Exercise 2: IDEAS SimpleHouse

Filip Jorissen, Damien Picard
*Jelger Jansen, Iago Cupeiro Figueroa, Lucas Verleyen

KU Leuven, September 20, 2024

Introduction

The goal of this exercise is to become familiar with Modelica and the IDEAS library. Since the IDEAS library components are typically used by combining several components graphically, the use of equations falls outside of the scope of this exercise.

For this exercise you will create a model of a simple house, consisting of a heating system, one building zone and a ventilation model. The exercise starts from a template file that should not produce any errors. This file will be extended in several steps, adding complexity. In between each step the user should be able to simulate the model, i.e. no errors should be produced and simulation results may be compared.

Prerequisites are that you should have the latest version of Dymola installed. You should have a working compiler, and a license. Dymola can be downloaded from this link. Installation instructions for Dymola and a C compiler can be found here. The latest version from the IDEAS library should be downloaded and opened in Dymola. To verify your installation, try to simulate `IDEAS.Fluid.Actuators.Dampers.Examples.Damper` by opening the simulation tab (tab bar at the top) and by clicking *Simulate* (). Finally, download *SimpleHouse-Template.mo* from the CrashCourse Github repository and load it into Dymola.

In the following sections the simple house model is discussed in several steps. The graphical representation of the final model is given in Figure 1. Each step first qualitatively explains the model part. Secondly, the names of the required IDEAS models are listed. Thirdly, we provide high-level instructions of how to set up the model. If these instructions are not clear immediately, have a look at the model documentation and at the type of connectors the model has, try out some things, make an educated guess, etc. Finally, we provide reference results that allow you to check if your implementation is correct. Depending on the parameter values that you choose, results may differ.

*Review

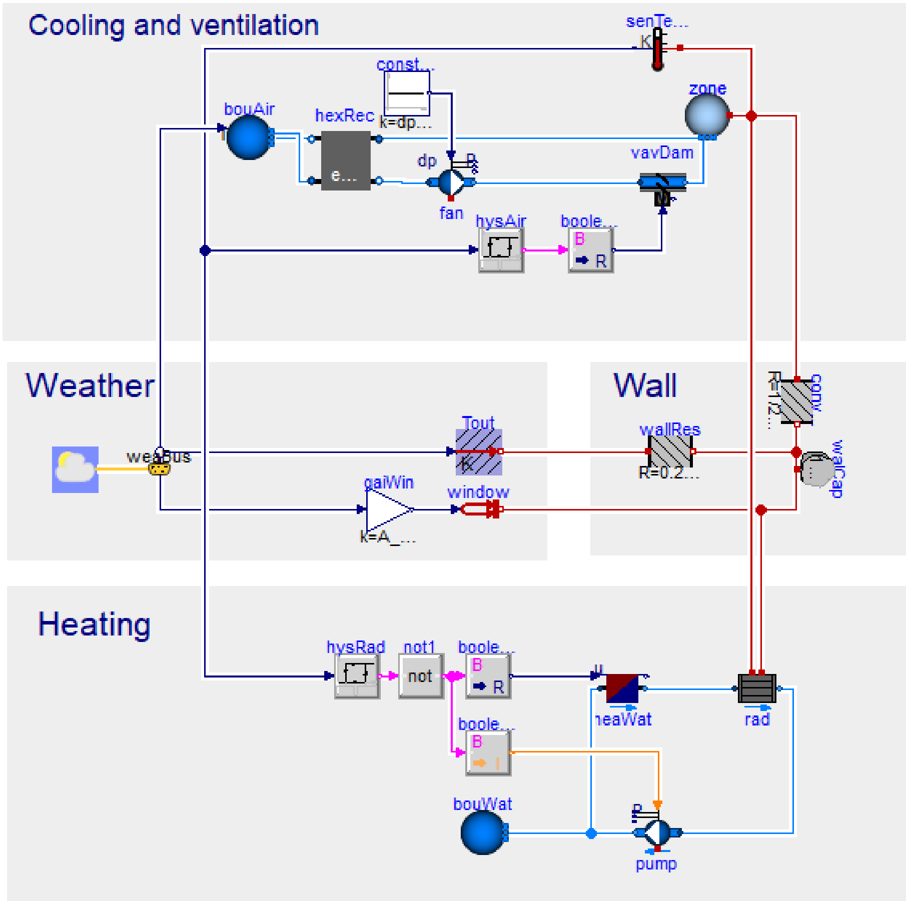

Figure 1: Graphical representation of the final simple house model.

1 Building wall model

Qualitative discussion Even though one of the goals of IDEAS is to provide detailed building envelope models, a very simple building envelope model will be constructed manually using thermal resistors and heat capacitors. The house consists of a wall represented by a single heat capacitor and a thermal resistor. The thermal resistor and boundary temperature are already added in the template. The wall has a surface area of $A_{wall} = 100\ m^2$, a thickness of $d_{wall} = 25\ cm$, a thermal conductivity of $k_{wall} = 0.04\ W/(m \cdot K)$, a density of $\rho_{wall} = 2000\ kg/m^3$ and a specific heat capacity $c_p = 1000\ J/(kg \cdot K)$ [1]. The conductive thermal resistance value of a wall may be computed as $R = d/(A \cdot k)$. The heat capacity value of a wall may be computed as $C = A \cdot d \cdot c_p \cdot \rho$.

Required models In this first step only the Modelica Standard Library (MSL) model `Modelica.Thermal.HeatTransfer.Components.HeatCapacitor` is required.

Connection instructions Connect the heat capacitor to the thermal resistor.

Reference result If you correctly added the model of the heat capacitor, connected it to the resistor and added the parameter values for C and R [2], then you should be able to simulate the model. To do this, go to the simulation tab (tab bar at the top), open the simulation options *Setup*, and set the model *Stop time* to 1e6 seconds. You can now simulate the model. You can plot individual variables values by clicking on their name in the variable browser on the left. Now plot the wall capacitor temperature value T. It should look like Figure 2.

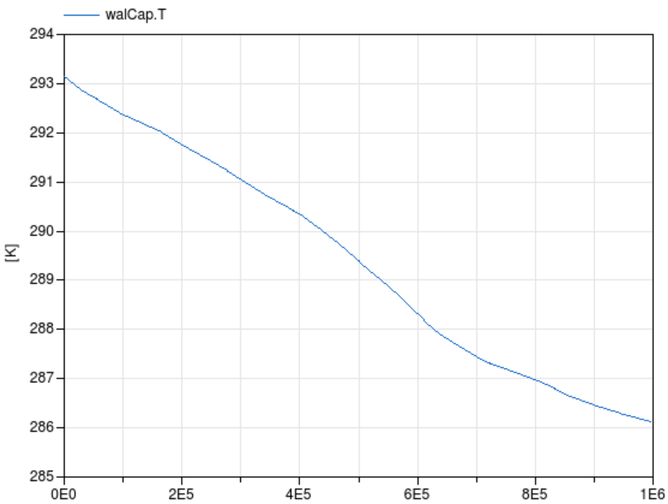

Figure 2: Wall temperature as function of time.

[1] We suggest to declare these parameters in the model via the text editor tab, similar to the example, but this is not required.
[2] Double-click on a component to see a list of its parameters. Gray values indicated default values.

2 Building window model

Qualitative discussion The window has a surface area of 2 m^2. As a simplified approach in this example, we are injecting two times the direct normal solar irradiance as heat onto the inside of the wall[3].

Required models

- `Modelica.Blocks.Math.Gain`
- `Modelica.Thermal.HeatTransfer.Sources.PrescribedHeatFlow`

Connection instructions To be able to use the value of the outdoor solar irradiance you will need to access our weather data reader. To do this, make a connection to the `weaBus` block. In the dialog box select <*add variable*> and here select *HDirNor*, which is the direct solar irradiance on a surface of 1 m^2, perpendicular to the sun rays. Set the gain factor k to 2, in order to get the solar irradiance through the window of 2 m^2. Make a connection with the `PrescribedHeatFlow` as well. This block makes the connection between the heat flow from the gain, represented as a real value, and a heat port that is compatible with the connections of the thermal capacitance and resistance.

Reference result The result with and without the window model is plotted in Figure 3.

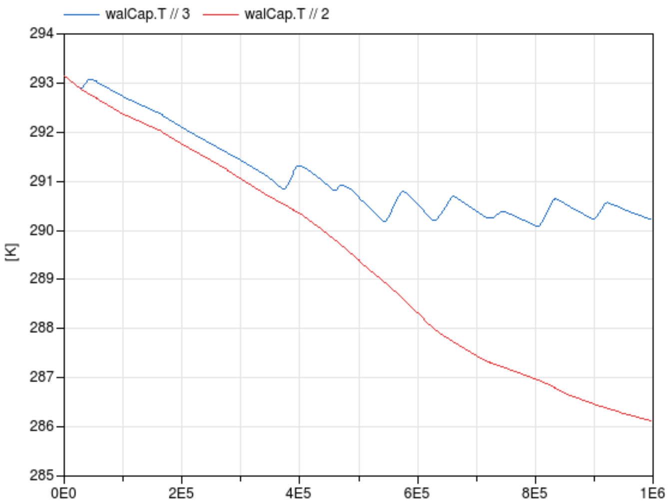

Figure 3: Wall temperature as function of time, with (blue) and without (red) window.

[3]This approach corresponds to a window which moves with the position of the sun and transmits only the direct radiation to 100% and neglects diffuse radiation.

3 Air model

Qualitative discussion To increase the model detail we now add an air model assuming the zone is 8 m x 8 m x 3 m in size. The air will exchange heat with the wall. This may be modelled using a thermal resistance representing the convective heat resistance, where $R_{conv} = \frac{1}{hA}$. A represents the heat exchange surface area and h represents the convective heat transfer coefficient and is equal to $h = 2\ W/(m^2 \cdot K)$.

Required models

- `IDEAS.Fluid.MixingVolumes.MixingVolume`
- `Modelica.Thermal.HeatTransfer.Components.ThermalResistor`

Connection instructions The `MixingVolume` *Medium* parameter contains information about the type of fluid and its properties that should be modelled by the `MixingVolume`. Set [4] its value to *MediumAir*, which is declared in the template. For the nominal mass flow rate you may assume a value of 1 kg/s. You will have to change this value once you add a ventilation system to the model (part 6). Finally, set the *energyDynamics* of the `MixingVolume`, which can be found in the *Dynamics* tab of the model parameter window, to *FixedInitial*. Your model should now look like Figure 4.

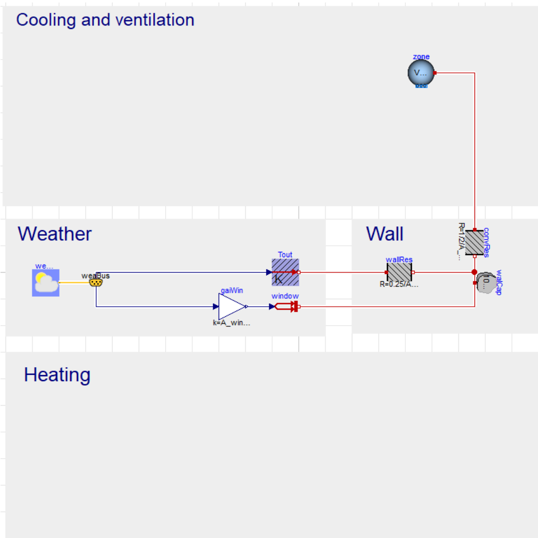

Figure 4: Simple house model with a wall, a window, and an air model.

Reference result The result with and without the air model is plotted in Figure 5.

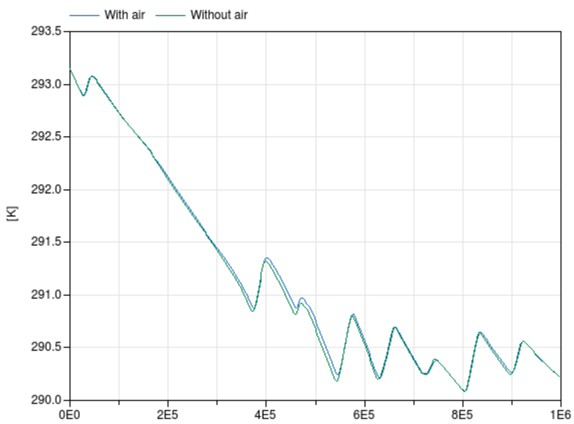

Figure 5: Wall temperature as function of time , with (blue) and without (green) air model.

[4]Click the right small arrow and 'edit text' in the parameter dialog box, or use the Modelica text view.

4 Heating model

Qualitative discussion The wall temperature (and therefore the room temperature) is quite low. In this step a heating system is added to resolve this. It consists of a radiator, a pump and a heater. The radiator has a nominal power of 3 kW for an inlet and outlet temperature of the radiator of $60°C$ and $40°C$ respectively. The pump has a (nominal) mass flow rate of 0.1 kg/s. Since the heating system uses water as a heat carrier fluid, the media for the models should be set to *MediumWater*.

Required models

- `IDEAS.Fluid.HeatExchangers.Radiators.RadiatorEN442_2`
- `IDEAS.Fluid.HeatExchangers.HeaterCooler_u`
- `IDEAS.Fluid.Movers.FlowControlled_m_flow`
- `IDEAS.Fluid.Sources.Boundary_pT`
- `Modelica.Blocks.Sources.Constant`

Connection instructions The radiator contains one port for convective heat transfer and one for radiative heat transfer. Connect both in a reasonable way. Since the heating system uses water as a heat carrier fluid, the media for the models should be set to *MediumWater*. The `Boundary_pT` model needs to be used to set an absolute pressure somewhere in the system. Otherwise the absolute pressure in the system is undefined. Pressure difference modelling may be disregarded since the chosen pump sets a fixed mass flow rate regardless of the pressure drop. Set the heater input to 1, meaning that it will produce 1 time its nominal power.

Reference result The result of the air temperature is plotted in Figure 6. The temperature rises very steeply since the wall is relatively well insulated ($k = 0.04\ W/(m \cdot K)$) and the heater is not disabled when it becomes too warm.

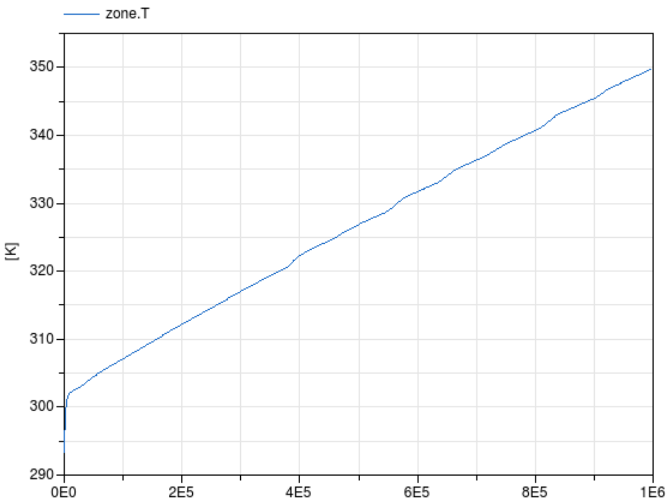

Figure 6: Air temperature as function of time.

5 Heating controller model

Qualitative discussion Since the zone becomes too warm, a controller is required that disables the heater when a set point is reached. We will implement a hysteresis controller with a set point of $295.15 \pm 1K$ (21-23 $^{\circ}C$). A temperature sensor will measure the zone air temperature.

Required models

- `Modelica.Blocks.Logical.Hysteresis`
- `Modelica.Blocks.Logical.Not`
- `Modelica.Blocks.Math.BooleanToReal`
- `Modelica.Thermal.HeatTransfer.Sensors.TemperatureSensor`

Connection instructions The heater modulation level should be set to one when the heater is on and to zero otherwise. Your model should now look like Figure 7.

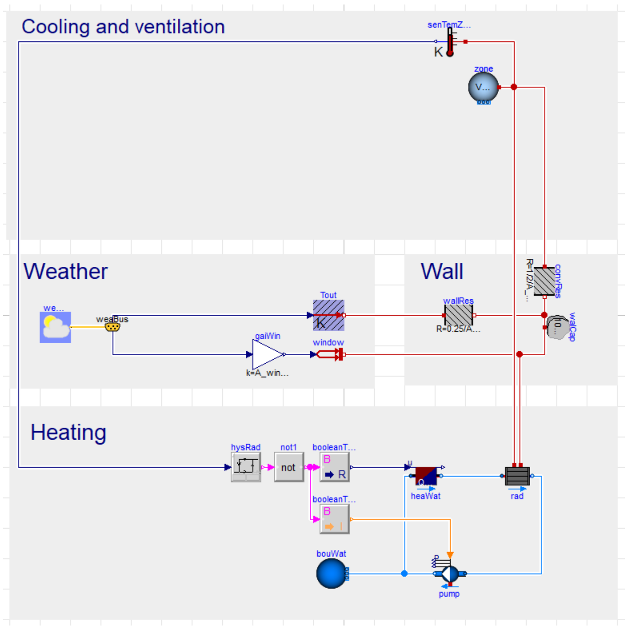

Figure 7: Simple house model with heating controller model.

Reference result Figure 8 shows the air temperature when the controller is added.

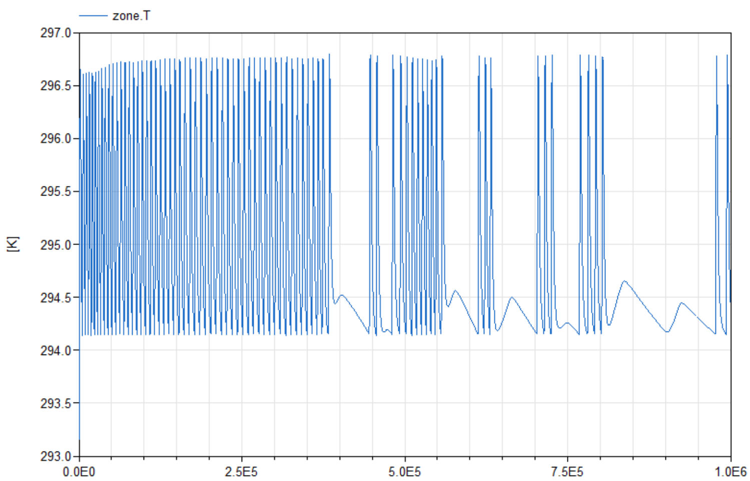

Figure 8: Air temperature as function of time.

6 Ventilation model

Qualitative discussion For this last exercise, we first increase the window size from 2 m^2 to 6 m^2. We will add a ventilation model that allows to perform free cooling using outside air when solar irradiation heats up the room too much. The system consists of a fan, a damper, a controller with an air temperature set point between 23°C and 25°C and a heat recovery unit with a constant effectivity of 85%. The damper and fan have a nominal pressure drop/raise of 200 Pa. The heat recovery unit has a nominal pressure drop of 10 Pa at both sides. The nominal mass flow rate of the complete cooling system is 0.1 kg/s.

Required models Use some of the previously used models and in addition to this:

- `IDEAS.Fluid.HeatExchangers.ConstantEffectiveness`
- `IDEAS.Fluid.Movers.FlowControlled_dp`
- `IDEAS.Fluid.Actuators.Dampers.VAV`

Connection instructions Connect the components such that they exchange mass (and therefore also energy) with the `MixingVolume` representing the zone air. Add a `boundary_pT` to draw air from the environment. Enable its temperature input and connect it to the $TDryBul$ variable in the weather data reader. Also reconsider the nominal mass flow rate parameter value in the `MixingVolume` given the flow rate information of the ventialtion system. The final model will look like Figure 1.

Reference result Figure 9 shows the result.

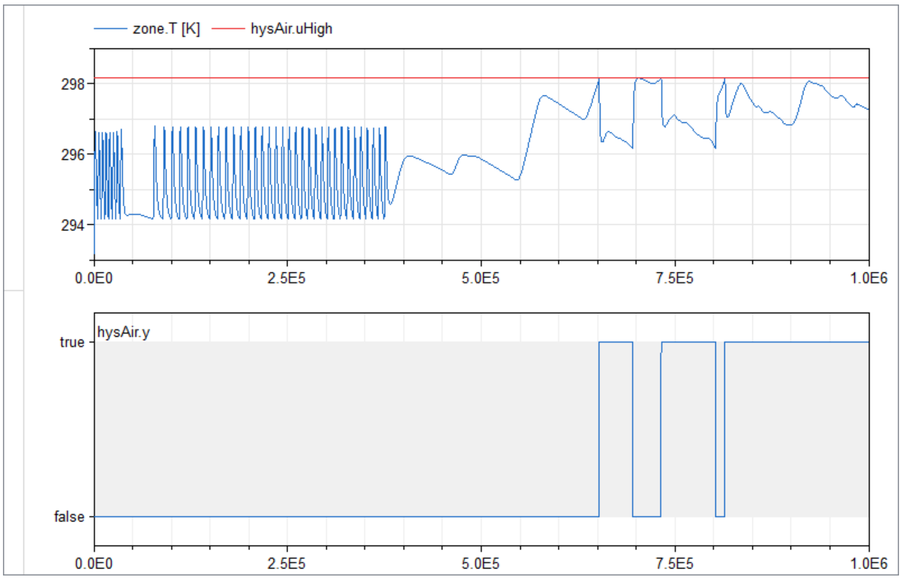

Figure 9: Top: Air temperature and upper bound of ventilation hysteresis controller as function of time. . . . . . . Bottom: Ventilation hysteresis output signal as function of time.