KU LEUVEN

Modelica, Dymola and IDEAS

Crash Course 2024

Modelica Language

Dymola

Naomi Adam, Jelger Jansen, Lucas Verleyen

THERMAL SYSTEMS SIMULATION

Who are we?

Naomi Adam

Jelger Jansen

Lucas Verleyen

The SySi Team

- Led by Professor Lieve Helsen

To sustainably use resources through integration and optimization of thermal systems performance in the built environment, including other energy vectors and sectors.

Motivation

- Why a crash course?
  - Introduction for our students and PhD students
  - Others who are interested in using Modelica
  - Broaden the user base and open-source community

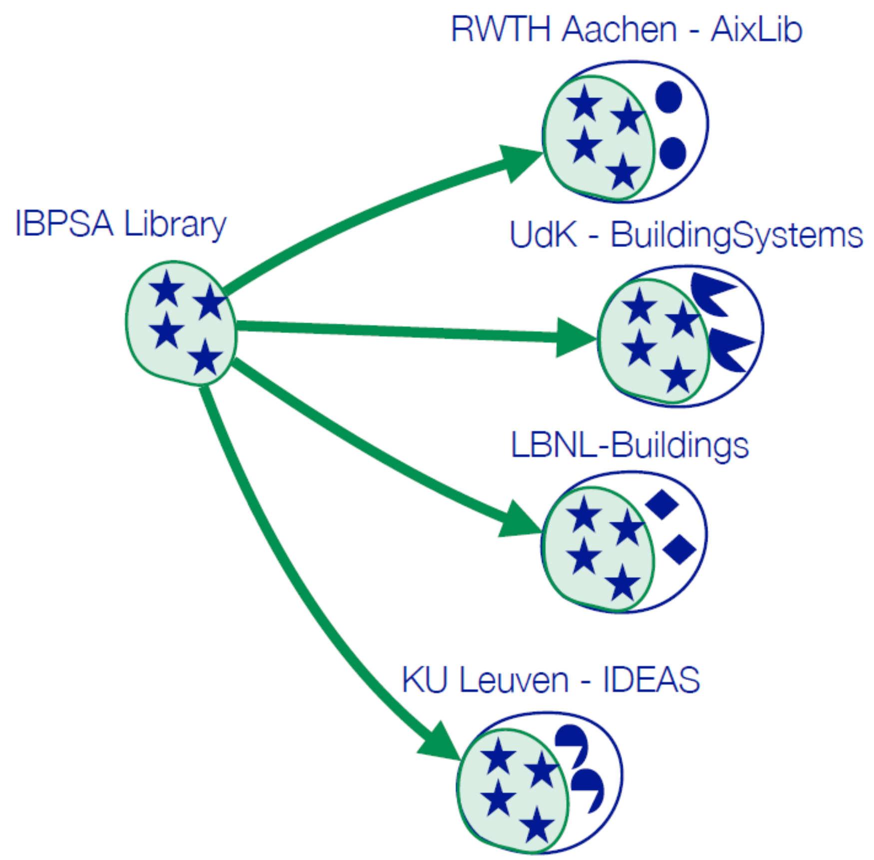
- About IDEAS
  - Integrated District Energy Assessment Simulations
  - Modelica users and library development since 2010
  - IDEAS was not the only one

Motivation

- In 2013
  - Modelica for buildings was very fragmented
  - Libraries were incompatible
  - Libraries replicated each other
  - Best practices were not understood

→ Joint efforts started (Annex 60, IBPSA Project 1)

  - Avoid fragmentation
  - Collaborate on development
  - Implement best practices
  - Share everything open-source and free



https://github.com/open-ideas/IDEAS
https://github.com/ibpsa/modelica-ibpsa

Motivation

- About IDEAS
  - IDEAS v3.0
  - BaseClasses inherited from IBPSA project 1
  - Focus on buildings (thermal)
  - Work on districts in other libraries (e.g. MoPED)
  - Many models are validated in academic research
  - Main user base: researchers, students

  - Builtwins: startup for sustainable control of buildings, using IDEAS

https://github.com/open-ideas/IDEAS
https://gitlab.kuleuven.be/positive-energy-districts/moped

Agenda

Morning: Dymola and Modelica

- 9:30 - 10:00 Lecture 1
  - What is Modelica? What is Dymola? What is OpenModelica?
  - Modelica/Dymola basics
- 10:00 - 10:30 Exercise 1
- 10:30 - 10:45 Break
- 10:45 - 11:30 Lecture 2
  - Create new models/packages
  - Modelling with several components
  - Use connectors
  - Set parameters/propagate parameters
- 11:30 - 12:30 Exercise 2
- 12:30 - 13:30 Lunch break

Afternoon: IDEAS

- 13:30 - 14:00 Lecture 3
  - What is IDEAS?
  - IDEAS building components
  - IDEAS workflow
- 14:00 - 16:00 Exercise 3
- 16:00 - 16:30 Break
- 16:30 - 16:45 Lecture 4
  - IDEAS HVAC components
  - Hydronic models
- 16:45 - 18:00 Exercise 4

Part 1: Introduction to Modelica and Dymola

Naomi Adam

Modelica

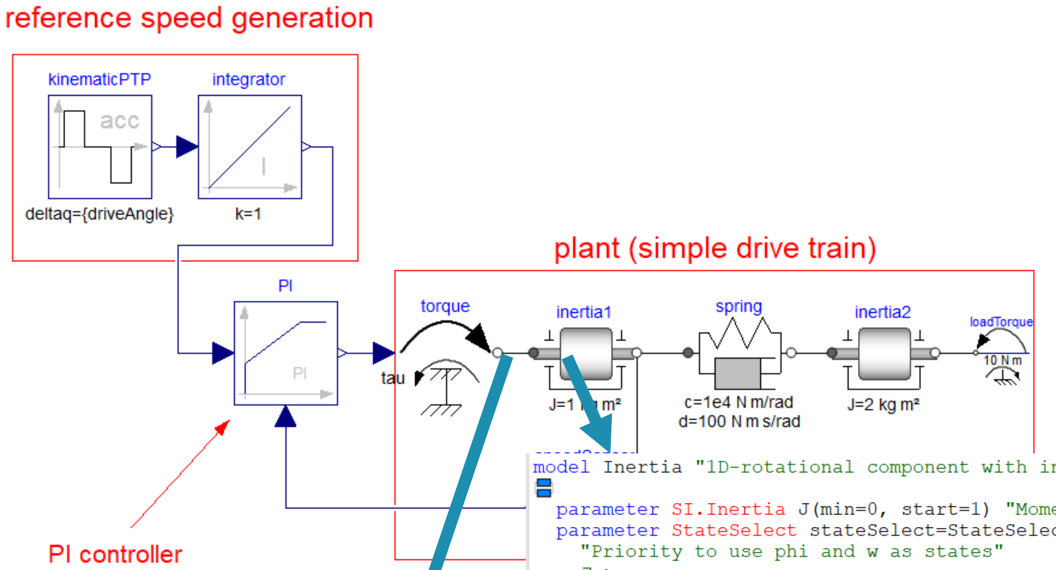
Modelica is a **modelling language** for modelling physical systems

- Language specification is open source
- object oriented
- Acausal modeling (equation-based)
- Multi-domain
- Primarily for simulation, but usable for optimization
- Small and large models (> 100 000 equations)
- Large community with many model libraries, especially in automotive industry (free and commercial)
- Textual and graphical modelling

Modelica

Object-oriented physical equation-based modelling

- A model represents a physical component
- Component is composed of sub-components and/or is described by equations
  → hierarchical structure
- Components can be connected to each other using connectors (=physical coupling)
- To simulate Modelica models, a Modelica simulation environment is needed




```
model Inertia "1D-rotational component with inertia"

  parameter SI.Inertia J(min=0, start=1) "Moment of inertia";
  parameter StateSelect stateSelect=StateSelect.default
    "Priority to use phi and w as states"
    ;
  SI.Angle phi(stateSelect=stateSelect)
    "Absolute rotation angle of component"
    ;
  SI.AngularVelocity w(stateSelect=stateSelect)
    "Absolute angular velocity of component (= der(phi))"
    ;
  SI.AngularAcceleration a
    "Absolute angular acceleration of component (= der(w))"
    ;

equation
  phi = flange_a.phi;
  phi = flange_b.phi;
  w = der(phi);
  a = der(w);
  J*a = flange_a.tau + flange_b.tau;

end Inertia;
```

```
connector Flange_a
  "1-dim. rotational flange of a shaft (filled square icon)"
  SI.Angle phi "Absolute rotation angle of flange";
  flow SI.Torque tau "Cut torque in the flange";

end Flange_a;
```

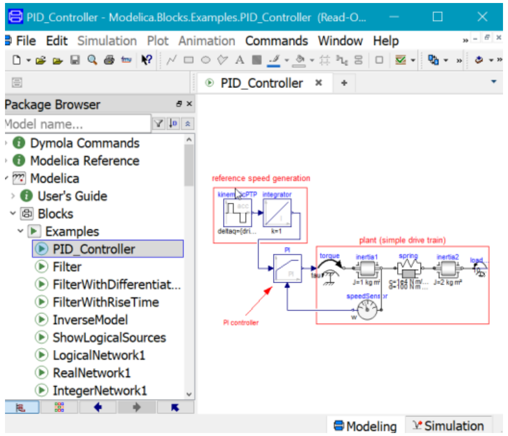

THERMAL SYSTEMS SIMULATION
KU LEUVEN

Dymola

Dymola is a **commercial Modelica simulation environment**

Live demo of features:

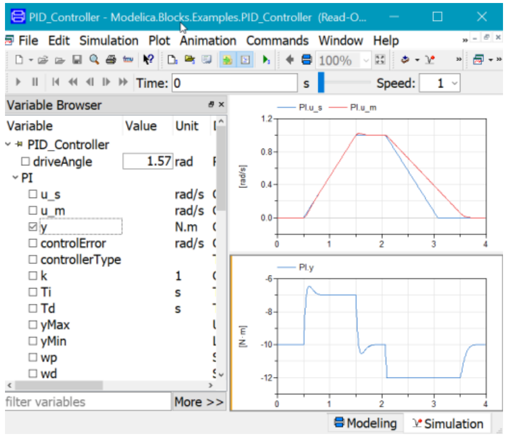
- Icon, Diagram, Editor, Info
- Package browser, modelling, simulation
- Set up (compiler), run
- Adapt parameter
- Load libraries
- Look at simulation results: plot, zoom, filter variable, plot as a function of other variable.
- Try Simulate and plot (IDEAS library)
- Open sub-components
- Documentation



- Graphical editor
- Modelica simulation environment

```
model PID_Controller
  "Demonstrates the usage of a Continuous.LimPID controller"
  extends Modelica.Icons.Example;
  parameter Modelica.SIunits.Angle driveAngle=1.57
    "Reference distance to move";
  Modelica.Blocks.Continuous.LimPID PI(
    k=100,
    Ti=0.1,
    yMax=12,
    Ni=0.1,
    initType=Modelica.Blocks.Types.InitPID.SteadyState,
    limitsAtInit=false,
    controllerType=Modelica.Blocks.Types.SimpleController.PI,
    Td=0.1) a ;
  Modelica.Mechanics.Rotational.Components.Inertia inertial(
    phi(fixed=true, start=0),
    J=1,
    a(fixed=true, start=0)) a ;

  Modelica.Mechanics.Rotational.Sources.Torque torque a ;
```

- Textual description (Modelica language)



- Translation of Modelica code into executable C-code
- Coupling with a solver
- Visualization of results

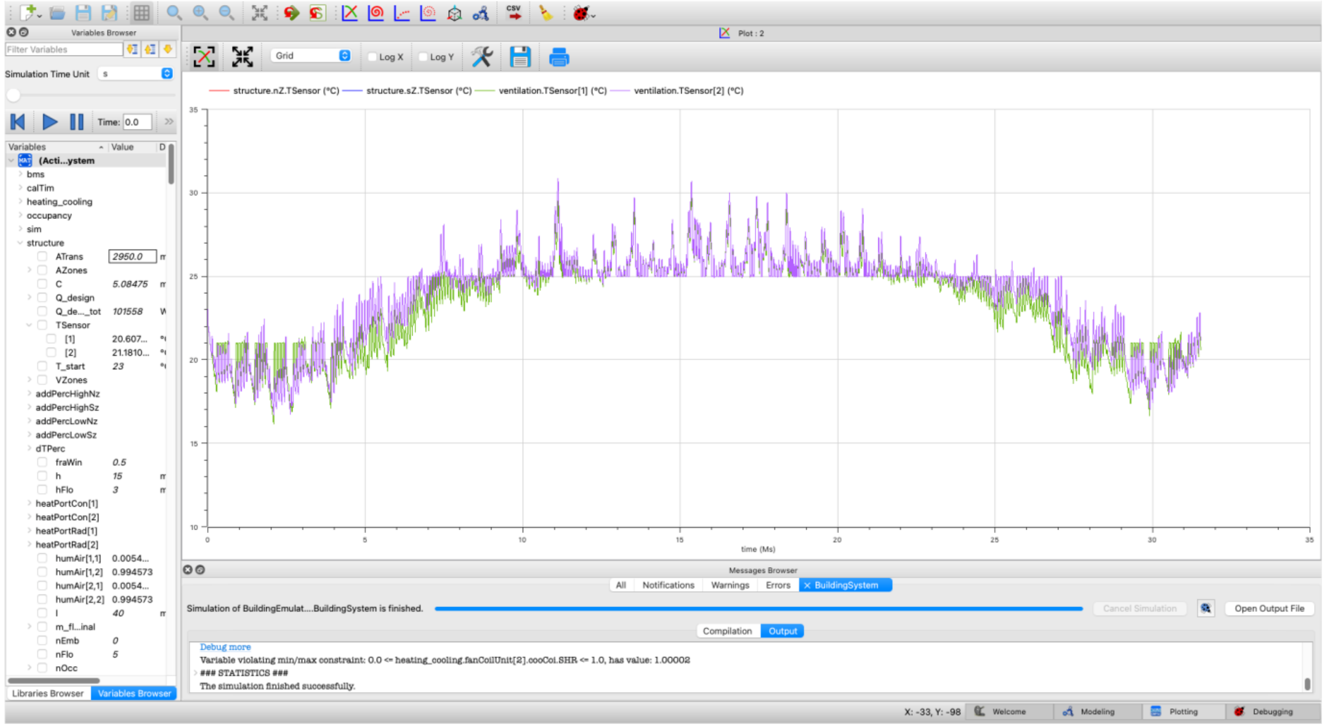
THERMAL SYSTEMS SIMULATION

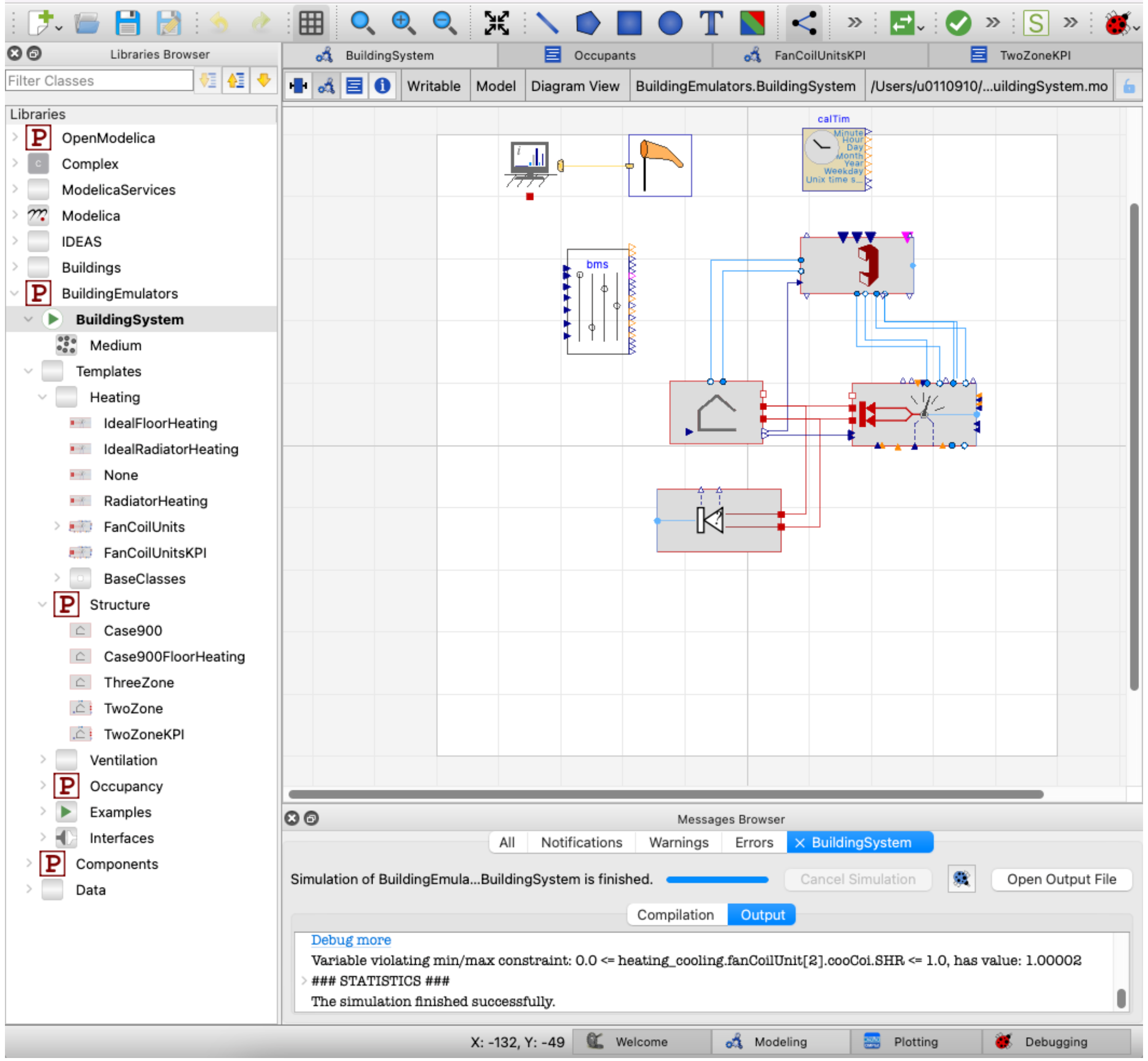
KU LEUVEN

OpenModelica

OM is a **free Modelica simulation environment**

Has very similar features to Dymola, like those we have just seen





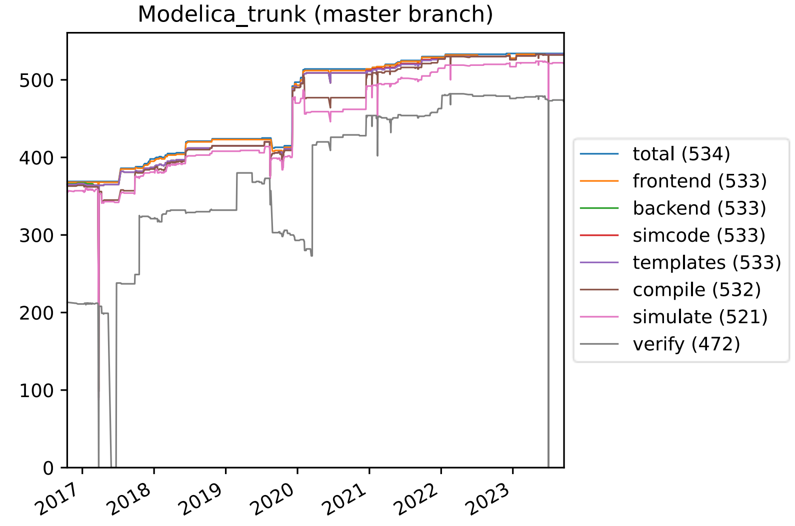
THERMAL SYSTEMS SIMULATION

KU LEUVEN

OpenModelica

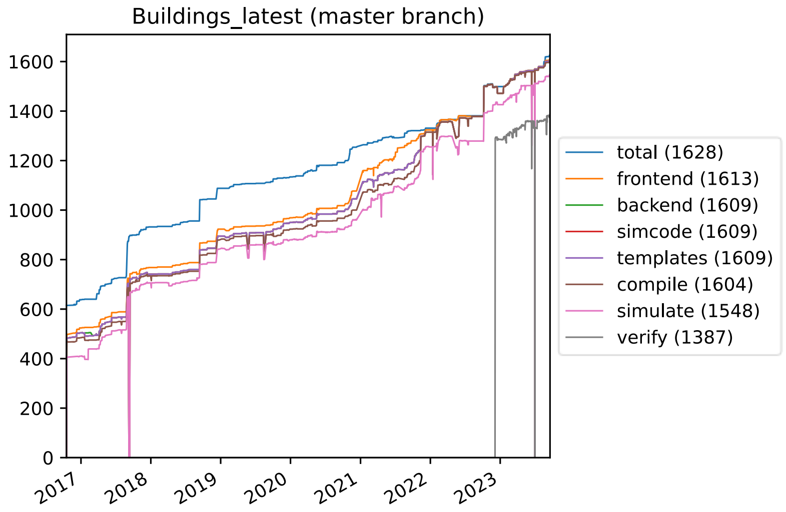
Historically less performant than Dymola, but has radically improved over the last years:

Modelica Standard Library (MSL) coverage



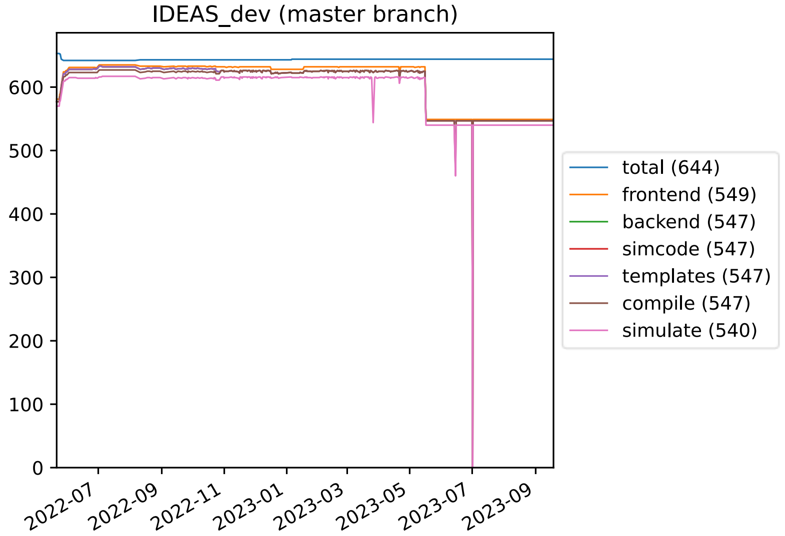
https://libraries.openmodelica.org/branches/history/master/Modelica_trunk.svg

Buildings library coverage



https://libraries.openmodelica.org/branches/history/master/Buildings_latest.svg

IDEAS coverage (recently added!)



https://libraries.openmodelica.org/branches/history/master/IDEAS_dev.svg

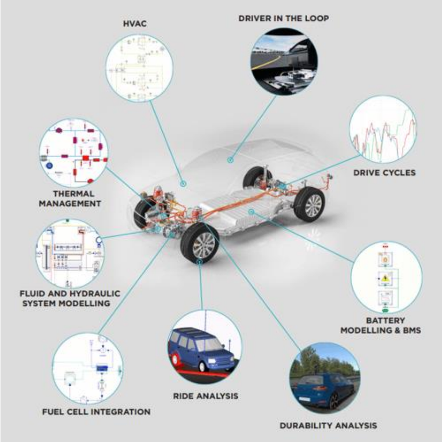
Reasons not to use it in the crash course: Dymola is still more performant…

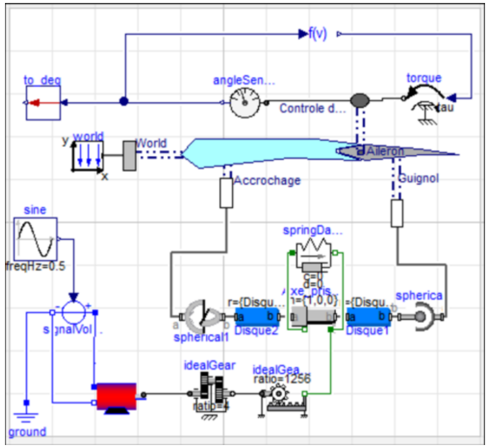
Applications

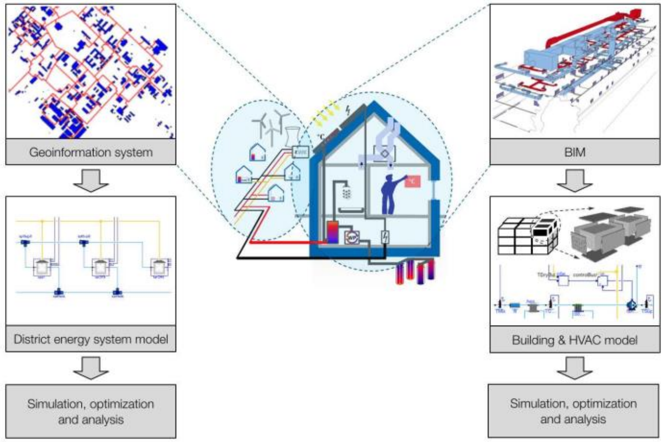
System modeling and simulation applications

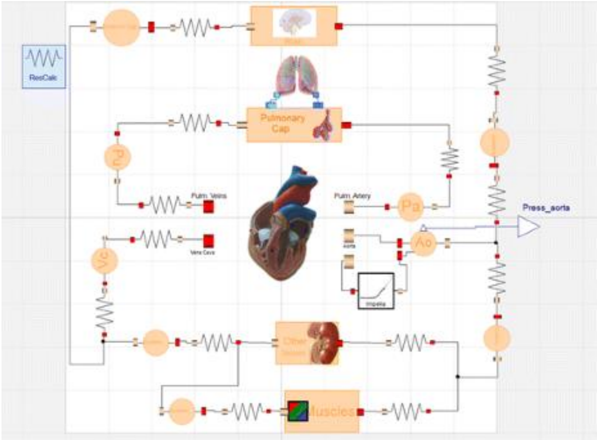
➔ Automotive, aerospace, architecture, energy, medicine, …

➔ In this crashcourse: thermal systems in buildings









[1] Dymola | TECHNIA (Finland)
[2] DOI:10.1109/MECATRONICS.2018.8495666
[3] Projects - International Building Performance Simulation Association (ibpsa.org)
[4] https://doi.org/10.1007/s11517-015-1384-6

Useful links

General

- https://modelica.org/
- https://openmodelica.org/
- https://jmodelica.org/ DISCONTINUED
- https://www.claytex.com/tech-blog/

Modelica language

- https://mbe.modelica.university/
- https://doc.modelica.org/
- https://specification.modelica.org/

Libraries:

- IDEAS
  https://github.com/open-ideas
- Buildings
  https://simulationresearch.lbl.gov/modelica
  (look at Buildings.Examples.Tutorial)
- IBPSA Project 1
  https://github.com/ibpsa/modelica-ibpsa

Dymola user guide

- Online
- Via Dymola > help

Exercise 1

- You can find the exercise on GitHub

  → Go to www.github.com/open-ideas

  → _CrashCourse_

  → Exercises

  → Exercise 1

  → Exercise1.pdf

  → https://github.com/open-ideas/__CrashCourse__/blob/master/Exercises/Exercise%201/Latex/Exercise1.pdf

Part 2: Modelling and simulating in Dymola

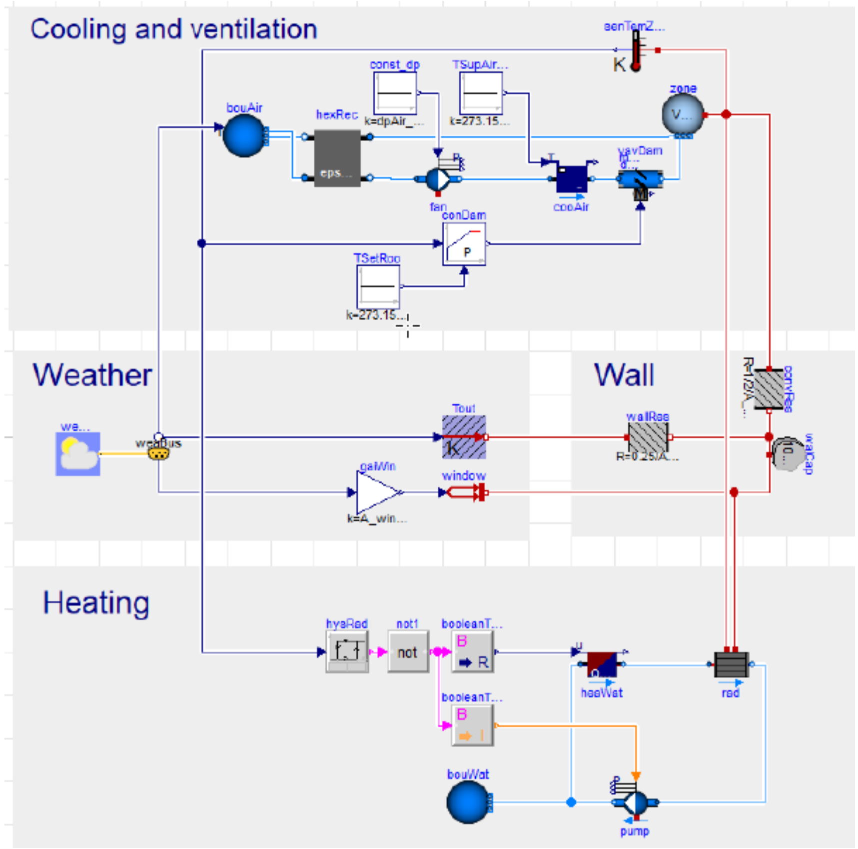
Lucas Verleyen

Live demonstration

- Create package, create model (+ save contents in one file)
- Search, drag and drop subcomponents. Instantiate model convention
- Connect components
- Graphical view and text view
- Simulation tab and adapt parameters
- Propagate parameters
- Units
- Use check/translate in Dymola and debug:
  - Syntax error
  - Modeling error: singularity
  - Model with external input

Exercise 2

- You can find the exercise on GitHub
  - → Go to www.github.com/open-ideas
  - → _CrashCourse_
  - → Exercises
  - → Exercise 2
  - → Latex
  - → Exercise2.pdf

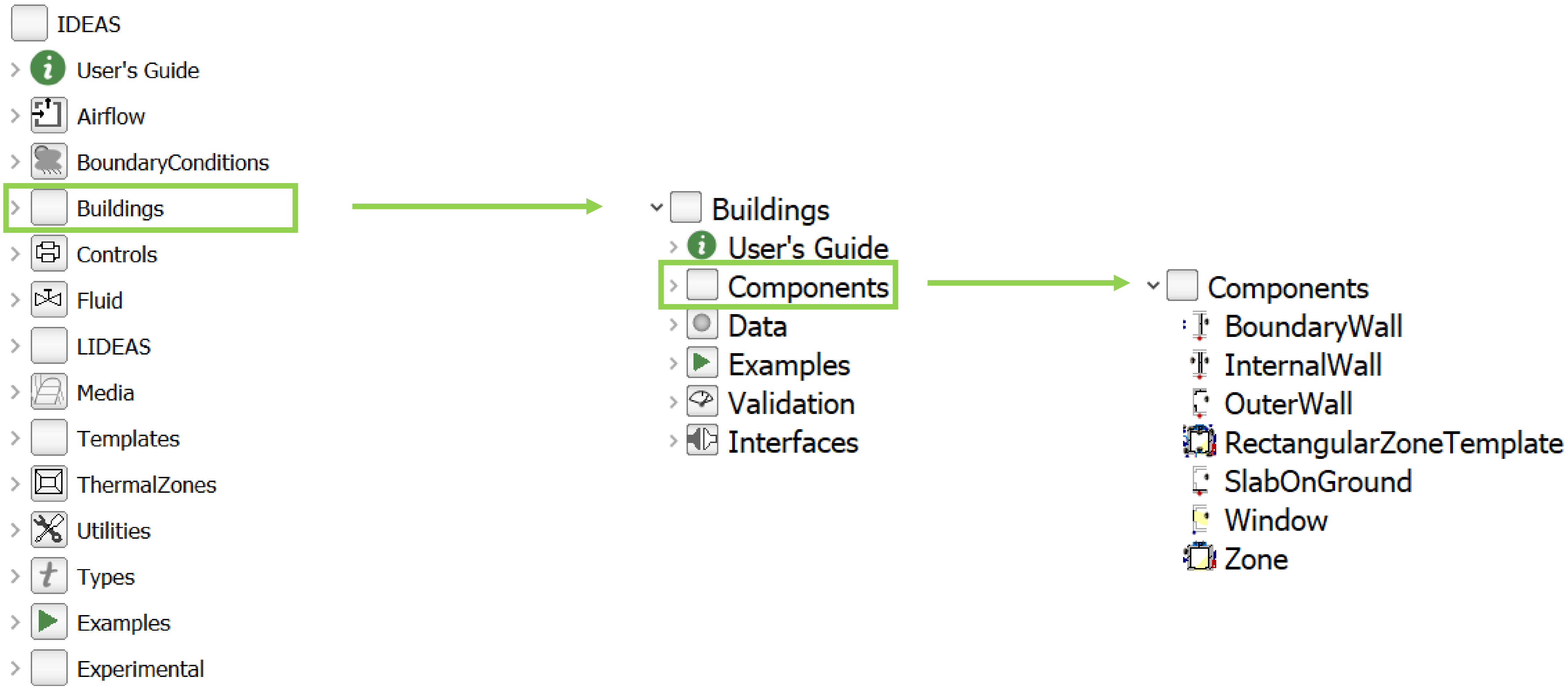


→ https://github.com/open-ideas/__CrashCourse__/blob/master/Exercises/Exercise%202/Latex/Exercise2.pdf

Part 3: IDEAS – Building envelope

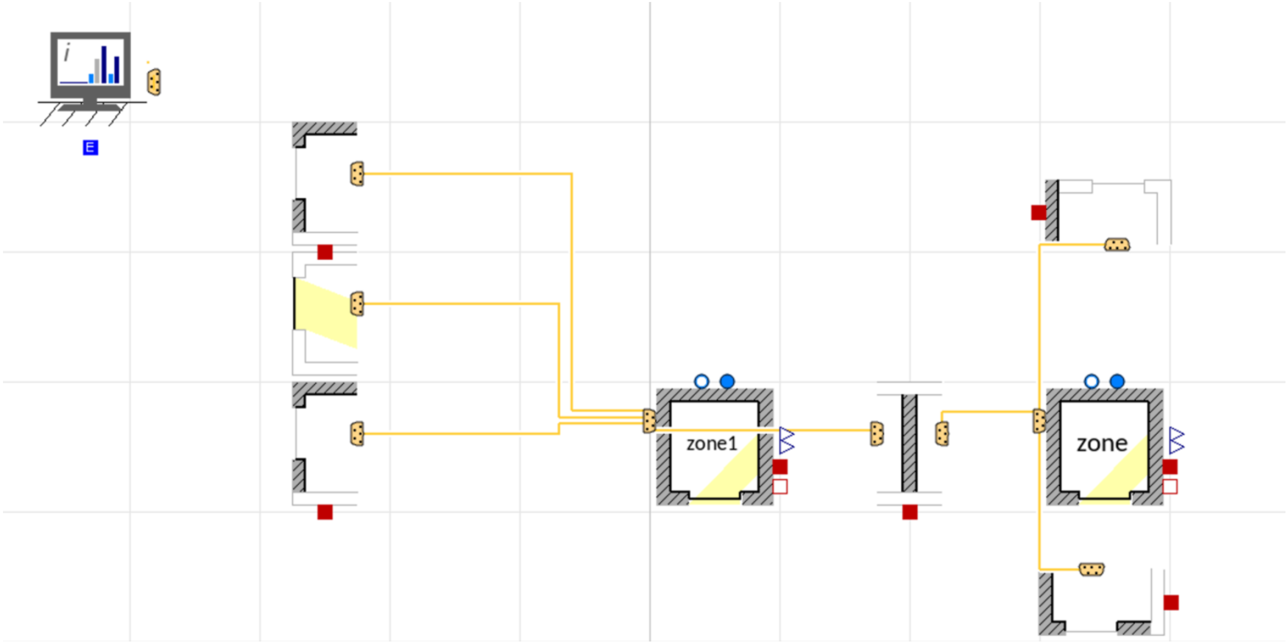
Jelger Jansen

IDEAS – Overview

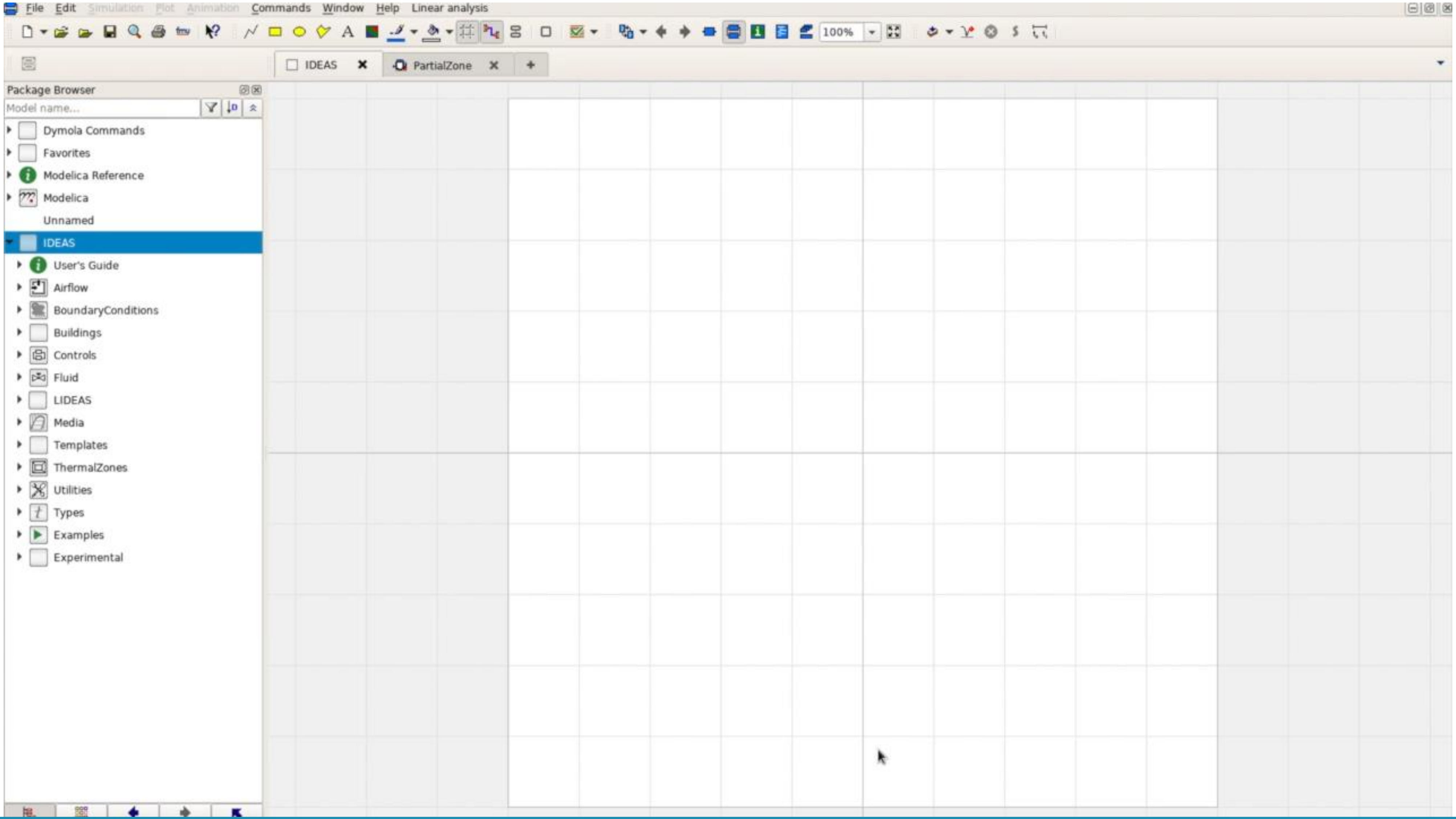


Philosophy

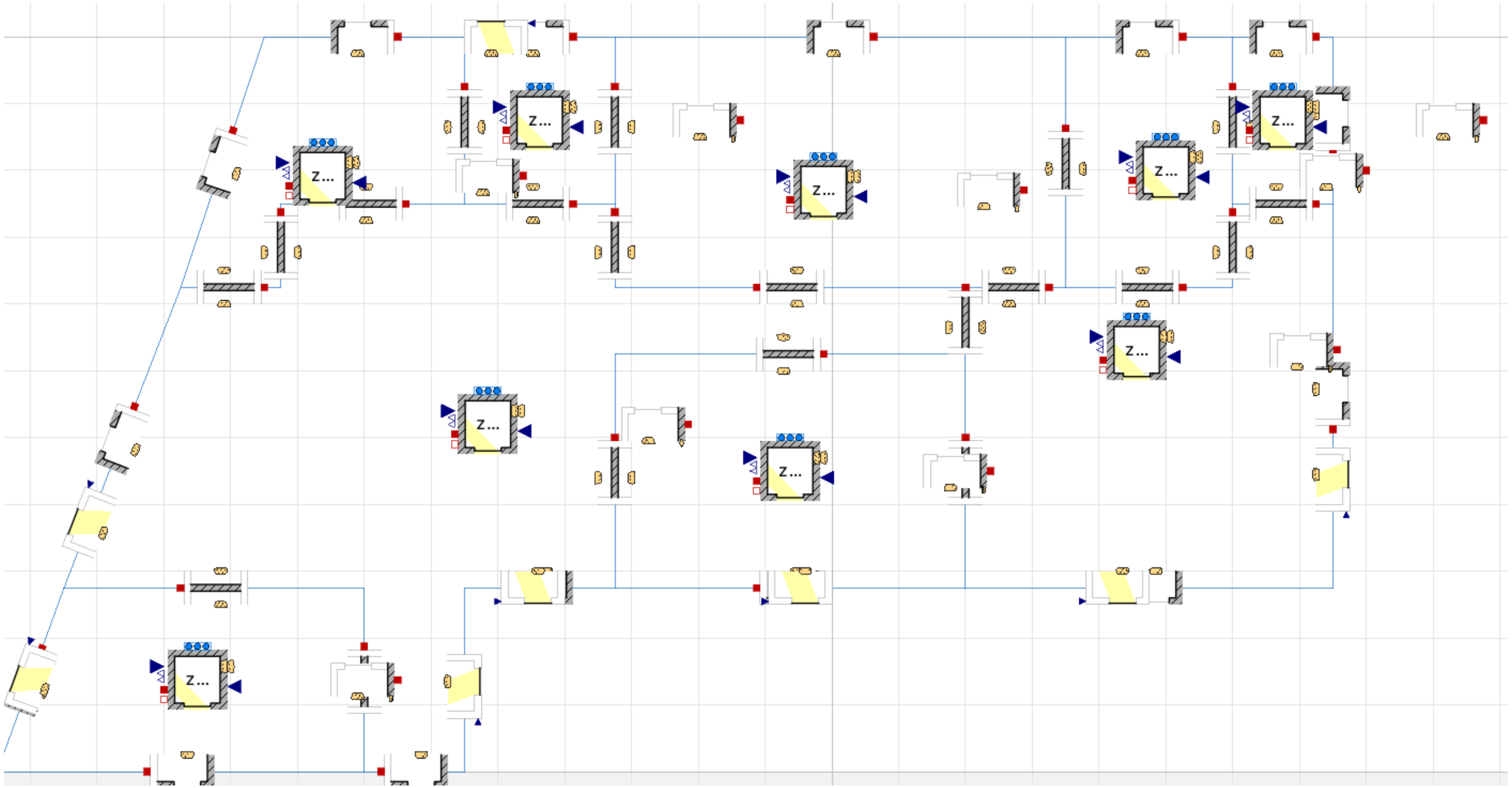
- Direct mapping between physical objects and components (white-box modelling)
- “What you see is what you get”
- Exception: SimInfoManager



IDEAS workflow

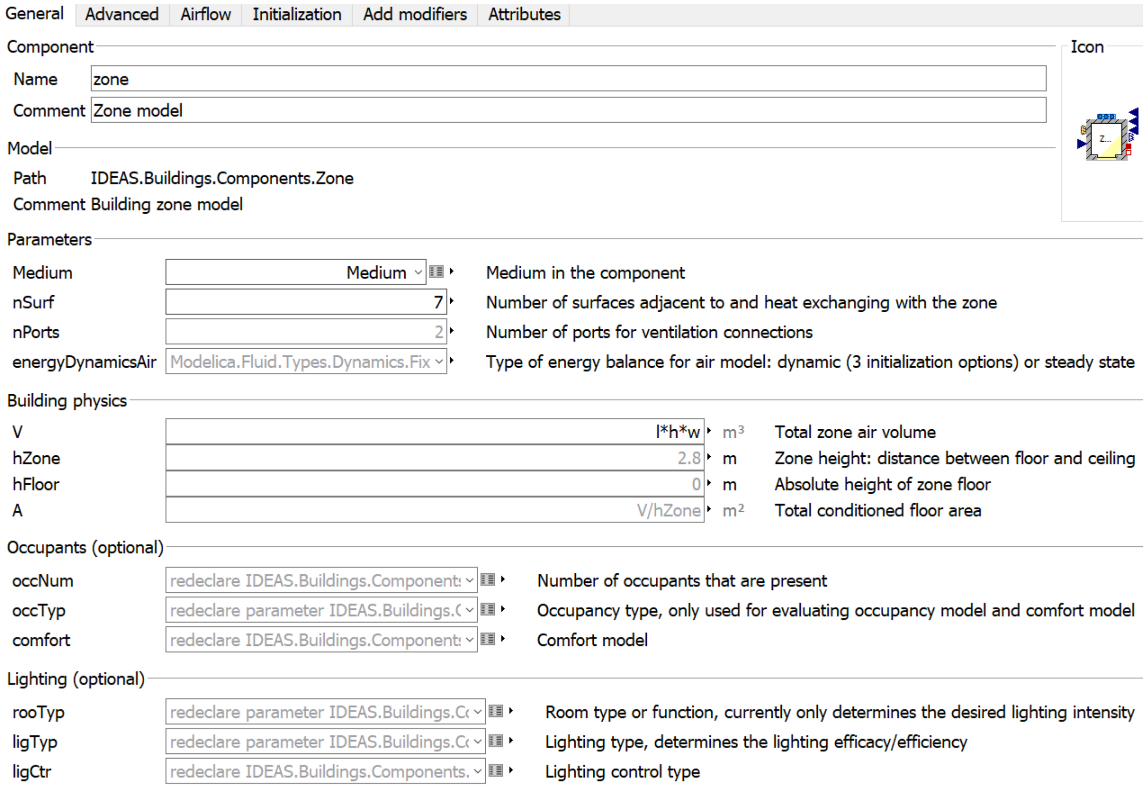


Multizone building models

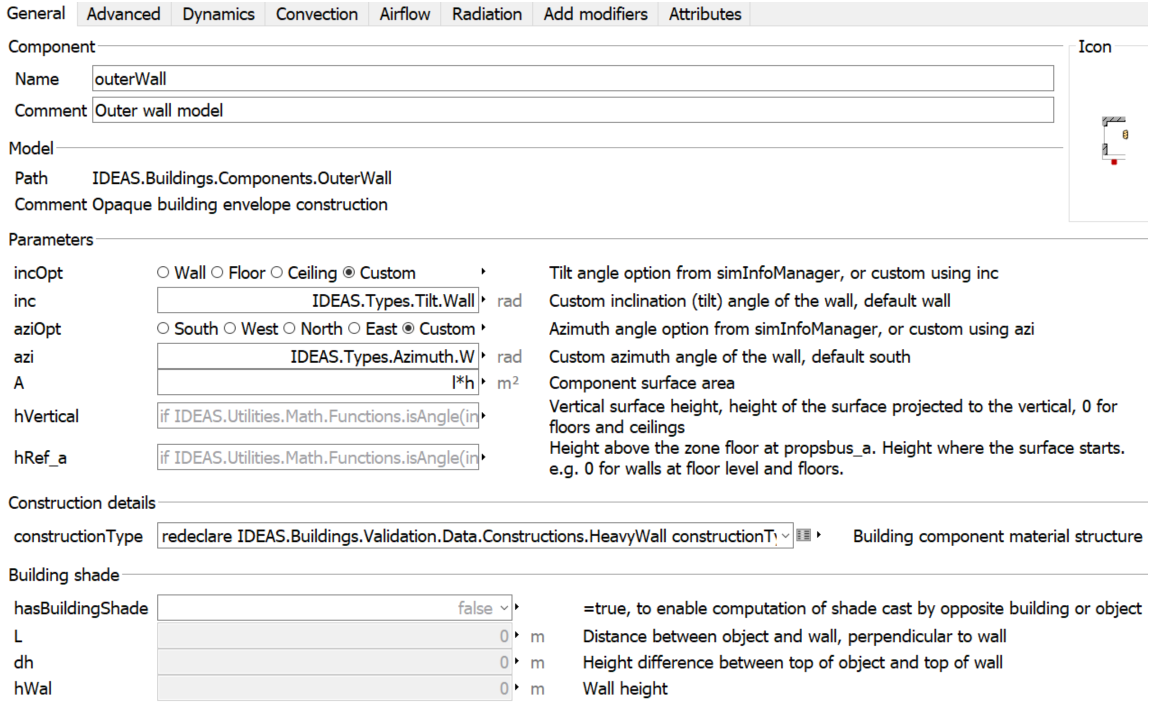


Parameters

Zone



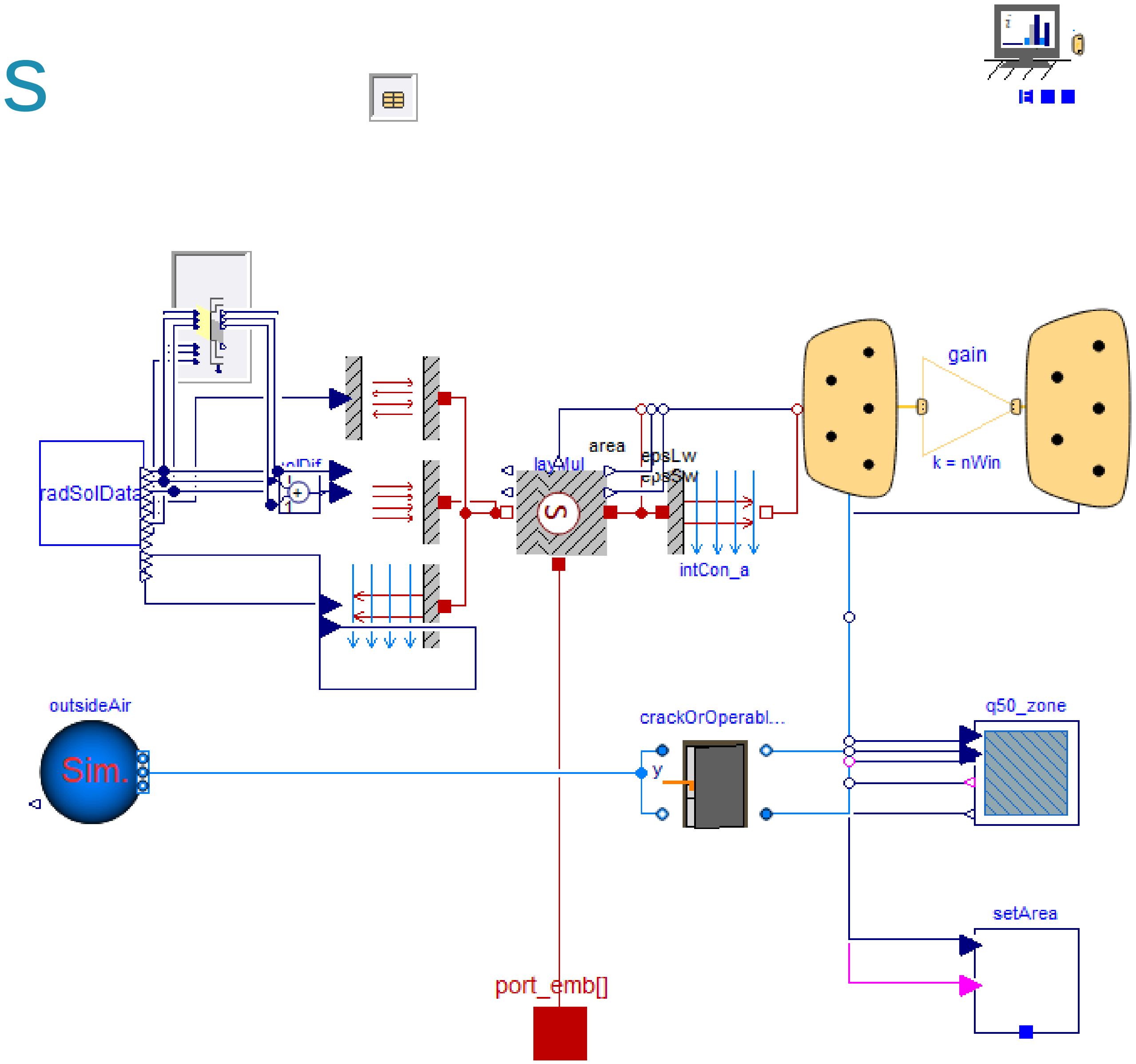
Wall



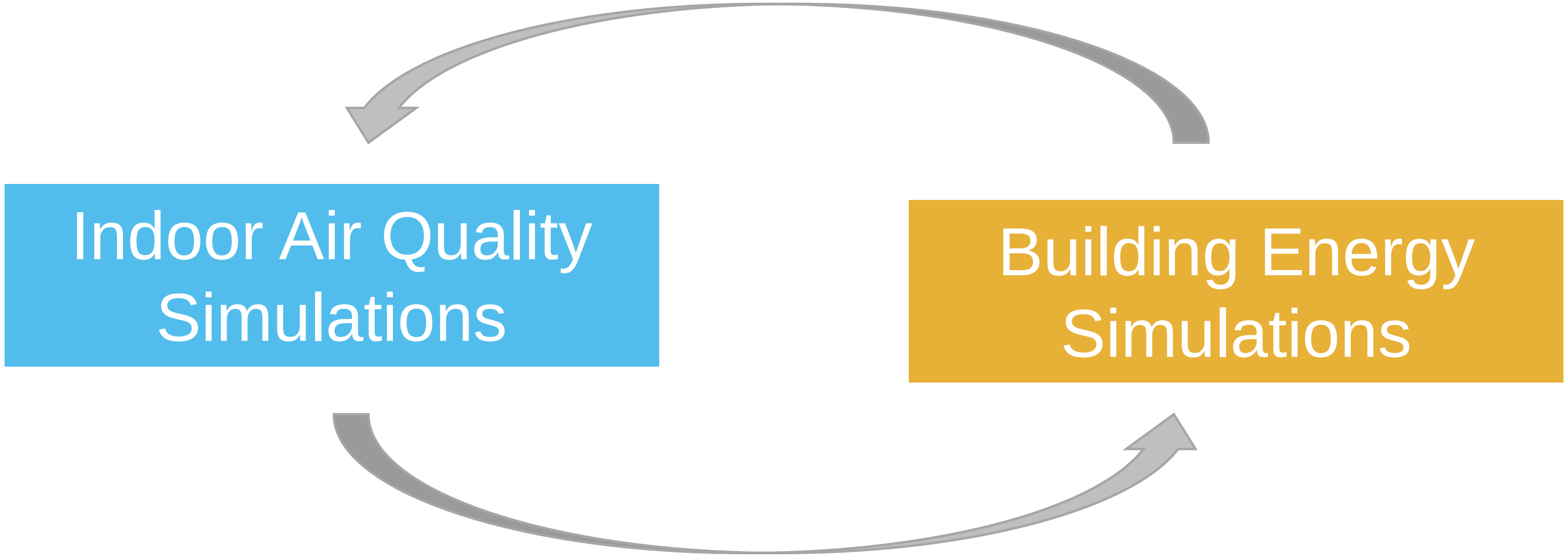
THERMAL SYSTEMS SIMULATION KU LEUVEN

Main building physics

- Conduction, thermal mass
- Convective heat transfer
- Radiative heat transfer
- Shortwave heat gains (incl. shading)
- Internal heat gains (occupants, lighting)
- **Integrated infiltration** and **interzonal airflow**

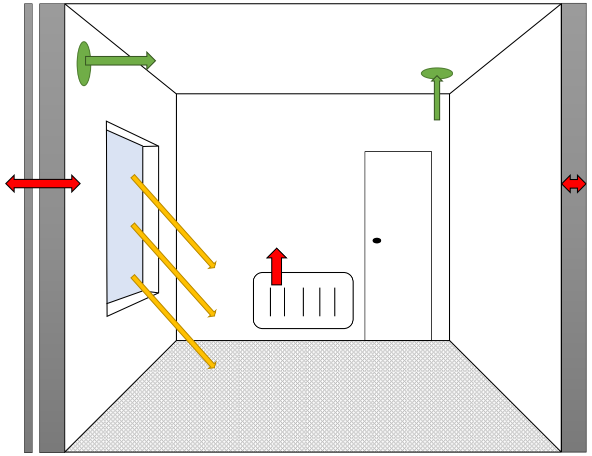


Integrated pressure-driven air flow modelling

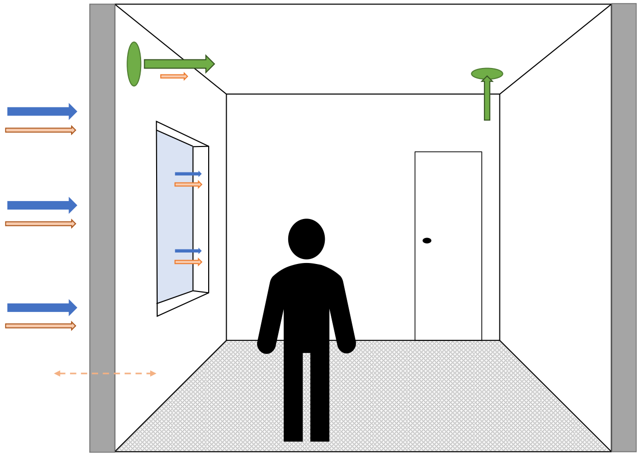


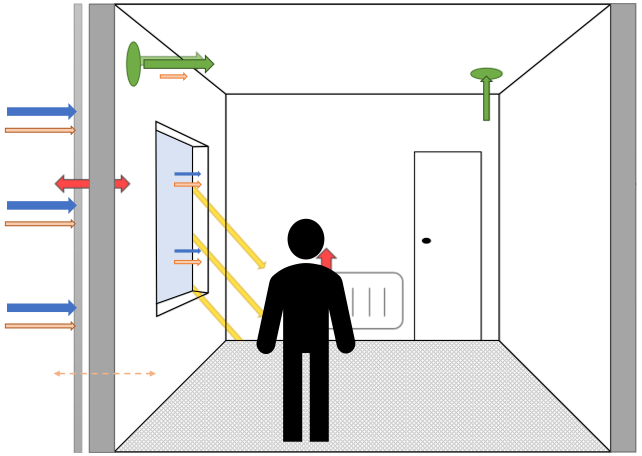
Integrated pressure-driven air flow modelling

Existing building energy simulation models
[IDEAS]



Airflow elements
[K. De Jonge -> IBPSA]





Integrated coupled energy
and
airflow solution

Exercise 3 – Building envelope model

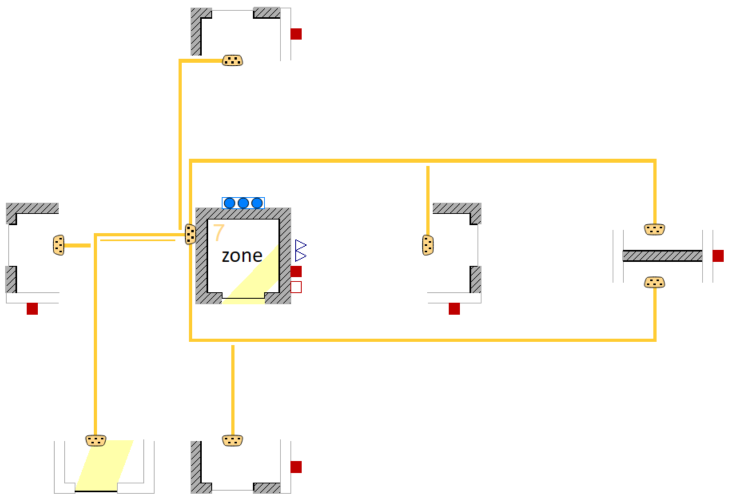
- See exercise sheet on Github
  https://github.com/open-ideas/__CrashCourse__/blob/master/Exercises/Exercise%203/Latex/Exercise3.pdf
- Make first assignment

1 Building model

Qualitative discussion We will now develop a simple building model that consists of one zone, four walls, a window, a floor and a ceiling. The zone dimensions are 8 m (walls with north and south orientation) by 4 m and the window is 3 m by 1.4 m. We use the default zone height (2.8 m). We use double glazing and a heavy wall, meaning they have high thermal mass.

Required models This step requires the main building envelope component models of IDEAS:

- `IDEAS.BoundaryConditions.SimInfoManager`
- `IDEAS.Buildings.Components.Zone`
- `IDEAS.Buildings.Components.OuterWall`
- `IDEAS.Buildings.Components.Window`
- `IDEAS.Buildings.Components.SlabOnGround`



Advanced Modelica concepts – ‘extend’

- Imports all equations from the extended model
- Allows modifications/extensions on top of that model

Valve 1

```
model TwoWayLinear "Two way valve with linear flow characteristics"
  extends BaseClasses.PartialTwoWayValveKv(phi=l + y_actual*(1 - l));
```

Valve 2

```
model TwoWayPolynomial "Two way valve with polynomial characteristic"
  extends IDEAS.Fluid.Actuators.BaseClasses.PartialTwoWayValveKv(
    phi=l + pol_y*(1 - l));

  parameter Real[:] c
    "Polynomial coefficients, starting with fixed offset";

protected
  constant Integer nP = 100
    "Number of points for initial algorithm test";
  Real pol_y = sum(c.*{y_actual^i for i in 0:size(c,1)-1})
    "Polynomial of valve control signal";
```

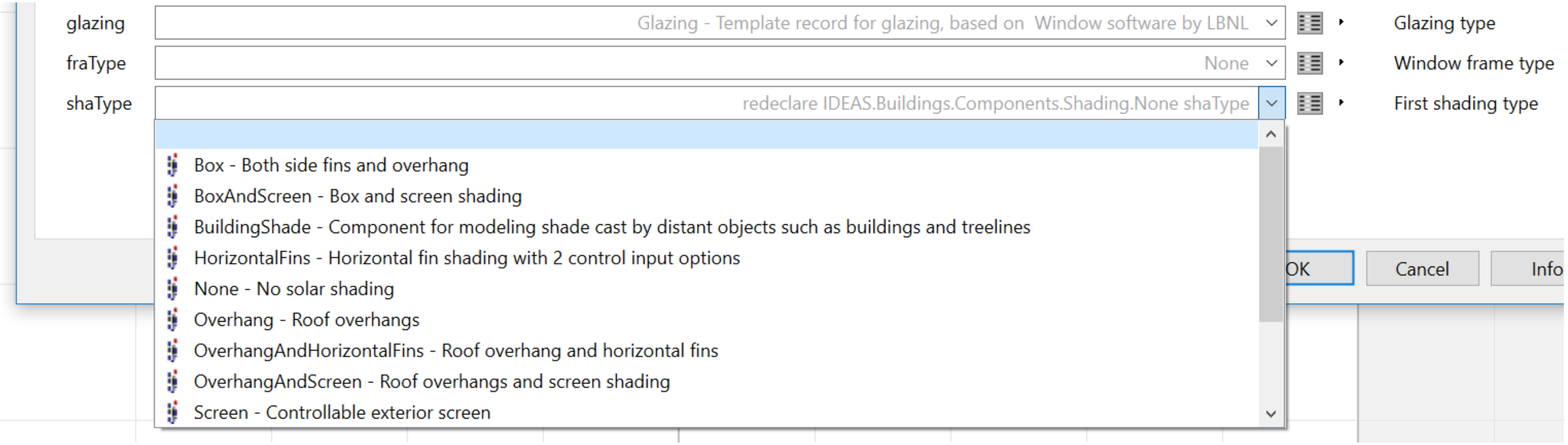
Advanced Modelica concepts – ‘replaceable’

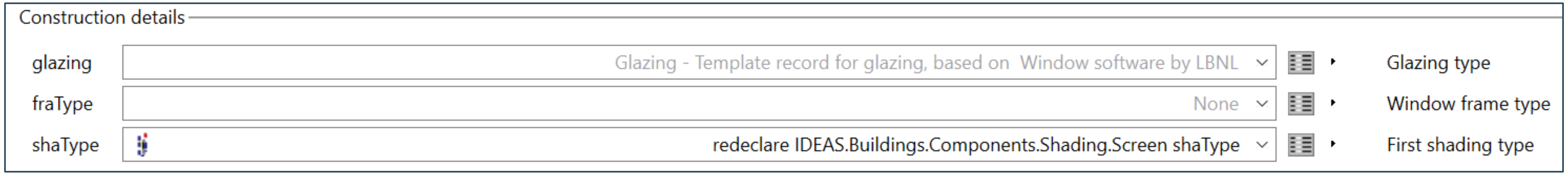
- ‘replaceable’: a component whose type can be changed
- ‘constrainedby’: specify a constraining type of a ‘replaceable’
- ‘redeclare’: changing the type of a replaceable component

https://mbe.modelica.university/components/architectures/replaceable/

Example: window shading type (assignment 2 of exercise 3)

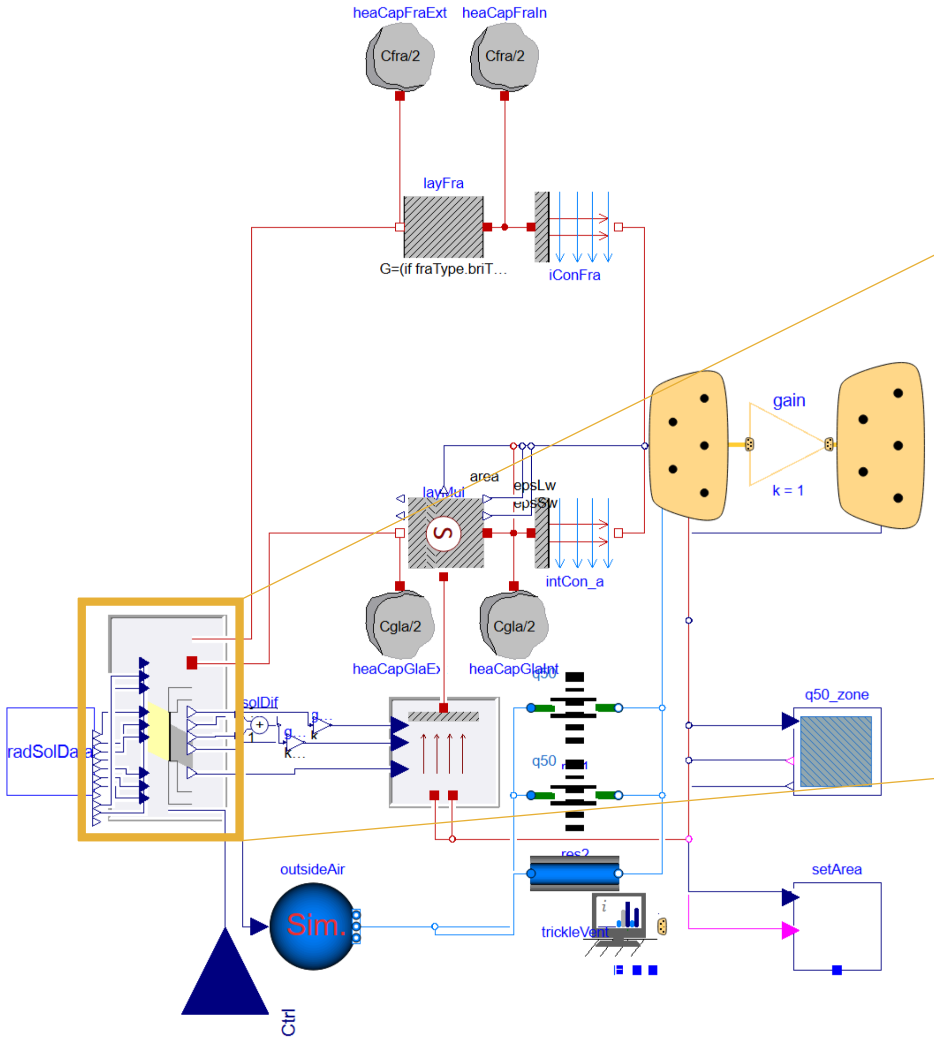
‘replaceable’ – window shading type






```
model Example2 "Adding closed screens"
  extends Example1(window(redeclare Buildings.Components.Shading.Screen shaType));
```

‘replaceable’ – window shading type

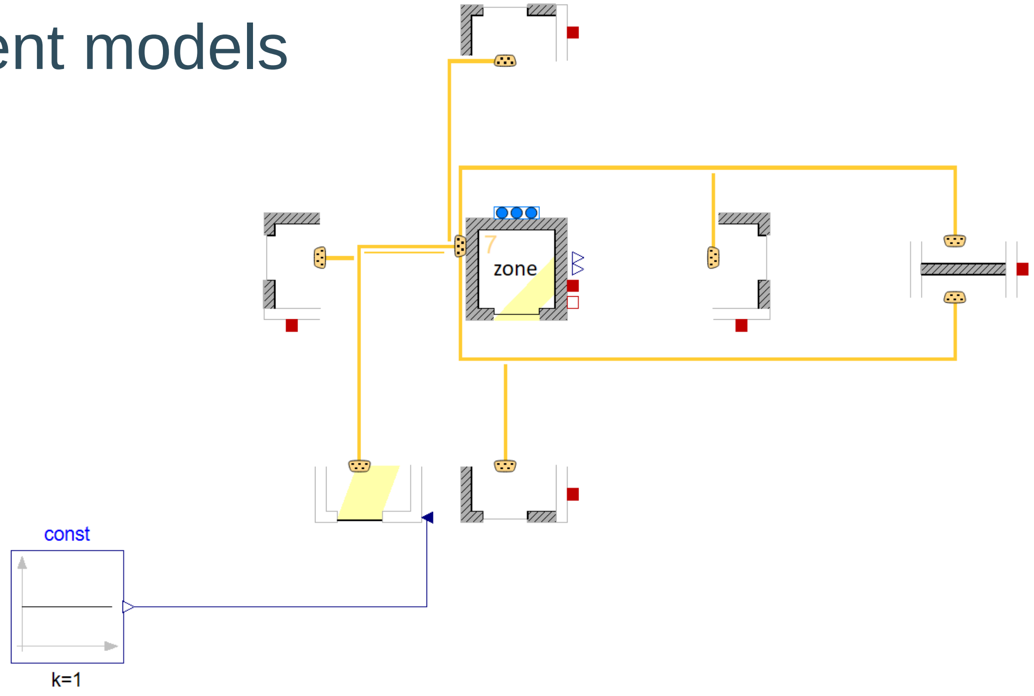



```
replaceable IDEAS.Buildings.Components.Shading.None shaType
  constrainedby Shading.Interfaces.PartialShading(
    haveFrame=fraType.present and A*frac > 0,
    A_frame = A * frac,
    A_glazing = A * (1 - frac),
    Tenv_nom = sim.Tenv_nom,
    epsLw_frame = fraType.mat.epsLw,
    epsLw_glazing = layMul.parEpsLw_b,
    epsSw_frame = fraType.mat.epsSw,
    g_glazing=glazing.g_value,
    inc = incInt,
    linCon = linExtCon or sim.linearise,
    linRad = linExtRad or sim.linearise,
    final azi=aziInt) "First shading type" ;
```

https://github.com/open-ideas/IDEAS/blob/master/IDEAS/Buildings/Components/Shading/Interfaces/PartialShading.mo

Exercise 3 – Building envelope model

- See exercise sheet on Github
  https://github.com/open-ideas/__CrashCourse__/blob/master/Exercises/Exercise%203/Latex/Exercise3.pdf
- Use 'extend' when moving to next assignment
- Use built-in replaceables of IDEAS component models (solar shading, occupancy)



Part 4: IDEAS – HVAC

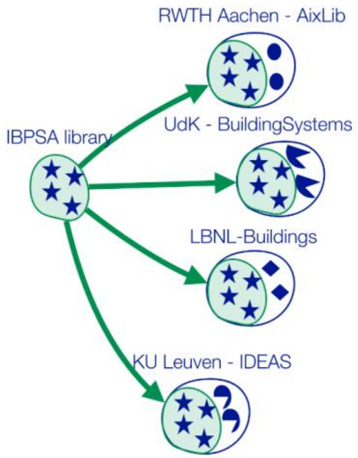
Lucas Verleyen

IDEAS – HVAC overview

International Building Performance Simulation Association

- IDEAS
  - User's Guide
  - Airflow
  - BoundaryConditions
  - Buildings
  - Controls
  - Fluid
  - LIDEAS
  - Media
  - Templates
  - ThermalZones
  - Utilities
  - Types
  - Examples
  - Experimental

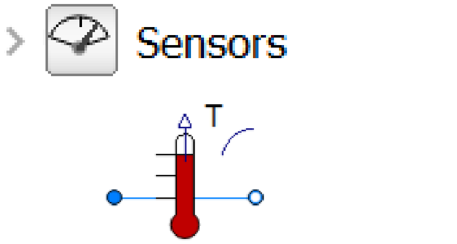
- Fluid
  - UsersGuide
  - Actuators
  - Chillers
  - Delays
  - FMI
  - FixedResistances
  - Geothermal
  - HeatExchangers
  - HeatPumps
  - Humidifiers
  - MassExchangers
  - MixingVolumes
  - Movers
  - Sensors
  - Sources
  - Storage
  - Taps
  - Types
  - Examples
  - Interfaces
  - BaseClasses

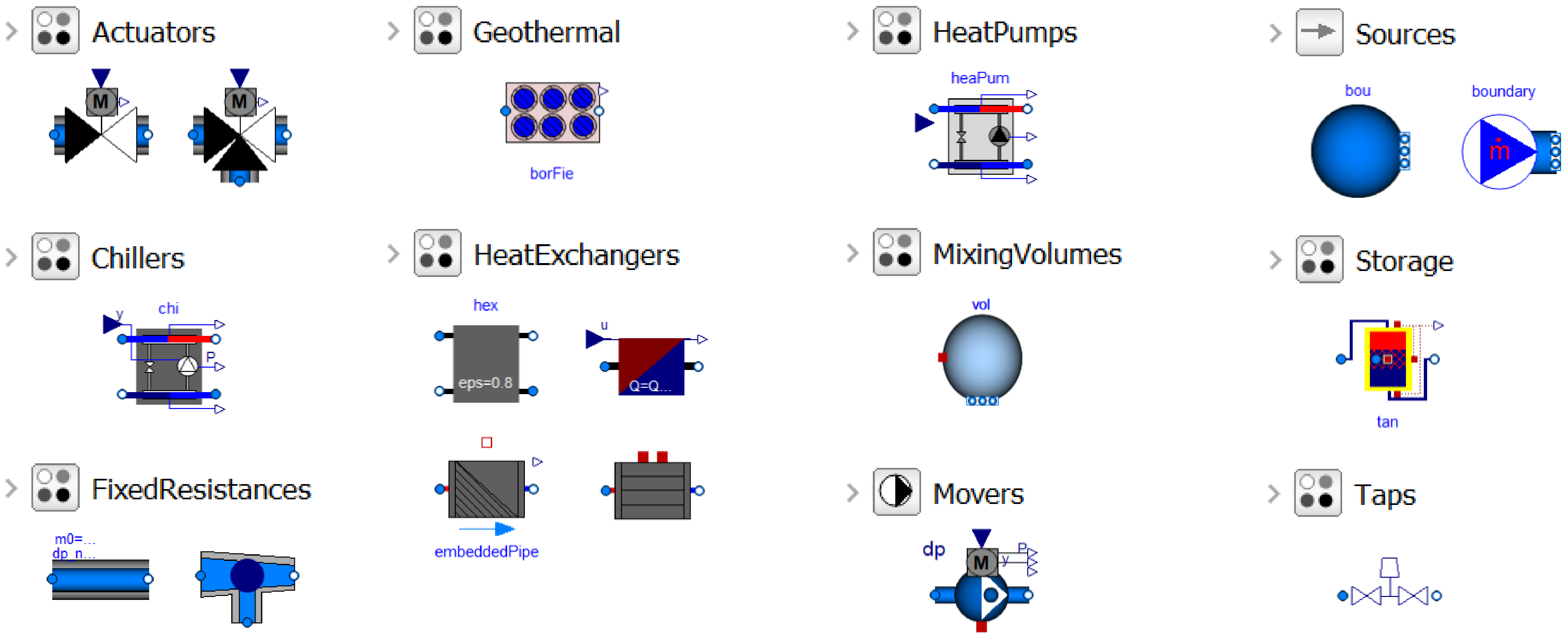


THERMAL SYSTEMS SIMULATION

KU LEUVEN

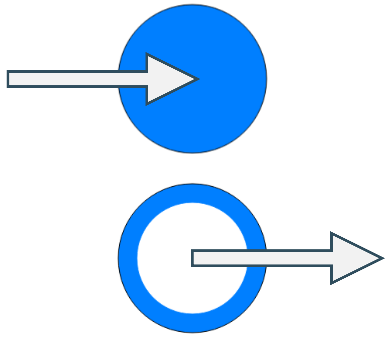
IDEAS – HVAC overview





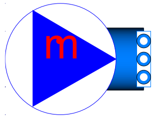
Modelica.Fluid.Interfaces.FluidPort

- Potential variable: pressure → unique value
- Flow variable: mass flow rate → Σ = 0
- Stream variable: enthalpy → characteristic of flow



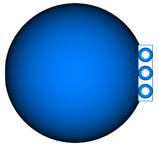
```
connector FluidPort
  "Interface for quasi one-dimensional fluid flow in a piping network (incompressible or comp

  replaceable package Medium = Modelica.Media.Interfaces.PartialMedium
    "Medium model" ;

  flow Medium.MassFlowRate m_flow
    "Mass flow rate from the connection point into the component";
  Medium.AbsolutePressure p "Thermodynamic pressure in the connection point";
  stream Medium.SpecificEnthalpy h_outflow
    "Specific thermodynamic enthalpy close to the connection point if m_flow < 0";
  stream Medium.MassFraction Xi_outflow[Medium.nXi]
    "Independent mixture mass fractions m_i/m close to the connection point if m_flow < 0";
  stream Medium.ExtraProperty C_outflow[Medium.nC]
    "Properties c_i/m close to the connection point if m_flow < 0";
end FluidPort;
```

Basic circuit



IDEAS.Fluid.Sources.MassFlowSource

→ Sets mass flow rate

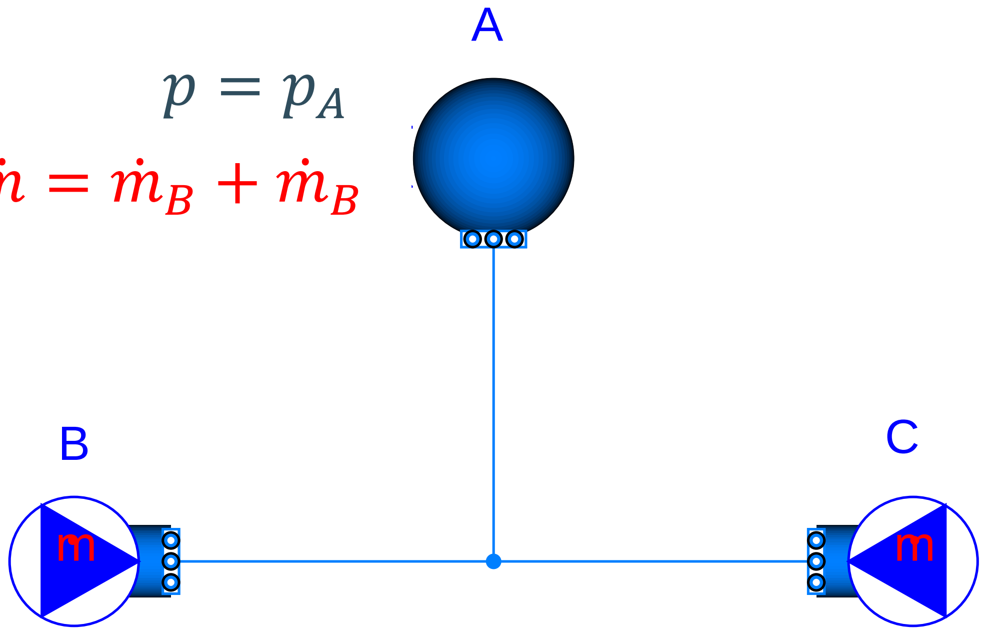


IDEAS.Fluid.Sources.Boundary_pT

→ Sets absolute pressure

$$p = p_A$$

$$\dot{m} = \dot{m}_B + \dot{m}_B$$



$$p = p_A$$

$$\dot{m} = \dot{m}_B$$

$$p = p_A$$

$$\dot{m} = \dot{m}_C$$

Pressure equals pressure set by Boundary A

Flow rate equals sum of flow rates set by Source B and Source C

Illegal circuits

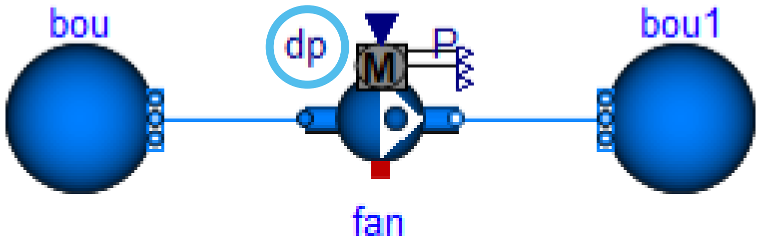
The problem is structurally singular

No mass flow rate

& problem when $p_{bou} \neq p_{bou1}$

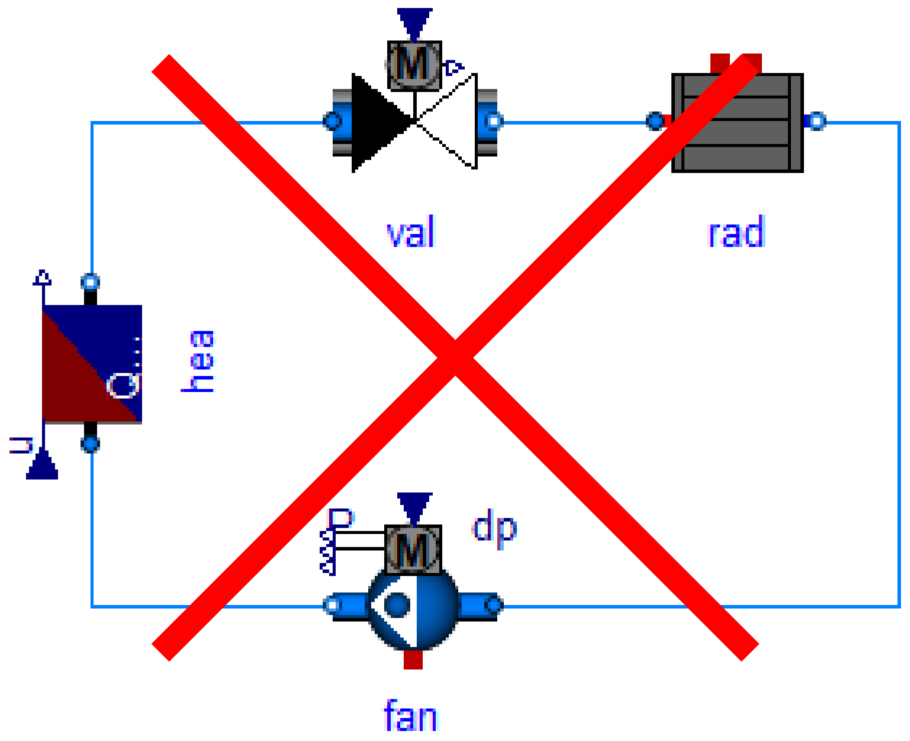
No absolute pressure

& problem when $\dot{m}_{boundary} \neq -\dot{m}_{boundary1}$

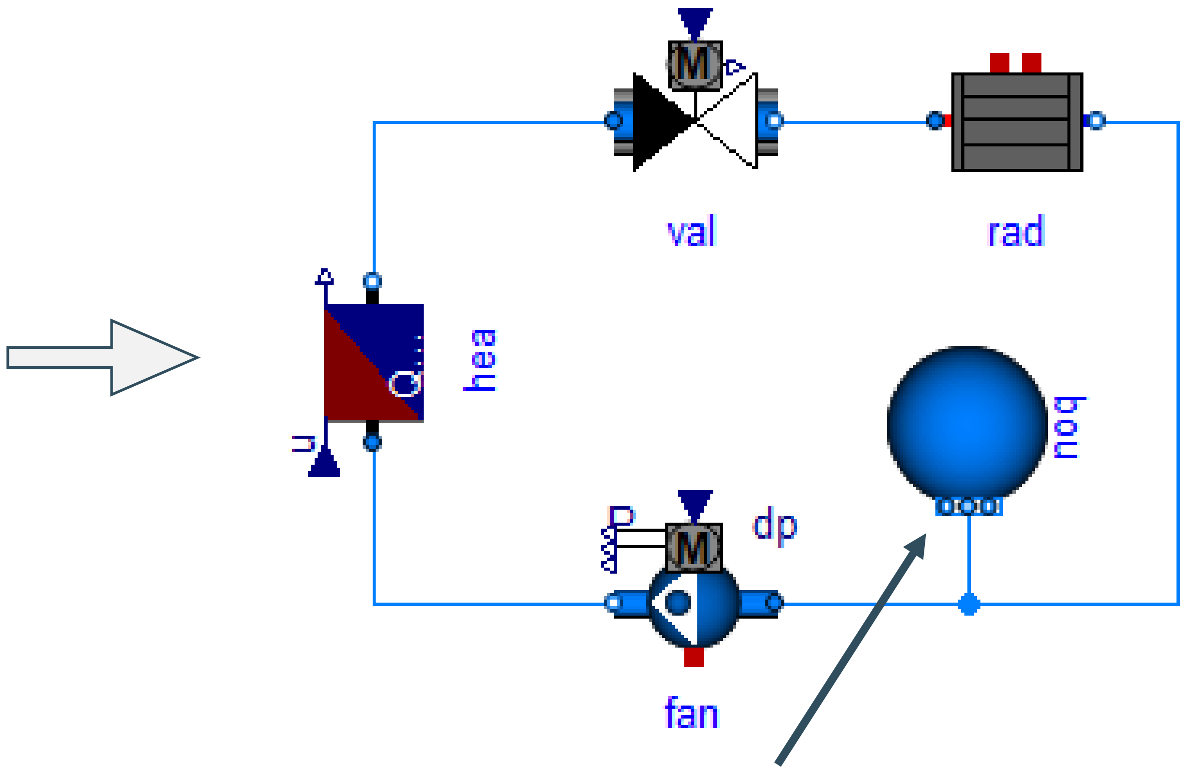


Problem when $p_{bou} + \Delta p_{fan} \neq p_{bou1}$

Simple HVAC circuit



Only pressure drops



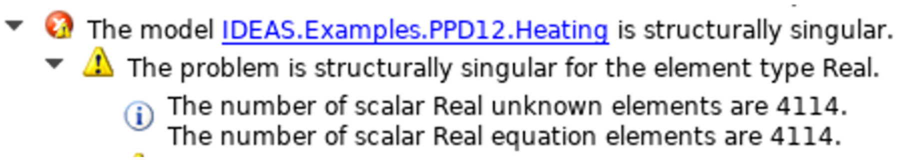
Sets absolute pressure

Singularity error

- Modelica is a generic modelling language → errors are general (unclear)
  i.e. not tailored to buildings / HVAC
- Important language requirements:
  - # equations = # variables
  - Equations should not contradict each other (e.g. adding both x = 1 and x = 0)
  - Solution to equations should exist (e.g. don't try $x^2 = -1$)
- These requirements are abstract and are hidden from the user by library developers. However, they pop up sometimes, leading to unclear errors like:

The model IDEAS.Examples.PPD12.Heating is structurally singular.
The problem is structurally singular for the element type Real.
The number of scalar Real unknown elements are 4114.
The number of scalar Real equation elements are 4114.

Singularity error



- You don’t have to worry about component models when dragging and dropping
- Singularity error occurs, when:
  - Dangling connectors / more than 1 connection
  - # equations ≠ # variables
  - Conflicting equations, equations without real solutions
  - Infinite number of solutions (e.g. no absolute pressure set)
- IDEAS.Buildings is fairly robust as long as each zone propsBus connector is connected to exactly one surface propsBus connector.
- IDEAS.Fluid pressure drop circuits can require some experience:
  - Set absolute pressure in flow circuits
  - Don’t oversimplify pressure drops

Further reading

- F. Jorissen, G. Reynders, R. Baetens, D. Picard, D. Saelens, and L. Helsen. Implementation and Verification of the IDEAS Building Energy Simulation Library. *Journal of Building Performance Simulation*, **11** (6), 669-688, 2018. doi: 10.1080/19401493.2018.1428361.
- F. Jorissen, M. Wetter, and L. Helsen. Simulation Speed Analysis and Improvements of Modelica Models for Building Energy Simulation. In 11th International Modelica Conference, pages 59–69, Paris, 2015. doi: 10.3384/ecp1511859.
- F. Jorissen, M. Wetter, and L. Helsen. Simplifications for Hydronic System Models in Modelica. *Journal of Building Performance Simulation*, **11** (6), 639-654, 2019.
- F. Jorissen. *Toolchain for Optimal Control and Design of Energy Systems in Buildings*. PhD thesis, Arenberg Doctoral School, KU Leuven, April 2018

Exercise 4 – HVAC model

- See exercise sheet on Github
  https://github.com/open-ideas/__CrashCourse__/blob/master/Exercises/Exercise%204/Latex/Exercise4.pdf
- Start from building envelope of step 5 (*see IDEAS.Examples.Tutorial*):
  1. Add a geothermal heat pump heating system
  2. Add a heat pump controller
  3. Compute the energy use and export it in a json file
  4. Add a CO_2-controlled ventilation system