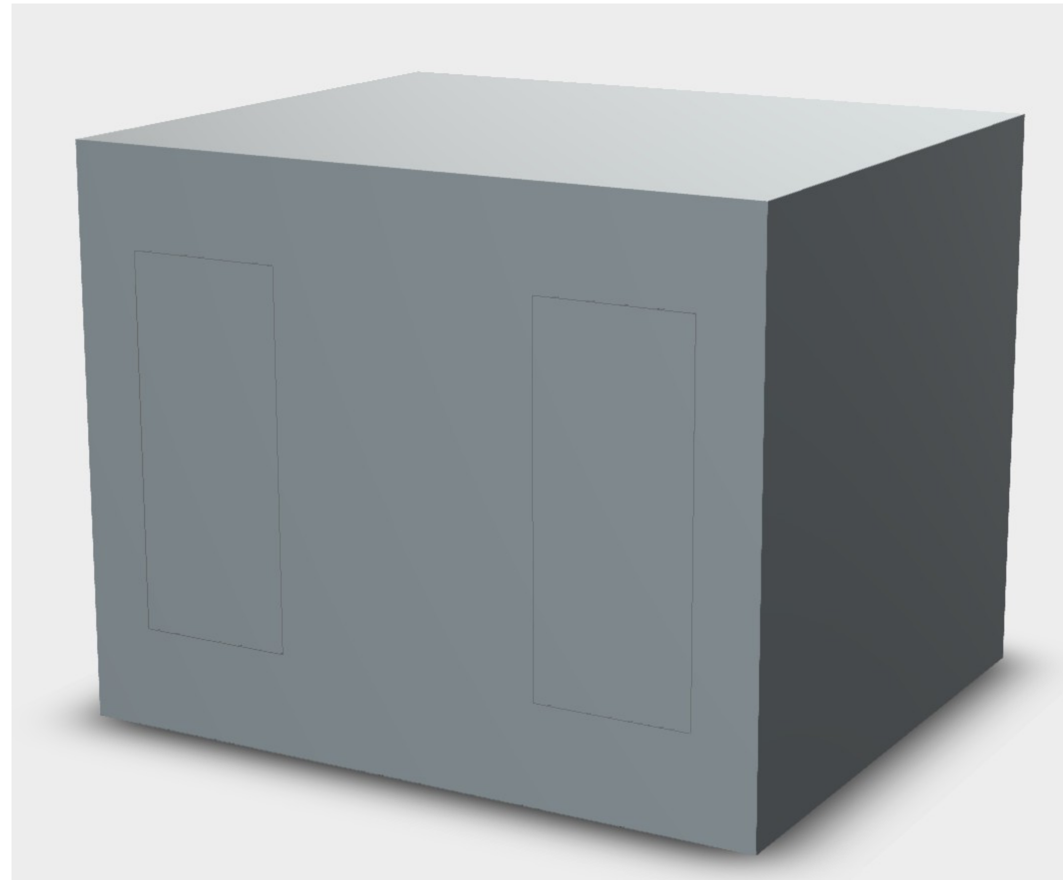
Those miniaturized components are designed in such a way that their internal reactance is very small. When correctly mounted and utilized, they function as almost pure resistors on a very large range of frequency, up to 50 GHz.

STANDARD ELECTRICAL SPECIFICATIONS

MODEL	SIZE	RESISTANCE RANGE Ω	RATED POWER Pn W	LIMITING ELEMENT VOLTAGE V	TOLERANCE ± %	TEMPERATURE COEFFICIENT ± ppm/°C
CH02016	02016	10 to 500	0.030	30	2, 5	100 (50 upon request)
CH0402	0402	10 to 500	0.050	37	1, 2, 5	100 (50 upon request)
CH0603	0603	10 to 500	0.125	50	1, 2, 5	100 (50 upon request)

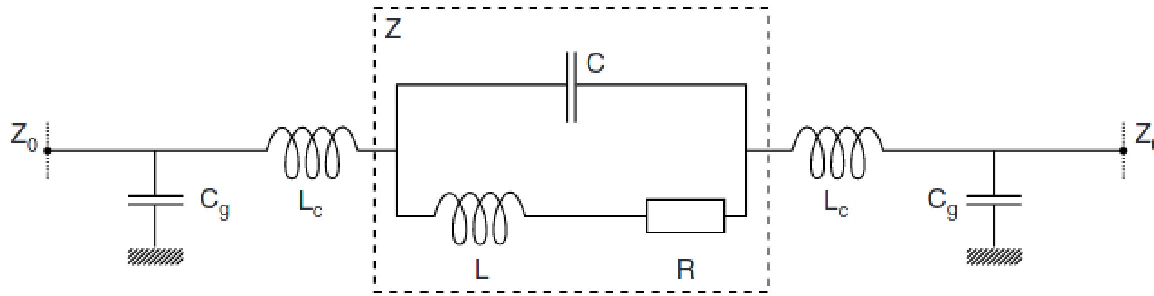
ISO 10303-21 STEP 3D CAD file

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION((
'CAx-IF Rec.Pracs.---Model Styling and Organization---1.5---2016-08-15',
'CAx-IF Rec.Pracs.---Geometric and Assembly Validation
Properties---4.4---2016-08-17',
    'CAx-IF Rec.Pracs.---User Defined Attributes---1.5---2016-08-15',
    'CAx-IF Rec.Pracs.---External References---2.1---2005-01-19'),'2;1'
  );
FILE_NAME('CH02016_P.stp','2017-11-14T12:13:16',('Unspecified'),(
    'Unspecified'),'CAD Exchanger 3.3.1 (www.cadexchanger.com)',
  'CAD Exchanger 3.3.1','');
FILE_SCHEMA(('AUTOMOTIVE_DESIGN { 1 0 10303 214 1 1 1 1 }'));
ENDSEC;
DATA;
#1 = APPLICATION_PROTOCOL_DEFINITION('international standard',
  'automotive_design',2000,#2);
#2 = APPLICATION_CONTEXT(
  'core data for automotive mechanical design processes');
#3 = SHAPE_DEFINITION_REPRESENTATION(#4,#10);
#4 = PRODUCT_DEFINITION_SHAPE('',$,#5);
#5 = PRODUCT_DEFINITION('design','',#6,#9);
#6 = PRODUCT_DEFINITION_FORMATION('','',#7);
#7 = PRODUCT('67','67','',(#8));
#8 = PRODUCT_CONTEXT('',#2,'mechanical');
#9 = PRODUCT_DEFINITION_CONTEXT('part definition',#2,'design');
#10 = ADVANCED_BREP_SHAPE_REPRESENTATION('',(#16,#154),#11);
#11 = ( GEOMETRIC_REPRESENTATION_CONTEXT(3)
GLOBAL_UNCERTAINTY_ASSIGNED_CONTEXT((#12)) GLOBAL_UNIT_ASSIGNED_CONTEXT(
(#13,#14,#15)) REPRESENTATION_CONTEXT('','') );
#12 = UNCERTAINTY_MEASURE_WITH_UNIT(LENGTH_MEASURE(1.E-007),#13,'',
  'maximum tolerance');
#13 = ( LENGTH_UNIT() NAMED_UNIT(*) SI_UNIT(.MILLI.,.METRE.) );
#14 = ( NAMED_UNIT(*) PLANE_ANGLE_UNIT() SI_UNIT($,.RADIAN.) );
#15 = ( NAMED_UNIT(*) SI_UNIT($,.STERADIAN.) SOLID_ANGLE_UNIT() );
#16 = MANIFOLD_SOLID_BREP('67',#17);
#17 = CLOSED_SHELL('',(#18,#54,#90,#110,#126,#142));
#18 = ADVANCED_FACE('',(#24),#19,.T.);
#19 = B_SPLINE_SURFACE_WITH_KNOTS('',1,1,(
    (#20,#21)
    ,(#22,#23
    )),.UNSPECIFIED.,.F.,.F.,.U.,(2,2),(2,2),(-0.251,0.251),(-0.201,
    0.201),.PIECEWISE_BEZIER_KNOTS.);
#20 = CARTESIAN_POINT('',(-1.E-003,-1.E-003,0.42));
#21 = CARTESIAN_POINT('',(-1.E-003,0.401,0.42));
#22 = CARTESIAN_POINT('',(0.501,-1.E-003,0.42));
#23 = CARTESIAN_POINT('',(0.501,0.401,0.42));
#24 = FACE_BOUND('',#25,.T.);
#25 = EDGE_LOOP('',(#26,#35,#42,#49));
#26 = ORIENTED_EDGE('',*,*,#27,.T.);
#27 = EDGE_CURVE('',#31,#33,#28,.T.);
#28 = B_SPLINE_CURVE_WITH_KNOTS('',1,(#29,#30),.UNSPECIFIED.,.F.,.U.,(2,
    2),(-0.2,0.2),.PIECEWISE_BEZIER_KNOTS.);
#29 = CARTESIAN_POINT('',(0.5,0.E+000,0.42));
#30 = CARTESIAN_POINT('',(0.5,0.4,0.42));
#31 = VERTEX_POINT('',#32);
#32 = CARTESIAN_POINT('',(0.5,0.E+000,0.42));
#33 = VERTEX_POINT('',#34);
#34 = CARTESIAN_POINT('',(0.5,0.4,0.42));
#35 = ORIENTED_EDGE('',*,*,#36,.T.);
#36 = EDGE_CURVE('',#33,#40,#37,.T.);
#37 = B_SPLINE_CURVE_WITH_KNOTS('',1,(#38,#39),.UNSPECIFIED.,.F.,.U.,(2,
    2),(-0.25,0.25),.PIECEWISE_BEZIER_KNOTS.);
#38 = CARTESIAN_POINT('',(0.5,0.4,0.42));
#39 = CARTESIAN_POINT('',(0.E+000,0.4,0.42));
#40 = VERTEX_POINT('',#41);
#41 = CARTESIAN_POINT('',(0.E+000,0.4,0.42));
#42 = ORIENTED_EDGE('',*,*,#43,.T.);
#43 = EDGE_CURVE('',#40,#47,#44,.T.);
#44 = B_SPLINE_CURVE_WITH_KNOTS('',1,(#45,#46),.UNSPECIFIED.,.F.,.U.,(2,
    2),(-0.2,0.2),.PIECEWISE_BEZIER_KNOTS.);
#45 = CARTESIAN_POINT('',(0.E+000,0.4,0.42));
#46 = CARTESIAN_POINT('',(0.E+000,0.E+000,0.42));
#47 = VERTEX_POINT('',#48);
...
```



AUTODESK® VIEWER > CH02016_P.stp

TYPICAL HIGH FREQUENCY PERFORMANCE ELECTRICAL MODEL



C	Internal shunt capacitance
L	Internal inductance
R	Resistance
Z	Internal impedance (R, L, C)
L_c	External connection inductance
C_g	External capacitance to ground

The complex impedance of the chip resistor is given by the following equations:

$$Z = \frac{R + j\omega(L - R^2C - L^2C\omega^2)}{1 + C[(R^2C - 2L)\omega^2 + L^2C\omega^4]}$$

$$\frac{[Z]}{R} = \frac{1}{1 + C[(R^2C - 2L)\omega^2 + L^2C\omega^4]} \times \sqrt{1 + \left[\frac{\omega(L - R^2C - L^2C\omega^2)}{R}\right]^2}$$

$$\theta = \tan^{-1}\frac{\omega(L - R^2C - L^2C\omega^2)}{R}$$

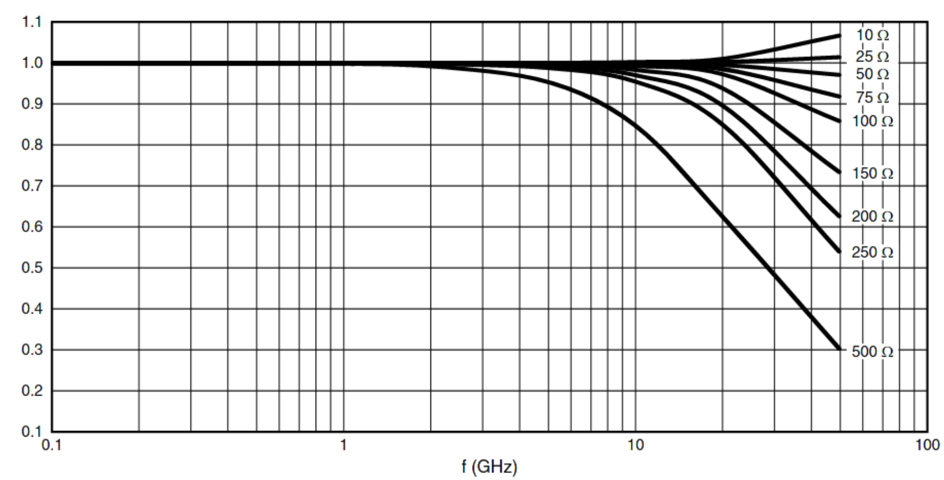
Notes

- $\omega = 2 \times \pi \times f$
- f: frequency

R, L and C are relevant to the chip resistor itself.
L_c and C_g also depend on the way the chip resistor is mounted.
It is important to notice that after assembly the external reactance of L_c and C_g will be combined to internal reactance of L and C. This combination can upgrade or downgrade the HF behavior of the component.
This is why we are displaying three sets of data:

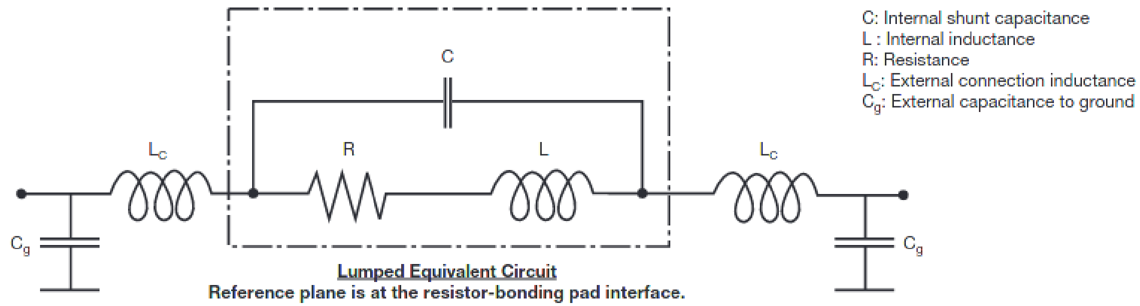
- $\frac{[Z]}{R}$ versus frequency curves which aim to show at a glance the intrinsic HF performance of a given chip resistor
- $\frac{[Z_{total}]}{R}$ versus frequency curves which aim to show the behavior of the chip resistor when mounted

INTERNAL IMPEDANCE CURVES



Internal impedance curve for 02016 size (F and P terminations)

TYPICAL HIGH FREQUENCY PERFORMANCE ELECTRICAL MODEL AND TESTING

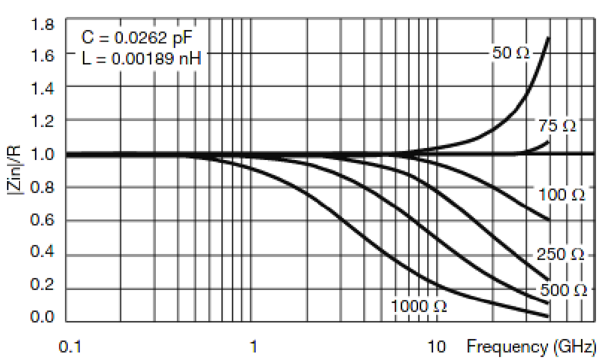


Lumped Equivalent Circuit
Reference plane is at the resistor-bonding pad interface.

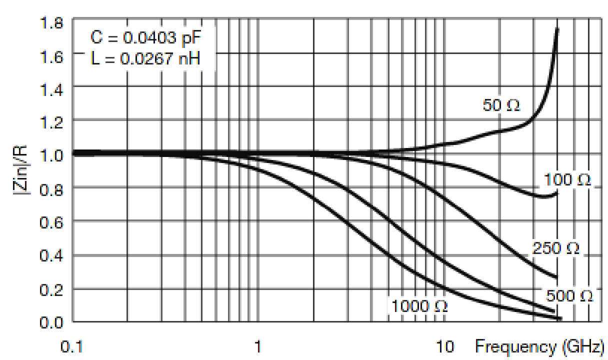
The lumped circuit above was used to model the data at the bonding pad-resistor reference plane. High frequency testing was performed by Modelithics, Inc. on parts mounted to quartz test boards. Quartz test boards were chosen to minimize the contribution of the board effects at high frequencies. Future testing will be performed on various industry standard board types. Vishay in partnership with Modelithics, Inc. will develop substrate scalable models for the FC series resistors. These models will be available for industry standard design software packages and will allow the designer to accurately model their wireless and microwave printed boards.

INTERNAL IMPEDANCE

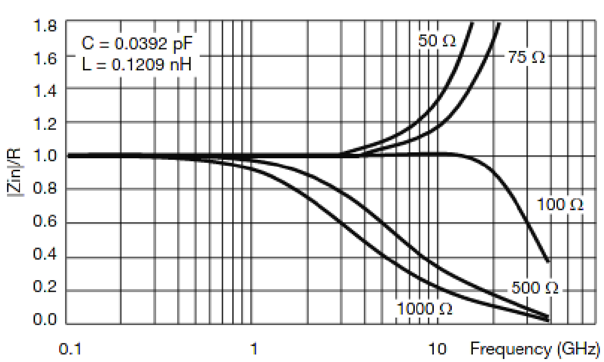
0402 Flip chip



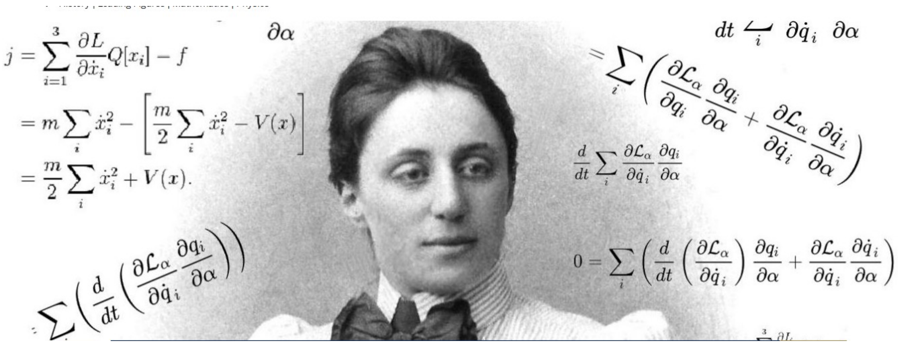
0603 Flip chip



0402 Wraparound



models based on Laws of Physics



Invariante Variationsprobleme.

(F. Klein zum fünfzigjährigen Doktorjubiläum.)

Von

Emmy Noether in Göttingen.

Vorgelegt von F. Klein in der Sitzung vom 26. Juli 1918 [1]).

Es handelt sich um Variationsprobleme, die eine kontinuierliche Gruppe (im Lieschen Sinne) gestatten; die daraus sich ergebenden Folgerungen für die zugehörigen Differentialgleichungen finden ihren allgemeinsten Ausdruck in den in § 1 formulierten, in den folgenden Paragraphen bewiesenen Sätzen. Über diese aus Variationsproblemen entspringenden Differentialgleichungen lassen sich viel präzisere Aussagen machen als über beliebige, eine Gruppe gestattende Differentialgleichungen, die den Gegenstand der Lieschen Untersuchungen bilden. Das folgende beruht also auf einer Verbindung der Methoden der formalen Variationsrechnung mit denen der Lieschen Gruppentheorie. Für spezielle Gruppen und Variationsprobleme ist diese Verbindung der Methoden nicht neu; ich erwähne Hamel und Herglotz für spezielle endliche, Lorentz und seine Schüler (z. B. Fokker), Weyl und Klein für spezielle unendliche Gruppen [2]). Insbesondere sind die zweite Kleinsche Note und die vorliegenden Ausführungen gegenseitig durch einander beein-

1) Die endgiltige Fassung des Manuskriptes wurde erst Ende September eingereicht.

2) Hamel: Math. Ann. Bd. 59 und Zeitschrift f. Math. u. Phys. Bd. 50. Herglotz: Ann. d. Phys. (4) Bd. 36, bes. § 9, S. 511. Fokker, Verslag d. Amsterdamer Akad., 27./1. 1917. Für die weitere Litteratur vergl. die zweite Note von Klein: Göttinger Nachrichten 19. Juli 1918.

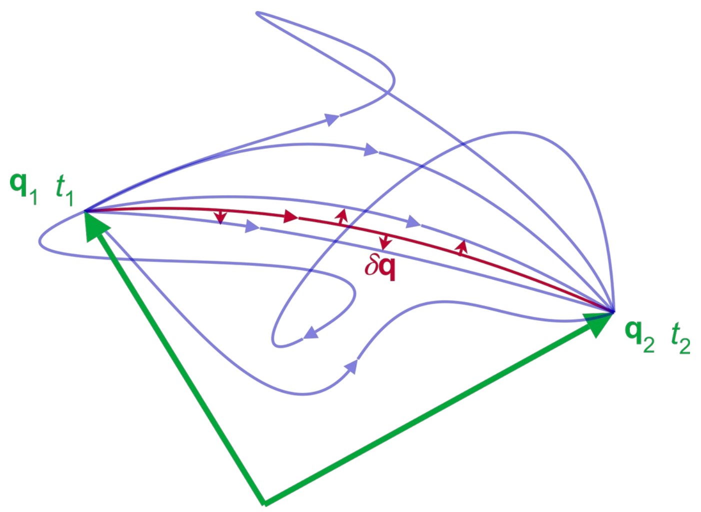
In einer eben erschienenen Arbeit von Kneser (Math. Zeitschrift Bd. 2) handelt es sich um Aufstellung von Invarianten nach ähnlicher Methode.

Noether's theorem or **Noether's first theorem** states that every differentiable symmetry of the action of a physical system has a corresponding conservation law.[1] The theorem was proven by mathematician Emmy Noether in 1915 and published in 1918,[2] after a special case was proven by E. Cosserat and F. Cosserat in 1909.[3] The action of a physical system is the integral over time of a Lagrangian function (which may be an integral over space of a Lagrangian density function), from which the system's behavior can be determined by the principle of least action. This theorem only applies to continuous and smooth symmetries over physical space.

Application of Noether's theorem allows physicists to gain powerful insights into any general theory in physics, by just analyzing the various transformations that would make the form of the laws involved invariant. For example:

- the invariance of physical systems with respect to spatial translation (in other words, that the laws of physics do not vary with locations in space) gives the law of conservation of linear momentum;
- invariance with respect to rotation gives the law of conservation of angular momentum;
- invariance with respect to time translation gives the well-known law of conservation of energy

In quantum field theory, the analog to Noether's theorem, the Ward–Takahashi identity, yields further conservation laws, such as the conservation of electric charge from the invariance with respect to a change in the phase factor of the complex field of the charged particle and the associated gauge of the electric potential and vector potential.



In non-relativistic physics, the **principle of least action** – or, more accurately, the **principle of stationary action** – is a variational principle that, when applied to the action of a mechanical system, can be used to obtain the equations of motion for that system by stating a system follows the path where the average difference between the kinetic energy and potential energy is minimized or maximized over any time period. It is called stable if minimized. In relativity, a different average must be minimized or maximized. The principle can be used to derive Newtonian, Lagrangian, and Hamiltonian equations of motion.

The starting point is the *action*, denoted $\mathcal{S}$ (calligraphic S), of a physical system. It is defined as the integral of the Lagrangian L between two instants of time t_1 and t_2 - technically a functional of the N generalized coordinates $\mathbf{q} = (q_1, q_2 \ldots q_N)$ which define the configuration of the system:

$$\mathcal{S}[\mathbf{q}(t)] = \int_{t_1}^{t_2} L(\mathbf{q}(t), \dot{\mathbf{q}}(t), t) dt$$

where the dot denotes the time derivative, and t is time.

Mathematically the principle is[11][12][13]

$$\delta \mathcal{S} = 0$$

where δ (Greek lowercase delta) means a *small* change. In words this reads:[10]

> *The path taken by the system between times t_1 and t_2 is the one for which the **action** is **stationary (no change)** to **first order**.*

* **Conservation of Mass-Energy:**

The total energy in a closed or isolated system is constant, no matter what happens.

* **Conservation of Momentum:**

The total momentum in a closed or isolated system remains constant. An alternative of this is the law of conservation of angular momentum.

* **Newton's Law of Gravity:**

Explains the attractive force between a pair of masses. In the twentieth century, it became clear that this is not the whole story, as **Einstein's theory of general relativity** has provided a more comprehensive explanation for the phenomenon of gravity.

* **Newton's Three Laws of Motion:**

Fundamental relationship between the acceleration of an object and the total forces acting upon it.

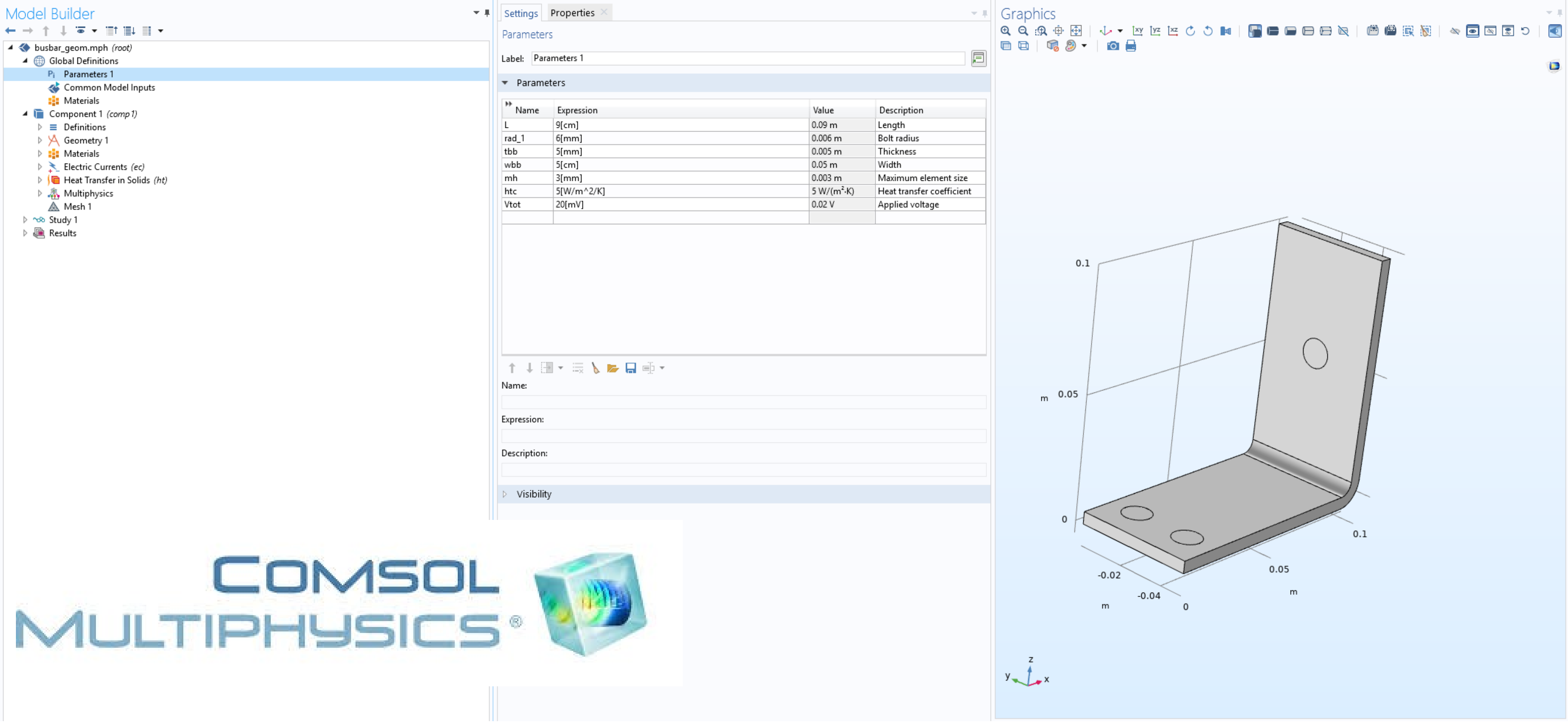
- First Law states that in order for the motion of an object to change, a force must act upon it, a concept generally called inertia.
- Second Law defines the relationship between acceleration, force, and mass. $F = m\,a$?
- Third Law states that any time a force acts from one object to another, there is an equal force acting back on the original object.

* …

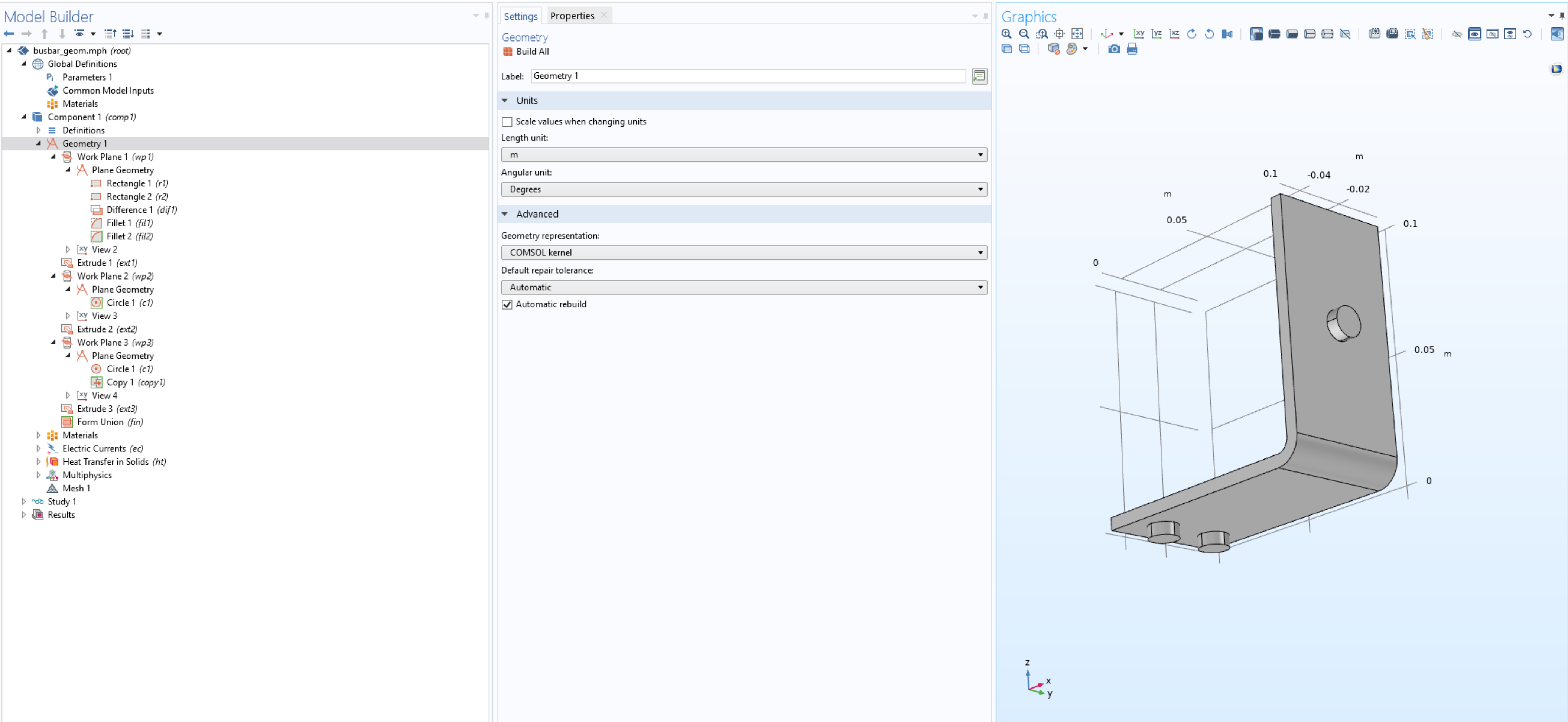
GPS: relativistic

Sodium lamp: quantum

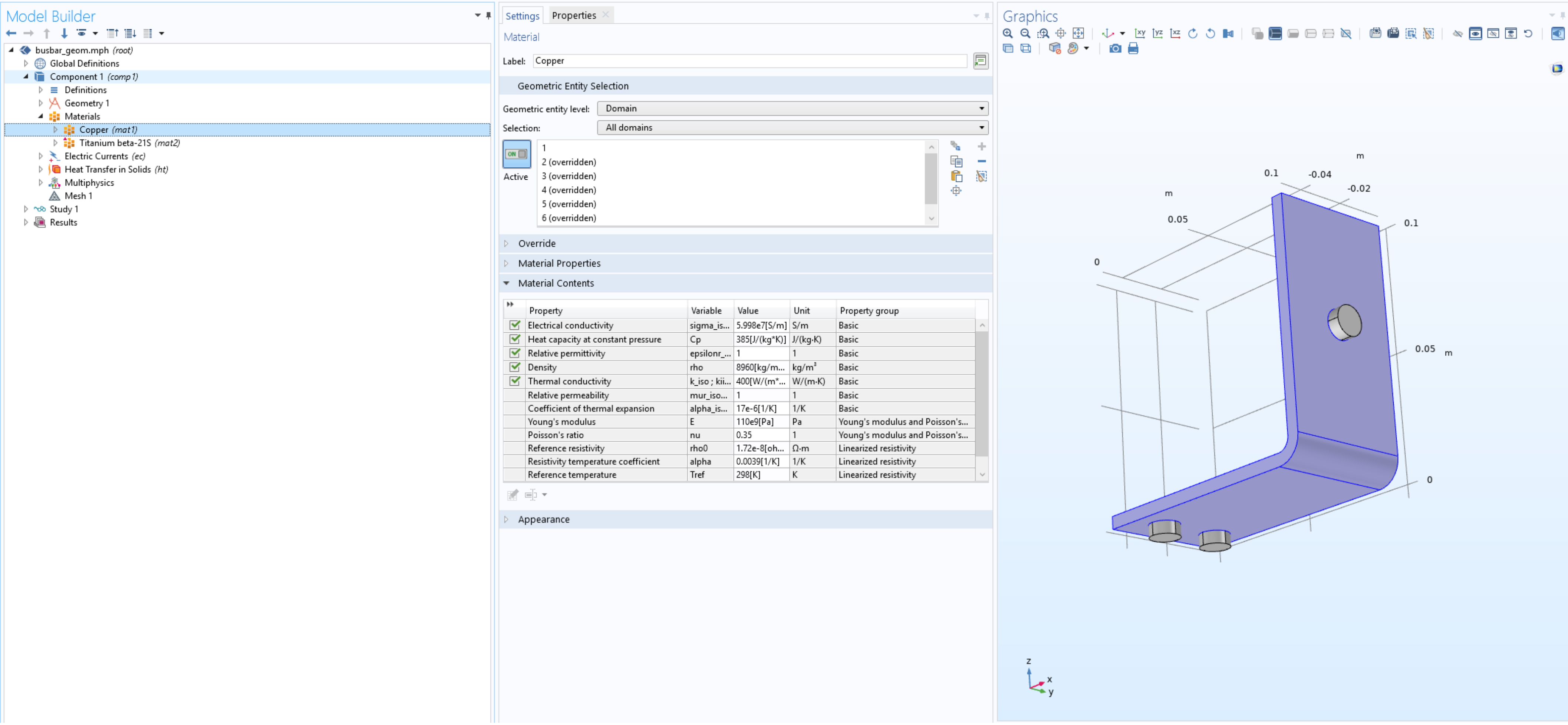
“distributed parameter” models (based on Laws of Physics)



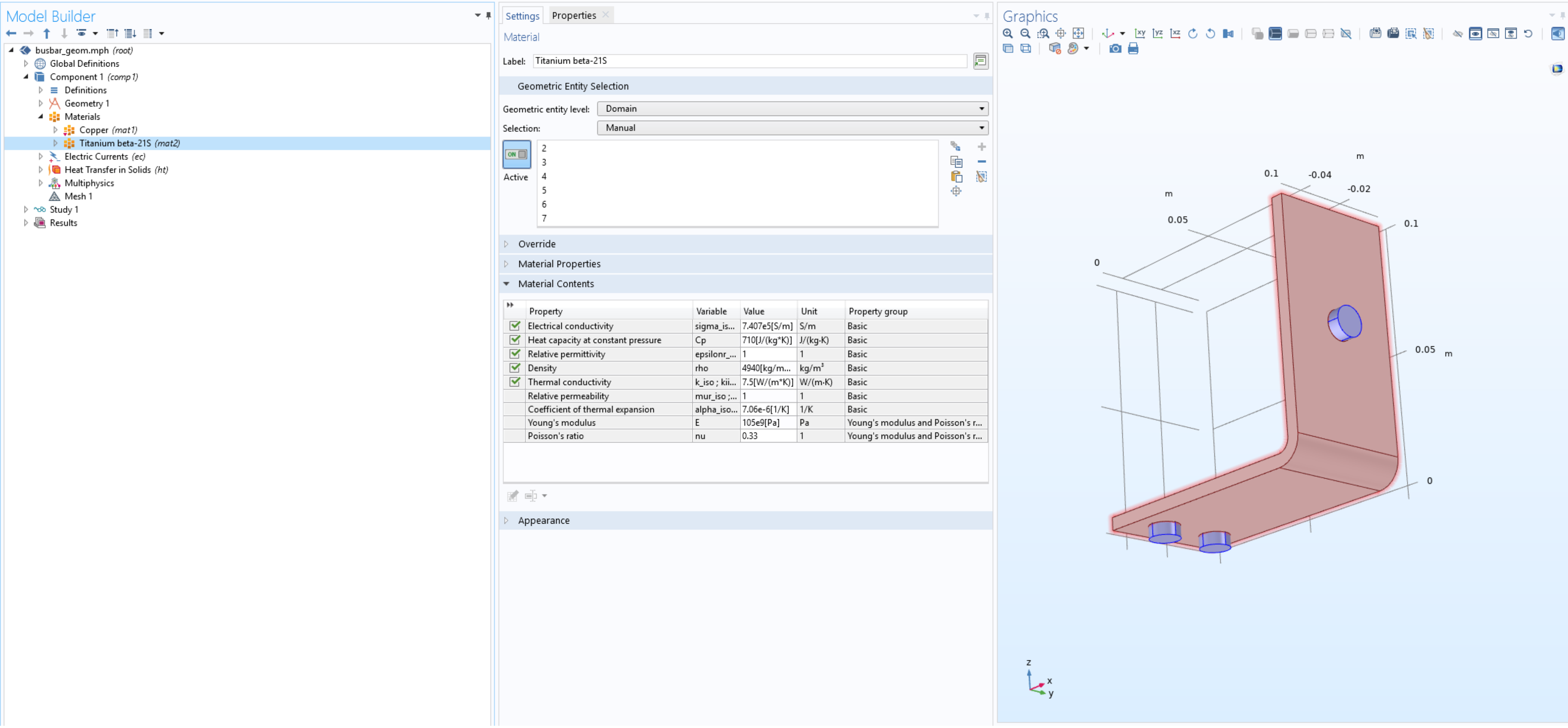
parametrized model


Model Builder
busbar_geom.mph (root)
Global Definitions
Parameters 1
Common Model Inputs
Materials
Component 1 (comp1)
Definitions
Geometry 1
Work Plane 1 (wp1)
Plane Geometry
Rectangle 1 (r1)
Rectangle 2 (r2)
Difference 1 (dif1)
Fillet 1 (fil1)
Fillet 2 (fil2)
View 2
Extrude 1 (ext1)
Work Plane 2 (wp2)
Plane Geometry
Circle 1 (c1)
View 3
Extrude 2 (ext2)
Work Plane 3 (wp3)
Plane Geometry
Circle 1 (c1)
Copy 1 (copy1)
View 4
Extrude 3 (ext3)
Form Union (fin)
Materials
Electric Currents (ec)
Heat Transfer in Solids (ht)
Multiphysics
Mesh 1
Study 1
Results
Settings
Properties
Geometry
Build All
Label: Geometry 1
Units
Scale values when changing units
Length unit:
m
Angular unit:
Degrees
Advanced
Geometry representation:
COMSOL kernel
Default repair tolerance:
Automatic
Automatic rebuild
Graphics
m
0.1
-0.04
-0.02
m
0.05
0.1
0
0.05
m
0
z
x
y

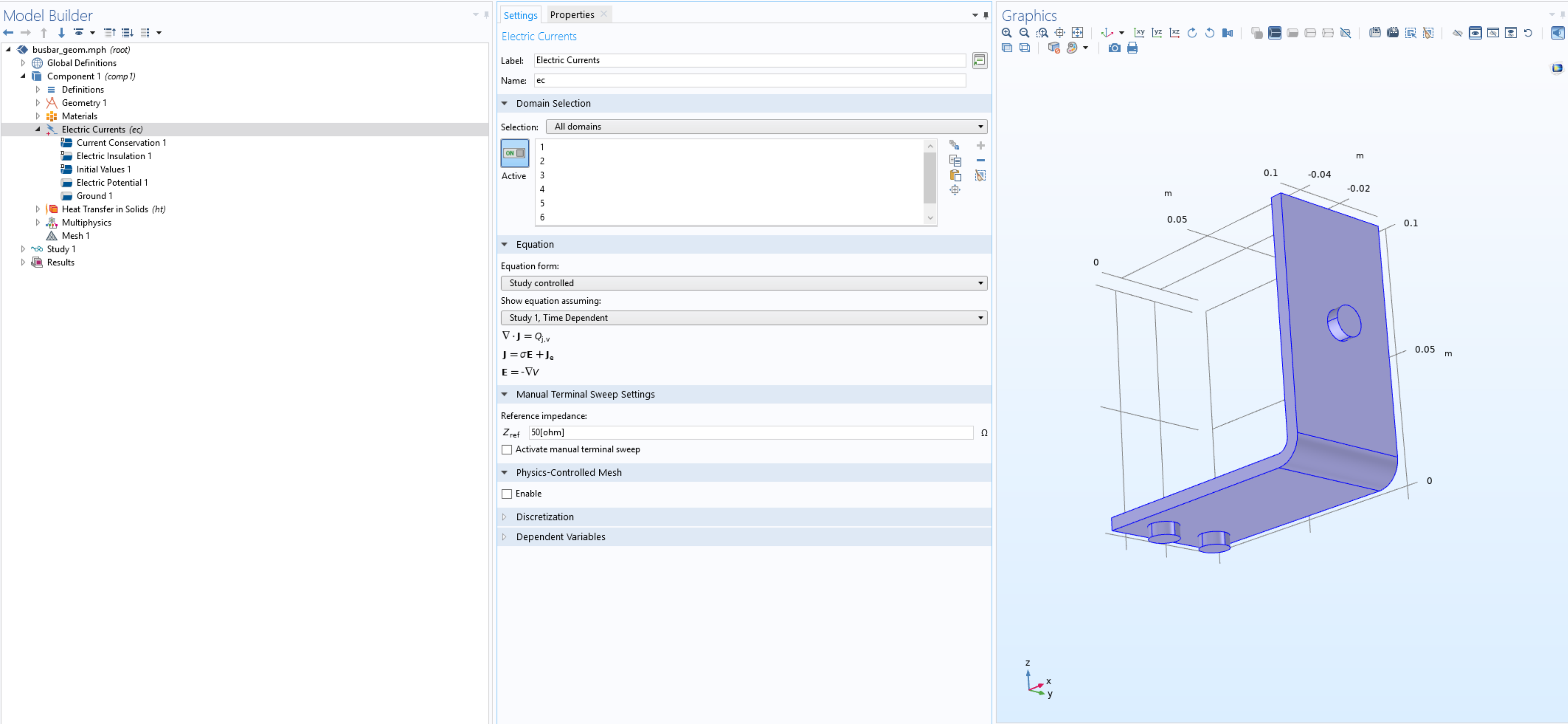
geometry



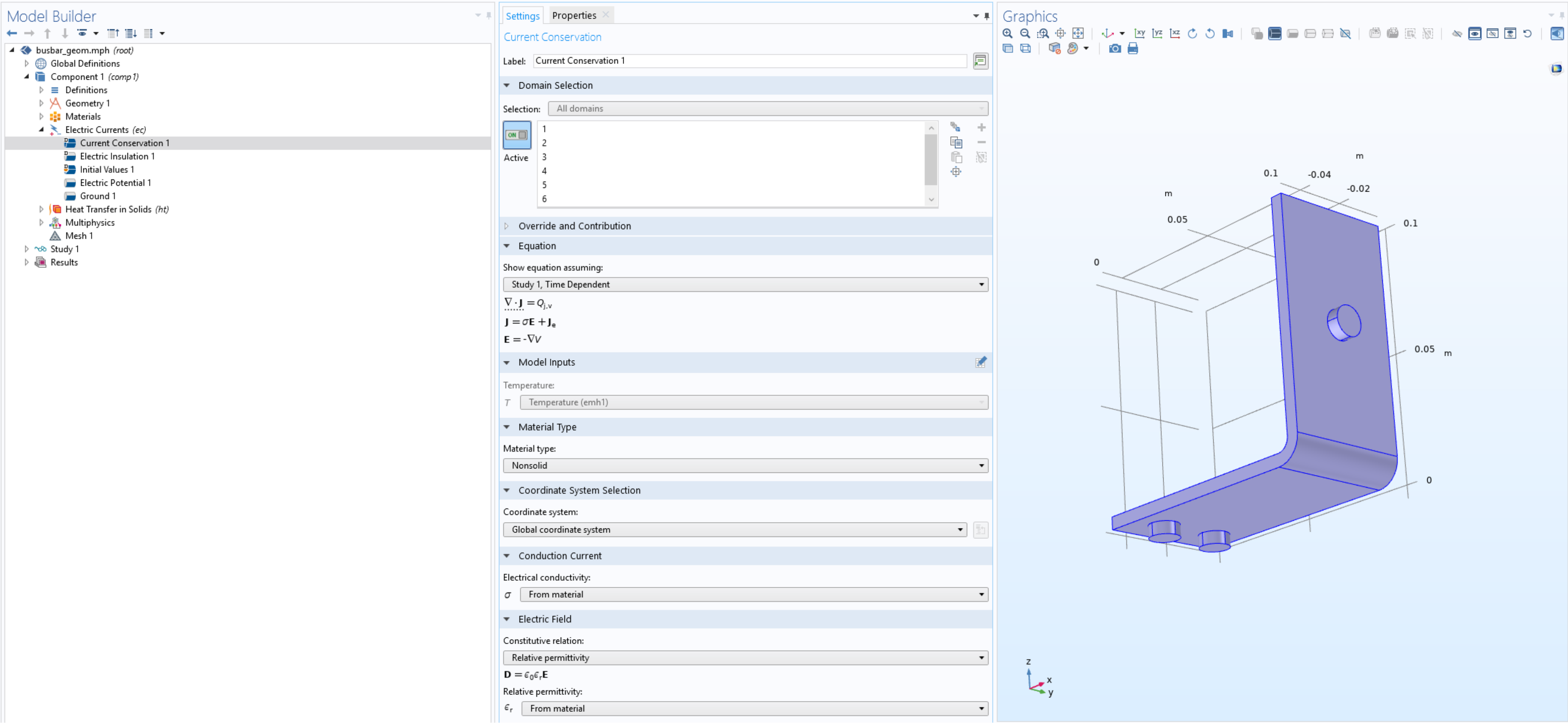
material properies



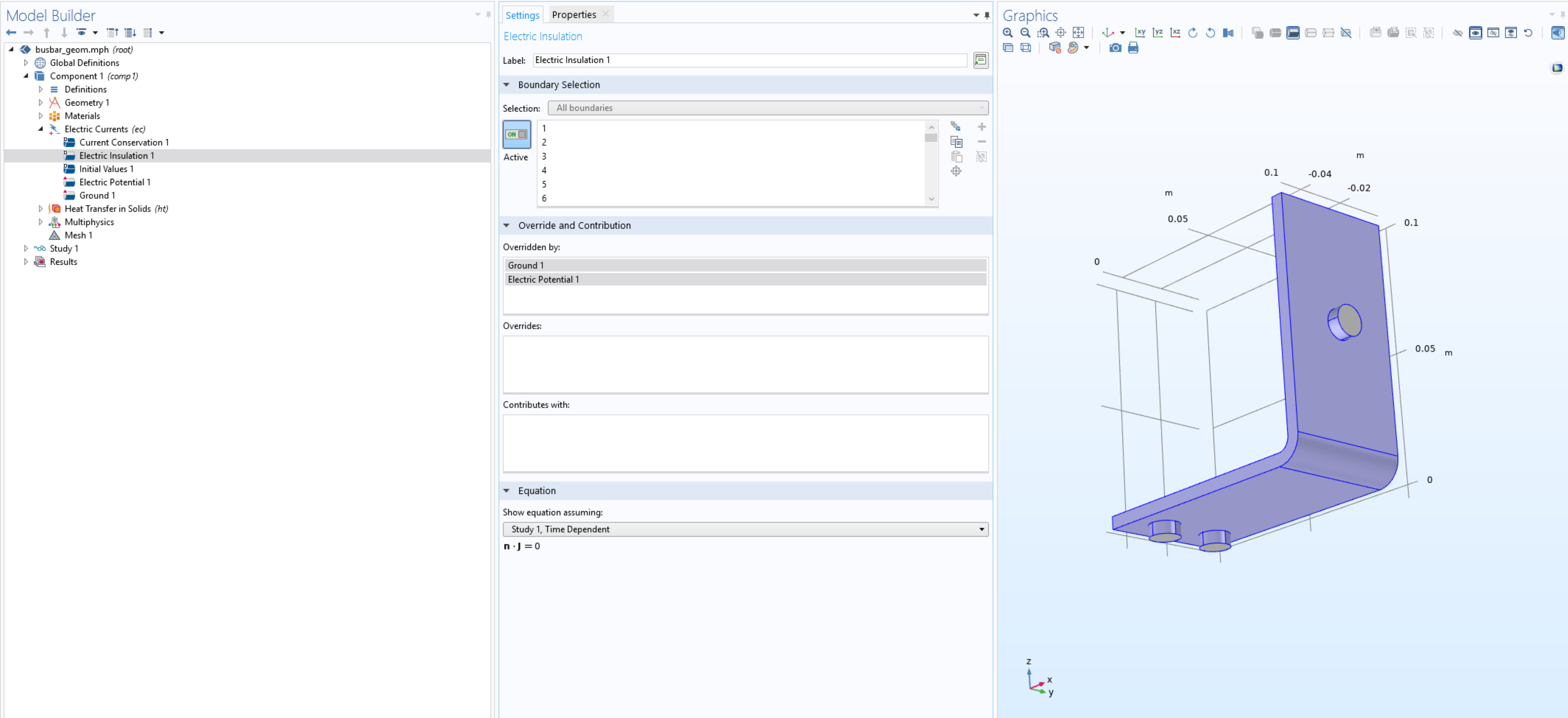
material properies



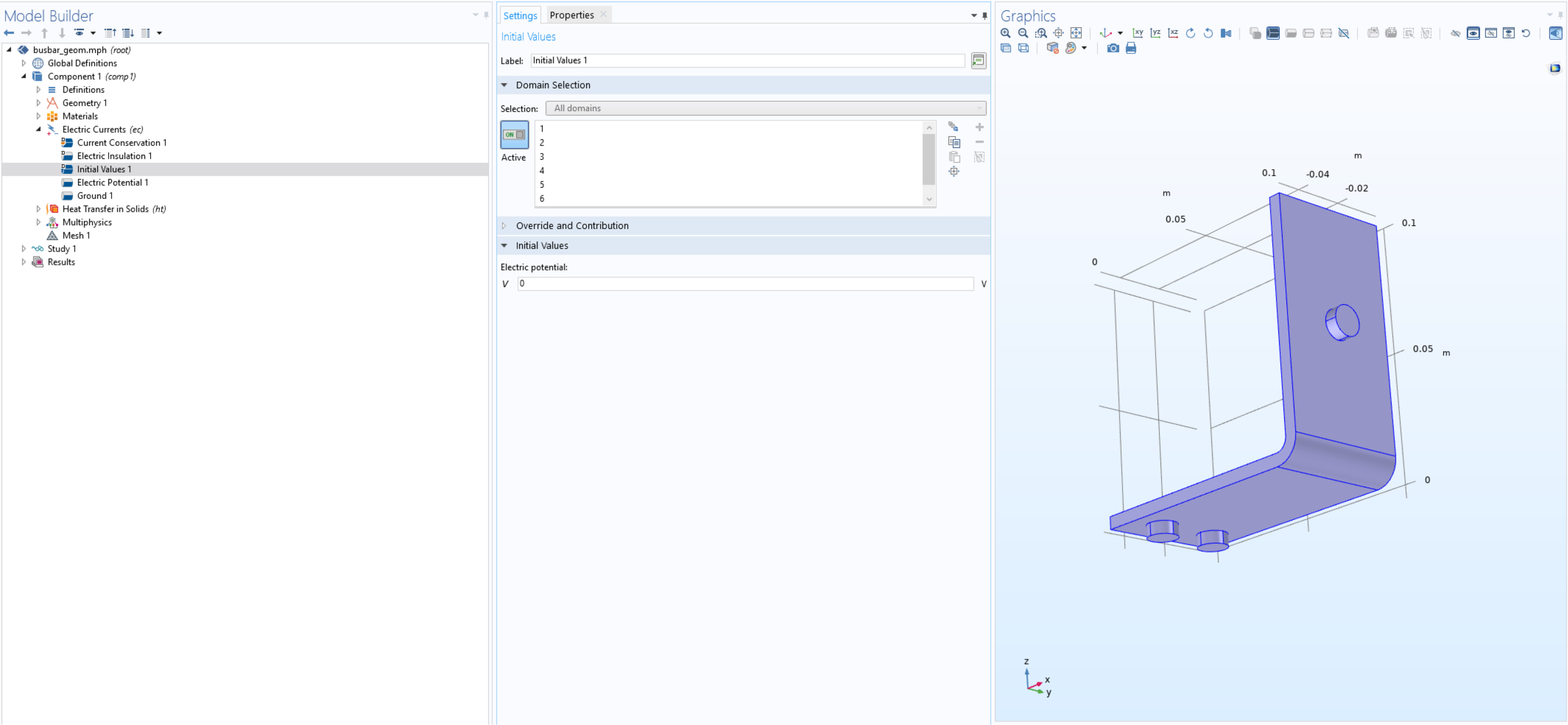
Laws of Physics



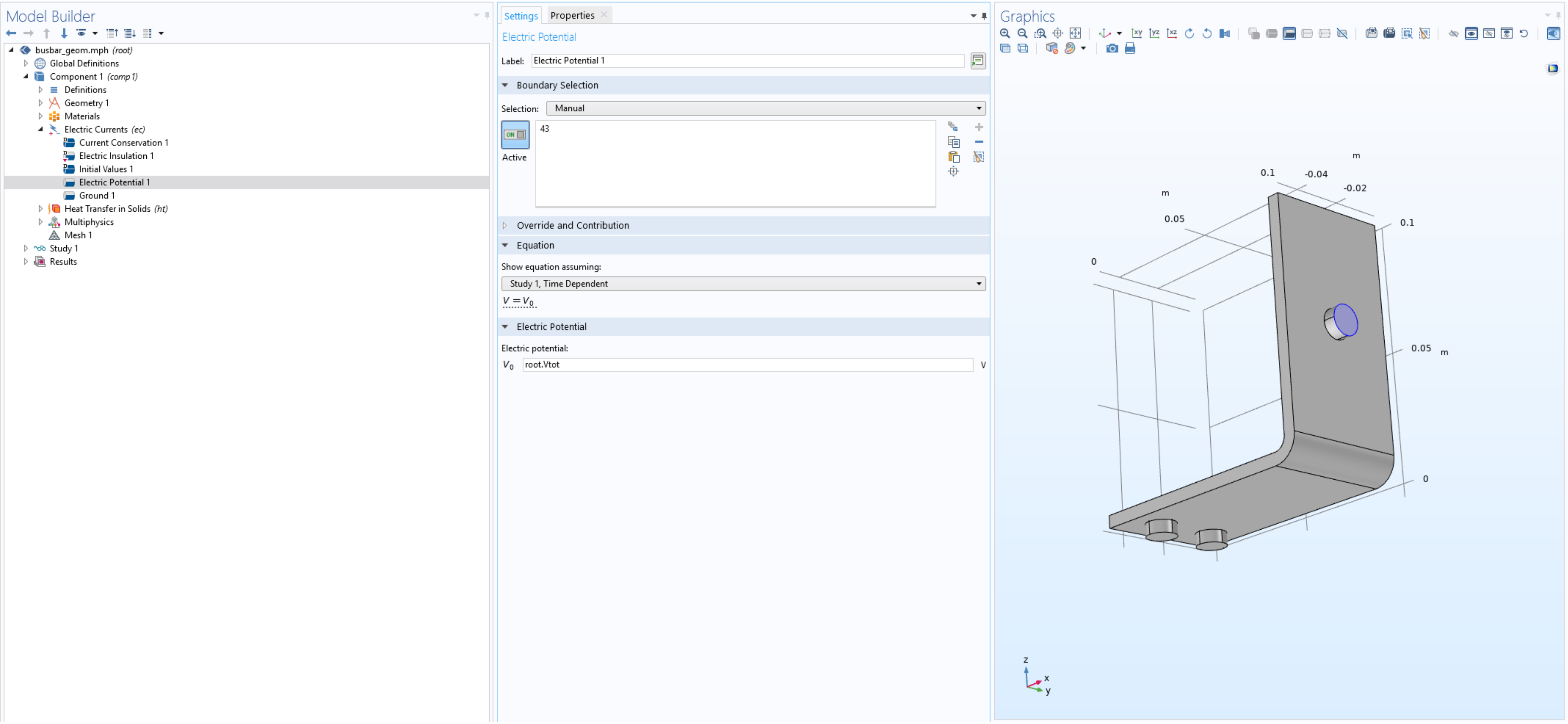
Laws of Physics



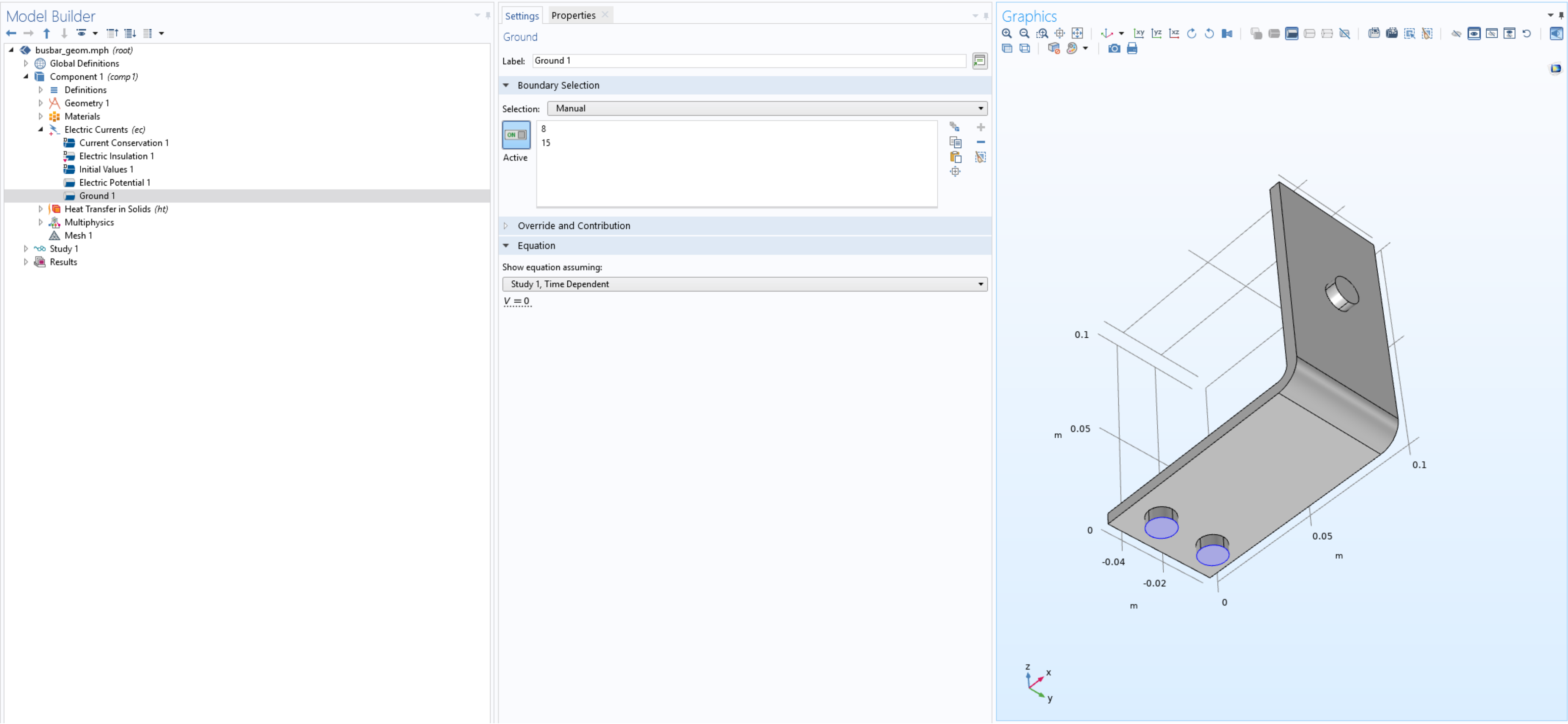
Laws of Physics



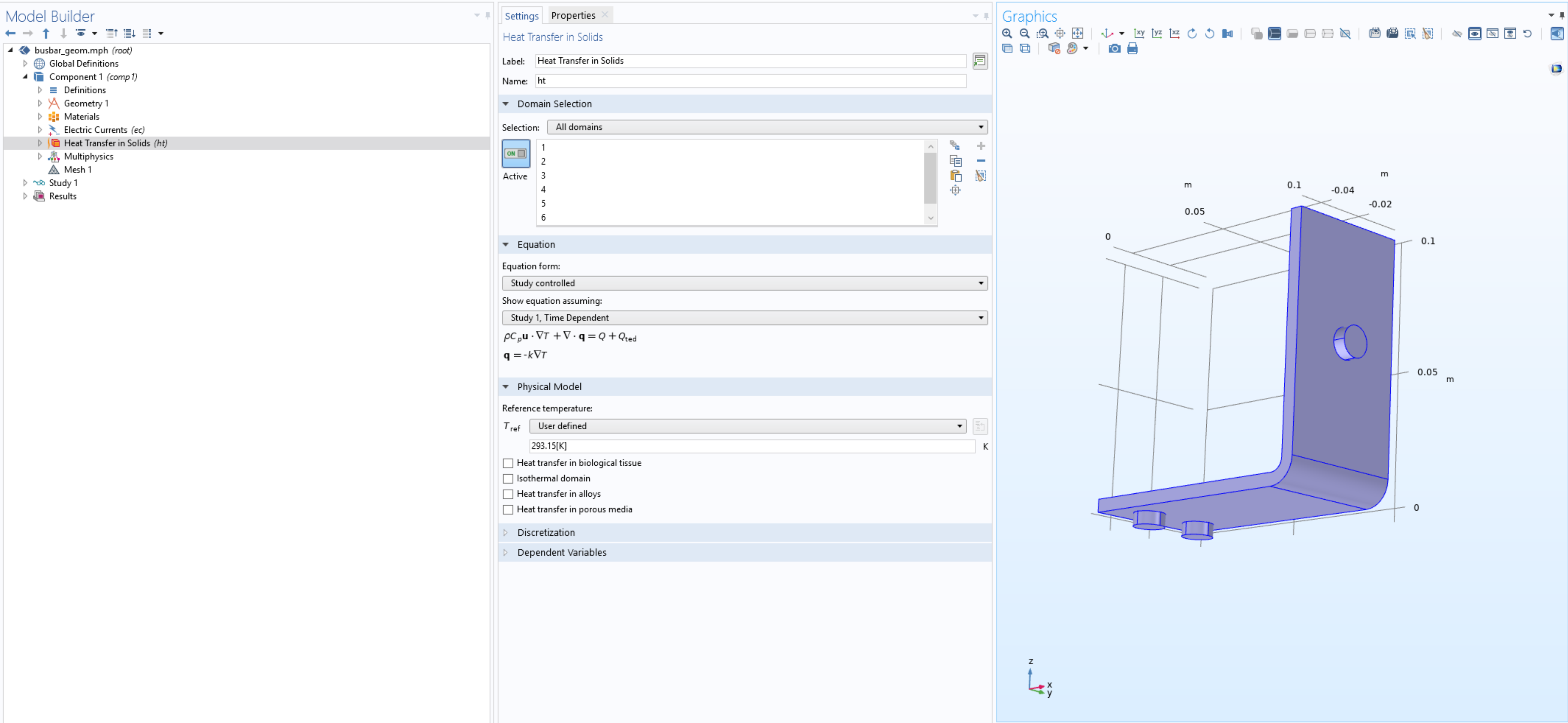
initial values



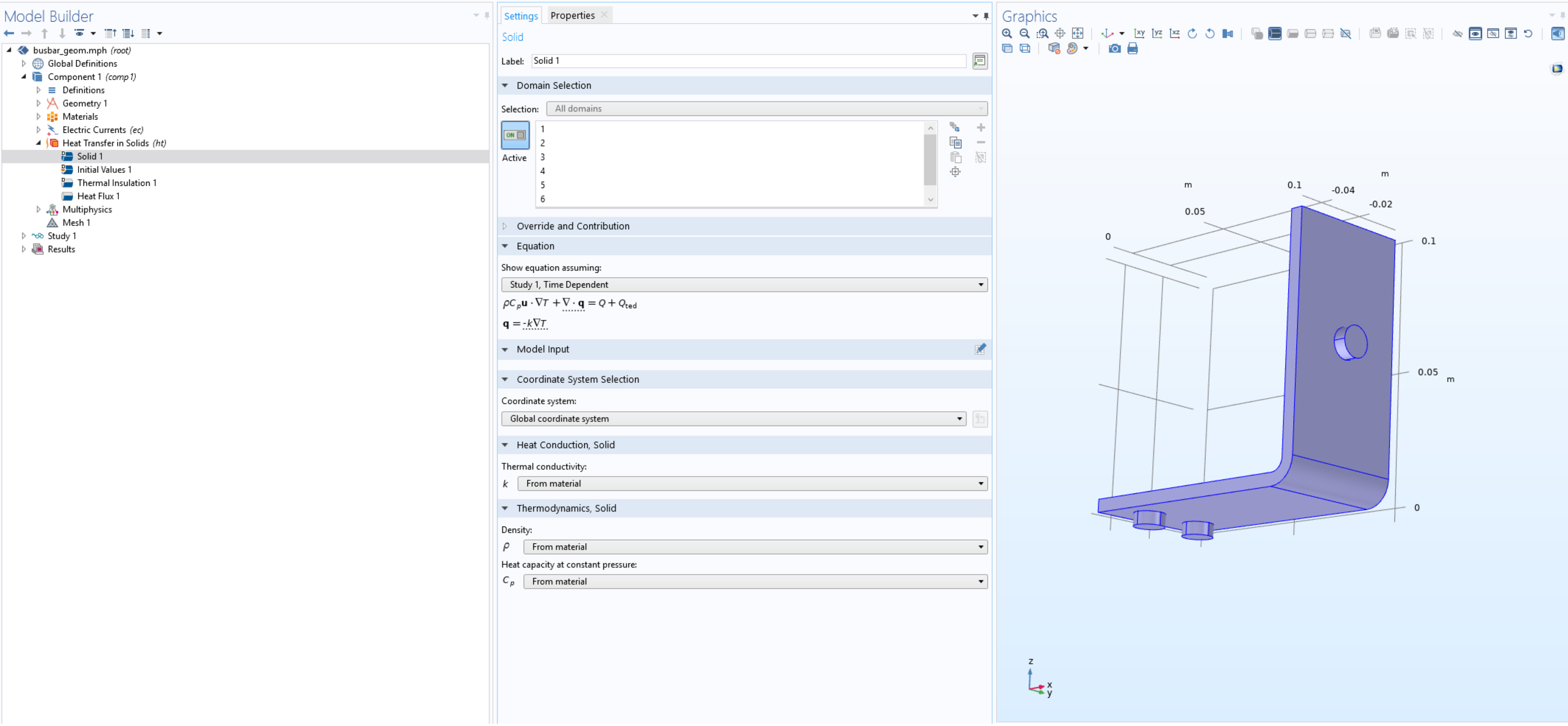
boundary conditions
(link with environment)

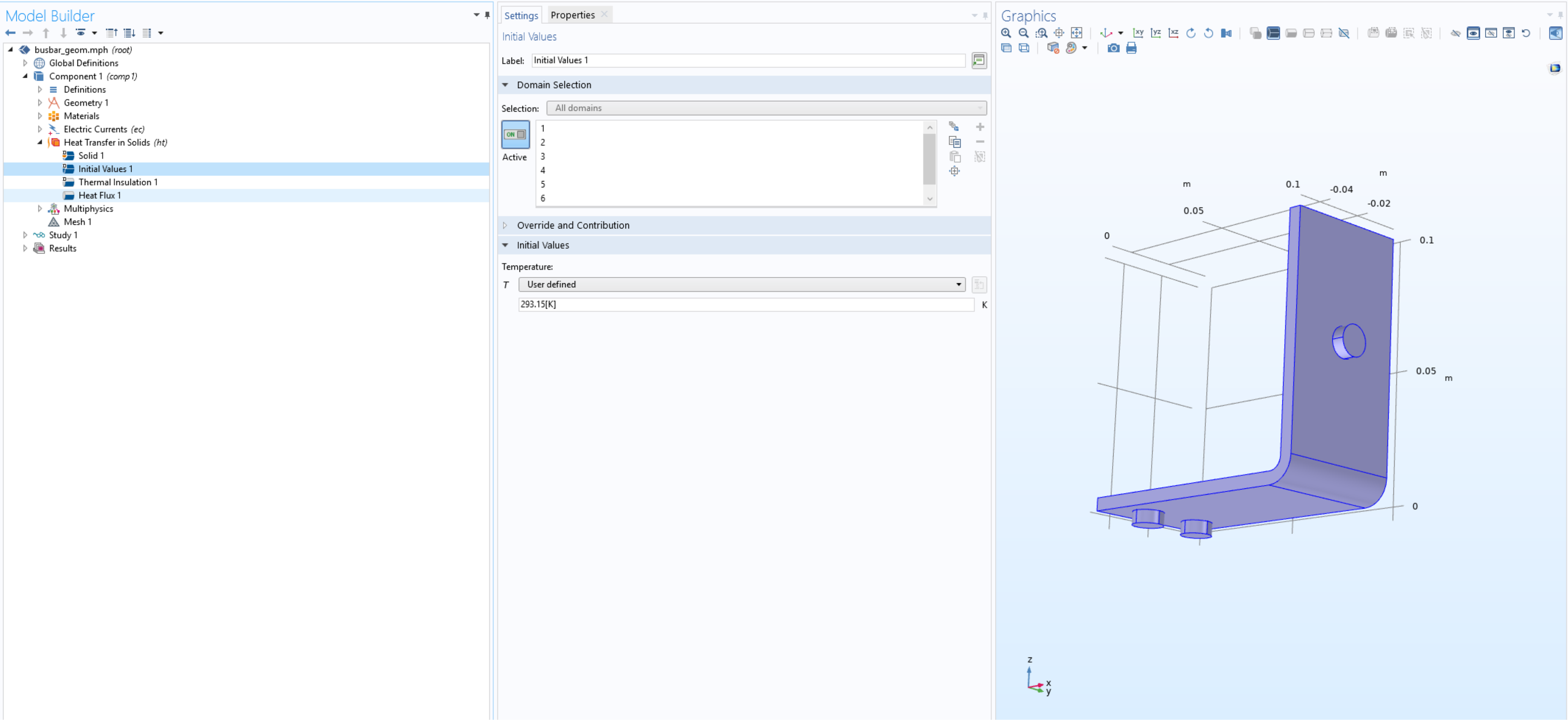


boundary conditions
(link with environment)

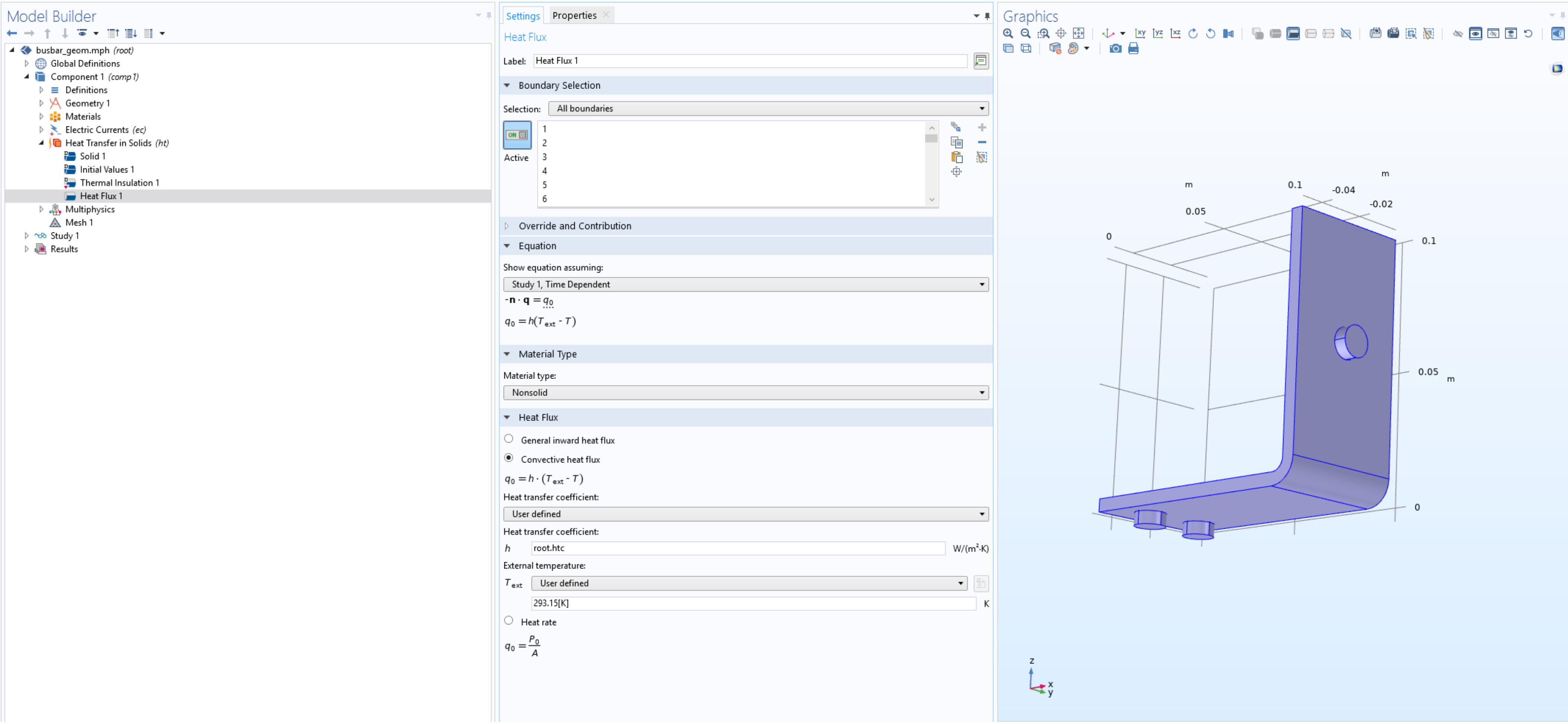


Laws of Physics

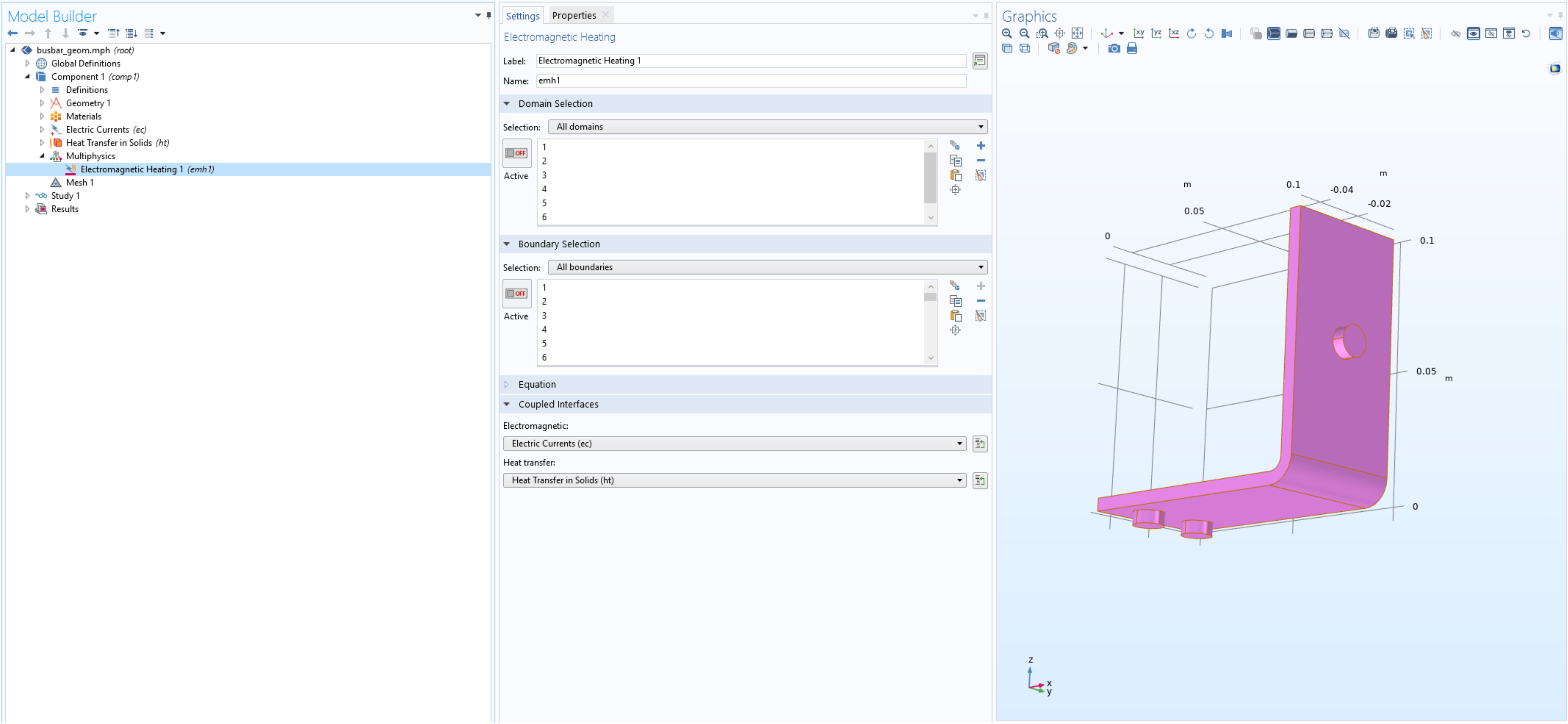

Model Builder
busbar_geom.mph (root)
Global Definitions
Component 1 (comp1)
Definitions
Geometry 1
Materials
Electric Currents (ec)
Heat Transfer in Solids (ht)
Solid 1
Initial Values 1
Thermal Insulation 1
Heat Flux 1
Multiphysics
Mesh 1
Study 1
Results
Settings
Properties
Solid
Label: Solid 1
Domain Selection
Selection: All domains
Active
1
2
3
4
5
6
Override and Contribution
Equation
Show equation assuming:
Study 1, Time Dependent
$\rho C_p \mathbf{u} \cdot \nabla T + \nabla \cdot \mathbf{q} = Q + Q_{ted}$
$\mathbf{q} = -k\nabla T$
Model Input
Coordinate System Selection
Coordinate system:
Global coordinate system
Heat Conduction, Solid
Thermal conductivity:
k From material
Thermodynamics, Solid
Density:
ρ From material
Heat capacity at constant pressure:
C_p From material
Graphics
m
0.1
-0.04
m
-0.02
0.05
0
0.1
0.05
m
0
z
x
y



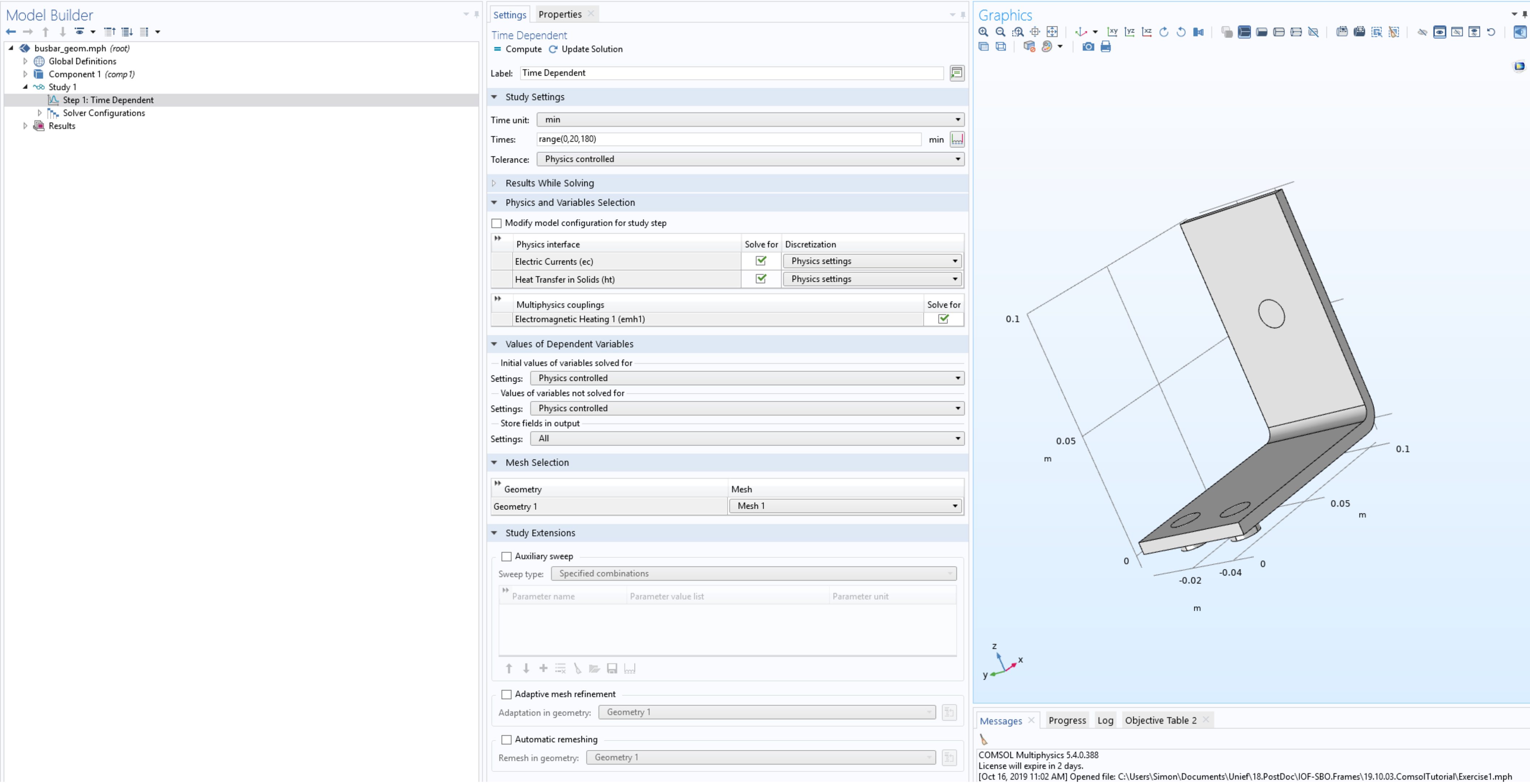
initial values



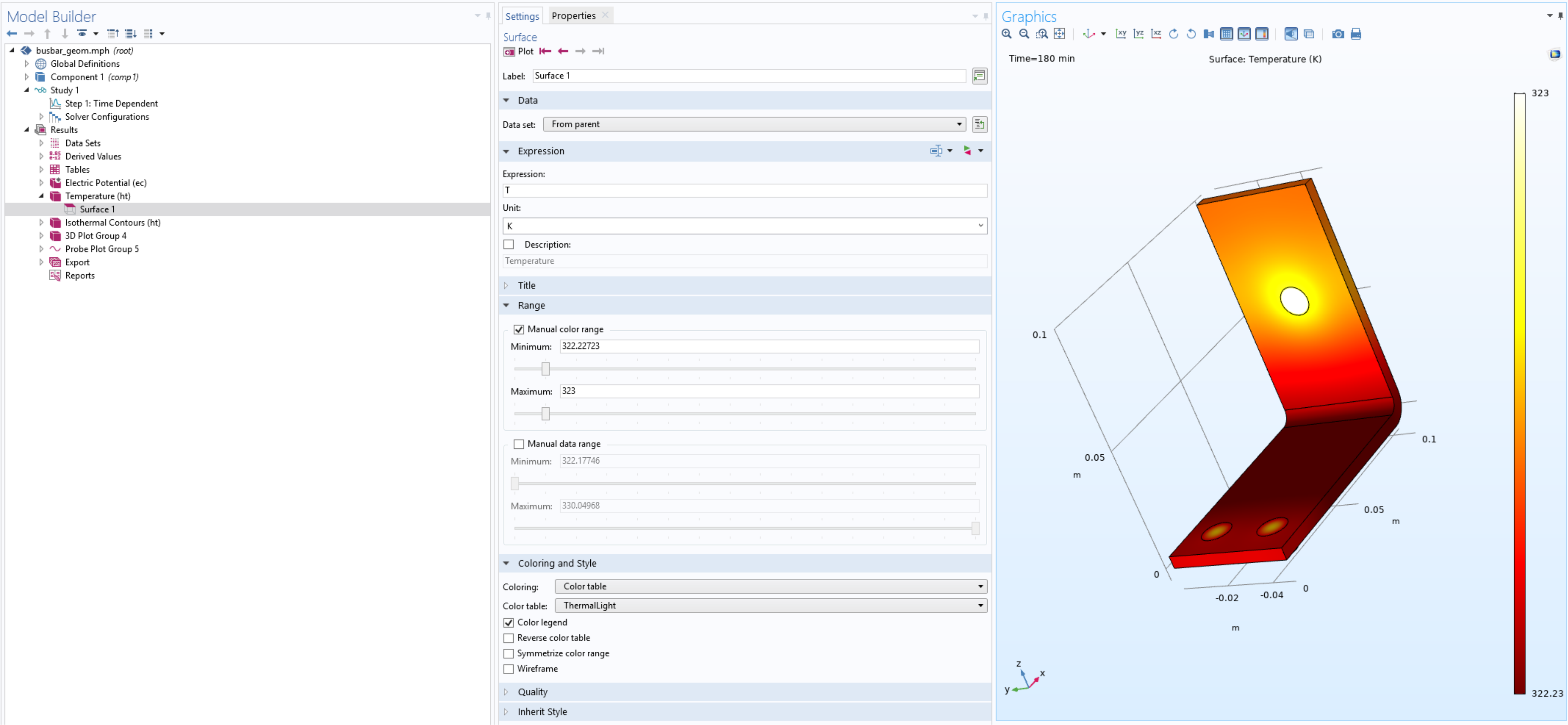
boundary conditions
(link with environment)



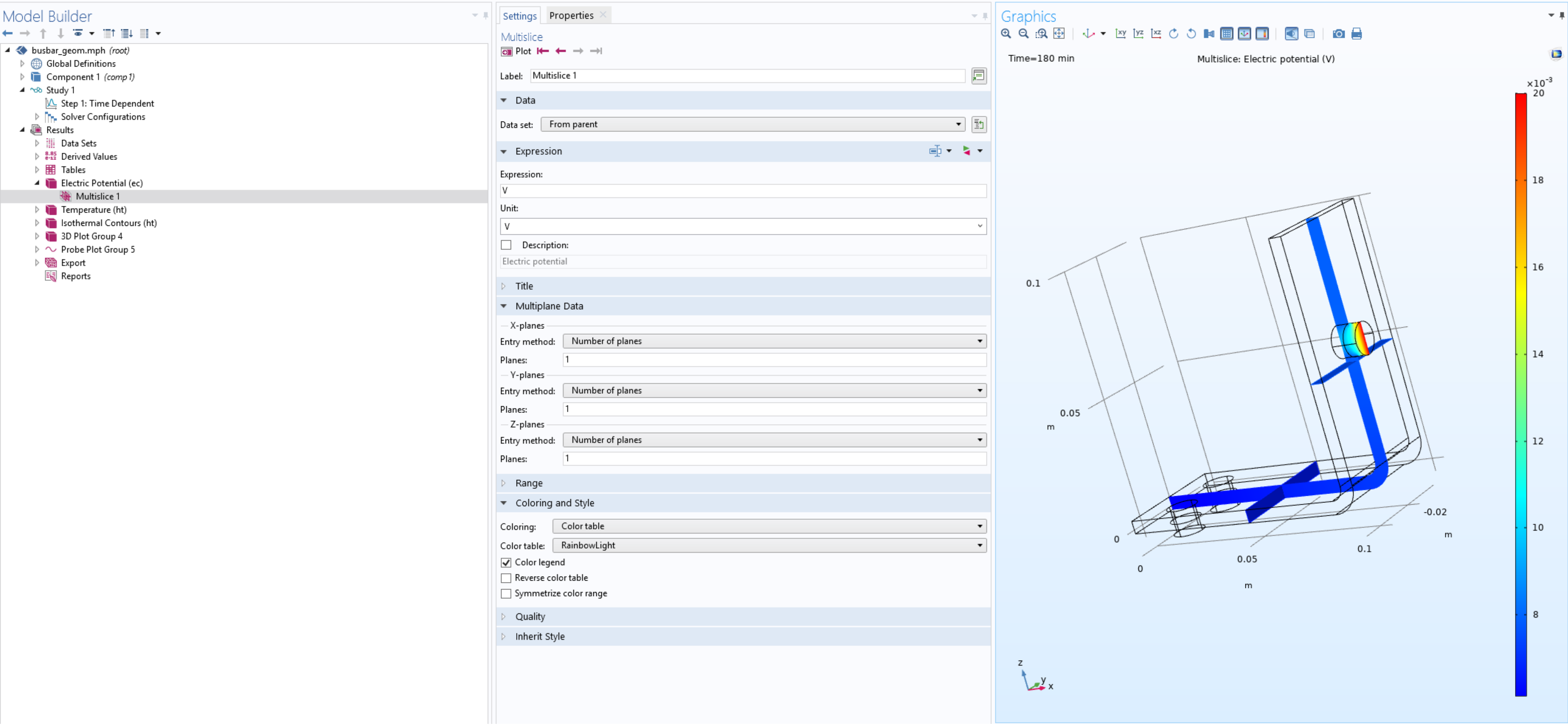
Laws of Physics


Model Builder
busbar_geom.mph (root)
Global Definitions
Component 1 (comp1)
Study 1
Step 1: Time Dependent
Solver Configurations
Results
Settings
Properties
Time Dependent
Compute
Update Solution
Label: Time Dependent
Study Settings
Time unit: min
Times: range(0,20,180) min
Tolerance: Physics controlled
Results While Solving
Physics and Variables Selection
Modify model configuration for study step
Physics interface
Solve for
Discretization
Electric Currents (ec)
Physics settings
Heat Transfer in Solids (ht)
Physics settings
Multiphysics couplings
Solve for
Electromagnetic Heating 1 (emh1)
Values of Dependent Variables
Initial values of variables solved for
Settings: Physics controlled
Values of variables not solved for
Settings: Physics controlled
Store fields in output
Settings: All
Mesh Selection
Geometry
Mesh
Geometry 1
Mesh 1
Study Extensions
Auxiliary sweep
Sweep type: Specified combinations
Parameter name
Parameter value list
Parameter unit
Adaptive mesh refinement
Adaptation in geometry: Geometry 1
Automatic remeshing
Remesh in geometry: Geometry 1
Graphics
0.1
0.05
m
0.1
0.05
m
0
-0.02
-0.04
0
m
z
y
x
Messages
Progress
Log
Objective Table 2
COMSOL Multiphysics 5.4.0.388
License will expire in 2 days.
[Oct 16, 2019 11:02 AM] Opened file: C:\Users\Simon\Documents\Unief\18.PostDoc\IOF-SBO.Frames\19.10.03.ComsolTutorial\Exercise1.mph

experiment

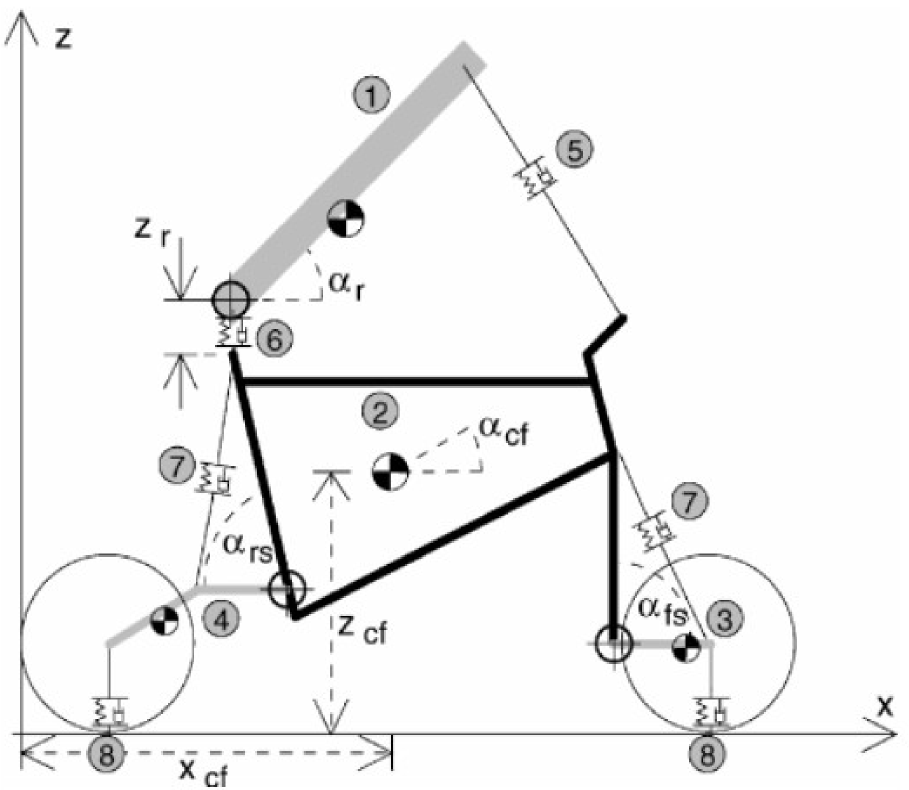


experiment result
(temperature distribution)



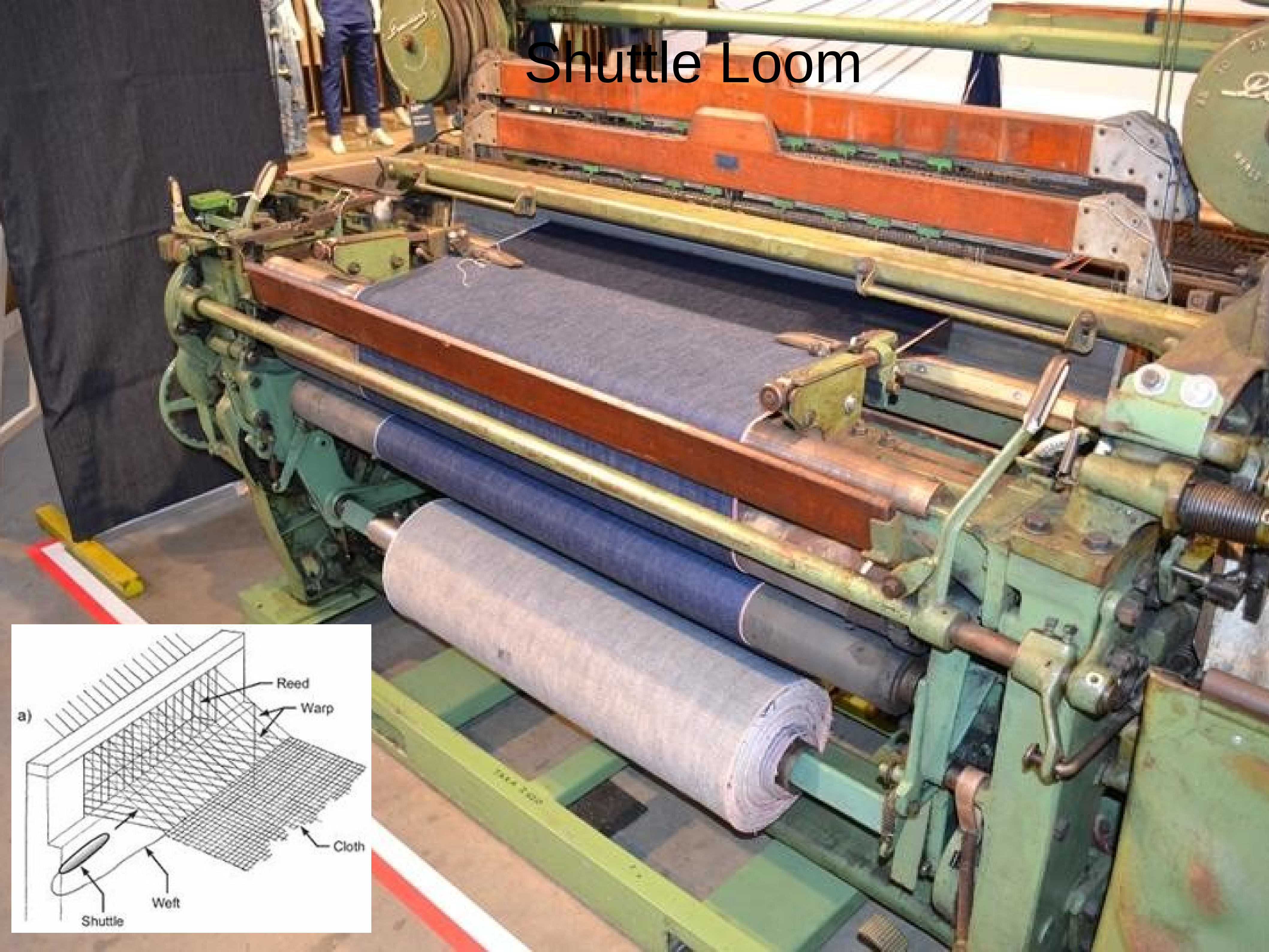
experiment result
(voltage distribution)

Parameters



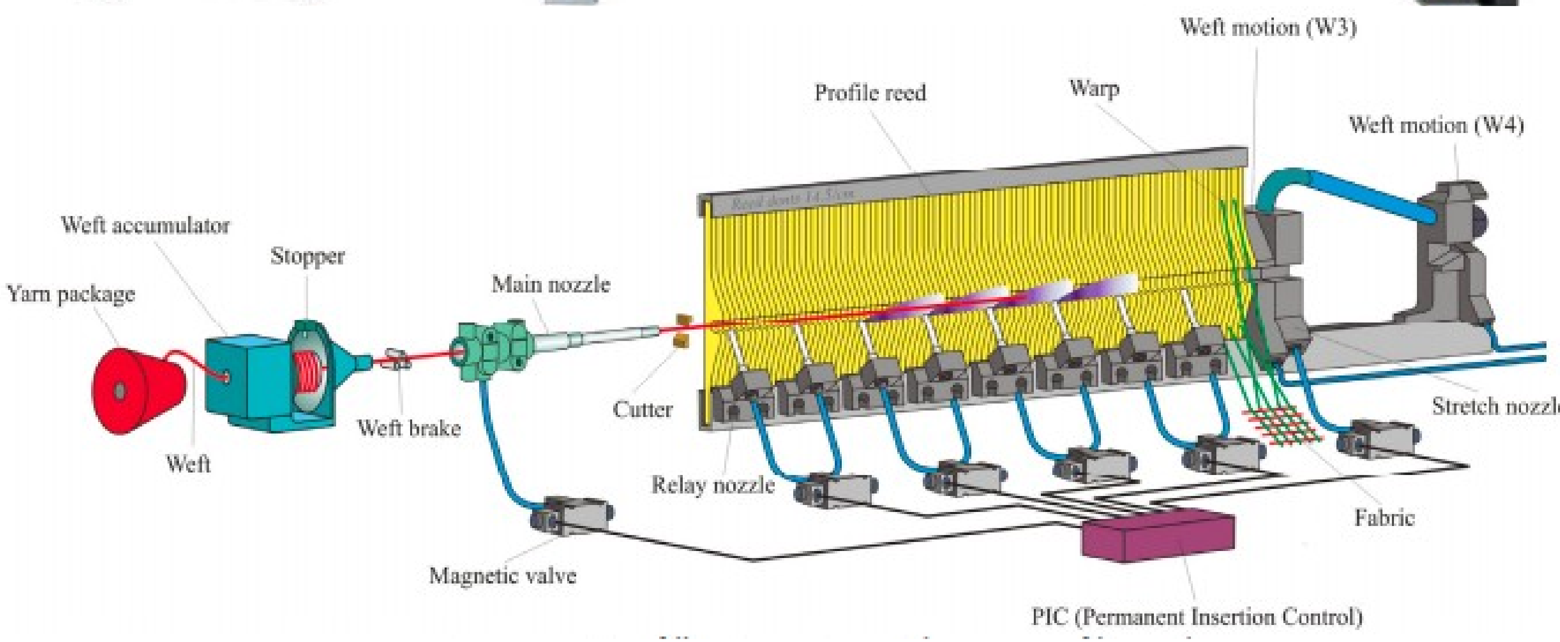
Distributed Lumped

Shuttle Loom



Air Jet Loom


PICANOL


Weft motion (W3)
Profile reed
Warp
Weft motion (W4)
Weft accumulator
Stopper
Yarn package
Main nozzle
Cutter
Weft brake
Weft
Relay nozzle
Stretch nozzle
Fabric
Magnetic valve
PIC (Permanent Insertion Control)

distributed + lumped parameter models

S. Haag, R. Anderl / Manufacturing Letters 15 (2018) 64–66

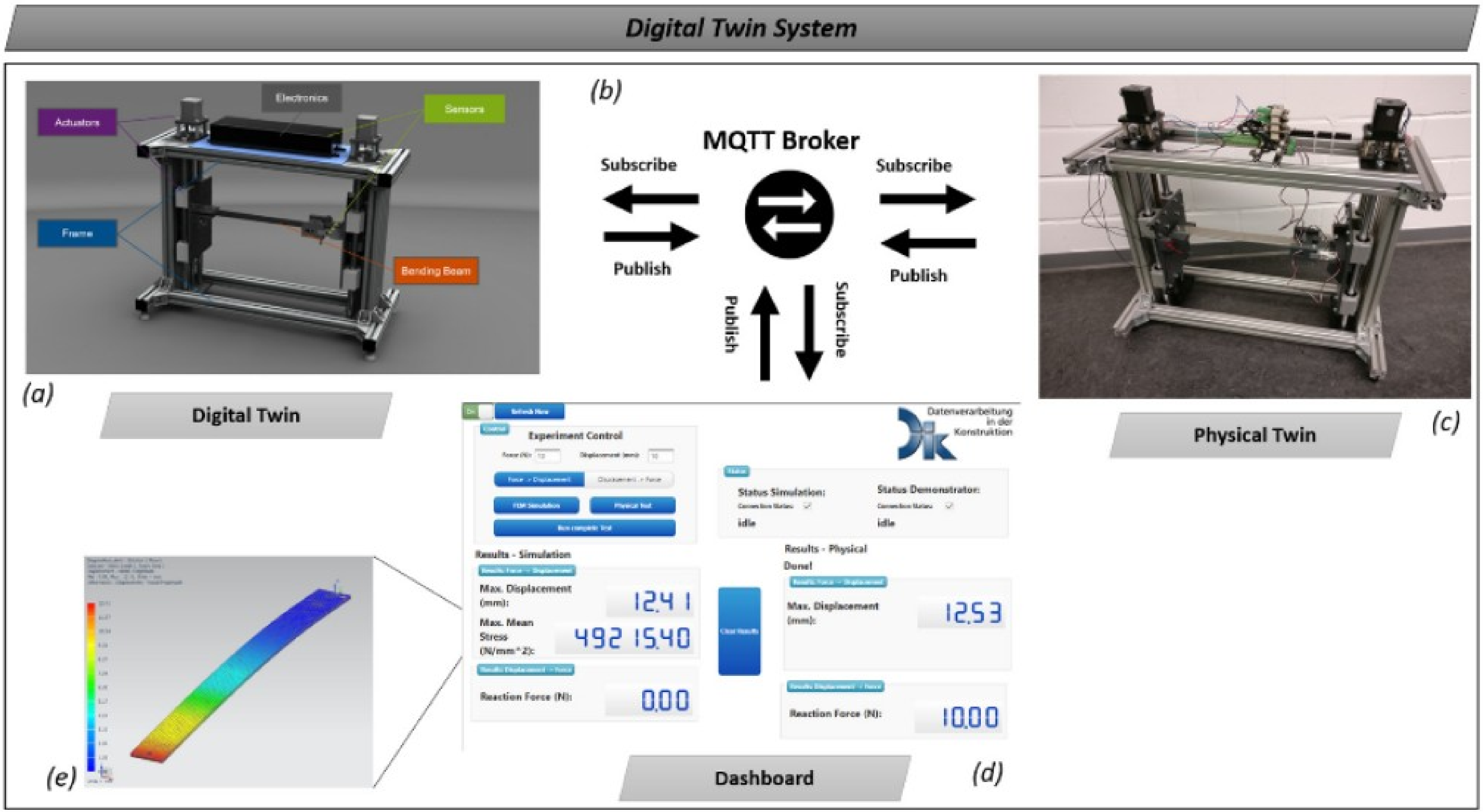
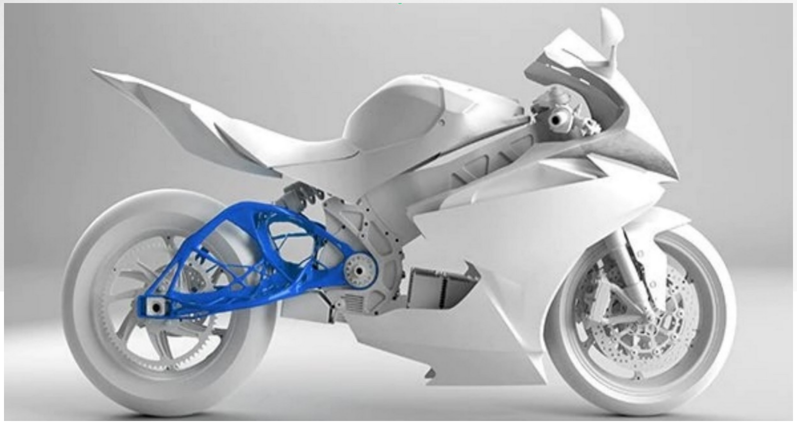


Fig. 1. Overview of the Digital Twin System. The Digital Twin (a) and the Physical Twin (c) are connected through a broker-client-architecture (b). The system is controlled via a web-based dashboard (d) accessible from any internet-capable device. Running a complete test through the dashboard will trigger the actuators of the physical twin as well as a FEM simulation. Results are shown numerically on the dashboard as well as graphically in the CAD system (e).

generative design
(Design-Space Exploration – DSE)

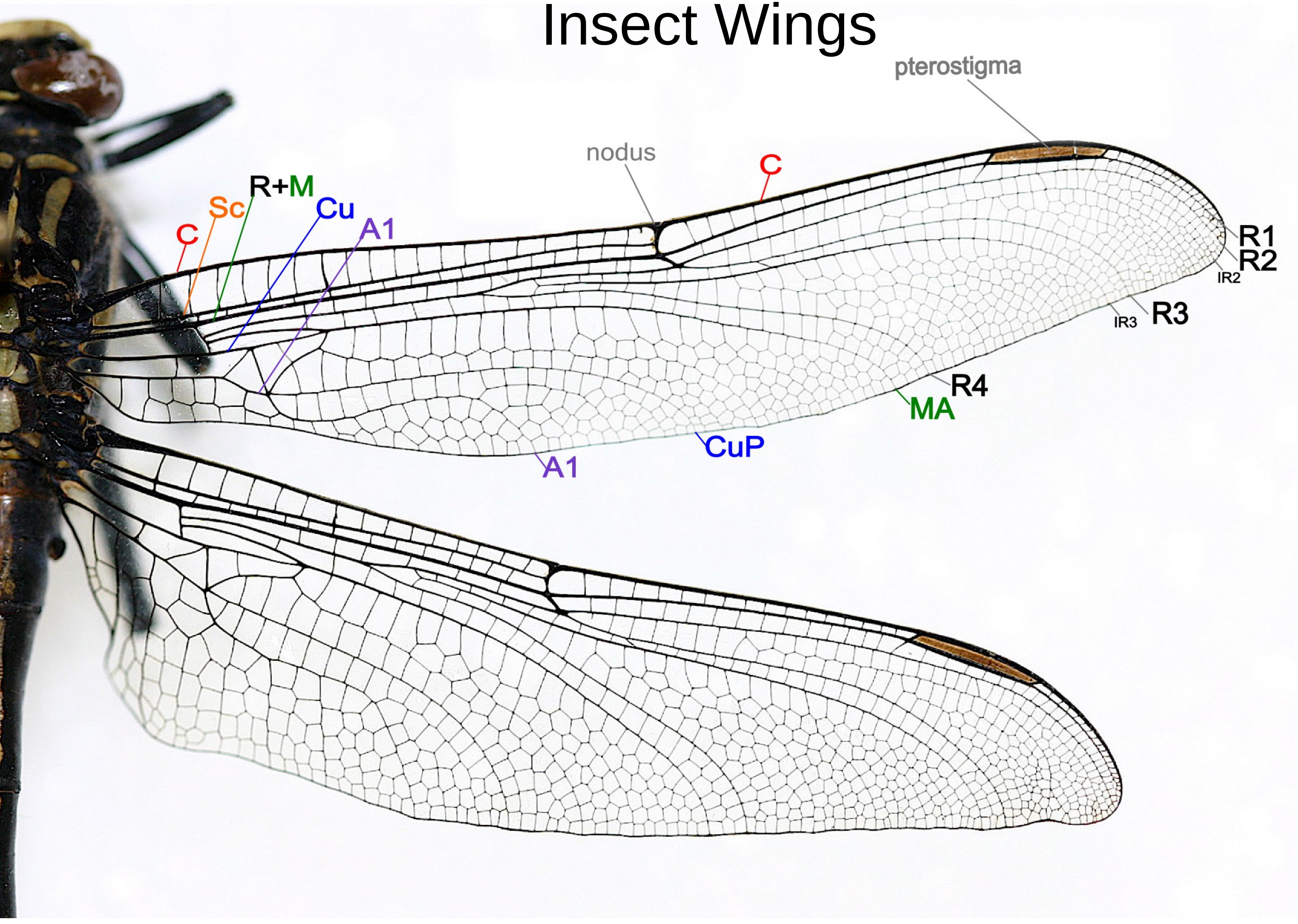

The World's Fastest Production
Electric Motorcycle - Lightning LS-218

AUTODESK.
Make anything.



https://www.autodesk.com/solutions/generative-design

Insect Wings

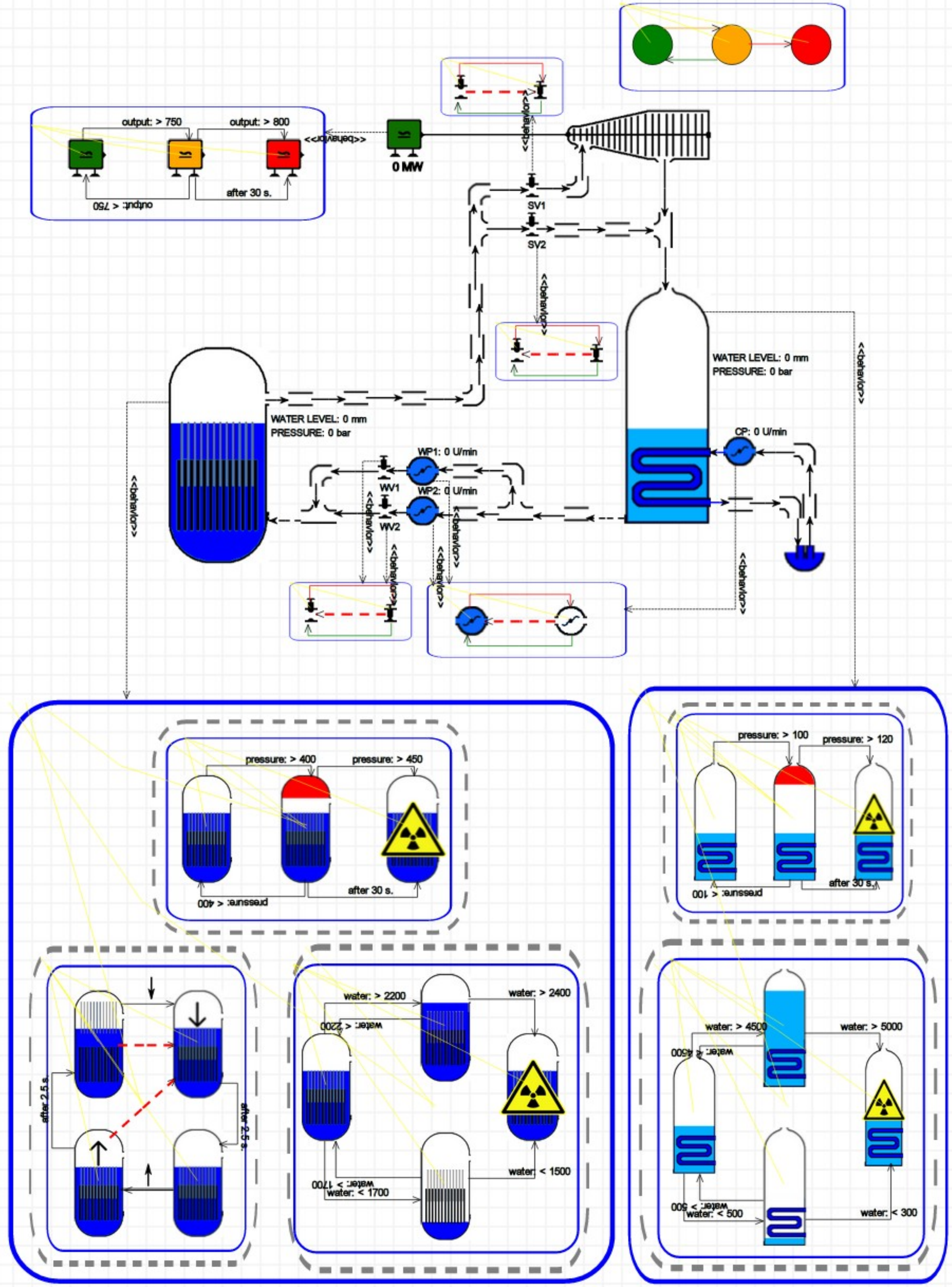


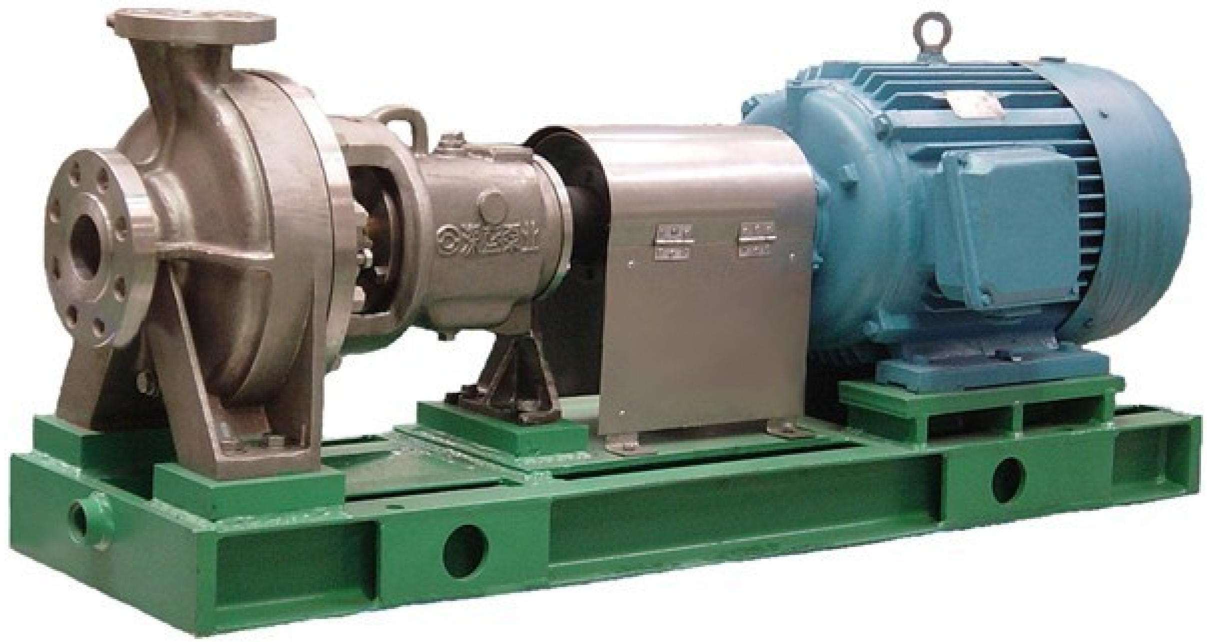
Physical Systems Modelling

- Problem-Specific (technological)
- Domain-Specific (e.g., translational mechanical)
- (general) Laws of Physics
- Power Flow/Bond Graphs (physical: energy/power)
- Computationally a-causal
  (Mathematical and Object-Oriented) ← **Modelica**
- Causal Block Diagrams (data flow)
- Numerical (Discrete) Approximations
- Computer Algorithmic + Numerical
  (Floating Point vs. Fixed Point)
- As-Fast-As-Possible vs. Real-time (XiL)
- Hybrid (discrete-continuous) modelling/simulation
- Hiding IP: Composition of Functional Mockup Units (FMI)
- Dynamic Structure

Physical Systems Modelling

- Problem-Specific (technological)
- Domain-Specific (e.g., translational mechanical)
- (general) Laws of Physics
- Power Flow/Bond Graphs (physical: energy/power)
- Computationally a-causal
  (Mathematical and Object-Oriented) ← **Modelica**
- Causal Block Diagrams (data flow)
- Numerical (Discrete) Approximations
- Computer Algorithmic + Numerical
  (Floating Point vs. Fixed Point)
- As-Fast-As-Possible vs. Real-time (XiL)
- Hybrid (discrete-continuous) modelling/simulation
- Hiding IP: Composition of Functional Mockup Units (FMI)
- Dynamic Structure


output: > 750
output: > 800
output: < 750
after 30 s.
<<behavior>>
0 MW
SV1
SV2
WATER LEVEL: 0 mm
PRESSURE: 0 bar
CP: 0 U/min
WATER LEVEL: 0 mm
PRESSURE: 0 bar
WP1: 0 U/min
WP2: 0 U/min
WV1
WV2
pressure: > 400
pressure: > 450
pressure: < 400
after 30 s.
pressure: > 100
pressure: > 120
pressure: < 100
after 30 s.
after 2.5 s.
after 2.5 s.
water: > 2200
water: > 2400
water: < 2200
water: < 1500
water: < 1700
water: < 1700
water: > 4500
water: > 5000
water: < 4500
water: < 500
water: < 500
water: < 300



Boric Acid Transportation Pump

Product parameters

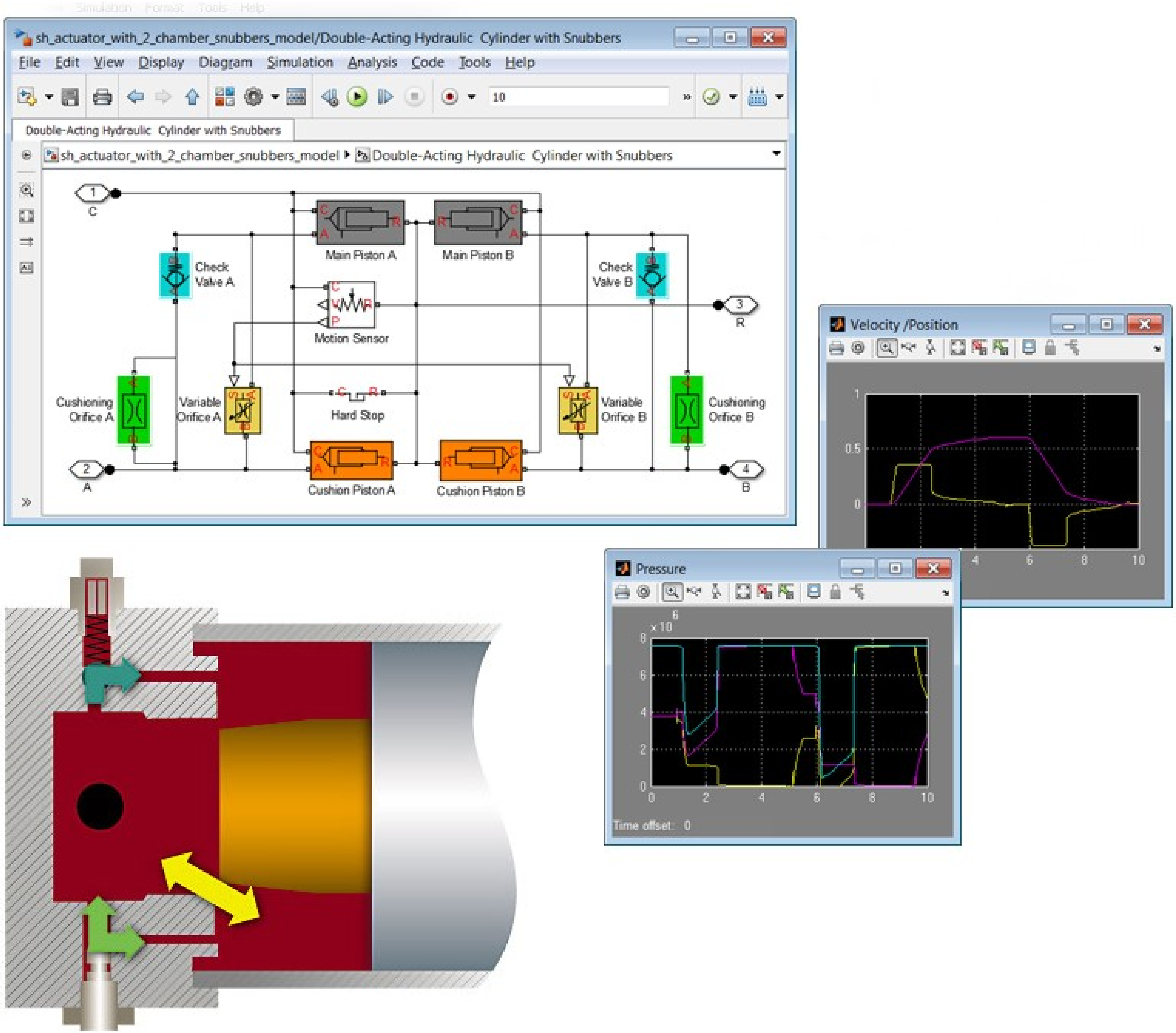
Design standards： RCC-M
Flow： 16.6m3/h
Head： 85m
Temperature： ~80℃
Pressure： 1.6MPa

Used in 600MWe 、 900MWe 、 1000MWe PWR nuclear power plant boric acid transportation system.

Physical Systems Modelling

- Problem-Specific (technological)
- Domain-Specific (e.g., translational mechanical)
- (general) Laws of Physics
- Power Flow/Bond Graphs (physical: energy/power)
- Computationally a-causal
  (Mathematical and Object-Oriented) ← **Modelica**
- Causal Block Diagrams (data flow)
- Numerical (Discrete) Approximations
- Computer Algorithmic + Numerical
  (Floating Point vs. Fixed Point)
- As-Fast-As-Possible vs. Real-time (XiL)
- Hybrid (discrete-continuous) modelling/simulation
- Hiding IP: Composition of Functional Mockup Units (FMI)
- Dynamic Structure

SimHydraulics



Why DS(V)M ?
(as opposed to General Purpose modelling)

- **match the user's mental model** of the problem domain
- **maximally constrain** the user (to the problem at hand)
  $\Rightarrow$ easier to learn
  $\Rightarrow$ avoid errors
- **separate** domain-expert's work
  from analysis/transformation expert's work

Anecdotal evidence of 5 to 10 times speedup

Steven Kelly and Juha-Pekka Tolvanen. Domain-Specific Modeling: Enabling Full Code Generation. Wiley, 2008.

Laurent Safa. The practice of deploying DSM, report from a Japanese appliance maker trenches. In Proceedings of the 6th OOPSLA Workshop on Domain-Specific Modeling (DSM'06), pp. 185-196, 2006.

++ more potential for optimization thanks to more (tighter) "type" information $V^2 = 4$

In der Beschränkung zeigt sich erst der Meister.

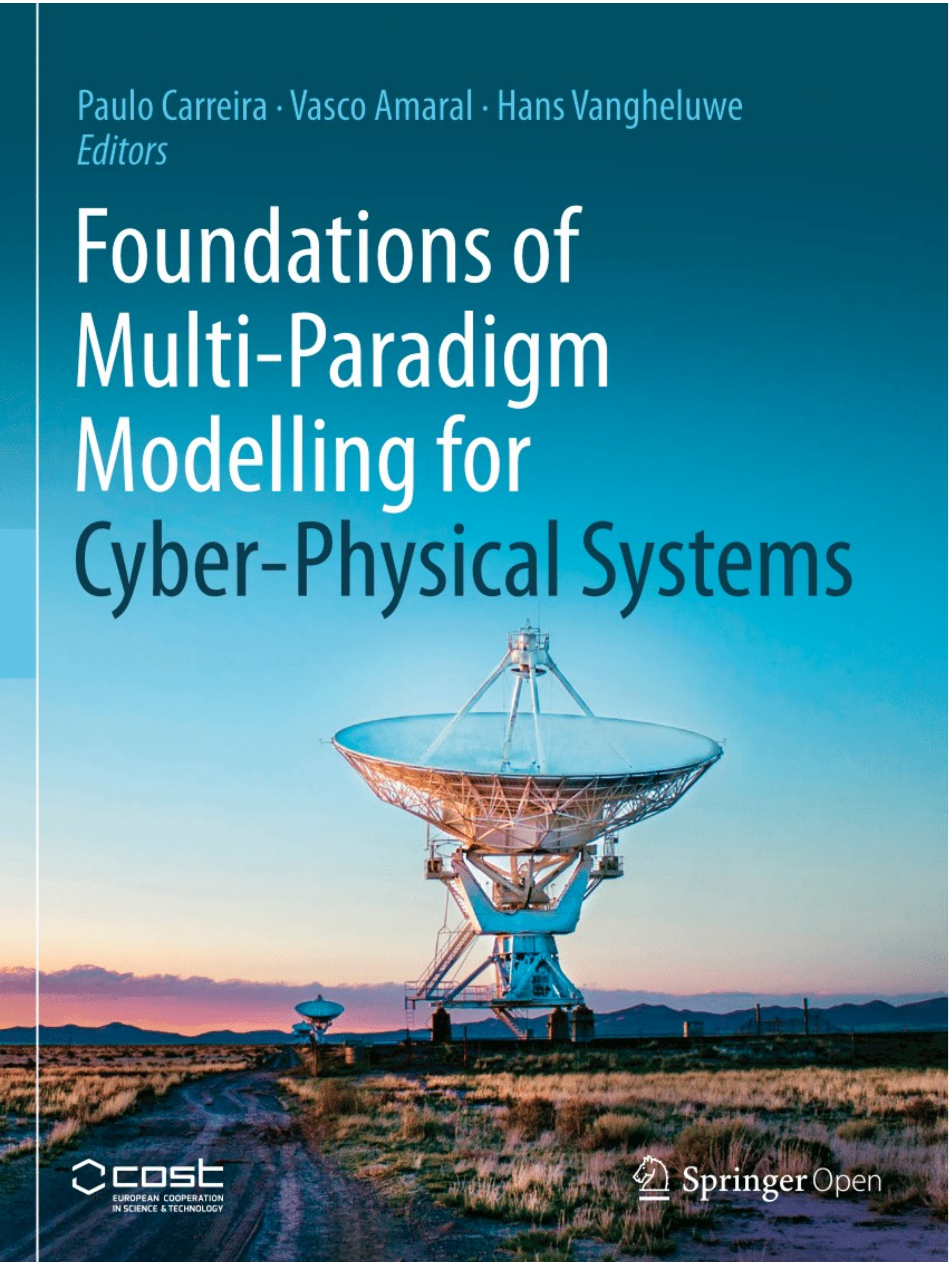
(Johann Wolfgang von Goethe)

Physical Systems Modelling

- Problem-Specific (technological)
- Domain-Specific (e.g., translational mechanical)
- (general) Laws of Physics
- Power Flow/Bond Graphs (physical: energy/power)
- Computationally a-causal
  (Mathematical and Object-Oriented) ← **Modelica**
- Causal Block Diagrams (data flow)
- Numerical (Discrete) Approximations
- Computer Algorithmic + Numerical
  (Floating Point vs. Fixed Point)
- As-Fast-As-Possible vs. Real-time (XiL)
- Hybrid (discrete-continuous) modelling/simulation
- Hiding IP: Composition of Functional Mockup Units (FMI)
- Dynamic Structure

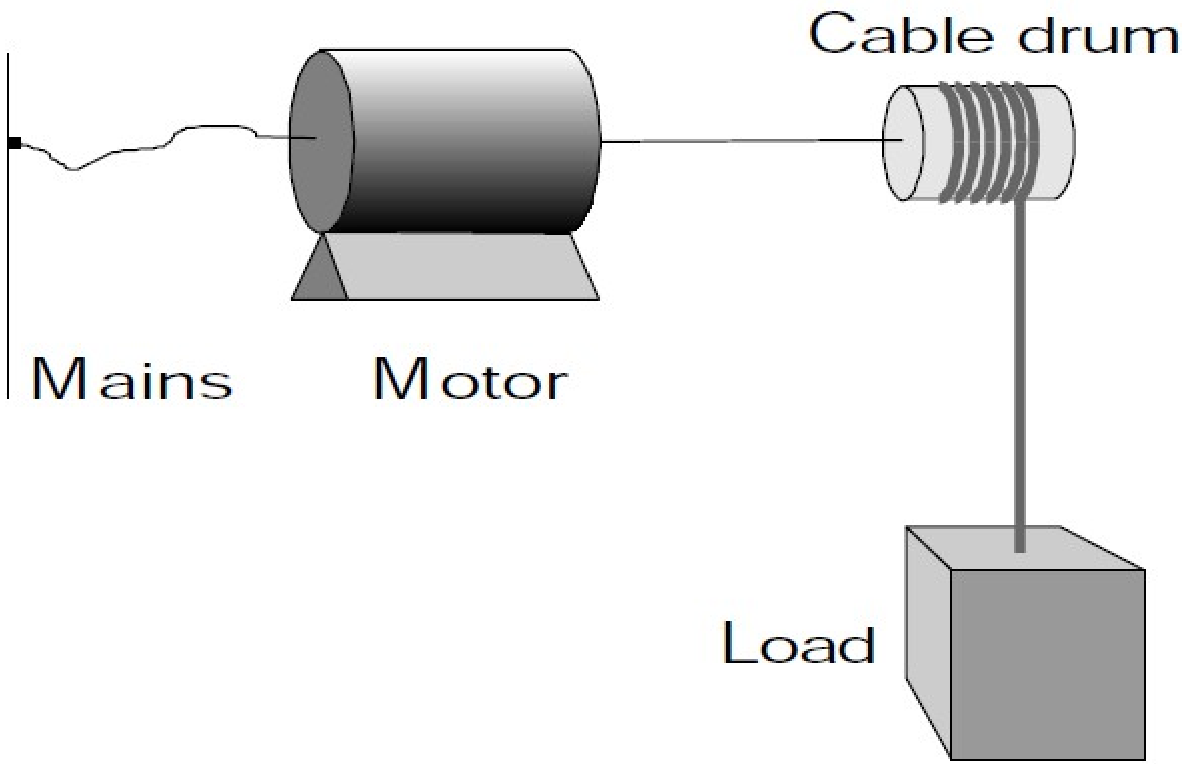
Physical Systems Modelling

- Problem-Specific (technological)
- Domain-Specific (e.g., translational mechanical)
- (general) Laws of Physics
- Power Flow/Bond Graphs (physical: energy/power)
- Computationally a-causal
  (Mathematical and Object-Oriented) ← **Modelica**
- Causal Block Diagrams (data flow)
- Numerical (Discrete) Approximations
- Computer Algorithmic + Numerical
  (Floating Point vs. Fixed Point)
- As-Fast-As-Possible vs. Real-time (XiL)
- Hybrid (discrete-continuous) modelling/simulation
- Hiding IP: Composition of Functional Mockup Units (FMI)
- Dynamic Structure

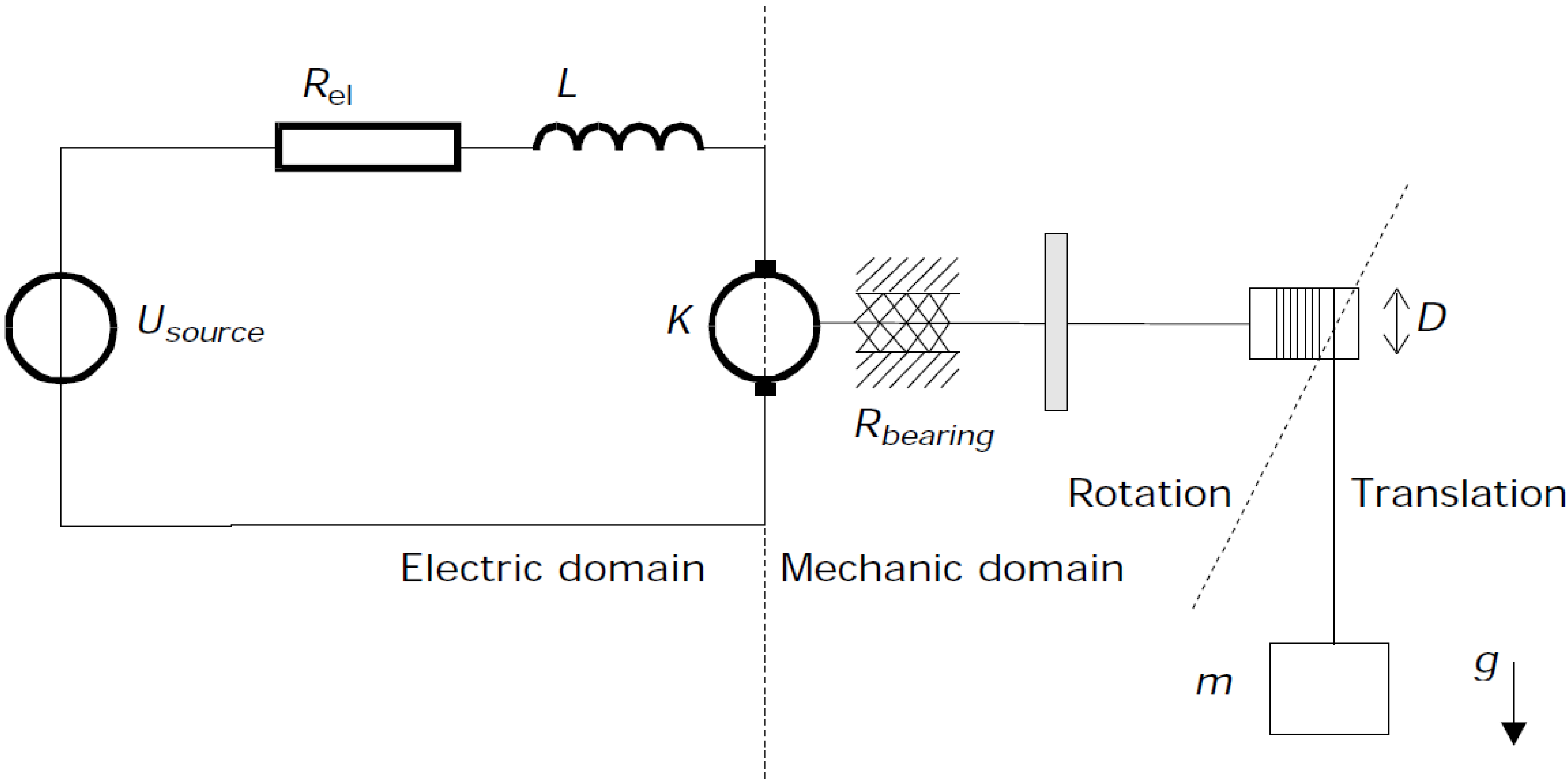


Broenink J.F. (2020) Bond Graphs: A Unifying Framework for Modelling of Physical Systems.
In: Carreira P., Amaral V., Vangheluwe H. (eds) Foundations of Multi-Paradigm Modelling for Cyber-Physical Systems. Springer, Cham.
https://doi.org/10.1007/978-3-030-43946-0_2

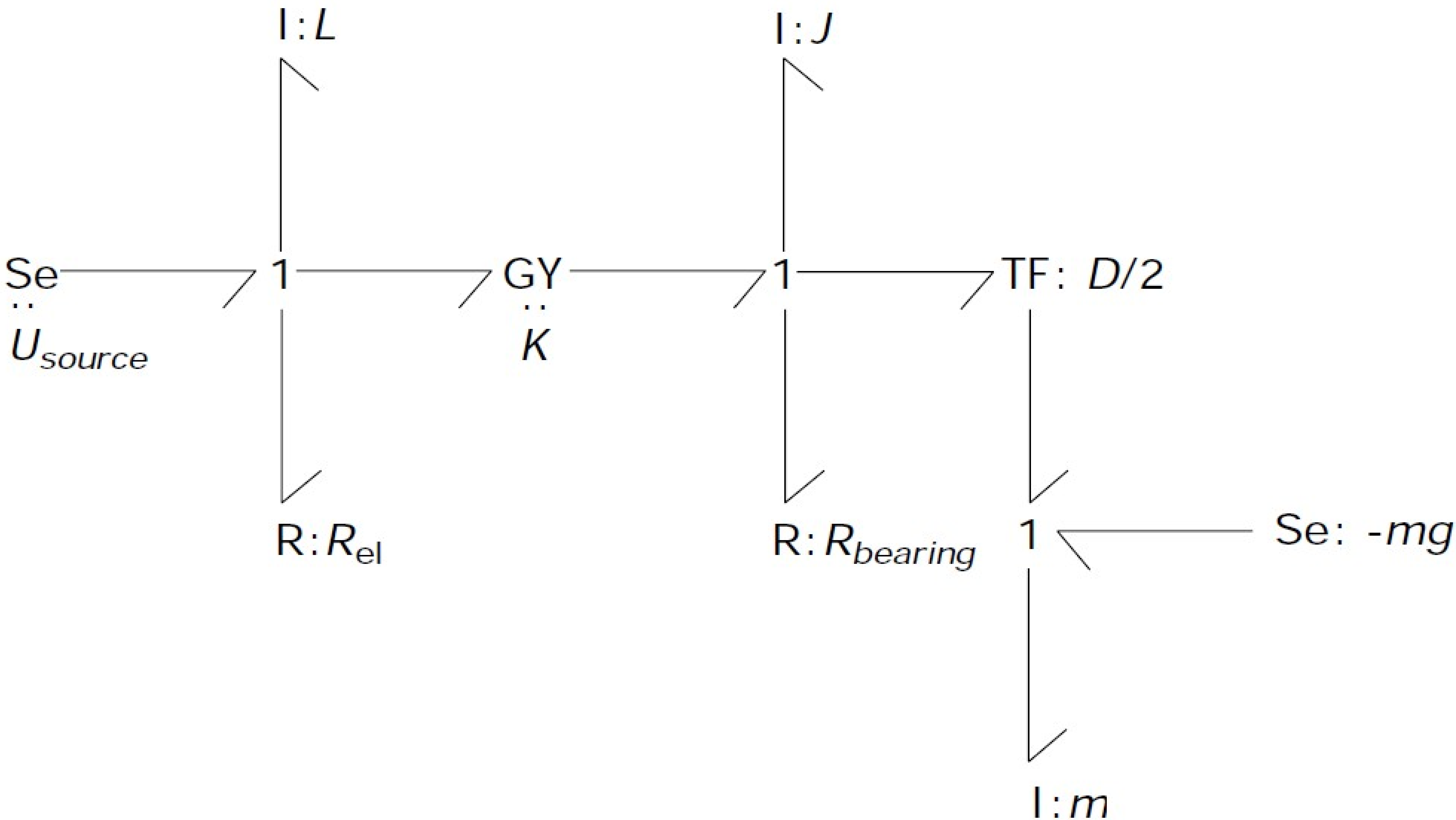
model using domain notation



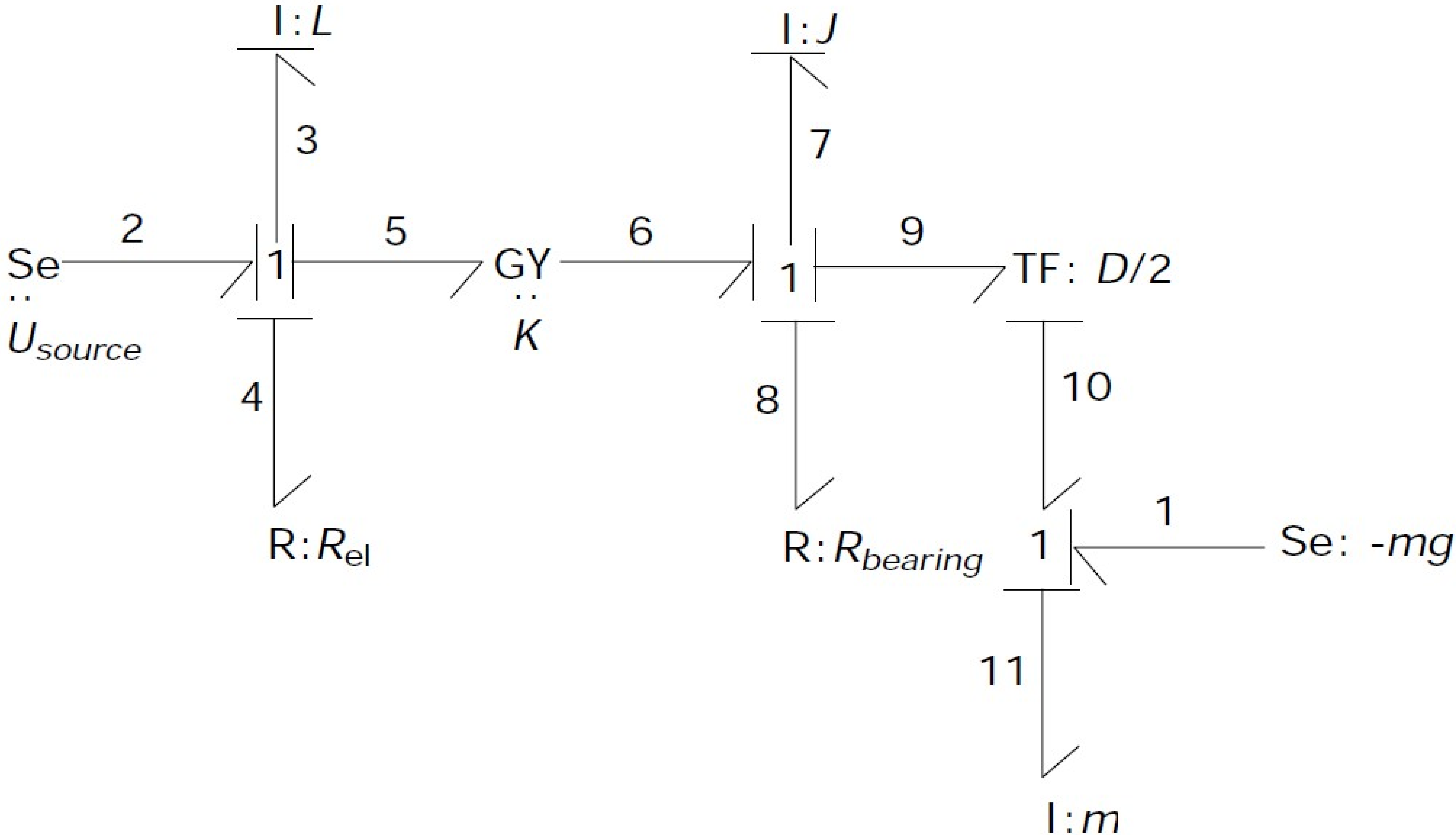
Idealized Physical Model (IPM)
1D aka “lumped parameter”



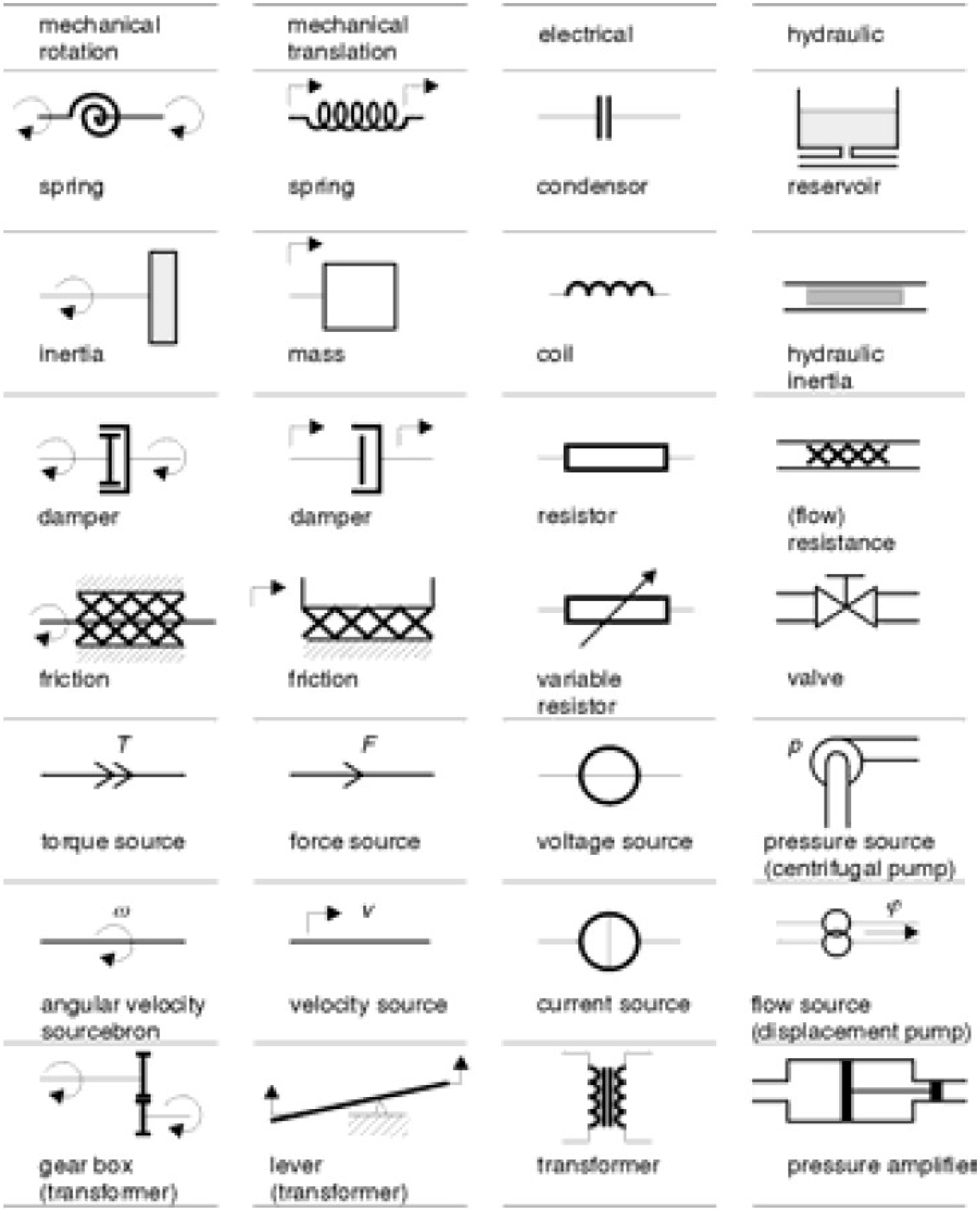
a-causal

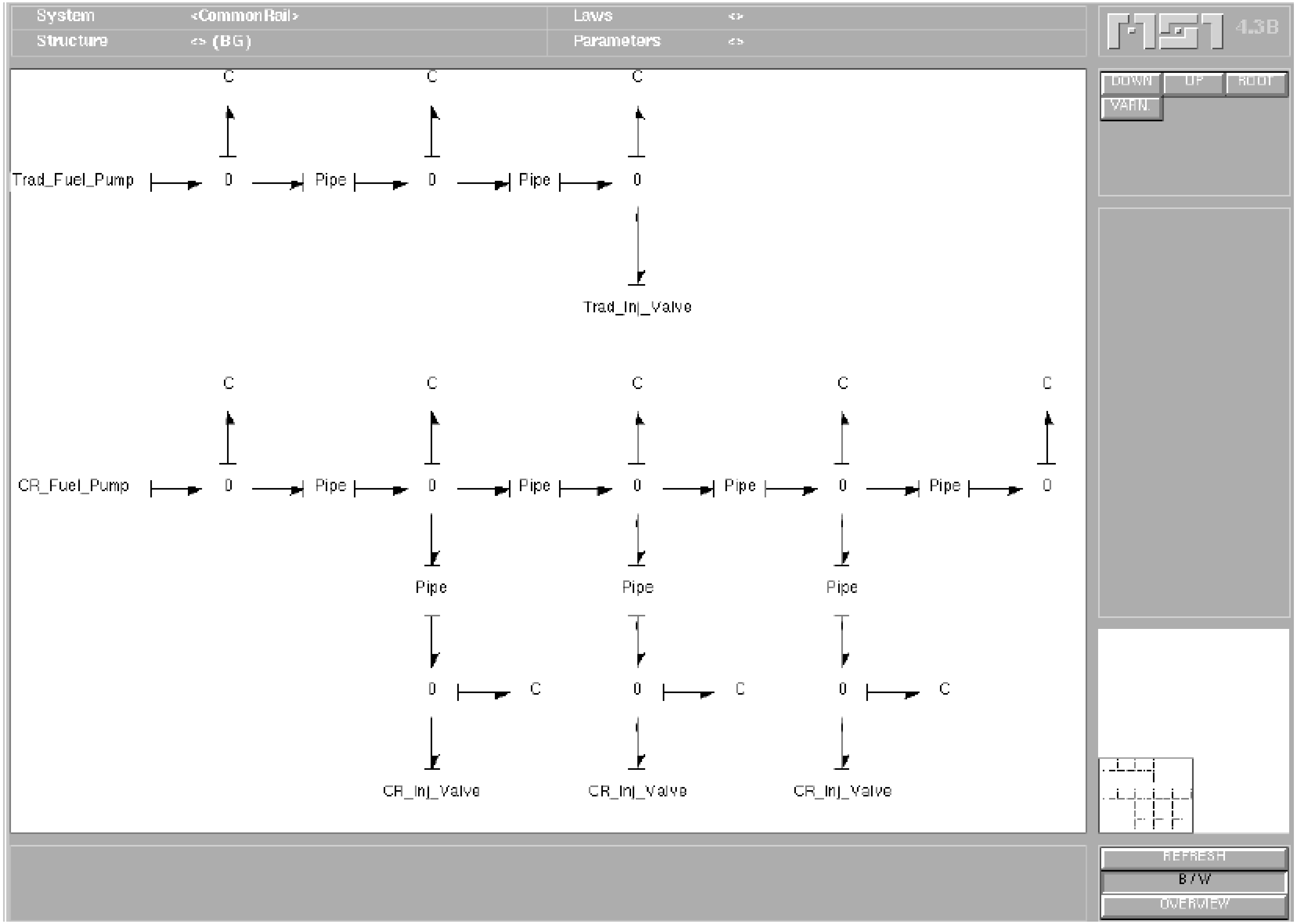


Causal (after “causality assignment” – propagation)



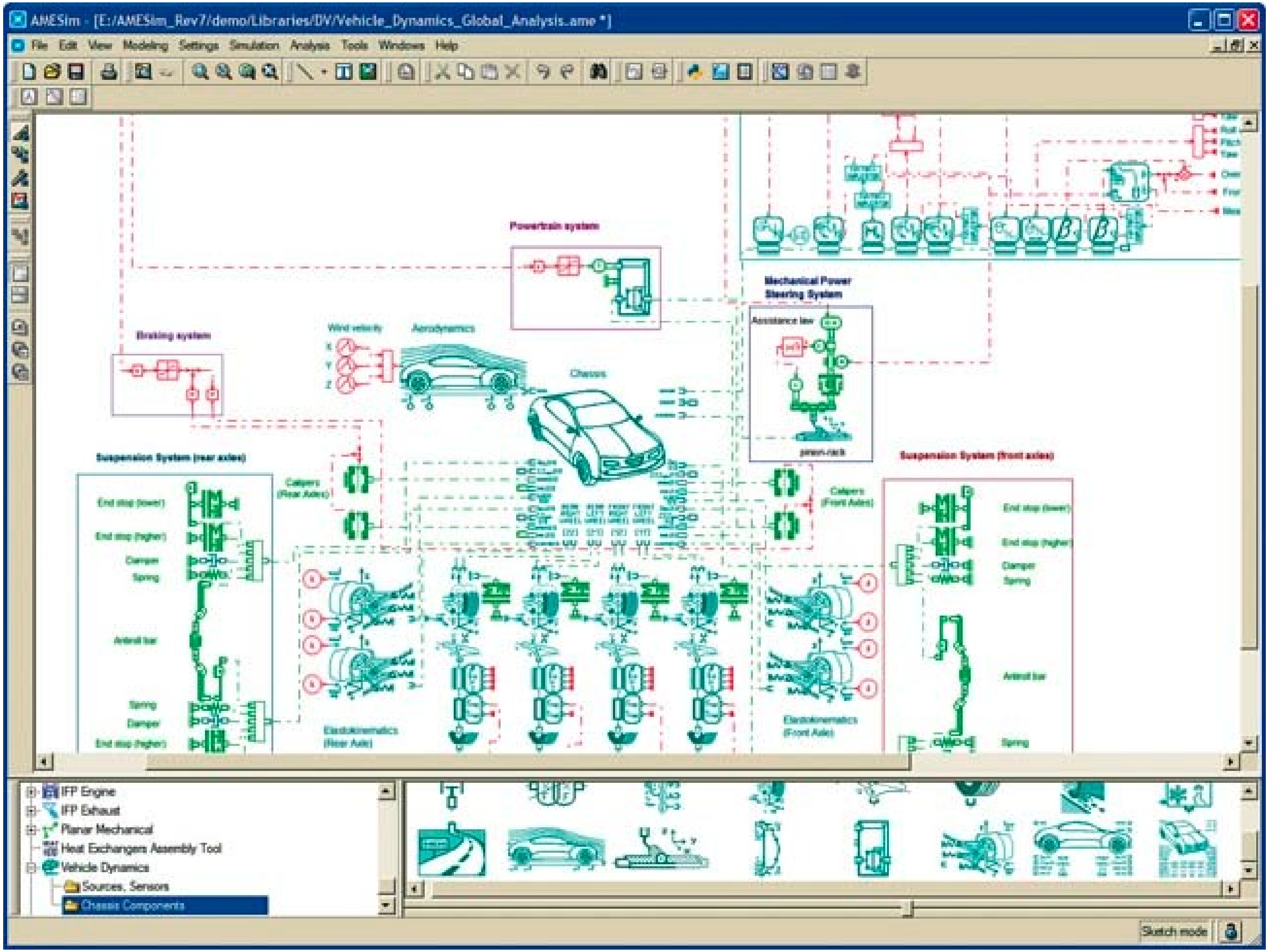
	f *flow*	E *effort*	$q = \int f\,dt$ *generalized displacement*	$p = \int e\,dt$ *generalized momentum*
Electromagnetic	i current	U voltage	$q = \int i dt$ charge	$\lambda = \int u dt$ magnetic flux linkage
mechanical translation	V velocity	F force	$x = \int v dt$ displacement	$p = \int F dt$ momentum
mechanical rotation	ω angular velocity	T torque	$\theta = \int \omega dt$ angular displacement	$b = \int T dt$ angular momentum
hydraulic/ pneumatic	φ volume flow	P pressure	$V = \int \varphi dt$ volume	$\Gamma = \int p dt$ momentum of a flow tube
Thermal	T temperature	F_S entropy flow	$S = \int f_S dt$ entropy	
Chemical	μ chemical potential	F_N molar flow	$N = \int f_N dt$ number of moles	


mechanical rotation
mechanical translation
electrical
hydraulic
spring
spring
condensor
reservoir
inertia
mass
coil
hydraulic inertia
damper
damper
resistor
(flow) resistance
friction
friction
variable resistor
valve
T
F
p
torque source
force source
voltage source
pressure source (centrifugal pump)
ω
v
φ
angular velocity sourcebron
velocity source
current source
flow source (displacement pump)
gear box (transformer)
lever (transformer)
transformer
pressure amplifier



System <CommonRail>
Structure <> (BG)
Laws <>
Parameters <>
4.3B
DOWN
UP
ROOT
VARN.
Trad_Fuel_Pump
0
Pipe
C
Trad_Inj_Valve
CR_Fuel_Pump
CR_Inj_Valve
REFRESH
B/W
OVERVIEW

Imagine.Lab AMESim



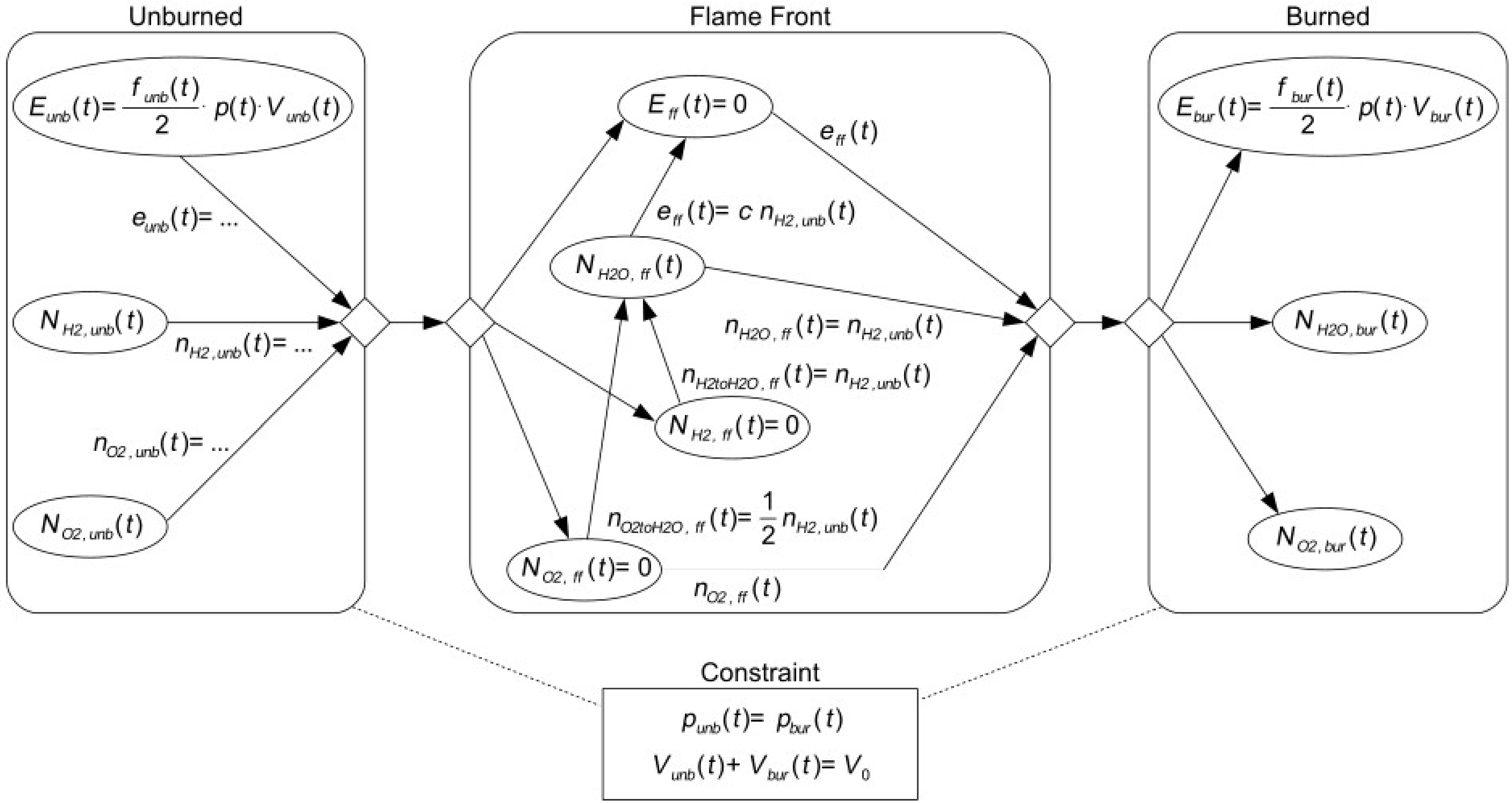
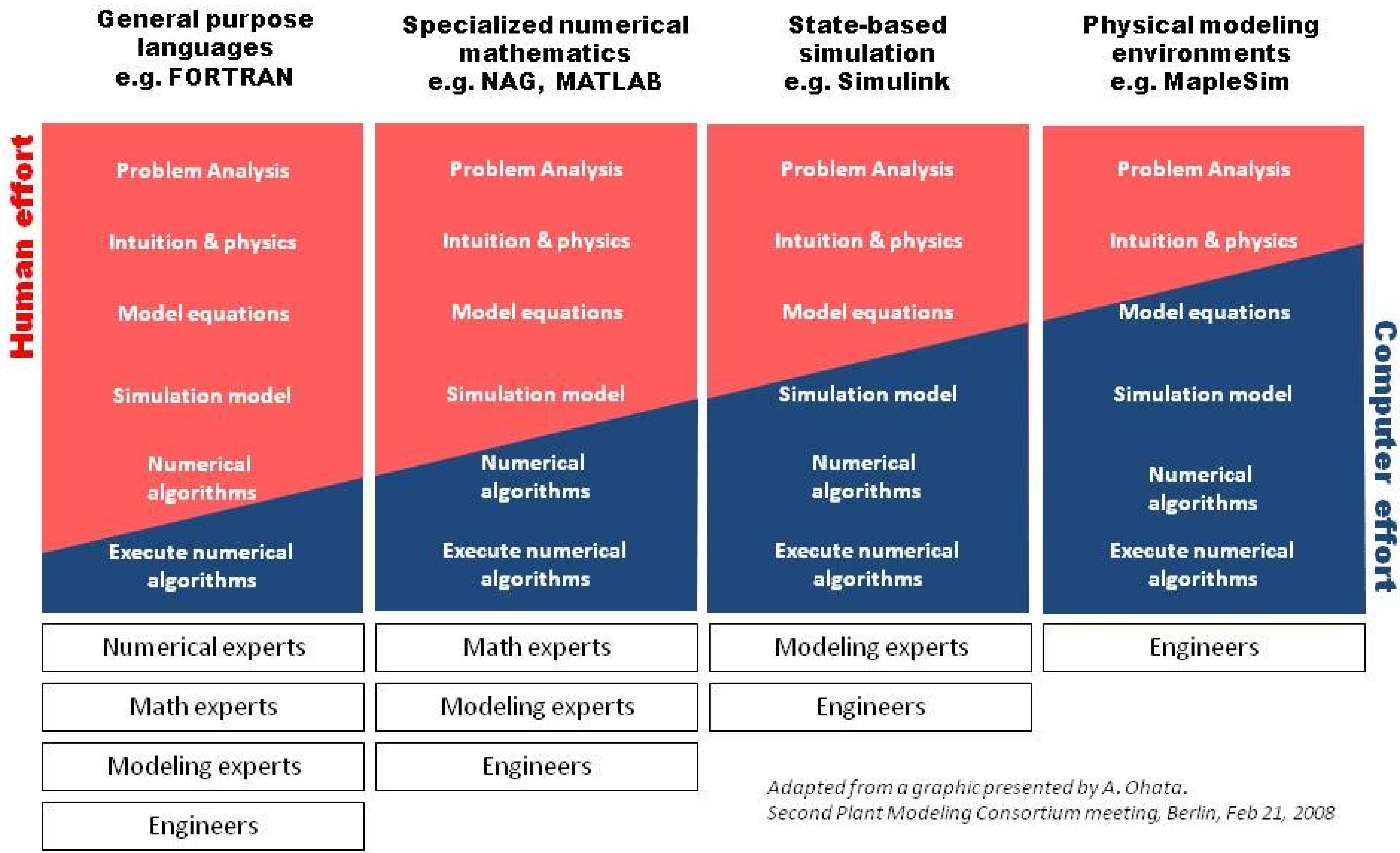


Figure 2. High Level Model Description (HLMD) example - hydrogen-oxygen combustion in a closed chamber.

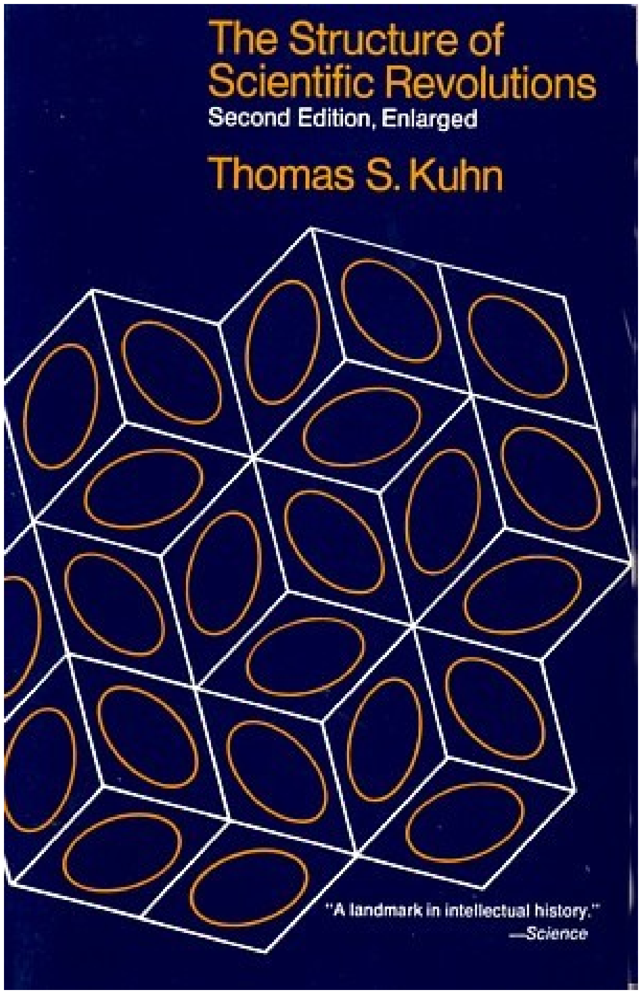
Akira Ohata @ Toyota

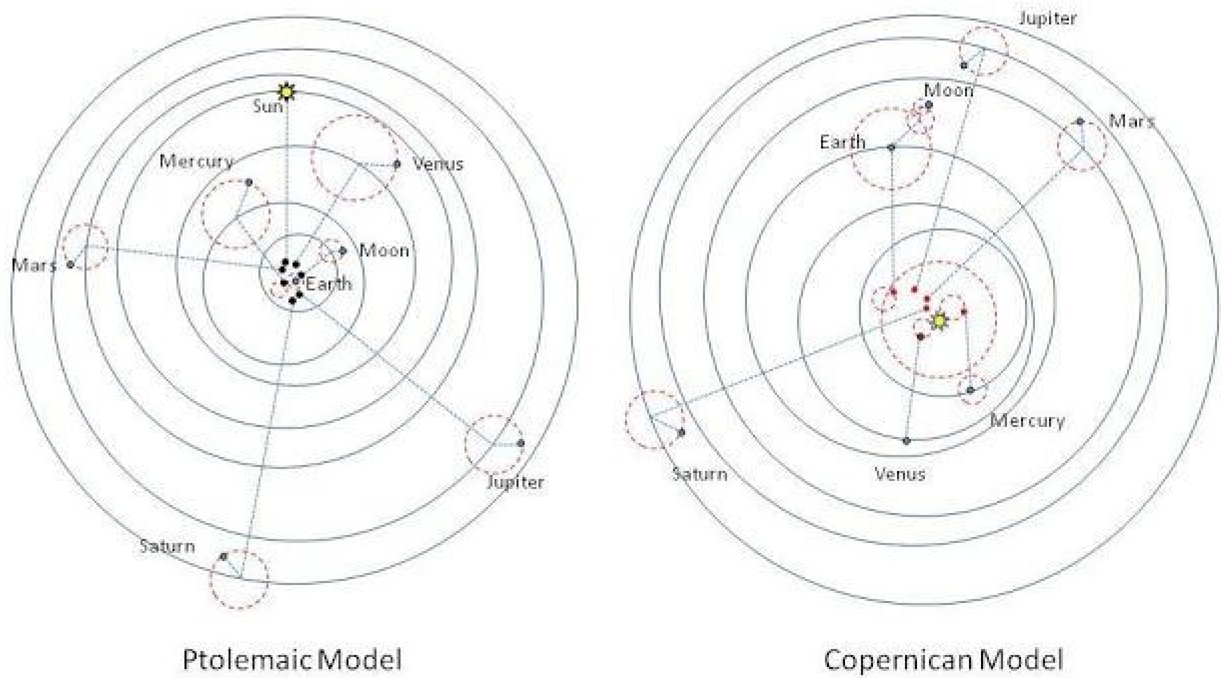

General purpose languages e.g. FORTRAN
Specialized numerical mathematics e.g. NAG, MATLAB
State-based simulation e.g. Simulink
Physical modeling environments e.g. MapleSim
Human effort
Computer effort
Problem Analysis
Intuition & physics
Model equations
Simulation model
Numerical algorithms
Execute numerical algorithms
Numerical experts
Math experts
Modeling experts
Engineers
Math experts
Modeling experts
Engineers
Modeling experts
Engineers
Engineers
Adapted from a graphic presented by A. Ohata.
Second Plant Modeling Consortium meeting, Berlin, Feb 21, 2008

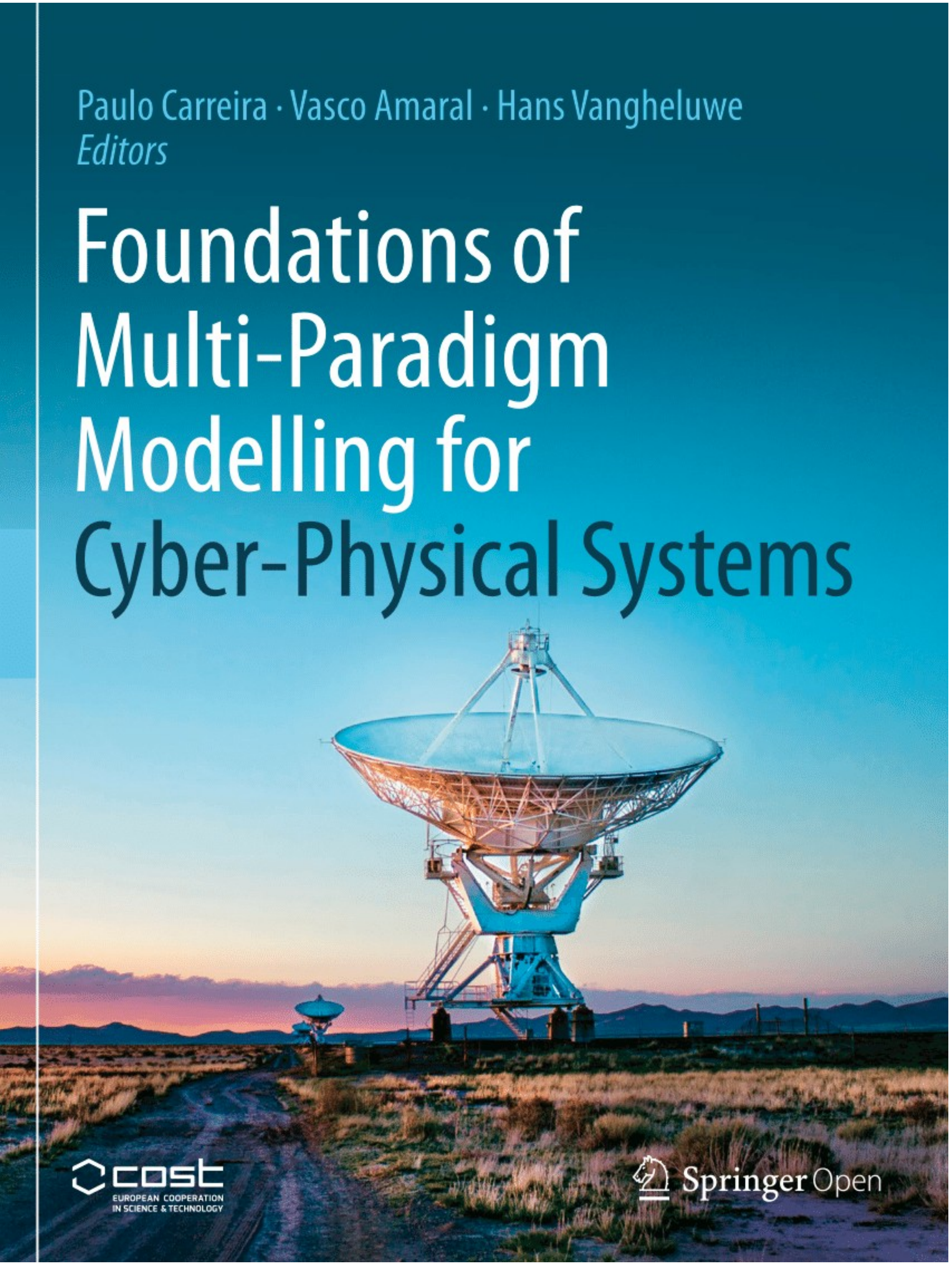
Physical Systems Modelling

- Problem-Specific (technological)
- Domain-Specific (e.g., translational mechanical)
- (general) Laws of Physics
- Power Flow/Bond Graphs (physical: energy/power)
- Computationally a-causal
  (Mathematical and Object-Oriented) ← **Modelica**
- Causal Block Diagrams (data flow)
- Numerical (Discrete) Approximations
- Computer Algorithmic + Numerical
  (Floating Point vs. Fixed Point)
- As-Fast-As-Possible vs. Real-time (XiL)
- Hybrid (discrete-continuous) modelling/simulation
- Hiding IP: Composition of Functional Mockup Units (FMI)
- Dynamic Structure

Thomas Kuhn: “paradigm shift”







Fritzson P. (2020) Modelica: Equation-Based, Object-Oriented Modelling of Physical Systems.
In: Carreira P., Amaral V., Vangheluwe H. (eds) Foundations of Multi-Paradigm Modelling for Cyber-Physical Systems. Springer, Cham.
https://doi.org/10.1007/978-3-030-43946-0_3

Dokumentutgivare
Lund Institute of Technology

Handläggare
Karl Johan Åström

Författare
Hilding Elmqvist

Dokumentnamn
REPORT

Dokumentbeteckning
LUTFD2/(TFRT-1015)/1-226/(1978)

Utgivningsdatum
May 1978

Ärendebeteckning

Dymola

Dokumenttitel och undertitel

A Structured Model Language for Large Continuous Systems

Referat (sammandrag)

A model language, called DYMOLA, for continuous dynamical systems is proposed. Large models are conveniently described hierarchically using a submodel concept. The ordinary differential equations and algebraic equations need not be converted to assignment statements. There is a concept, cut, which corresponds to connection mechanisms of complex types, and there are facilities to describe the connection structure of a system. A model can be manipulated for different purposes such as simulation and static calculations. The model equations are sorted and they are converted to assignment statements using formula manipulation. A translator for the model language is also included.

Referat skrivet av
Author

Förslag till ytterligare nyckelord
nonlinear systems, compiler, permutations, graph theory

Klassifikationssystem och -klass(er)

Indextermer (ange källa)
Mathematical models, Simulation languages, Computerized simulation, Nonlinear systems, Ordinary differential equations, Compilers. (Thesaurus of Engineering and Scientific Terms, Eng. Joint Council,USA)

Omfång
226 pages

Övriga bibliografiska uppgifter

Språk
English

Sekretessuppgifter

ISSN

ISBN

Dokumentet kan erhållas från
Department of Automatic Control
Lund Institute of Technology
P O Box 725, S-220 07 Lund 7, Sweden

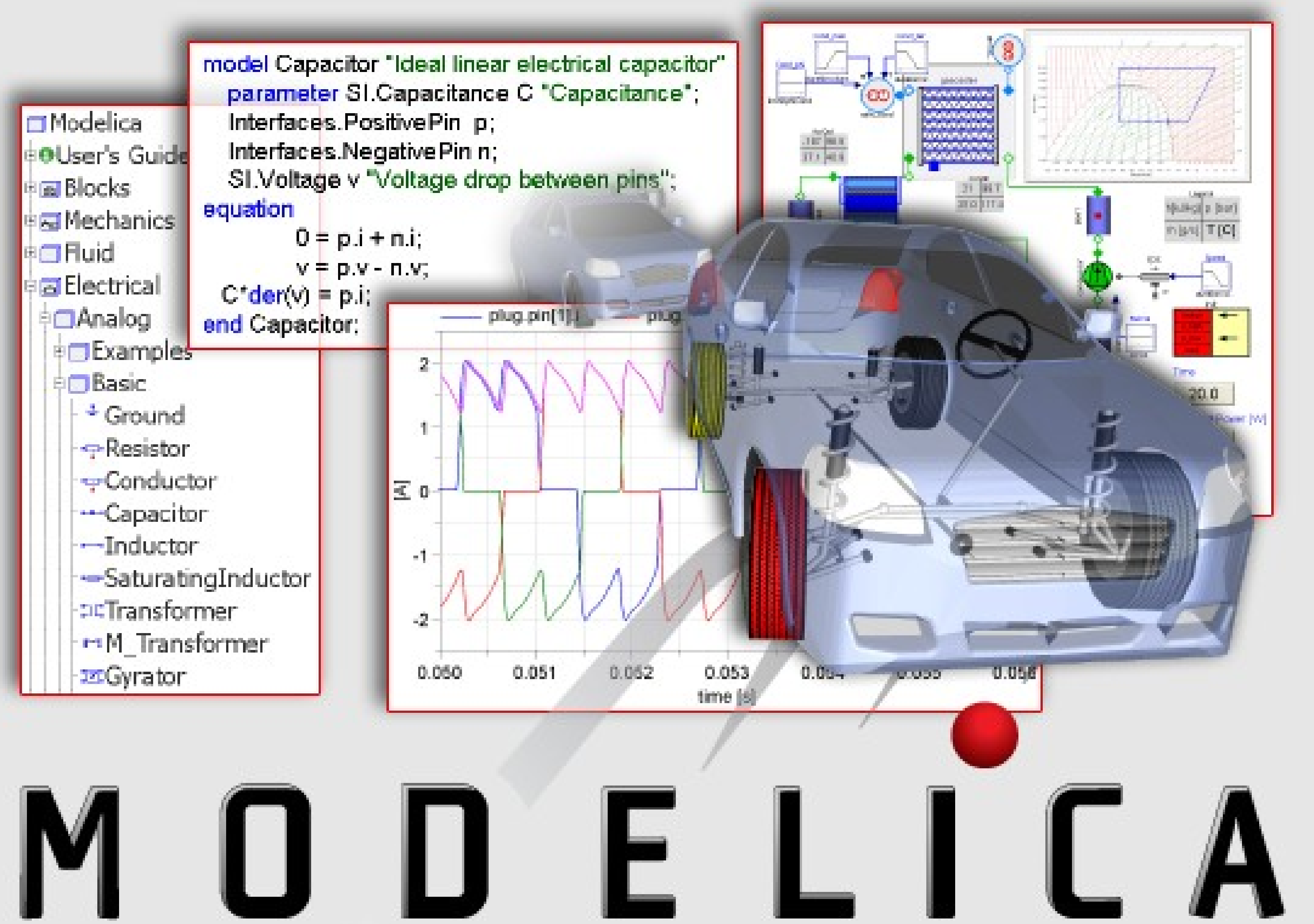
Mottagarens uppgifter

Pris

SIS-DB 1

DOKUMENTDATABLAD enligt SIS 62 10 12

Blankett LU 11:25 1976–07



Modelica
User's Guide
Blocks
Mechanics
Fluid
Electrical
Analog
Examples
Basic
Ground
Resistor
Conductor
Capacitor
Inductor
SaturatingInductor
Transformer
M_Transformer
Gyrator
model Capacitor "Ideal linear electrical capacitor"
parameter SI.Capacitance C "Capacitance";
Interfaces.PositivePin p;
Interfaces.NegativePin n;
SI.Voltage v "Voltage drop between pins";
equation
0 = p.i + n.i;
v = p.v - n.v;
C*der(v) = p.i;
end Capacitor;
plug.pin[1].i
[A]
2
1
0
-1
-2
0.050
0.051
0.052
0.053
time [s]
MODELICA

Simulation in Europe

SiE

ESPRIT Basic Research Working Group 8467
Simulation for the Future: New Concepts, Tools and Applications

Keywords:
simulation technologies, multi-paradigm modelling, solvers, standards, interoperability, industrial deployment, demonstrators, user-simulator interfaces

OpenModelica

Equation-Based Object-Oriented Modeling Languages and Tools

News

EOOLT 2017
The EOOLT workshop took successfully place in Munich, Germany on December 1. Proceedings are now available on ACM Digital Library

Modelica Scalable Test Suite
A new suite of scalable test models can be found here.

Welcome to the EOOLT community!

This site is intended to be a meeting point for researchers and practitioners working in the area of equation-based object-oriented modeling languages and tools. The site's main purpose is to host the workshop pages for the EOOLT workshop series. Below you can find links to the current and past events, together with links to the open access workshop proceedings.

This site is maintained by David Broman. If you have any questions or comments, please send an email.

EOOLT 2017, December 1, Munich, Germany
8th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools

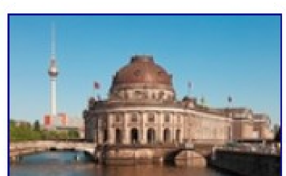
EOOLT 2017 Proceedings (ACM Digital Library)

Workshop site

EOOLT 2016, April 18, Milano, Italy
7th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools

EOOLT 2016 Proceedings (ACM Digital Library)

Workshop site (archived)



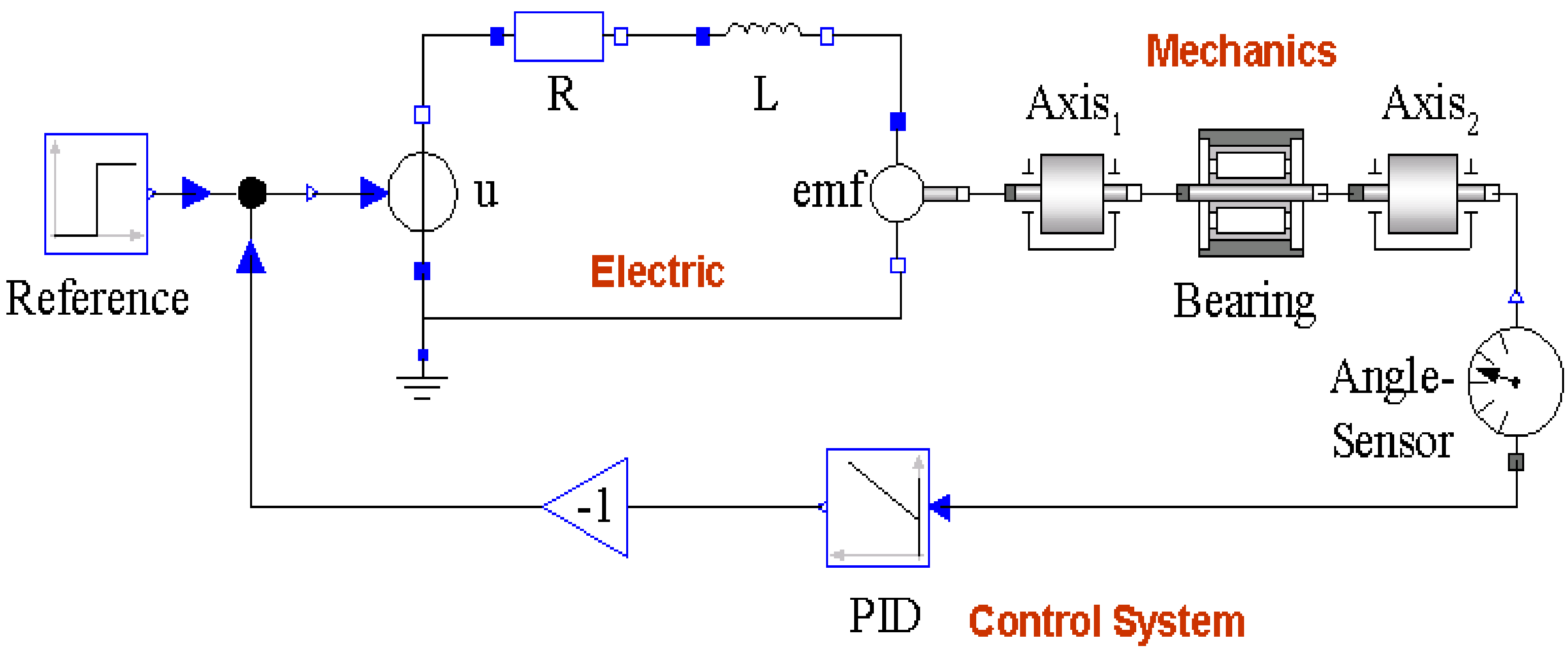
EOOLT 2014, Berlin, Germany
6th International Workshop on Equation-Based Object-Oriented Modeling Languages and Tools

EOOLT 2014 Proceedings (ACM Digital Library)

Workshop site (archived)

Multi-Domain Modeling

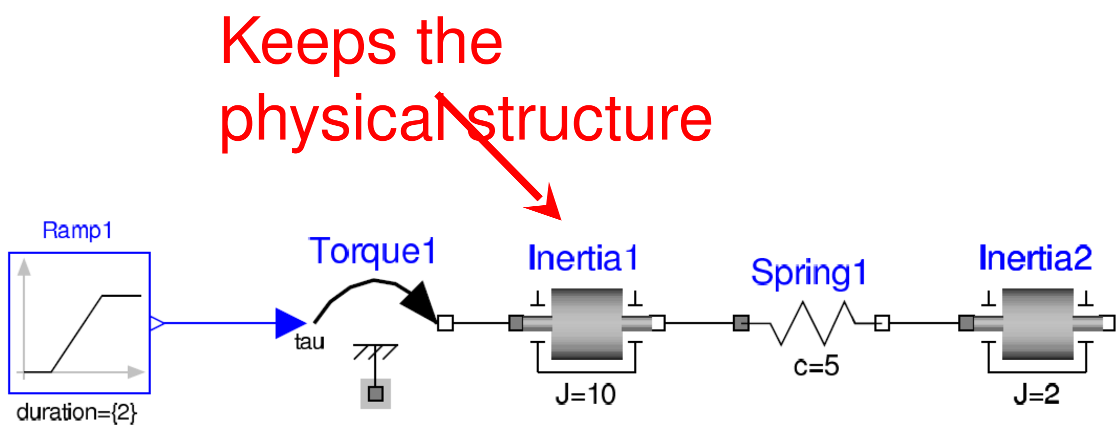
http://www.modelica.org



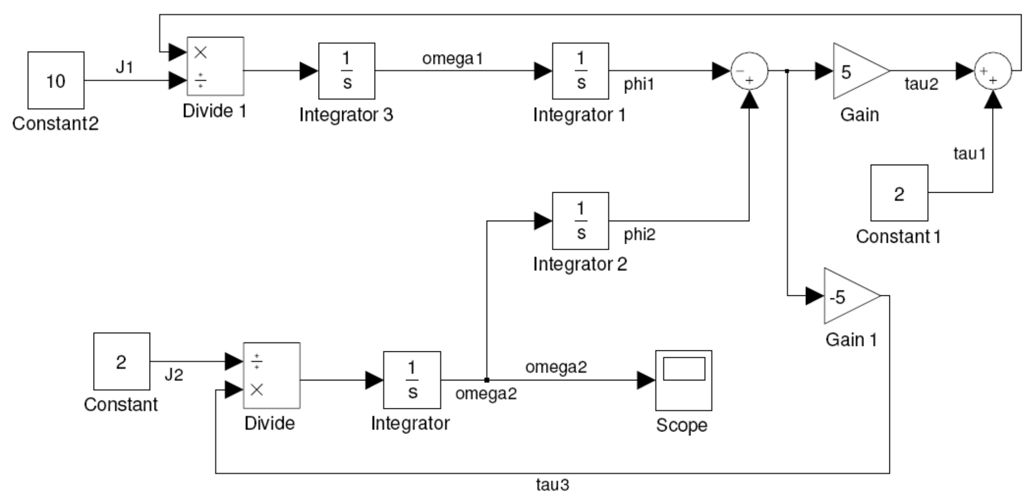
this slide from Peter Fritzson's Modelica tutorial

Multi-Domain Modeling

Visual Acausal Hierarchical Component Modeling



Acausal model (Modelica)

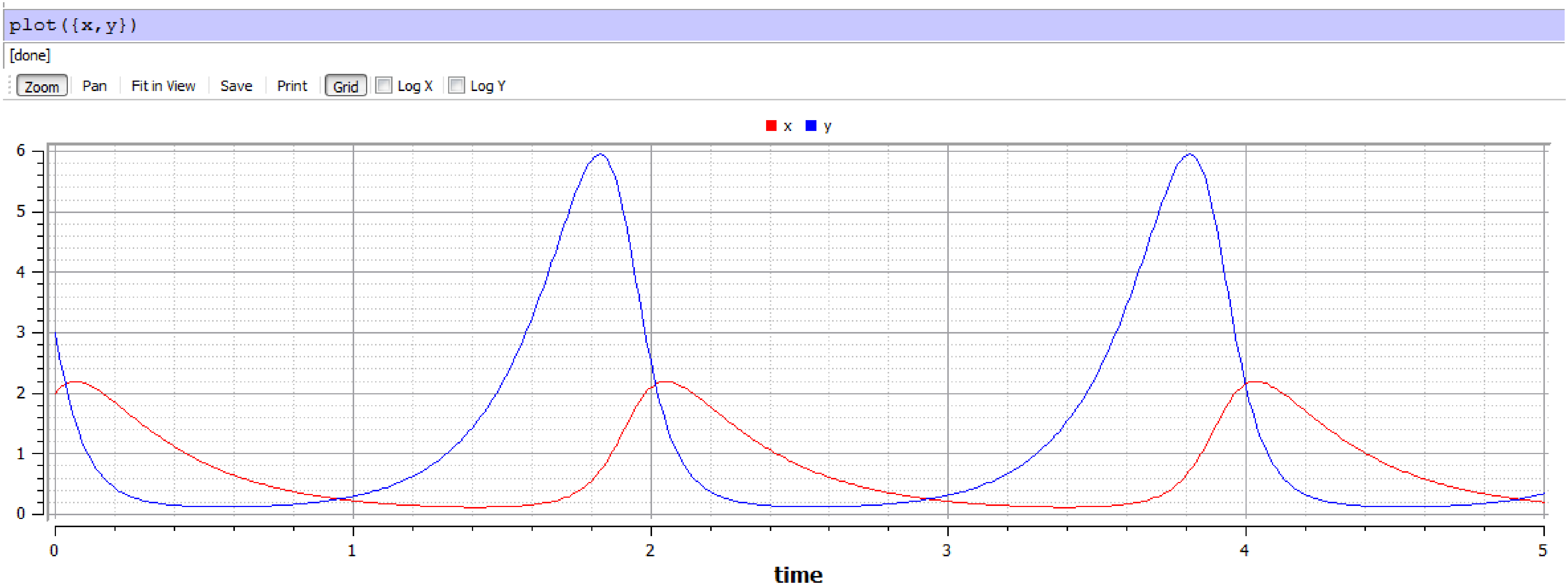


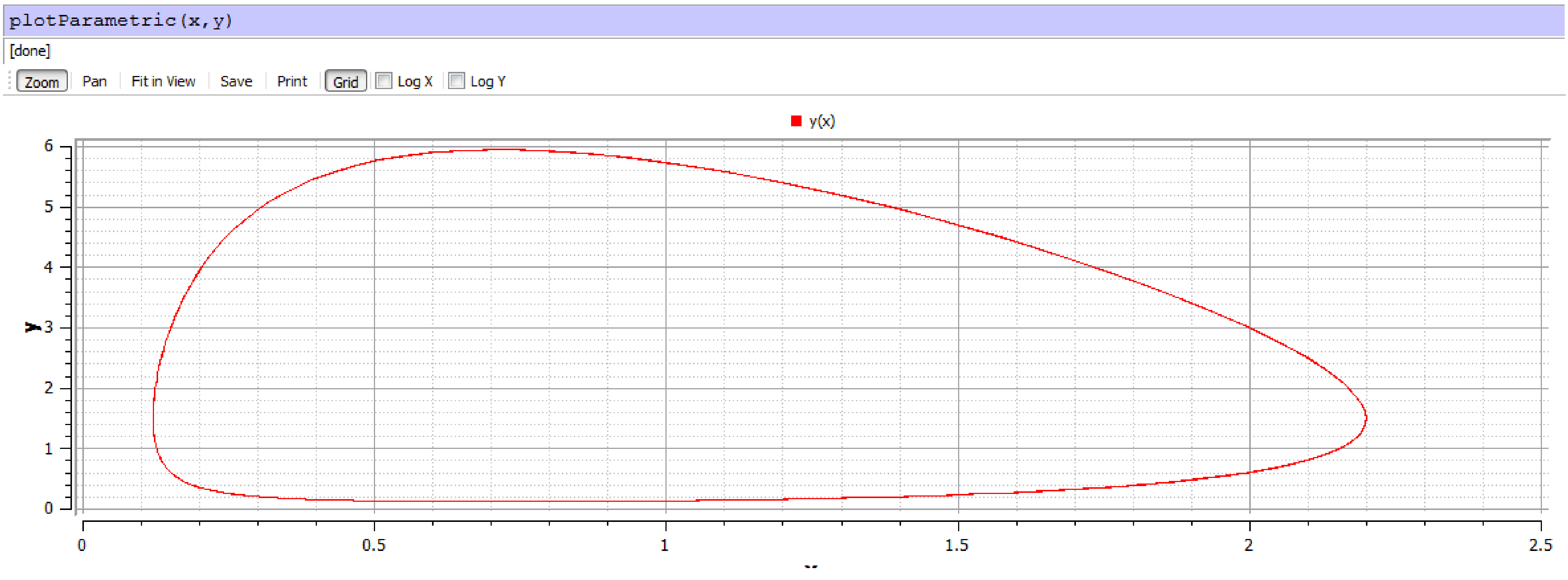
Causal block-based model (Simulink)

this slide from Peter Fritzson's Modelica tutorial

Equation-based (computationally a-causal) modelling

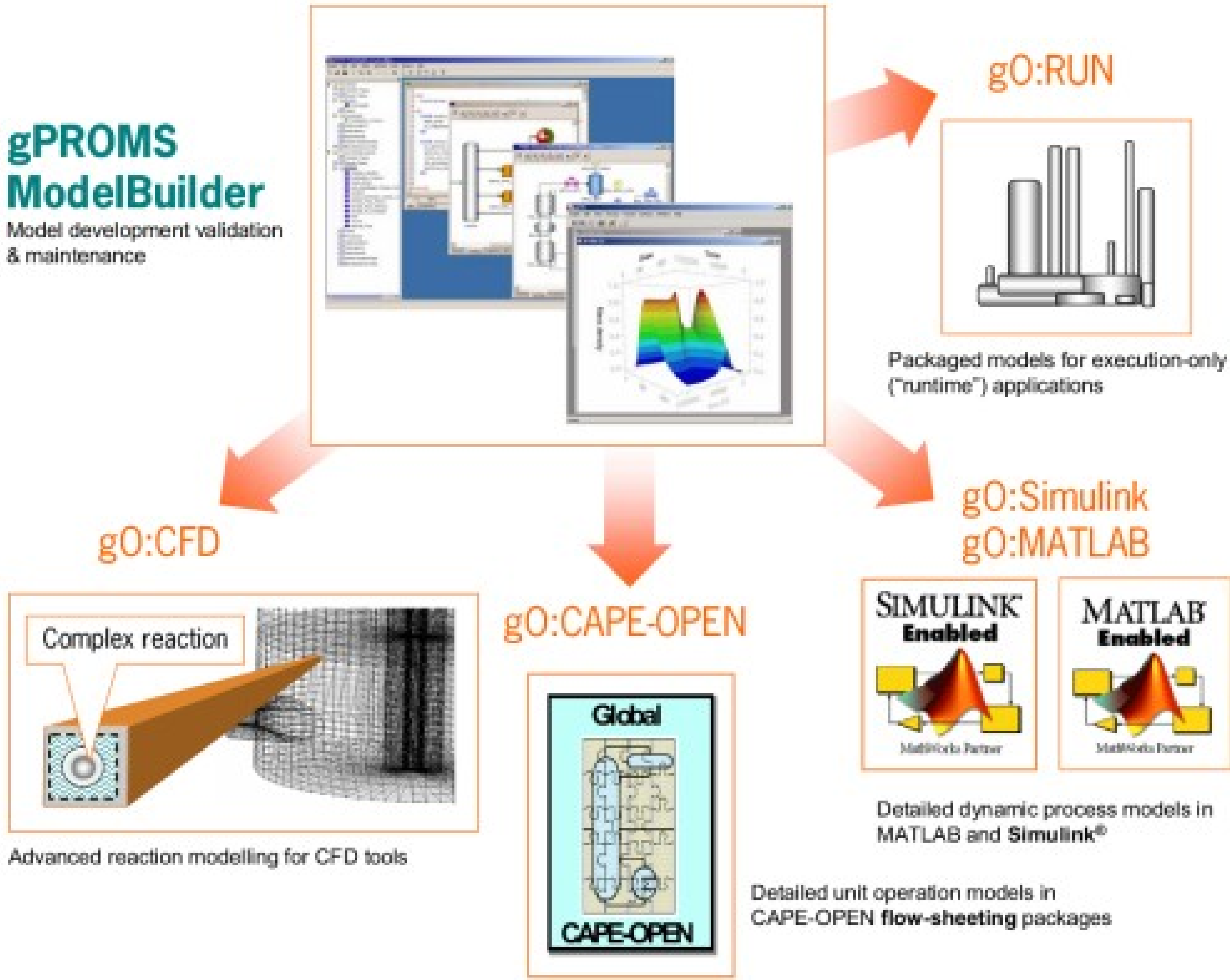
```
model mySimpleEqnSet "simple equation set"
  Real x(start=2, fixed=true);
  Real y(start=3, fixed=true);
equation
  der(x) = 2*x*y-3*x;
  der(y) = 5*y-7*x*y;
end mySimpleEqnSet;
```

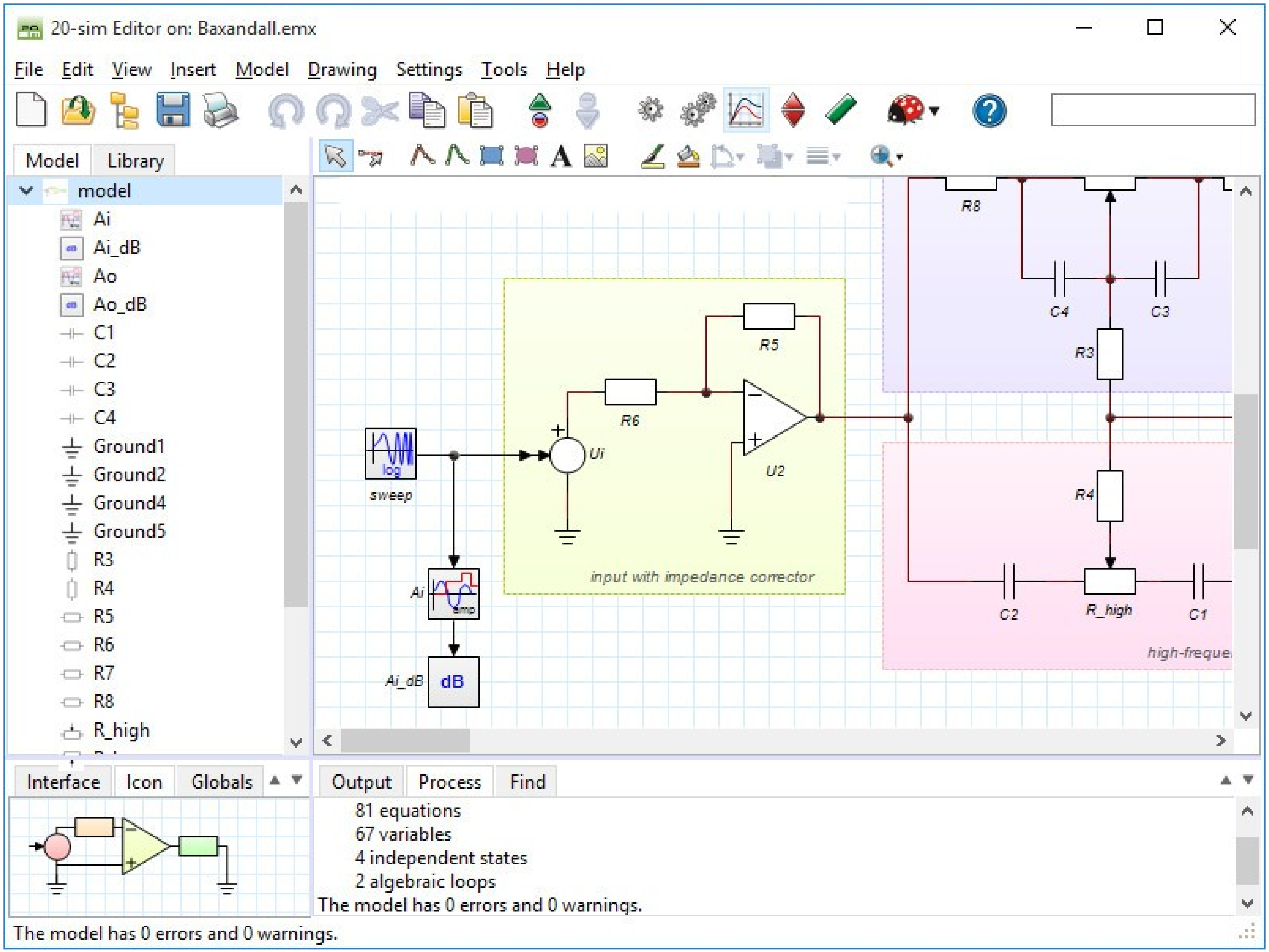

plot({x,y})
[done]
Zoom
Pan
Fit in View
Save
Print
Grid
Log X
Log Y
x
y
0
1
2
3
4
5
6
time

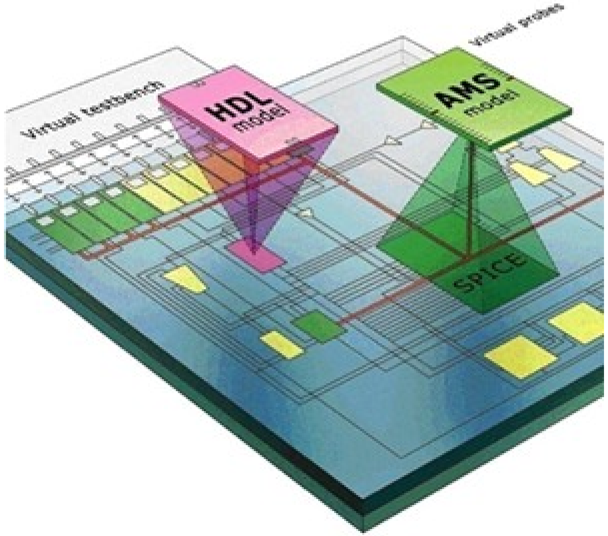

plotParametric(x,y)
[done]
Zoom
Pan
Fit in View
Save
Print
Grid
Log X
Log Y
y(x)
0
0.5
1
1.5
2
2.5
3
4
5
6

- Model exchange/re-use standard (Modelica Association)
- Modelica Standard Library (MSL)
- Object-oriented, hierarchical; semantics based on flattening
- Computationally a-causal modelling; semantics based on DAEs
- Originated in Hilding Elmquist's 1978 PhD thesis @ Lund
- Early 1990's: Modelica Design Team
  (started in SiE – Simulation in Europe ESPRIT Basic Research Working Group 8467)

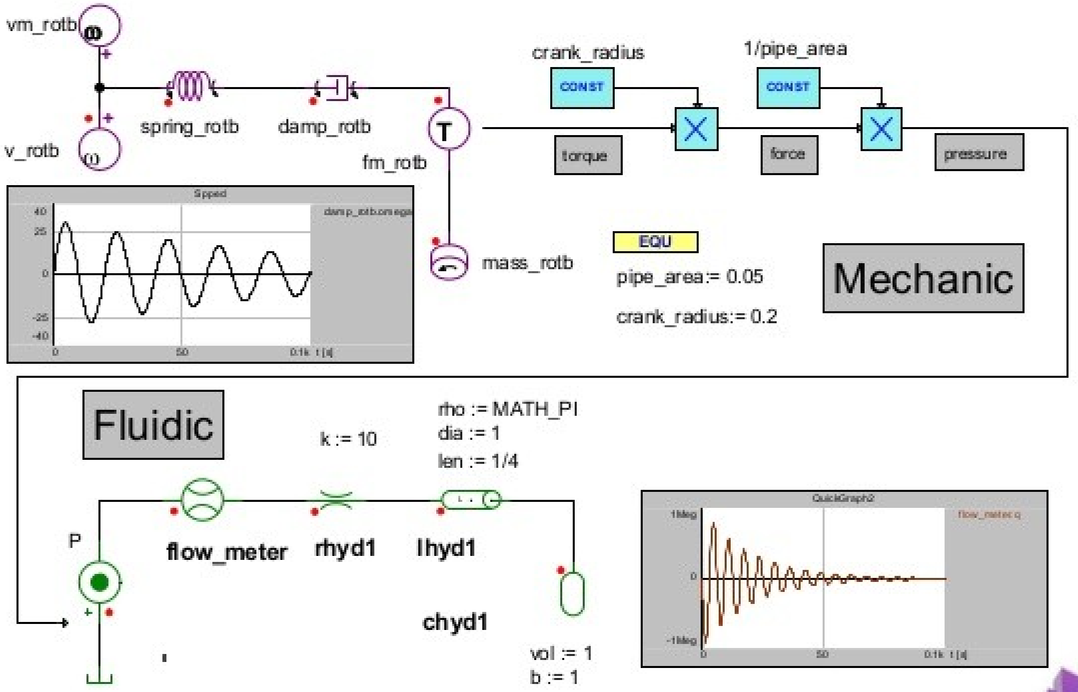
- hybrid (discrete-time/discrete-event) constructs (e.g., used to model network protocols based on TrueTime http://www.control.lth.se/truetime/)
- Limited support for Dynamic Structure models (i.e., no “agents”)
- Separate model from its (numerical) solution ...
- Generate Functional Mockup Interface (FMI) compliant simulation units
- Currently: many commercial and open (e.g., OpenModelica) tools
- Related: Mathworks Simscape, EcosimPro, NMF, gProms, ...


gPROMS
ModelBuilder
Model development validation
& maintenance
gO:RUN
Packaged models for execution-only
("runtime") applications
gO:CFD
Complex reaction
Advanced reaction modelling for CFD tools
gO:CAPE-OPEN
Global
CAPE-OPEN
Detailed unit operation models in
CAPE-OPEN flow-sheeting packages
gO:Simulink
gO:MATLAB
SIMULINK
Enabled
MATLAB
Enabled
Detailed dynamic process models in
MATLAB and Simulink®

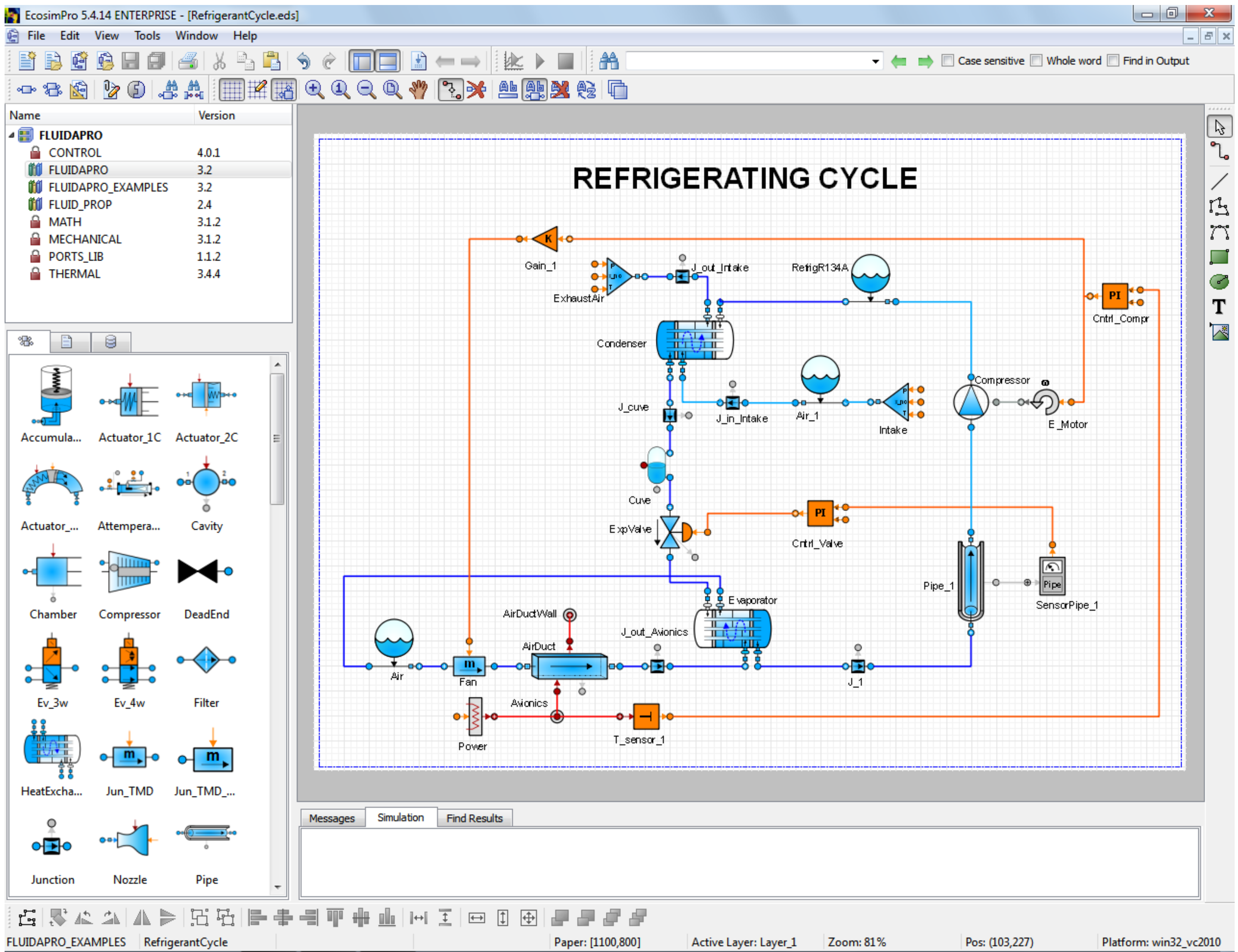

20-sim Editor on: Baxandall.emx
File Edit View Insert Model Drawing Settings Tools Help
Model Library
model
Ai
Ai_dB
Ao
Ao_dB
C1
C2
C3
C4
Ground1
Ground2
Ground4
Ground5
R3
R4
R5
R6
R7
R8
R_high
R8
C4
C3
R3
R5
R6
Ui
U2
sweep
R4
input with impedance corrector
C2
R_high
C1
Ai
Ai_dB
dB
Interface Icon Globals
Output Process Find
81 equations
67 variables
4 independent states
2 algebraic loops
The model has 0 errors and 0 warnings.
The model has 0 errors and 0 warnings.



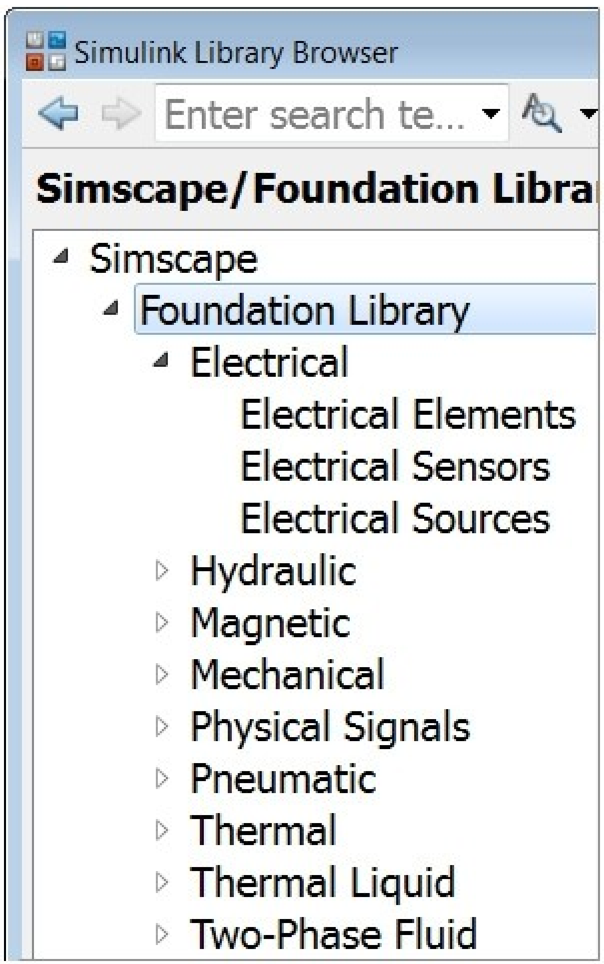
VHDL-AMS Multi Domain Design

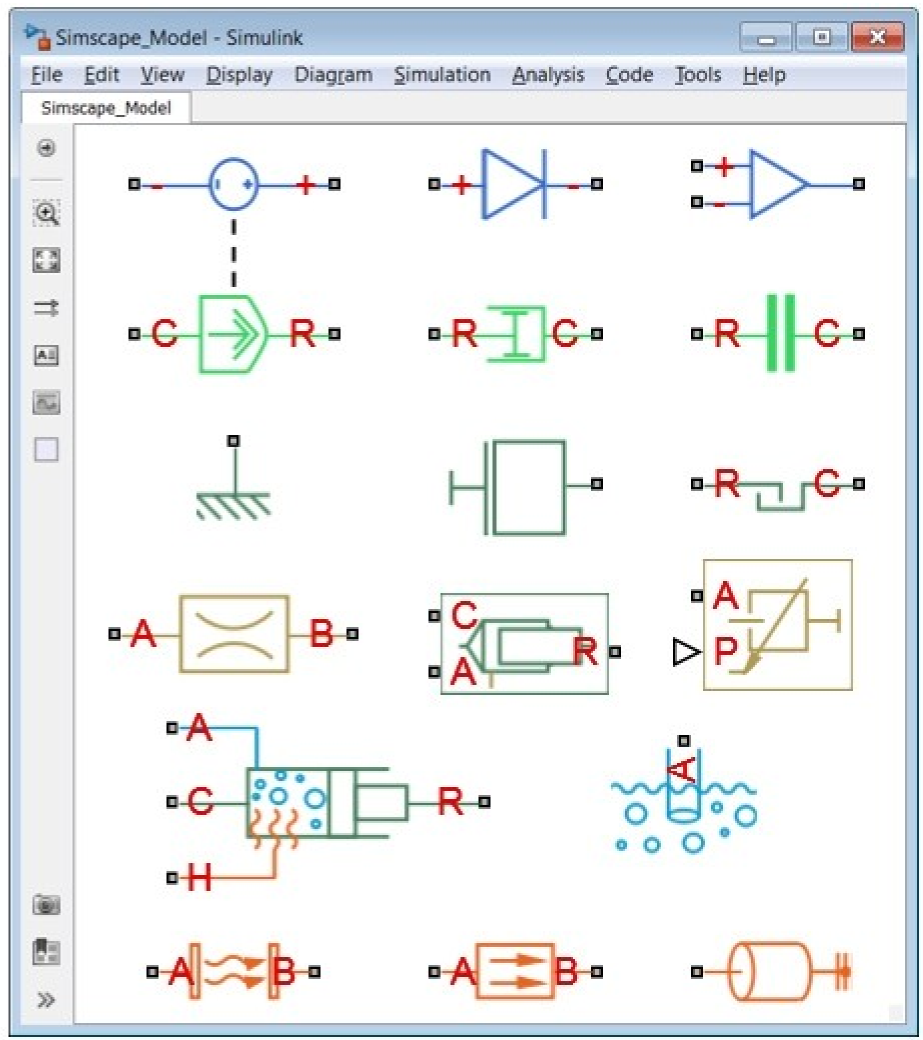


EcosimPro
Modelling and Simulation Software



Simscape



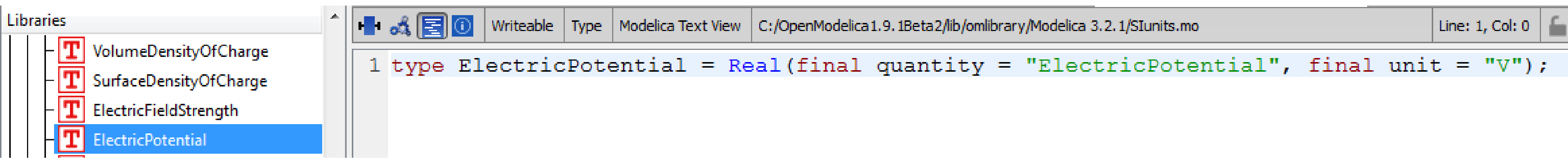


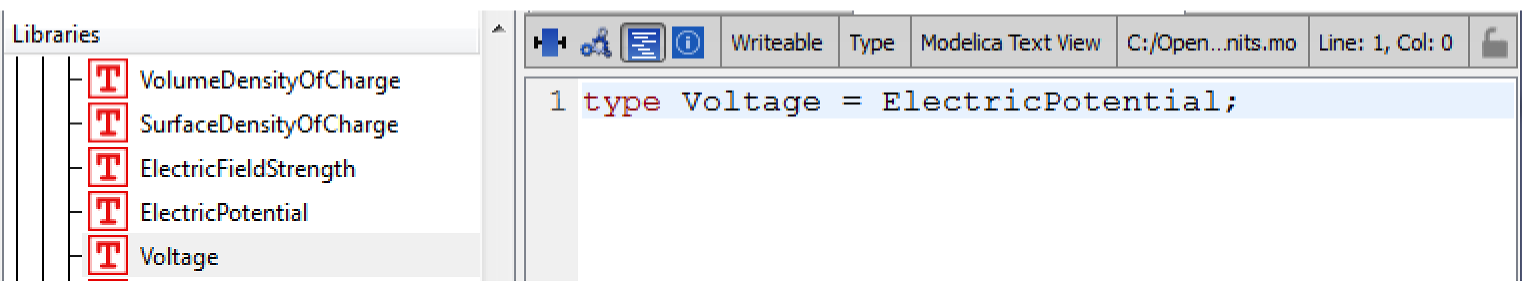
Steven Xu

Electrical Types

```
type Time = Real (final quantity="Time", final unit="s");
type ElectricPotential = Real (final quantity="ElectricPotential",
                               final unit="V");
type Voltage = ElectricPotential;
type ElectricCurrent = Real (final quantity="ElectricCurrent",
                             final unit="A");
type Current = ElectricCurrent;
```

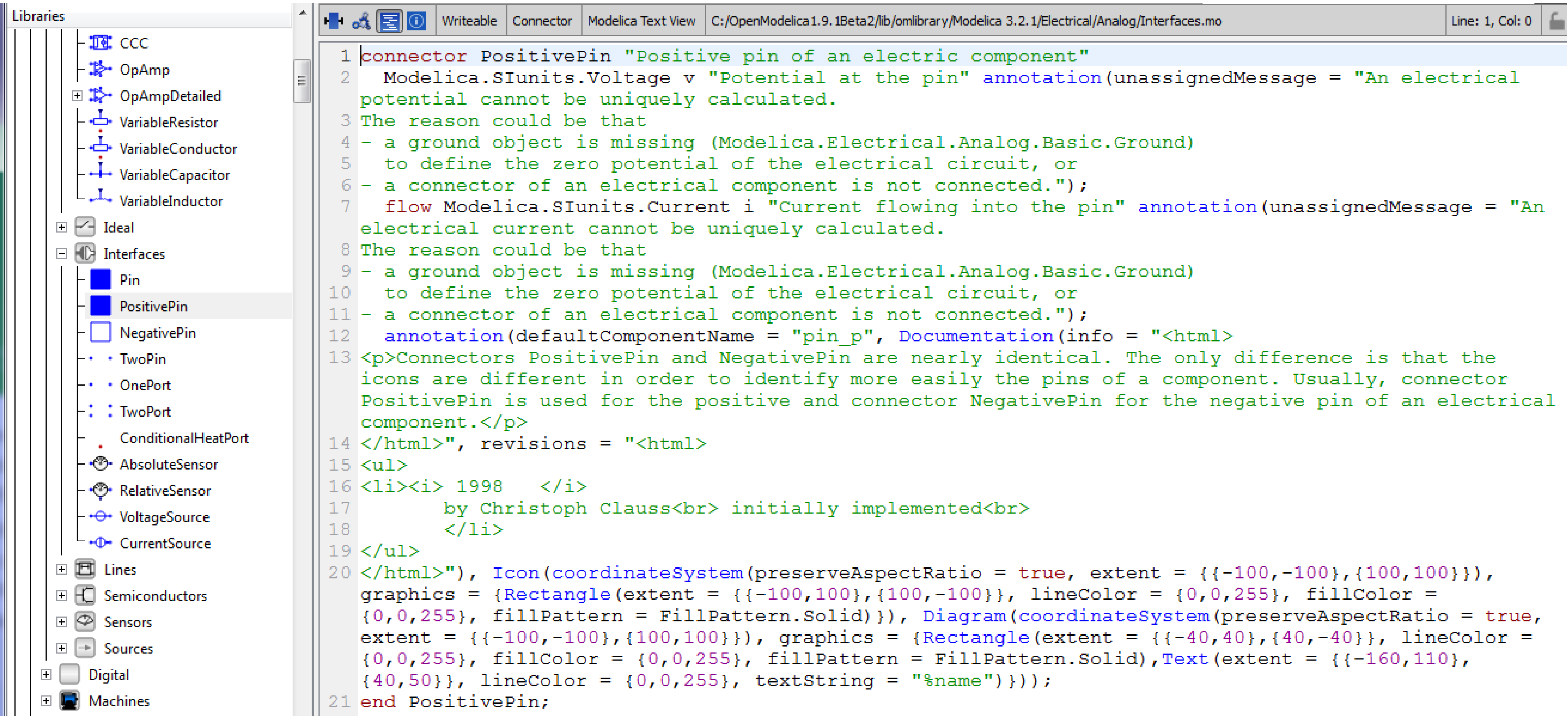
Beware: variables are **signals** (functions of **time**)!


Libraries
VolumeDensityOfCharge
SurfaceDensityOfCharge
ElectricFieldStrength
ElectricPotential
Writeable
Type
Modelica Text View
C:/OpenModelica1.9.1Beta2/lib/omlibrary/Modelica 3.2.1/SIunits.mo
Line: 1, Col: 0
1 type ElectricPotential = Real(final quantity = "ElectricPotential", final unit = "V");


Libraries
VolumeDensityOfCharge
SurfaceDensityOfCharge
ElectricFieldStrength
ElectricPotential
Voltage
Writeable
Type
Modelica Text View
C:/Open...nits.mo
Line: 1, Col: 0
1 type Voltage = ElectricPotential;

Electrical Pin Interface

```
connector PositivePin "Positive pin of an electric component"
       Voltage v "Potential at the pin";
  flow Current i "Current flowing into the pin";
end PositivePin;
```

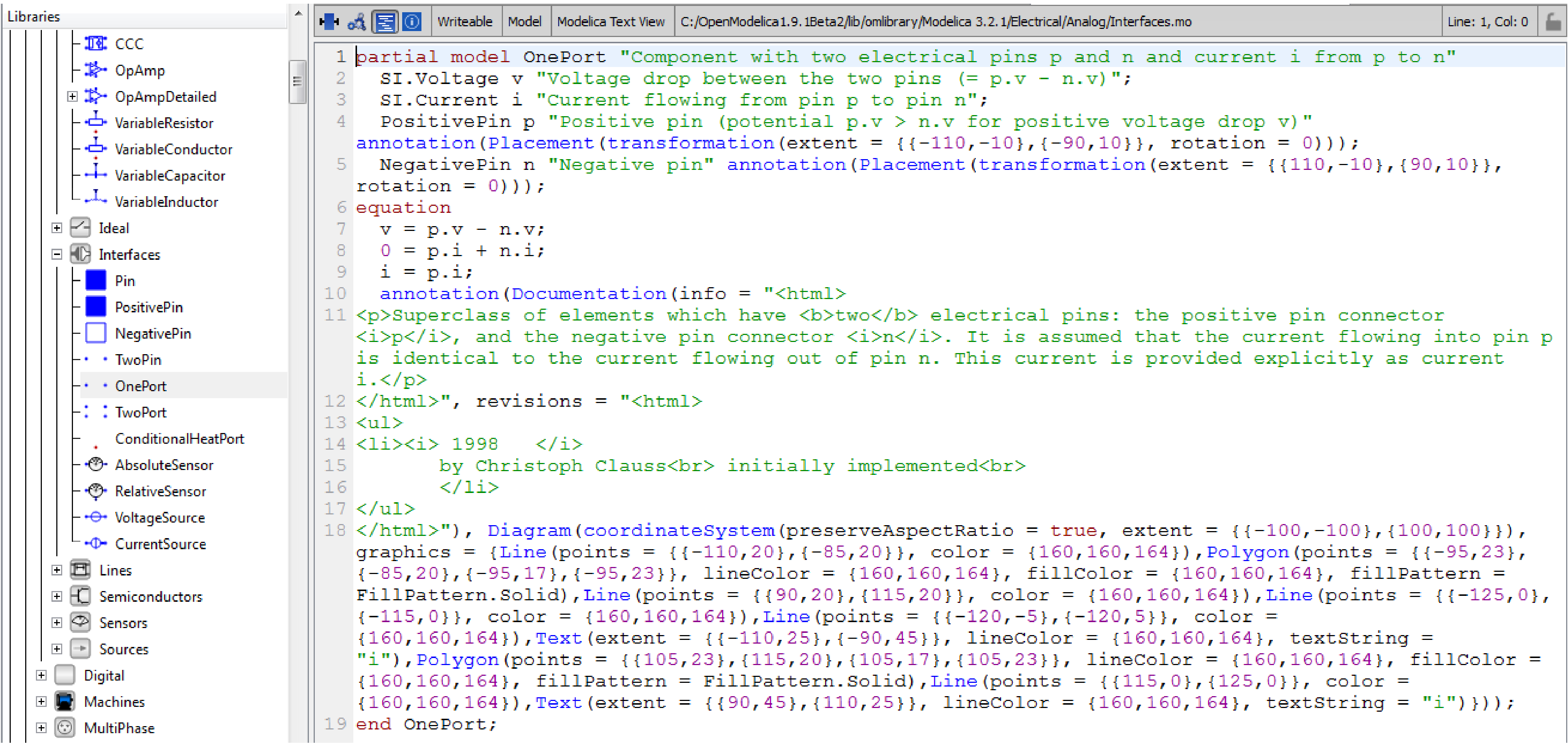

Libraries
CCC
OpAmp
OpAmpDetailed
VariableResistor
VariableConductor
VariableCapacitor
VariableInductor
Ideal
Interfaces
Pin
PositivePin
NegativePin
TwoPin
OnePort
TwoPort
ConditionalHeatPort
AbsoluteSensor
RelativeSensor
VoltageSource
CurrentSource
Lines
Semiconductors
Sensors
Sources
Digital
Machines
Writeable
Connector
Modelica Text View
C:/OpenModelica1.9.1Beta2/lib/omlibrary/Modelica 3.2.1/Electrical/Analog/Interfaces.mo
Line: 1, Col: 0
connector PositivePin "Positive pin of an electric component"
Modelica.SIunits.Voltage v "Potential at the pin" annotation(unassignedMessage = "An electrical potential cannot be uniquely calculated.
The reason could be that
- a ground object is missing (Modelica.Electrical.Analog.Basic.Ground)
to define the zero potential of the electrical circuit, or
- a connector of an electrical component is not connected.");
flow Modelica.SIunits.Current i "Current flowing into the pin" annotation(unassignedMessage = "An electrical current cannot be uniquely calculated.
The reason could be that
- a ground object is missing (Modelica.Electrical.Analog.Basic.Ground)
to define the zero potential of the electrical circuit, or
- a connector of an electrical component is not connected.");
annotation(defaultComponentName = "pin_p", Documentation(info = "<html>
<p>Connectors PositivePin and NegativePin are nearly identical. The only difference is that the icons are different in order to identify more easily the pins of a component. Usually, connector PositivePin is used for the positive and connector NegativePin for the negative pin of an electrical component.</p>
</html>", revisions = "<html>

<i> 1998 </i>
by Christoph Clauss
 initially implemented

</html>"), Icon(coordinateSystem(preserveAspectRatio = true, extent = {{-100,-100},{100,100}}), graphics = {Rectangle(extent = {{-100,100},{100,-100}}, lineColor = {0,0,255}, fillColor = {0,0,255}, fillPattern = FillPattern.Solid)}), Diagram(coordinateSystem(preserveAspectRatio = true, extent = {{-100,-100},{100,100}}), graphics = {Rectangle(extent = {{-40,40},{40,-40}}, lineColor = {0,0,255}, fillColor = {0,0,255}, fillPattern = FillPattern.Solid),Text(extent = {{-160,110},{40,50}}, lineColor = {0,0,255}, textString = "%name")}));
end PositivePin;

Electrical Port

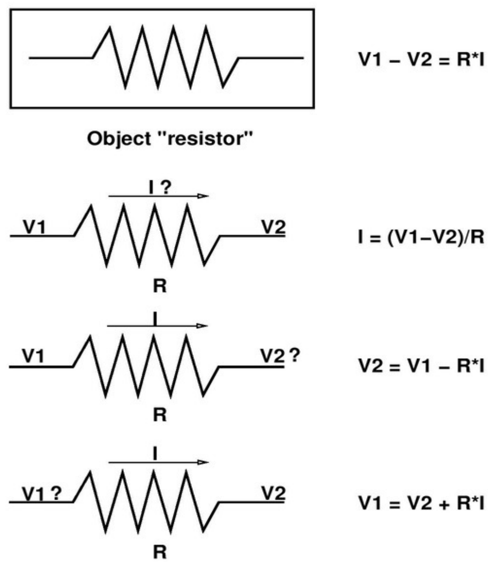
```
partial model OnePort
  "Component with two electrical pins p and n
   and current i from p to n"
  Voltage v "Voltage drop between the two pins (= p.v - n.v)";
  Current i "Current flowing from pin p to pin n";
  PositivePin p;
  NegativePin n;
equation
  v = p.v - n.v;
  0 = p.i + n.i;
  i = p.i;
end OnePort;
```


Libraries
CCC
OpAmp
OpAmpDetailed
VariableResistor
VariableConductor
VariableCapacitor
VariableInductor
Ideal
Interfaces
Pin
PositivePin
NegativePin
TwoPin
OnePort
TwoPort
ConditionalHeatPort
AbsoluteSensor
RelativeSensor
VoltageSource
CurrentSource
Lines
Semiconductors
Sensors
Sources
Digital
Machines
MultiPhase
Writeable
Model
Modelica Text View
C:/OpenModelica1.9.1Beta2/lib/omlibrary/Modelica 3.2.1/Electrical/Analog/Interfaces.mo
Line: 1, Col: 0
partial model OnePort "Component with two electrical pins p and n and current i from p to n"
SI.Voltage v "Voltage drop between the two pins (= p.v - n.v)";
SI.Current i "Current flowing from pin p to pin n";
PositivePin p "Positive pin (potential p.v > n.v for positive voltage drop v)" annotation(Placement(transformation(extent = {{-110,-10},{-90,10}}, rotation = 0)));
NegativePin n "Negative pin" annotation(Placement(transformation(extent = {{110,-10},{90,10}}, rotation = 0)));
equation
v = p.v - n.v;
0 = p.i + n.i;
i = p.i;
annotation(Documentation(info = "<html>
<p>Superclass of elements which have two electrical pins: the positive pin connector <i>p</i>, and the negative pin connector <i>n</i>. It is assumed that the current flowing into pin p is identical to the current flowing out of pin n. This current is provided explicitly as current i.</p>
</html>", revisions = "<html>

<i> 1998 </i>
by Christoph Clauss
 initially implemented

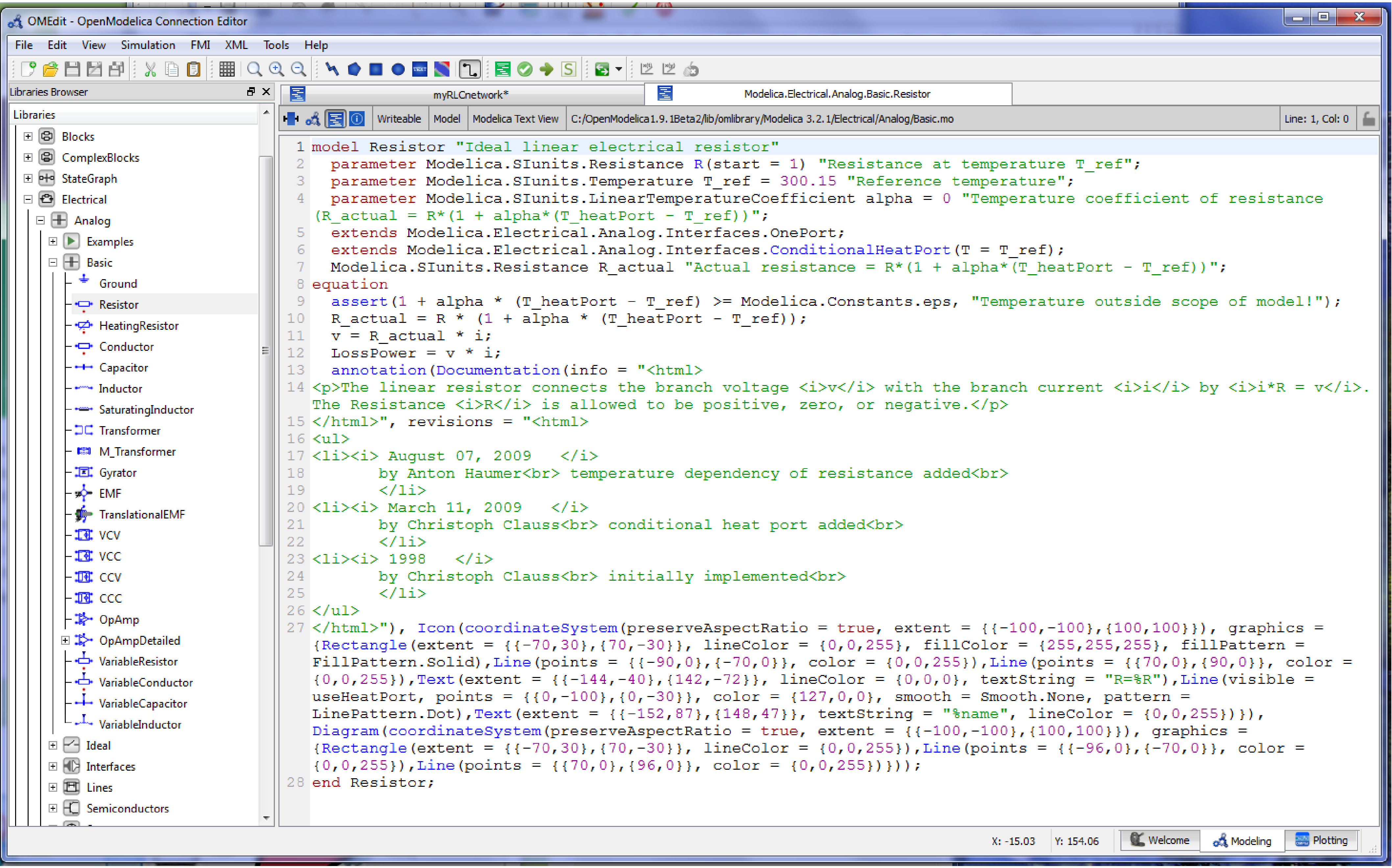
</html>"), Diagram(coordinateSystem(preserveAspectRatio = true, extent = {{-100,-100},{100,100}}), graphics = {Line(points = {{-110,20},{-85,20}}, color = {160,160,164}),Polygon(points = {{-95,23},{-85,20},{-95,17},{-95,23}}, lineColor = {160,160,164}, fillColor = {160,160,164}, fillPattern = FillPattern.Solid),Line(points = {{90,20},{115,20}}, color = {160,160,164}),Line(points = {{-125,0},{-115,0}}, color = {160,160,164}),Line(points = {{-120,-5},{-120,5}}, color = {160,160,164}),Text(extent = {{-110,25},{-90,45}}, lineColor = {160,160,164}, textString = "i"),Polygon(points = {{105,23},{115,20},{105,17},{105,23}}, lineColor = {160,160,164}, fillColor = {160,160,164}, fillPattern = FillPattern.Solid),Line(points = {{115,0},{125,0}}, color = {160,160,164}),Text(extent = {{90,45},{110,25}}, lineColor = {160,160,164}, textString = "i")}));
end OnePort;

Object-oriented re-use and causality



Electrical Resistor

```
model Resistor "Ideal linear electrical resistor"
  extends OnePort;
  parameter Resistance R=1 "Resistance";
  equation
    R*i = v;
end Resistor;
```

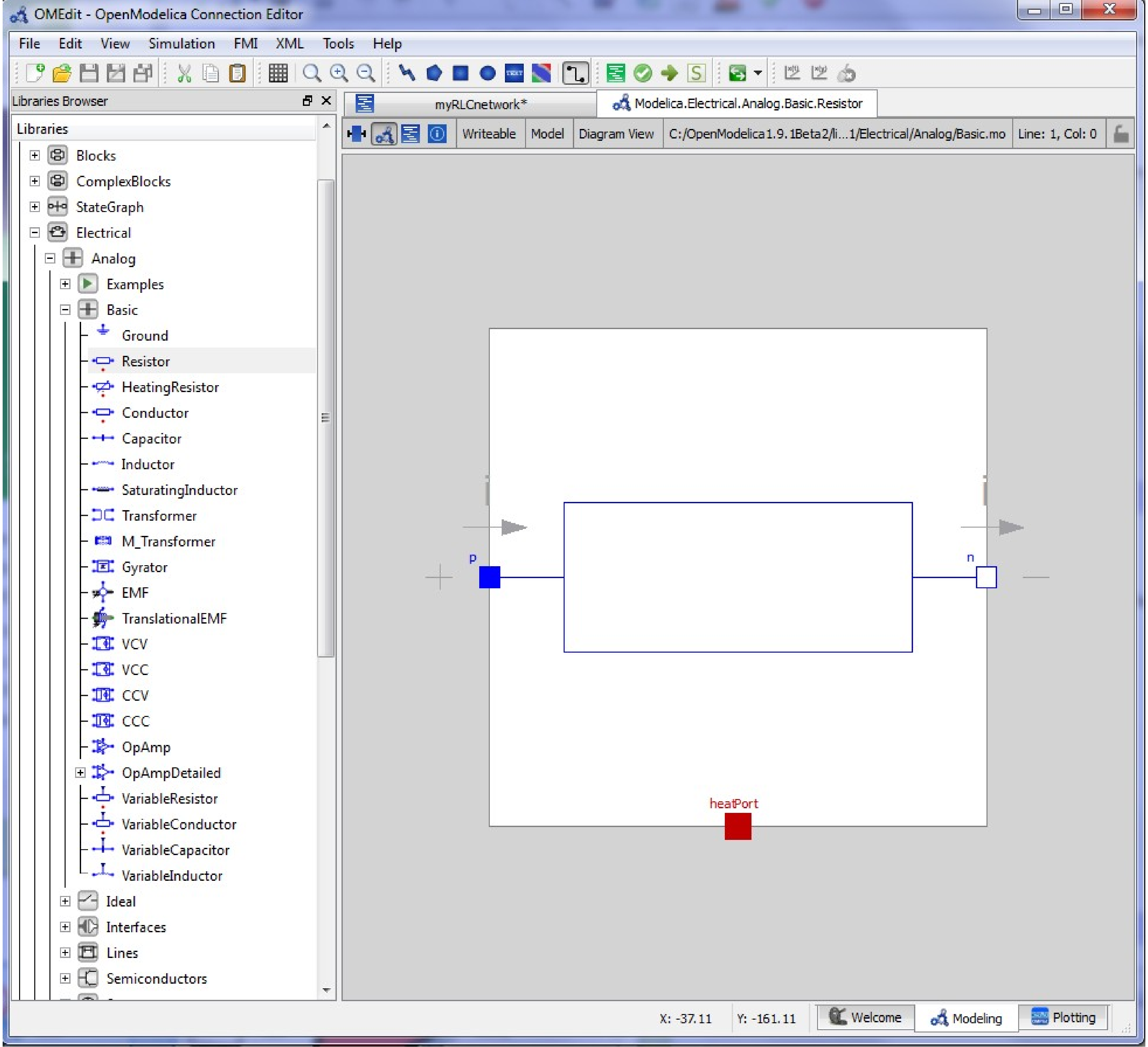

OMEdit - OpenModelica Connection Editor
File Edit View Simulation FMI XML Tools Help
Libraries Browser
Libraries
Blocks
ComplexBlocks
StateGraph
Electrical
Analog
Examples
Basic
Ground
Resistor
HeatingResistor
Conductor
Capacitor
Inductor
SaturatingInductor
Transformer
M_Transformer
Gyrator
EMF
TranslationalEMF
VCV
VCC
CCV
CCC
OpAmp
OpAmpDetailed
VariableResistor
VariableConductor
VariableCapacitor
VariableInductor
Ideal
Interfaces
Lines
Semiconductors
myRLCnetwork*
Modelica.Electrical.Analog.Basic.Resistor
Writeable
Model
Modelica Text View
C:/OpenModelica1.9.1Beta2/lib/omlibrary/Modelica 3.2.1/Electrical/Analog/Basic.mo
Line: 1, Col: 0
model Resistor "Ideal linear electrical resistor"
parameter Modelica.SIunits.Resistance R(start = 1) "Resistance at temperature T_ref";
parameter Modelica.SIunits.Temperature T_ref = 300.15 "Reference temperature";
parameter Modelica.SIunits.LinearTemperatureCoefficient alpha = 0 "Temperature coefficient of resistance (R_actual = R*(1 + alpha*(T_heatPort - T_ref))";
extends Modelica.Electrical.Analog.Interfaces.OnePort;
extends Modelica.Electrical.Analog.Interfaces.ConditionalHeatPort(T = T_ref);
Modelica.SIunits.Resistance R_actual "Actual resistance = R*(1 + alpha*(T_heatPort - T_ref))";
equation
assert(1 + alpha * (T_heatPort - T_ref) >= Modelica.Constants.eps, "Temperature outside scope of model!");
R_actual = R * (1 + alpha * (T_heatPort - T_ref));
v = R_actual * i;
LossPower = v * i;
annotation(Documentation(info = "<html>
<p>The linear resistor connects the branch voltage <i>v</i> with the branch current <i>i</i> by <i>i*R = v</i>. The Resistance <i>R</i> is allowed to be positive, zero, or negative.</p>
</html>", revisions = "<html>

<i> August 07, 2009 </i>
by Anton Haumer
 temperature dependency of resistance added

<i> March 11, 2009 </i>
by Christoph Clauss
 conditional heat port added

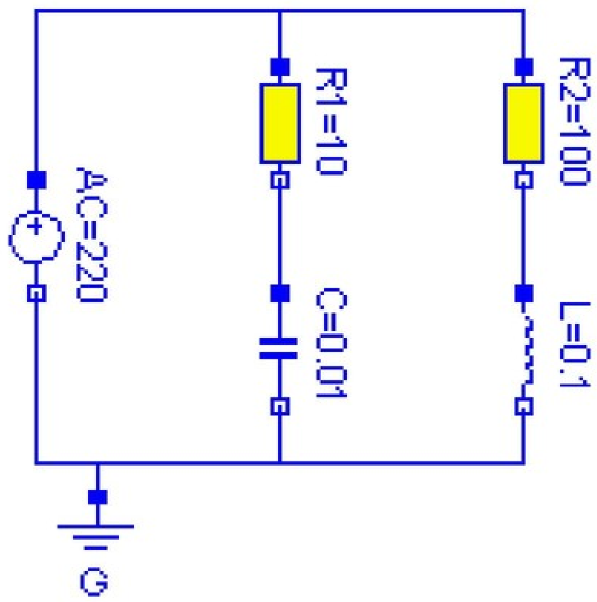
<i> 1998 </i>
by Christoph Clauss
 initially implemented

</html>"), Icon(coordinateSystem(preserveAspectRatio = true, extent = {{-100,-100},{100,100}}), graphics = {Rectangle(extent = {{-70,30},{70,-30}}, lineColor = {0,0,255}, fillColor = {255,255,255}, fillPattern = FillPattern.Solid),Line(points = {{-90,0},{-70,0}}, color = {0,0,255}),Line(points = {{70,0},{90,0}}, color = {0,0,255}),Text(extent = {{-144,-40},{142,-72}}, lineColor = {0,0,0}, textString = "R=%R"),Line(visible = useHeatPort, points = {{0,-100},{0,-30}}, color = {127,0,0}, smooth = Smooth.None, pattern = LinePattern.Dot),Text(extent = {{-152,87},{148,47}}, textString = "%name", lineColor = {0,0,255})}), Diagram(coordinateSystem(preserveAspectRatio = true, extent = {{-100,-100},{100,100}}), graphics = {Rectangle(extent = {{-70,30},{70,-30}}, lineColor = {0,0,255}),Line(points = {{-96,0},{-70,0}}, color = {0,0,255}),Line(points = {{70,0},{96,0}}, color = {0,0,255})}));
end Resistor;
X: -15.03
Y: 154.06
Welcome
Modeling
Plotting


OMEdit - OpenModelica Connection Editor
File Edit View Simulation FMI XML Tools Help
Libraries Browser
Libraries
Blocks
ComplexBlocks
StateGraph
Electrical
Analog
Examples
Basic
Ground
Resistor
HeatingResistor
Conductor
Capacitor
Inductor
SaturatingInductor
Transformer
M_Transformer
Gyrator
EMF
TranslationalEMF
VCV
VCC
CCV
CCC
OpAmp
OpAmpDetailed
VariableResistor
VariableConductor
VariableCapacitor
VariableInductor
Ideal
Interfaces
Lines
Semiconductors
myRLCnetwork*
Modelica.Electrical.Analog.Basic.Resistor
Writeable
Model
Diagram View
C:/OpenModelica1.9.1Beta2/li...1/Electrical/Analog/Basic.mo
Line: 1, Col: 0
p
n
heatPort
X: -37.11
Y: -161.11
Welcome
Modeling
Plotting

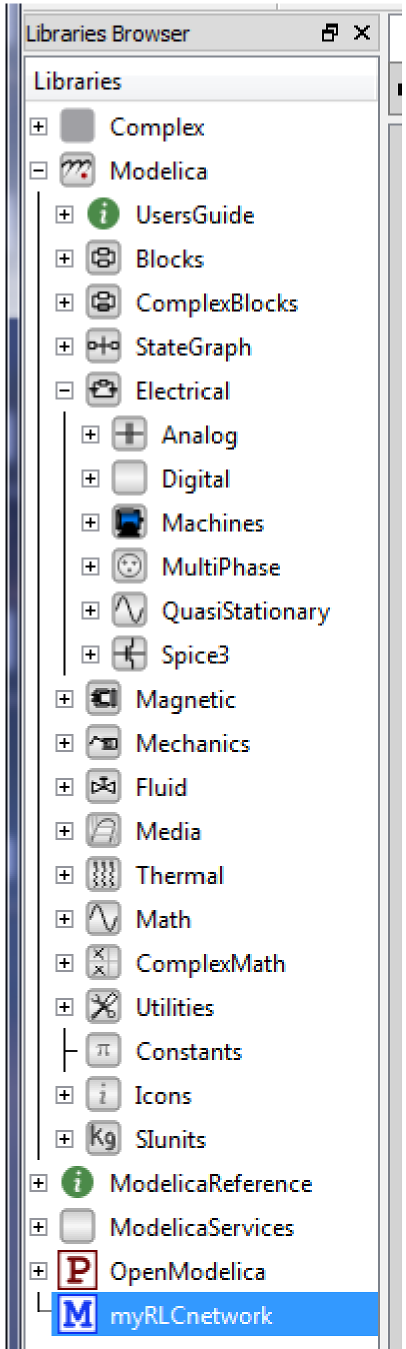
The circuit

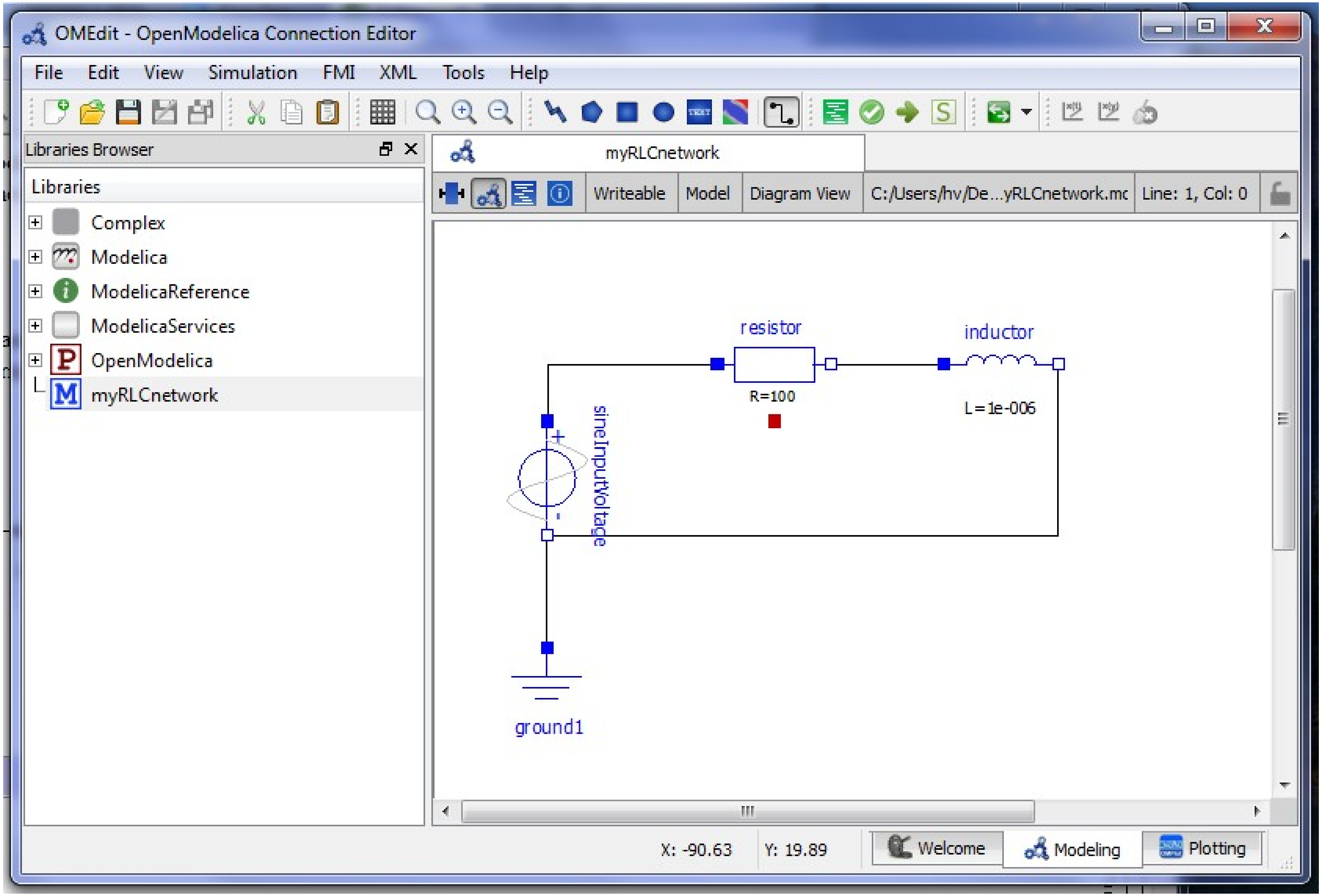
```
model circuit
 Resistor  R1(R=10);
 Capacitor C(C=0.01);
 Resistor  R2(R=100);
 Inductor  L(L=0.1);
 VsourceAC AC;
 Ground    G;
equation
 connect(AC.p, R1.p);
 connect(R1.n, C.p);
 connect(C.n, AC.n);
 connect(R1.p, R2.p);
 connect(R2.n, L.p);
 connect(L.n, C.n);
 connect(AC.n, G.p);
end circuit;
```

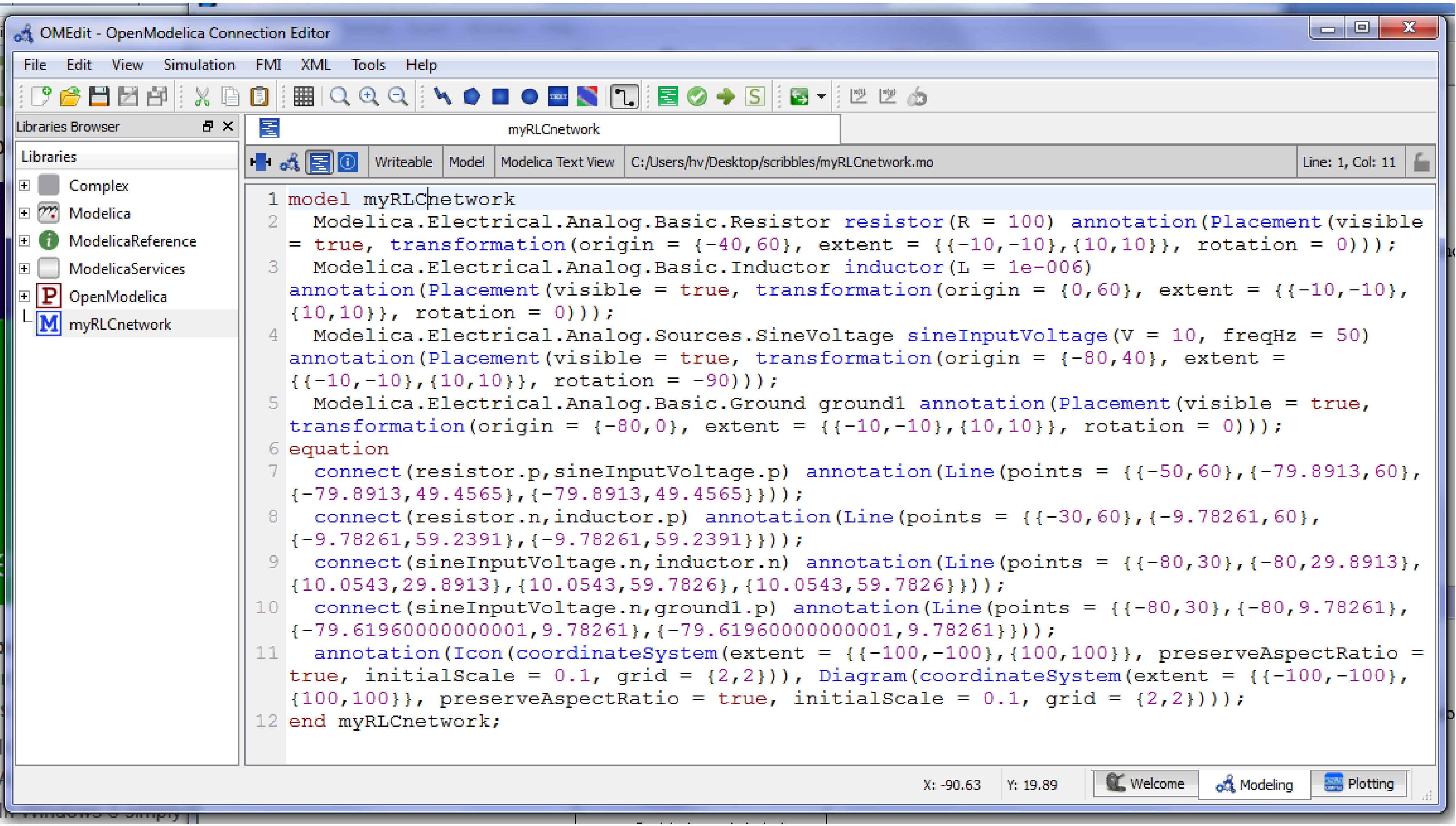


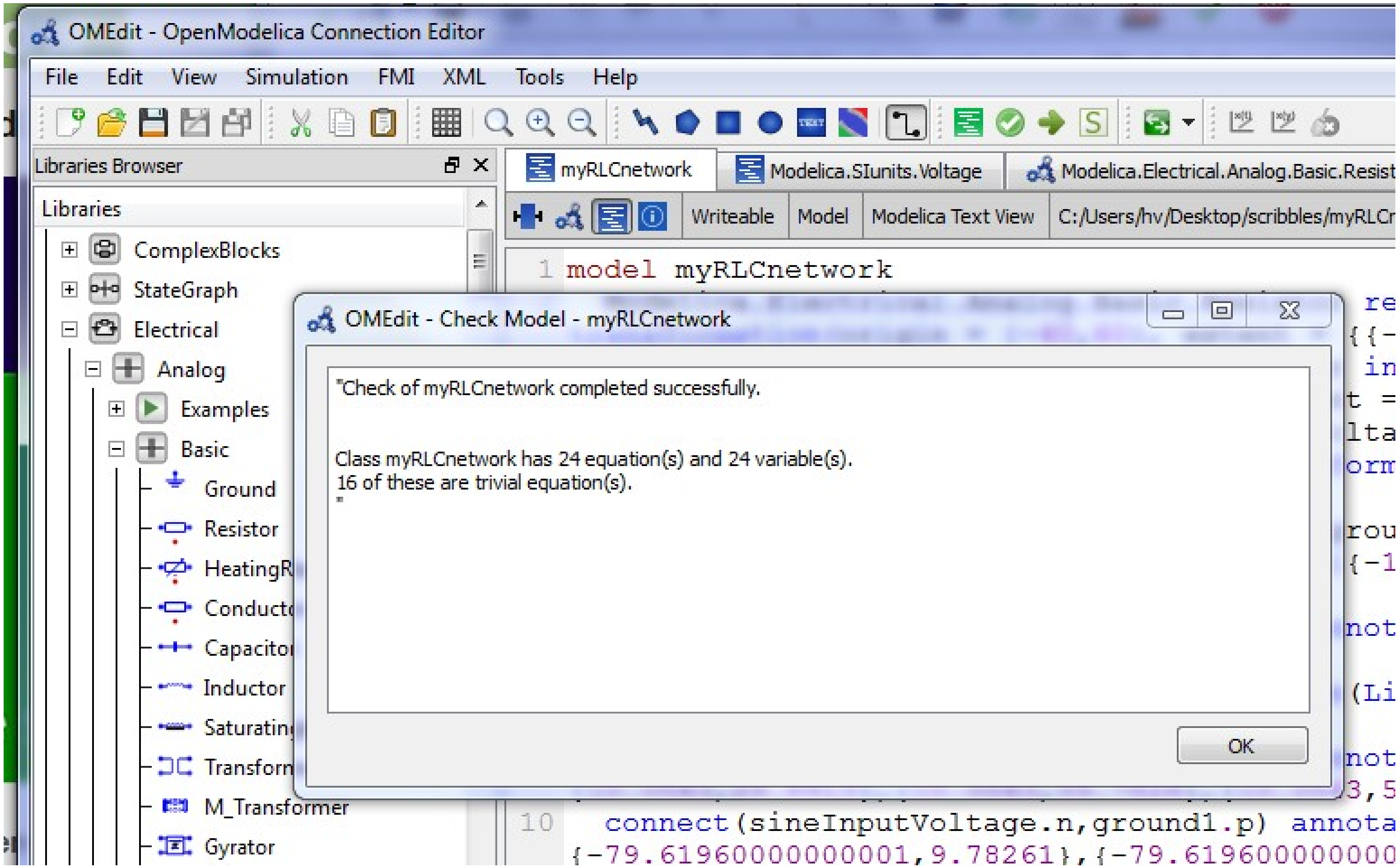
Meaning: set of Differential Algebraic Equations (DAEs) obtained by

1.a. expanding inheritance
1.b. instantiation
2. flattening hierarchy, construct unique names
3. expanding connect() into equations (across vs. flow)


Libraries Browser
Libraries
Complex
Modelica
UsersGuide
Blocks
ComplexBlocks
StateGraph
Electrical
Analog
Digital
Machines
MultiPhase
QuasiStationary
Spice3
Magnetic
Mechanics
Fluid
Media
Thermal
Math
ComplexMath
Utilities
Constants
Icons
SIunits
ModelicaReference
ModelicaServices
OpenModelica
myRLCnetwork


OMEdit - OpenModelica Connection Editor
File
Edit
View
Simulation
FMI
XML
Tools
Help
Libraries Browser
Libraries
Complex
Modelica
ModelicaReference
ModelicaServices
OpenModelica
myRLCnetwork
myRLCnetwork
Writeable
Model
Diagram View
C:/Users/hv/De...yRLCnetwork.mc
Line: 1, Col: 0
resistor
inductor
R=100
L=1e-006
sineInputVoltage
ground1
X: -90.63
Y: 19.89
Welcome
Modeling
Plotting


OMEdit - OpenModelica Connection Editor
File Edit View Simulation FMI XML Tools Help
Libraries Browser
Libraries
Complex
Modelica
ModelicaReference
ModelicaServices
OpenModelica
myRLCnetwork
myRLCnetwork
Writeable Model Modelica Text View C:/Users/hv/Desktop/scribbles/myRLCnetwork.mo Line: 1, Col: 11
model myRLCnetwork
Modelica.Electrical.Analog.Basic.Resistor resistor(R = 100) annotation(Placement(visible = true, transformation(origin = {-40,60}, extent = {{-10,-10},{10,10}}, rotation = 0)));
Modelica.Electrical.Analog.Basic.Inductor inductor(L = 1e-006) annotation(Placement(visible = true, transformation(origin = {0,60}, extent = {{-10,-10},{10,10}}, rotation = 0)));
Modelica.Electrical.Analog.Sources.SineVoltage sineInputVoltage(V = 10, freqHz = 50) annotation(Placement(visible = true, transformation(origin = {-80,40}, extent = {{-10,-10},{10,10}}, rotation = -90)));
Modelica.Electrical.Analog.Basic.Ground ground1 annotation(Placement(visible = true, transformation(origin = {-80,0}, extent = {{-10,-10},{10,10}}, rotation = 0)));
equation
connect(resistor.p,sineInputVoltage.p) annotation(Line(points = {{-50,60},{-79.8913,60},{-79.8913,49.4565},{-79.8913,49.4565}}));
connect(resistor.n,inductor.p) annotation(Line(points = {{-30,60},{-9.78261,60},{-9.78261,59.2391},{-9.78261,59.2391}}));
connect(sineInputVoltage.n,inductor.n) annotation(Line(points = {{-80,30},{-80,29.8913},{10.0543,29.8913},{10.0543,59.7826},{10.0543,59.7826}}));
connect(sineInputVoltage.n,ground1.p) annotation(Line(points = {{-80,30},{-80,9.78261},{-79.61960000000001,9.78261},{-79.61960000000001,9.78261}}));
annotation(Icon(coordinateSystem(extent = {{-100,-100},{100,100}}, preserveAspectRatio = true, initialScale = 0.1, grid = {2,2})), Diagram(coordinateSystem(extent = {{-100,-100},{100,100}}, preserveAspectRatio = true, initialScale = 0.1, grid = {2,2})));
end myRLCnetwork;
X: -90.63 Y: 19.89
Welcome Modeling Plotting


OMEdit - OpenModelica Connection Editor
File
Edit
View
Simulation
FMI
XML
Tools
Help
Libraries Browser
Libraries
ComplexBlocks
StateGraph
Electrical
Analog
Examples
Basic
Ground
Resistor
M_Transformer
Gyrator
myRLCnetwork
Modelica.SIunits.Voltage
Writeable
Model
Modelica Text View
model myRLCnetwork
connect(sineInputVoltage.n,ground1.p) annota
OMEdit - Check Model - myRLCnetwork
"Check of myRLCnetwork completed successfully.
Class myRLCnetwork has 24 equation(s) and 24 variable(s).
16 of these are trivial equation(s).
"
OK

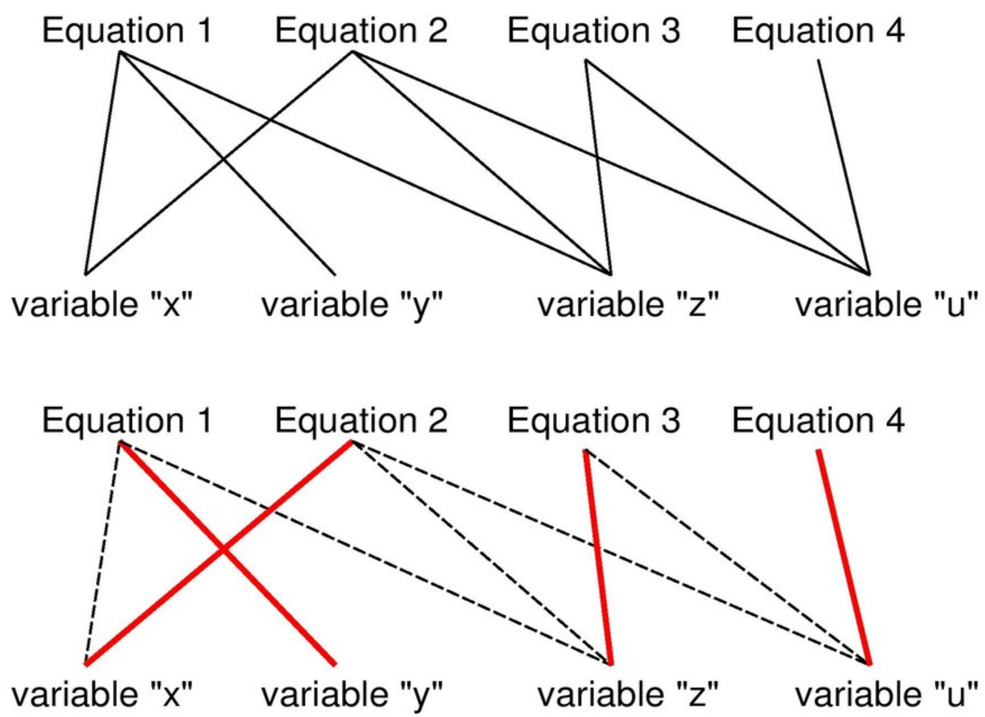
```
class myRLCnetwork
  Real resistor.v(quantity = "ElectricPotential", unit = "V") "Voltage drop between the two pins (= p.v - n.v)";
  Real resistor.i(quantity = "ElectricCurrent", unit = "A") "Current flowing from pin p to pin n";
  Real resistor.p.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real resistor.p.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  Real resistor.n.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real resistor.n.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  parameter Boolean resistor.useHeatPort = false "=true, if HeatPort is enabled";
  Real resistor.LossPower(quantity = "Power", unit = "W") "Loss power leaving component via HeatPort";
  Real resistor.T_heatPort(quantity = "ThermodynamicTemperature", unit = "K", displayUnit = "degC", min = 0.0, start = 288.15, nominal =
300.0) "Temperature of HeatPort";
  parameter Real resistor.R(quantity = "Resistance", unit = "Ohm", start = 1.0) = 100.0 "Resistance at temperature T_ref";
  parameter Real resistor.T_ref(quantity = "ThermodynamicTemperature", unit = "K", displayUnit = "degC", min = 0.0, start = 288.15,
nominal = 300.0) = 300.15 "Reference temperature";
  parameter Real resistor.alpha(quantity = "LinearTemperatureCoefficient", unit = "1/K") = 0.0 "Temperature coefficient of resistance
(R_actual = R*(1 + alpha*(T_heatPort - T_ref))";
  Real resistor.R_actual(quantity = "Resistance", unit = "Ohm") "Actual resistance = R*(1 + alpha*(T_heatPort - T_ref))";
  parameter Real resistor.T(quantity = "ThermodynamicTemperature", unit = "K", displayUnit = "degC", min = 0.0, start = 288.15, nominal =
300.0) = resistor.T_ref "Fixed device temperature if useHeatPort = false";
  Real inductor.v(quantity = "ElectricPotential", unit = "V") "Voltage drop between the two pins (= p.v - n.v)";
  Real inductor.i(quantity = "ElectricCurrent", unit = "A", start = 0.0) "Current flowing from pin p to pin n";
  Real inductor.p.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real inductor.p.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  Real inductor.n.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real inductor.n.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  parameter Real inductor.L(quantity = "Inductance", unit = "H", start = 1.0) = 1e-006 "Inductance";
  Real sineInputVoltage.v(quantity = "ElectricPotential", unit = "V") "Voltage drop between the two pins (= p.v - n.v)";
  Real sineInputVoltage.i(quantity = "ElectricCurrent", unit = "A") "Current flowing from pin p to pin n";
  Real sineInputVoltage.p.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real sineInputVoltage.p.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  Real sineInputVoltage.n.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real sineInputVoltage.n.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
  parameter Real sineInputVoltage.offset(quantity = "ElectricPotential", unit = "V") = 0.0 "Voltage offset";
  parameter Real sineInputVoltage.startTime(quantity = "Time", unit = "s") = 0.0 "Time offset";
  parameter Real sineInputVoltage.V(quantity = "ElectricPotential", unit = "V", start = 1.0) = 10.0 "Amplitude of sine wave";
  parameter Real sineInputVoltage.phase(quantity = "Angle", unit = "rad", displayUnit = "deg") = 0.0 "Phase of sine wave";
  parameter Real sineInputVoltage.freqHz(quantity = "Frequency", unit = "Hz", start = 1.0) = 50.0 "Frequency of sine wave";
  output Real sineInputVoltage.signalSource.y "Connector of Real output signal";
  parameter Real sineInputVoltage.signalSource.amplitude = sineInputVoltage.V "Amplitude of sine wave";
  parameter Real sineInputVoltage.signalSource.freqHz(quantity = "Frequency", unit = "Hz", start = 1.0) = sineInputVoltage.freqHz
"Frequency of sine wave";
  parameter Real sineInputVoltage.signalSource.phase(quantity = "Angle", unit = "rad", displayUnit = "deg") = sineInputVoltage.phase
"Phase of sine wave";
  parameter Real sineInputVoltage.signalSource.offset = sineInputVoltage.offset "Offset of output signal";
  parameter Real sineInputVoltage.signalSource.startTime(quantity = "Time", unit = "s") = sineInputVoltage.startTime "Output = offset for
time < startTime";
  protected constant Real sineInputVoltage.signalSource.pi = 3.141592653589793;
  Real ground1.p.v(quantity = "ElectricPotential", unit = "V") "Potential at the pin";
  Real ground1.p.i(quantity = "ElectricCurrent", unit = "A") "Current flowing into the pin";
```

```
equation
  assert(1.0 + resistor.alpha * (resistor.T_heatPort - resistor.T_ref) >= 1e-015,"Temperature outside scope of model!");
  resistor.R_actual = resistor.R * (1.0 + resistor.alpha * (resistor.T_heatPort - resistor.T_ref));
  resistor.v = resistor.R_actual * resistor.i;
  resistor.LossPower = resistor.v * resistor.i;
  resistor.v = resistor.p.v - resistor.n.v;
  0.0 = resistor.p.i + resistor.n.i;
  resistor.i = resistor.p.i;
  resistor.T_heatPort = resistor.T;
  inductor.L * der(inductor.i) = inductor.v;
  inductor.v = inductor.p.v - inductor.n.v;
  0.0 = inductor.p.i + inductor.n.i;
  inductor.i = inductor.p.i;
  sineInputVoltage.signalSource.y = sineInputVoltage.signalSource.offset + (if time <
sineInputVoltage.signalSource.startTime then 0.0 else sineInputVoltage.signalSource.amplitude * sin(6.283185307179586 *
sineInputVoltage.signalSource.freqHz * (time - sineInputVoltage.signalSource.startTime) +
sineInputVoltage.signalSource.phase));
  sineInputVoltage.v = sineInputVoltage.signalSource.y;
  sineInputVoltage.v = sineInputVoltage.p.v - sineInputVoltage.n.v;
  0.0 = sineInputVoltage.p.i + sineInputVoltage.n.i;
  sineInputVoltage.i = sineInputVoltage.p.i;
  ground1.p.v = 0.0;
  resistor.p.i + sineInputVoltage.p.i = 0.0;
  resistor.n.i + inductor.p.i = 0.0;
  inductor.n.i + sineInputVoltage.n.i + ground1.p.i = 0.0;
  resistor.p.v = sineInputVoltage.p.v;
  inductor.p.v = resistor.n.v;
  ground1.p.v = inductor.n.v;
  ground1.p.v = sineInputVoltage.n.v;
end myRLCnetwork;
```

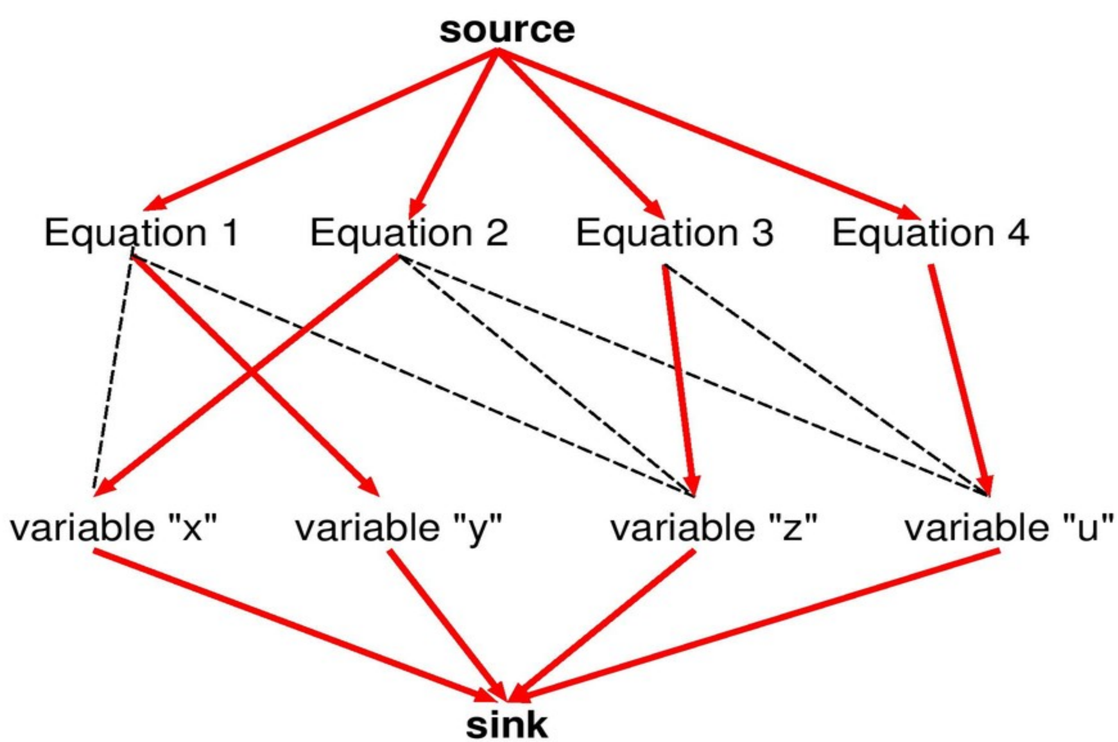
Non-causal model
(*e.g.*, from physical conservation laws)

$$\begin{cases} x + y + z & = & 0 & \text{Equation 1} \\ x + 3z + u^2 & = & 0 & \text{Equation 2} \\ z - u - 16 & = & 0 & \text{Equation 3} \\ u - 5 & = & 0 & \text{Equation 4} \end{cases}$$

Causality assignment: bipartite graph, maximum cardinality matching



Causality assignment: network flow



+ weights for “bad inverses”

Causality assigned

$$\begin{cases} x + \underline{y} + z & = & 0 & \text{Equation 1} \\ \underline{x} + 3z + u^2 & = & 0 & \text{Equation 2} \\ \underline{z} - u - 16 & = & 0 & \text{Equation 3} \\ \underline{u} - 5 & = & 0 & \text{Equation 4} \end{cases}$$

re-write in causal form

$$\begin{cases} \underline{y} & = & -x - z \\ \underline{x} & = & -3z - u^2 \\ \underline{z} & = & u + 16 \\ \underline{u} & = & 5 \end{cases}$$

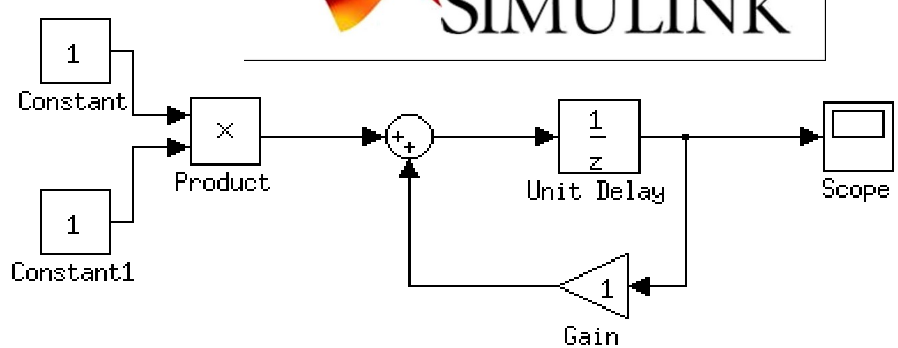
Set of Algebraic Eqns (no cyclic dependencies)

WRONG:

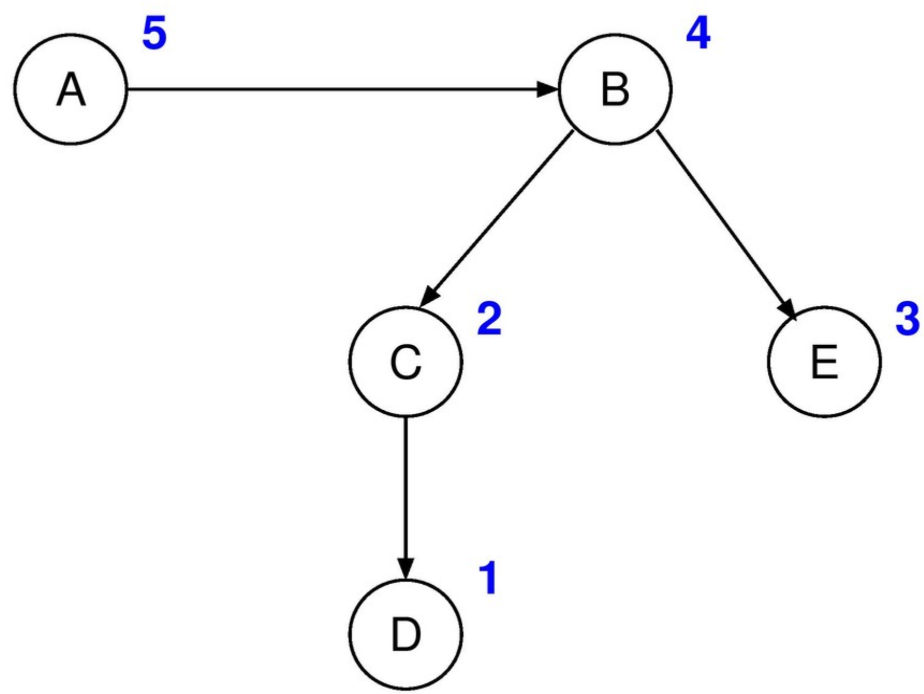
$$\left\{ \begin{array}{rcl} a & = & b^2 + 3 \\ b & = & sin(c \times e) \\ c & = & \sqrt{d - 4.5} \\ d & = & \pi/2 \\ e & = & u() \end{array} \right.$$

$$\left[\begin{array}{rcl} a & = & b^2 + 3 = 3 \\ b & = & sin(c \times e) = 0 \\ c & = & \sqrt{d - 4.5} = error \\ d & = & \pi/2 \\ e & = & u() \end{array} \right.$$

MATLAB SIMULINK



Sorting (no cyclic dependencies)
DFS, postorder numbering of dependency graph



Dependency Cycle (aka Algebraic Loop)

$$\begin{cases} x &= y + 16 \\ y &= -x - z \\ z &= 5 \end{cases}$$

Can *never* be sorted
due to a dependency *cycle* aka *strong component*
(every vertex in the component is reachable from every other)

$$x \rightarrow y \rightarrow x$$

May be solved implicitly

$$\left[\begin{array}{c} z = 5 \\ \left\{\begin{array}{rcl} x - y & = & -6 \\ x + y & = & -z \end{array}\right. \end{array}\right.$$

Implicit set of n equations in n unknowns.

- non-linear → non-linear solver.
- linear → numerical or symbolic solution.

Linear: may be solved symbolically (Cramer)

$$x = \frac{\begin{vmatrix} -6 & -1 \\ -z & 1 \end{vmatrix}}{\begin{vmatrix} 1 & -1 \\ 1 & 1 \end{vmatrix}} = \frac{-6-z}{2} \; ; \; y = \frac{\begin{vmatrix} 1 & -6 \\ 1 & -z \end{vmatrix}}{\begin{vmatrix} 1 & -1 \\ 1 & 1 \end{vmatrix}} = \frac{6-z}{2}$$

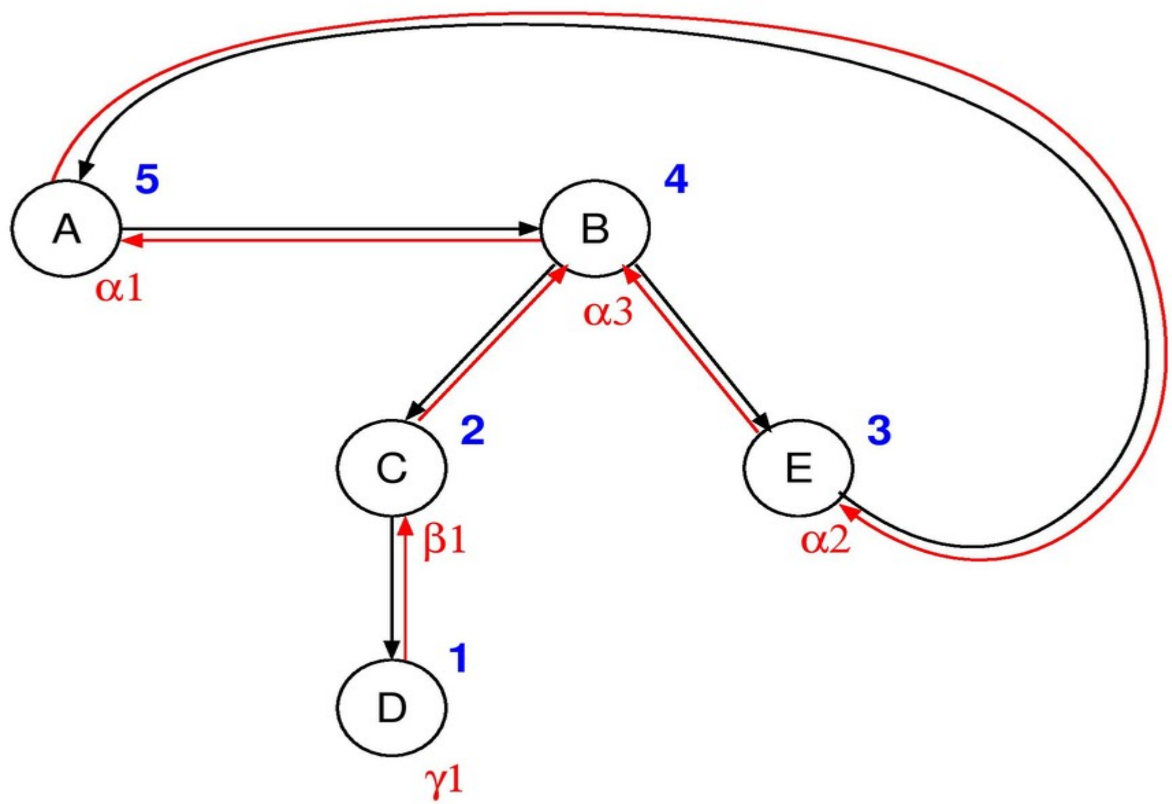
$$\left[\begin{array}{lcl} z & = & 5 \\ x & = & \frac{-6-z}{2} \\ y & = & \frac{6-z}{2} \end{array} \right.$$

Tarjan's algorithm for Cycle Detection

"strong components"

$$
\begin{cases}
a & = & b^2 + 3 \\
b & = & sin(c \times e) \\
c & = & \sqrt{d - 4.5} \\
d & = & \pi/2 \\
e & = & a^2 + u()
\end{cases}
$$

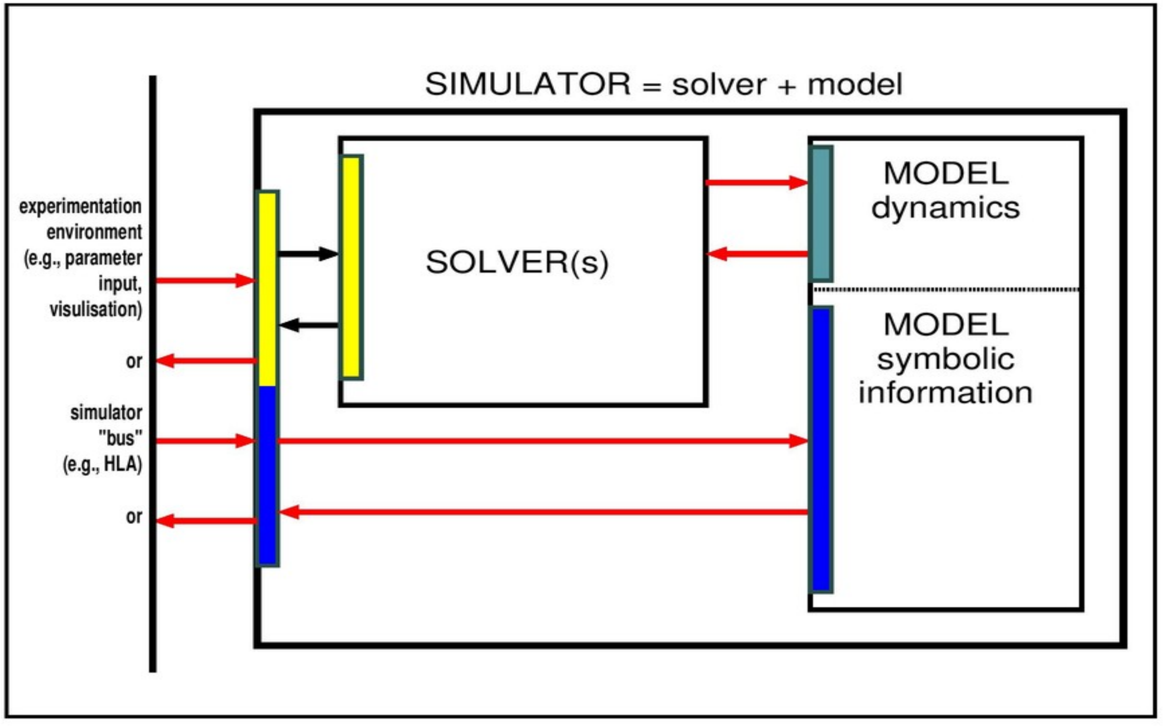
Algebraic Loop (Cycle) Detection

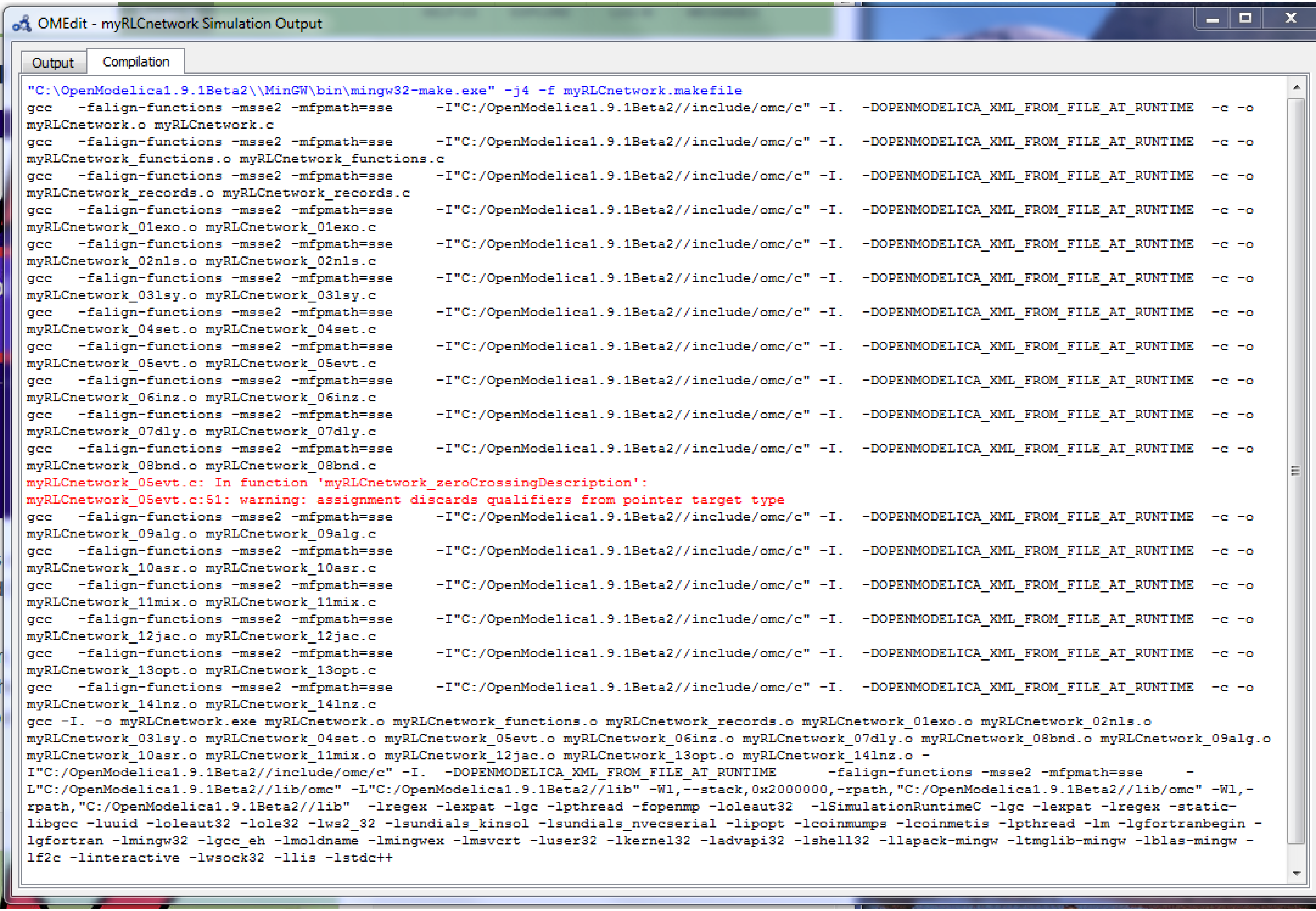


Algebraic Loop (Cycle) Detection Result

$$\left[\begin{array}{l} d = \pi/2 \\ c = \sqrt{d - 4.5} \\ \left\{\begin{array}{lcl} b &=& sin(c \times e) \\ a &=& b^2 + 3 \\ e &=& a^2 + u() \end{array}\right. \end{array}\right. ; \left[\begin{array}{l} d = \pi/2 \\ c = \sqrt{d - 4.5} \\ \left\{\begin{array}{cccccc} & b & -sin(c \times e) & & = & 0 \\ a & -b^2 & & -3 & = & 0 \\ a^2 & & -e & +u() & = & 0 \end{array}\right. \end{array}\right.$$

Model-Solver Interface
Simulator-Environment Interface



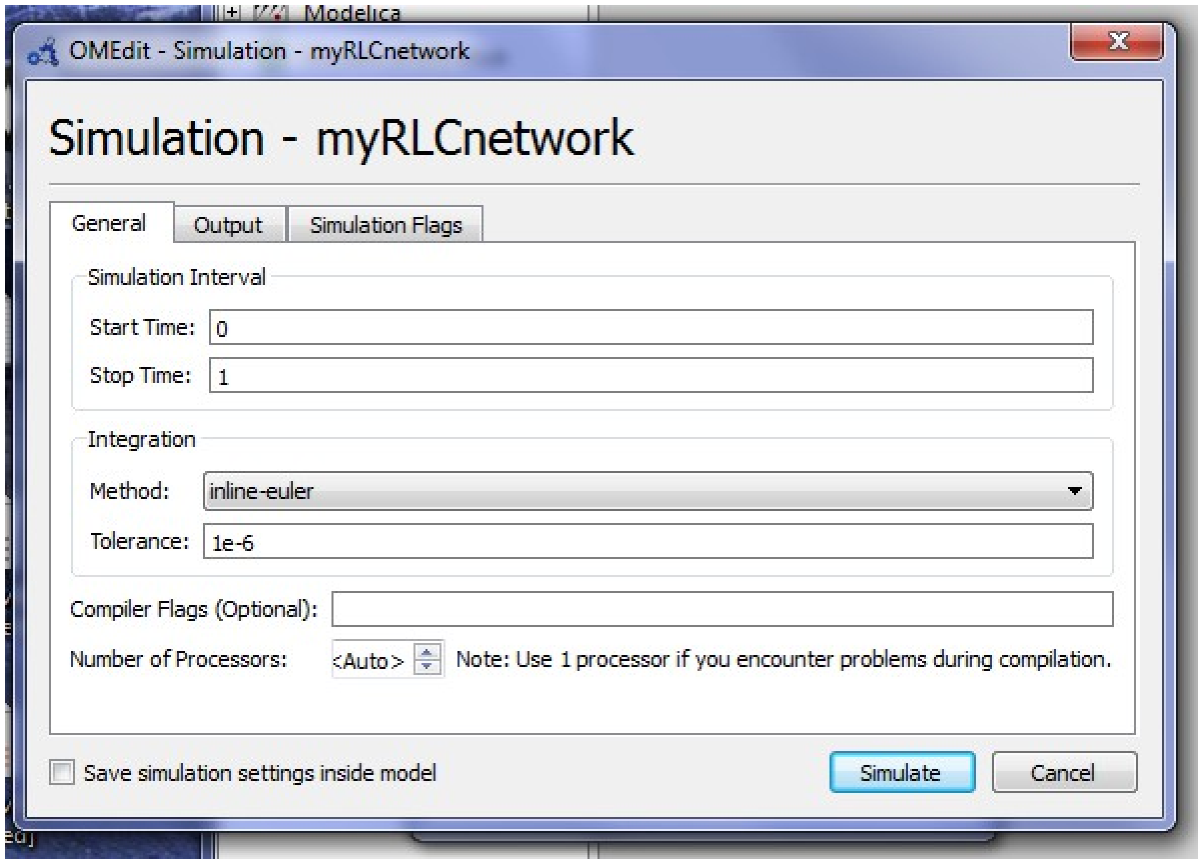


OMEdit - myRLCnetwork Simulation Output

Output | Compilation

```
"C:\OpenModelica1.9.1Beta2\\MinGW\bin\mingw32-make.exe" -j4 -f myRLCnetwork.makefile
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork.o myRLCnetwork.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_functions.o myRLCnetwork_functions.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_records.o myRLCnetwork_records.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_01exo.o myRLCnetwork_01exo.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_02nls.o myRLCnetwork_02nls.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_03lsy.o myRLCnetwork_03lsy.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_04set.o myRLCnetwork_04set.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_05evt.o myRLCnetwork_05evt.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_06inz.o myRLCnetwork_06inz.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_07dly.o myRLCnetwork_07dly.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_08bnd.o myRLCnetwork_08bnd.c
myRLCnetwork_05evt.c: In function 'myRLCnetwork_zeroCrossingDescription':
myRLCnetwork_05evt.c:51: warning: assignment discards qualifiers from pointer target type
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_09alg.o myRLCnetwork_09alg.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_10asr.o myRLCnetwork_10asr.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_11mix.o myRLCnetwork_11mix.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_12jac.o myRLCnetwork_12jac.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_13opt.o myRLCnetwork_13opt.c
gcc   -falign-functions -msse2 -mfpmath=sse     -I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME  -c -o
myRLCnetwork_14lnz.o myRLCnetwork_14lnz.c
gcc -I. -o myRLCnetwork.exe myRLCnetwork.o myRLCnetwork_functions.o myRLCnetwork_records.o myRLCnetwork_01exo.o myRLCnetwork_02nls.o
myRLCnetwork_03lsy.o myRLCnetwork_04set.o myRLCnetwork_05evt.o myRLCnetwork_06inz.o myRLCnetwork_07dly.o myRLCnetwork_08bnd.o myRLCnetwork_09alg.o
myRLCnetwork_10asr.o myRLCnetwork_11mix.o myRLCnetwork_12jac.o myRLCnetwork_13opt.o myRLCnetwork_14lnz.o -
I"C:/OpenModelica1.9.1Beta2//include/omc/c" -I.  -DOPENMODELICA_XML_FROM_FILE_AT_RUNTIME      -falign-functions -msse2 -mfpmath=sse     -
L"C:/OpenModelica1.9.1Beta2//lib/omc" -L"C:/OpenModelica1.9.1Beta2//lib" -Wl,--stack,0x2000000,-rpath,"C:/OpenModelica1.9.1Beta2//lib/omc" -Wl,-
rpath,"C:/OpenModelica1.9.1Beta2//lib"  -lregex -lexpat -lgc -lpthread -fopenmp -loleaut32  -lSimulationRuntimeC -lgc -lexpat -lregex -static-
libgcc -luuid -loleaut32 -lole32 -lws2_32 -lsundials_kinsol -lsundials_nvecserial -lipopt -lcoinmumps -lcoinmetis -lpthread -lm -lgfortranbegin -
lgfortran -lmingw32 -lgcc_eh -lmoldname -lmingwex -lmsvcrt -luser32 -lkernel32 -ladvapi32 -lshell32 -llapack-mingw -ltmglib-mingw -lblas-mingw -
lf2c -linteractive -lwsock32 -llis -lstdc++
```

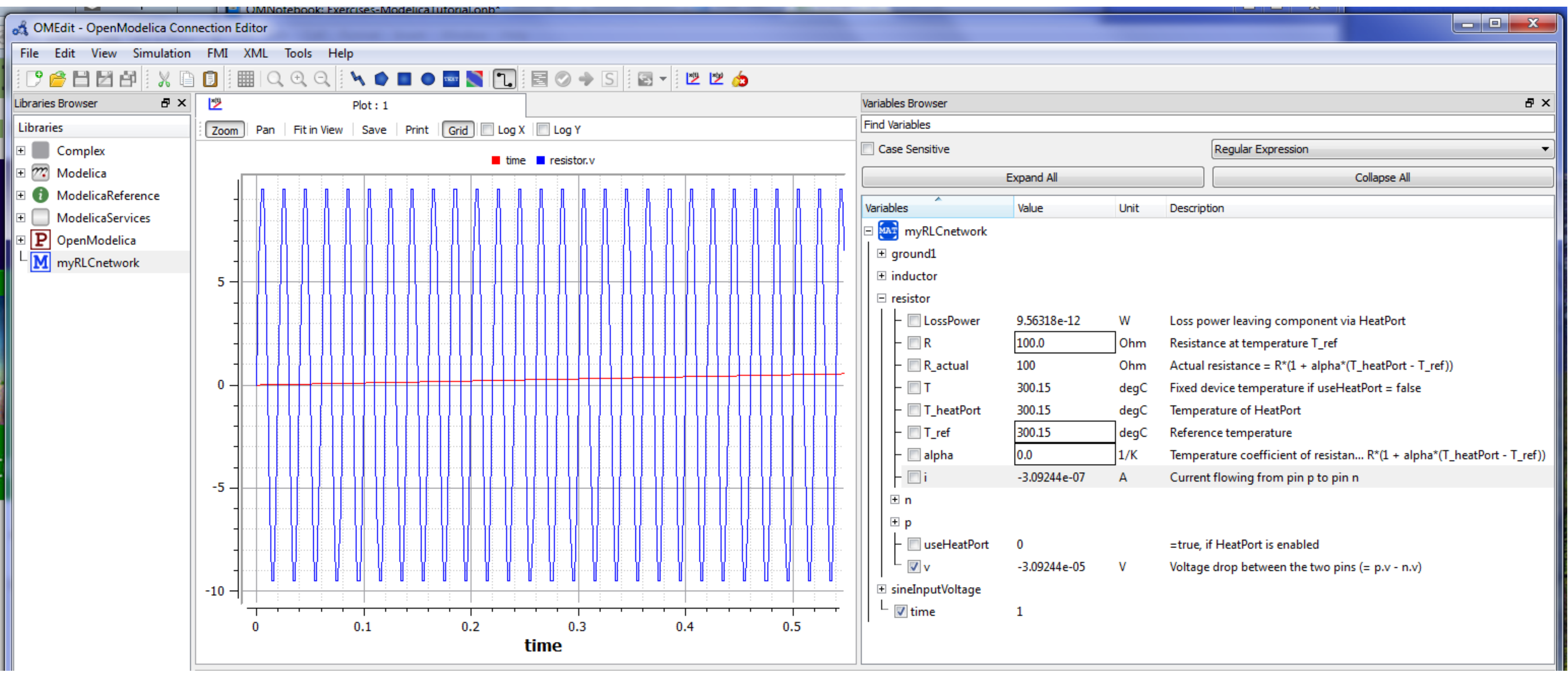
29/09/20...	C File	myRLCnetwork	14 KB
29/09/20...	Application	myRLCnetwork	9,960 KB
29/09/20...	LIBS File	myRLCnetwork.libs	0 KB
29/09/20...	Text Document	myRLCnetwork	0 KB
29/09/20...	MAKEFILE File	myRLCnetwork.makefile	2 KB
29/09/20...	O File	myRLCnetwork.o	17 KB
29/09/20...	C File	myRLCnetwork_01exo	2 KB
29/09/20...	O File	myRLCnetwork_01exo.o	2 KB
29/09/20...	C File	myRLCnetwork_02nls	2 KB
29/09/20...	O File	myRLCnetwork_02nls.o	1 KB
29/09/20...	C File	myRLCnetwork_03lsy	2 KB
29/09/20...	O File	myRLCnetwork_03lsy.o	1 KB
29/09/20...	C File	myRLCnetwork_04set	2 KB
29/09/20...	O File	myRLCnetwork_04set.o	1 KB
29/09/20...	C File	myRLCnetwork_05evt	3 KB
29/09/20...	O File	myRLCnetwork_05evt.o	2 KB
29/09/20...	C File	myRLCnetwork_06inz	7 KB
29/09/20...	O File	myRLCnetwork_06inz.o	5 KB
29/09/20...	C File	myRLCnetwork_07dly	2 KB
29/09/20...	O File	myRLCnetwork_07dly.o	1 KB
29/09/20...	C File	myRLCnetwork_08bnd	7 KB
29/09/20...	O File	myRLCnetwork_08bnd.o	5 KB
29/09/20...	C File	myRLCnetwork_09alg	2 KB
29/09/20...	O File	myRLCnetwork_09alg.o	1 KB
29/09/20...	C File	myRLCnetwork_10asr	2 KB
29/09/20...	O File	myRLCnetwork_10asr.o	1 KB
29/09/20...	C File	myRLCnetwork_11mix	2 KB
29/09/20...	H File	myRLCnetwork_11mix.h	0 KB
29/09/20...	O File	myRLCnetwork_11mix.o	1 KB
29/09/20...	C File	myRLCnetwork_12jac	4 KB
29/09/20...	H File	myRLCnetwork_12jac.h	2 KB


Modelica
OMEdit - Simulation - myRLCnetwork
Simulation - myRLCnetwork
General
Output
Simulation Flags
Simulation Interval
Start Time:
0
Stop Time:
1
Integration
Method:
inline-euler
Tolerance:
1e-6
Compiler Flags (Optional):
Number of Processors:
<Auto>
Note: Use 1 processor if you encounter problems during compilation.
Save simulation settings inside model
Simulate
Cancel

OMEdit - myRLCnetwork Simulation Output

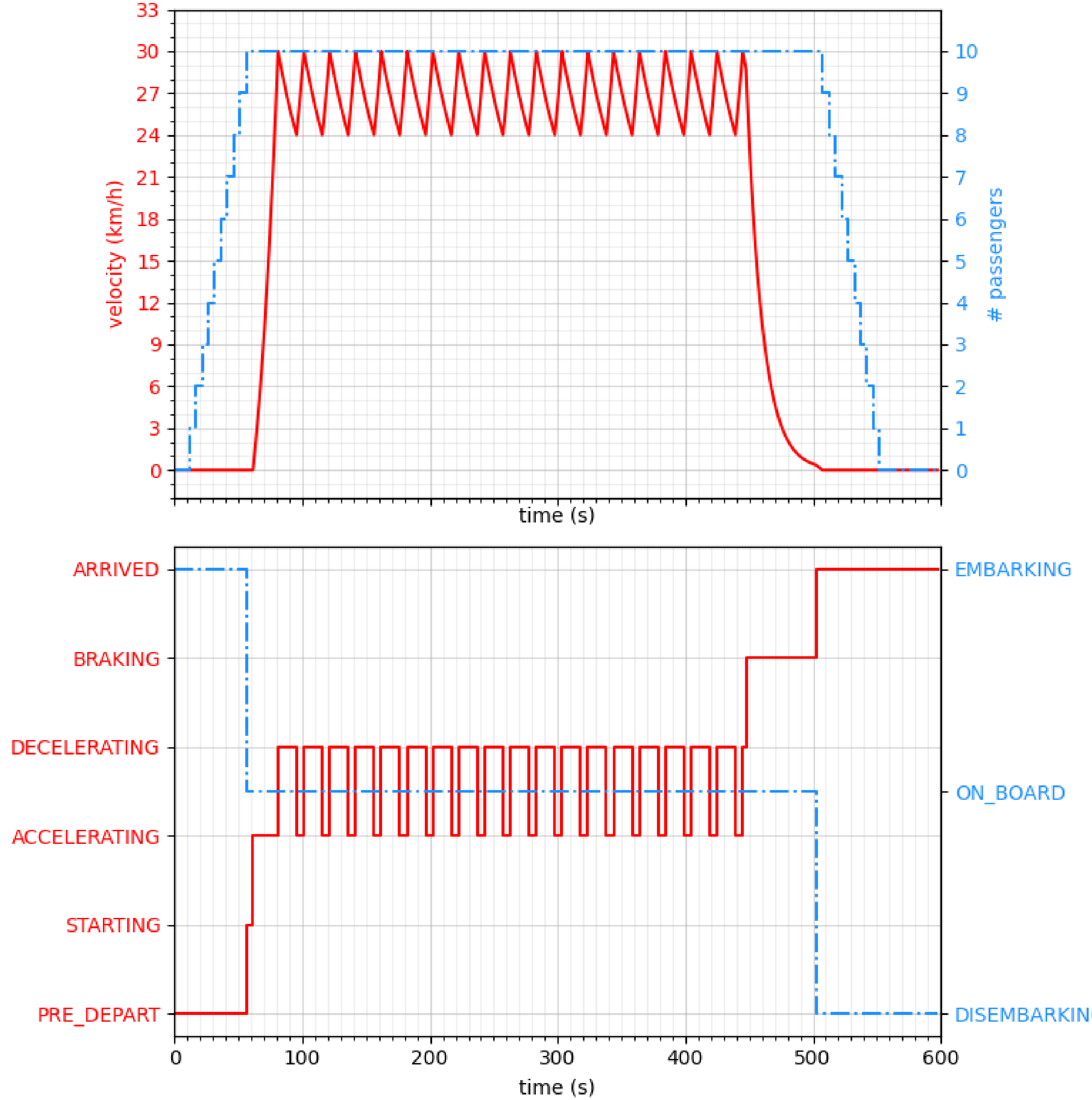
Output | Compilation

```
C:/Users/hv/AppData/Local/Temp/OpenModelica/OMEdit/myRLCnetwork.exe -port=49502 -logFormat=xml -w -lv=LOG_STATS
LOG_STATS                | info       | ### STATISTICS ###
LOG_STATS                | info       | timer
|                        | |          | |     0.0150538s [ 46.9%] pre-initialization
|                        | |          | | 4.18139e-005s [  0.1%] initialization
|                        | |          | |  2.0907e-005s [  0.1%] steps
|                        | |          | |     0.0157118s [ 49.0%] creating output-file
|                        | |          | |   0.000115558s [  0.4%] event-handling
|                        | |          | |   0.000295738s [  0.9%] overhead
|                        | |          | |   0.000824114s [  2.6%] simulation
|                        | |          | |     0.0320637s [100.0%] total
LOG_STATS                | info       | events
|                        | |          | |      0 state events
|                        | |          | |      0 time events
LOG_STATS                | info       | solver: DASSL
|                        | |          | |   2431 steps taken
|                        | |          | |   3266 calls of functionODE
|                        | |          | |    165 evaluations of jacobian
|                        | |          | |     73 error test failures
|                        | |          | |      0 convergence test failures
LOG_STATS                | info       | ### END STATISTICS ###
```

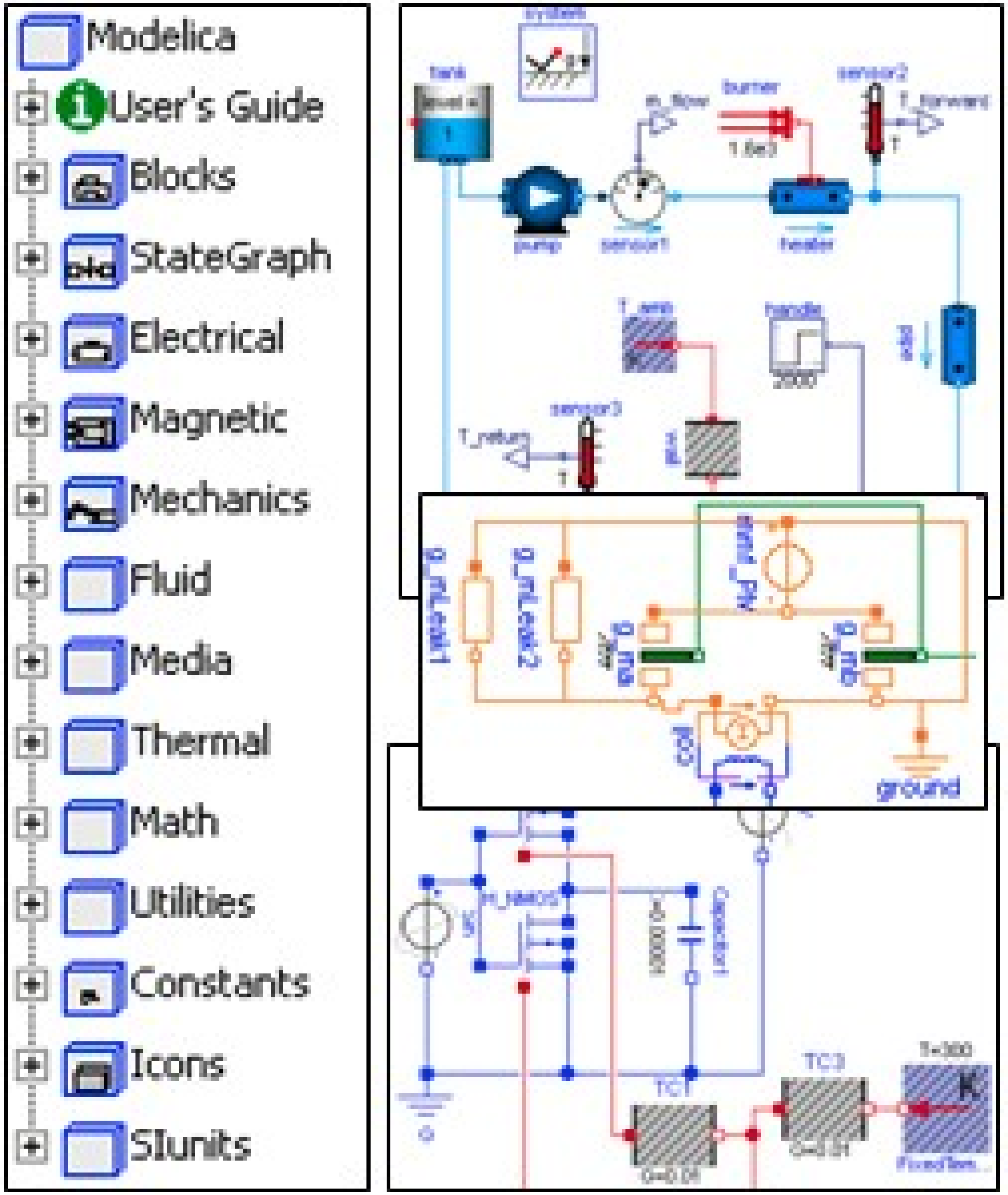

OMEdit - OpenModelica Connection Editor
File
Edit
View
Simulation
FMI
XML
Tools
Help
Libraries Browser
Libraries
Complex
Modelica
ModelicaReference
ModelicaServices
OpenModelica
myRLCnetwork
Plot : 1
Zoom
Pan
Fit in View
Save
Print
Grid
Log X
Log Y
time
resistor.v
5
0
-5
-10
0
0.1
0.2
0.3
0.4
0.5
time
Variables Browser
Find Variables
Case Sensitive
Regular Expression
Expand All
Collapse All
Variables
Value
Unit
Description
myRLCnetwork
ground1
inductor
resistor
LossPower 9.56318e-12 W Loss power leaving component via HeatPort
R 100.0 Ohm Resistance at temperature T_ref
R_actual 100 Ohm Actual resistance = R*(1 + alpha*(T_heatPort - T_ref))
T 300.15 degC Fixed device temperature if useHeatPort = false
T_heatPort 300.15 degC Temperature of HeatPort
T_ref 300.15 degC Reference temperature
alpha 0.0 1/K Temperature coefficient of resistan... R*(1 + alpha*(T_heatPort - T_ref))
i -3.09244e-07 A Current flowing from pin p to pin n
n
p
useHeatPort 0 =true, if HeatPort is enabled
v -3.09244e-05 V Voltage drop between the two pins (= p.v - n.v)
sineInputVoltage
time 1

Hybrid


Morgantown
Personal Rapid Transit


velocity (km/h)
33
30
27
24
21
18
15
12
9
6
3
0
passengers
10
9
8
7
6
5
4
3
2
1
0
time (s)
ARRIVED
BRAKING
DECELERATING
ACCELERATING
STARTING
PRE_DEPART
EMBARKING
ON_BOARD
DISEMBARKING
0
100
200
300
400
500
600
time (s)

MODELICA Standard Library (MSL)



Controller Design and Tuning