# Modelica Library for Building Heating, Ventilation and Air-Conditioning Systems

Michael Wetter
Simulation Research Group, Building Technologies Department,
Environmental Energy Technologies Division, Lawrence Berkeley National Laboratory,
Berkeley, CA 94720, USA

## Abstract

This paper presents a freely available Modelica library for building heating, ventilation and air conditioning systems. The library is based on the `Modelica.Fluid` library. It has been developed to support research and development of integrated building energy and control systems. The primary applications are controls design, energy analysis and model-based operation.

The library contains dynamic and steady-state component models that are applicable for analyzing fast transients when designing control algorithms and for conducting annual simulations when assessing energy performance. For most models, dimensional analysis is used to compute the performance for operating points that differ from nominal conditions. This allows parameterizing models in the absence of detailed geometrical information which is often impractical to obtain during the conceptual design phase of building systems.

In the first part of this paper, the library architecture and the main classes are described. In the second part, an example is presented in which we implemented a model of a hydronic heating system with thermostatic radiator valves and thermal energy storage.

*Keywords: building energy systems, heating, ventilation, air-conditioning, controls*

## 1 Introduction

Buildings account for a large fraction of carbon dioxide emission and energy consumption. For example, in the United States, buildings account for 38% of total carbon dioxide emissions, 70% of electricity consumption and 50% of natural gas consumption, while less than 2% of the building sector's energy consumption is from renewable energy [5]. Several government bodies and professional societies have set the goal to mandate Net Zero Energy Buildings (ZEB) in the next 15 to 20 years. Such buildings should produce as much energy as they consume on an annual average. The challenges inherent in designing and operating high performance buildings and ZEBs demand a number of breakthroughs, both in technology, including software and information technology, and in the fundamental knowledge of optimizing whole building performance through integration and component operation [4]. To accelerate innovation towards ZEB, we started the development of a freely available open-source Modelica library for building energy and control systems that is available from `http://simulationresearch.lbl.gov`. For the early applications, we are particularly interested in enabling:

1. Rapid prototyping of new building components and systems.

2. Development of advanced control systems.

3. Reuse of models during operation for energy-minimizing controls, fault detection and diagnostics.

The current development is focused on the development of models for building heating, ventilation and air-conditioning equipment and their control systems, as opposed to the building envelope. However, the library contains an interface that allows coupling Modelica with the EnergyPlus whole building energy simulation program [3] for co-simulation. This allows the use of the detailed, extensively validated EnergyPlus program for modeling the heat transfer through the envelope and the daylight illuminance in rooms, while using Modelica for rapid prototyping and analysis of innovative energy and control systems. The coupling is done through the Building Controls Virtual Test Bed (BCVTB) that is currently under development at the Lawrence Berkeley National Laboratory [13]. The BCVTB is a middleware that is

based on Ptolemy II [1, 6]. Ptolemy II is an open-source software framework to study modeling, simulation, and design of concurrent, real-time, embedded systems, with focus on the assembly of concurrent components and the use of heterogeneous mixtures of models of computation that govern the interactions between components. Ptolemy II allows modeling, analysis and simulation of systems that communicate and interact in a variety of ways such as synchronous or asynchronous, buffered or unbuffered. The BCVTB adds functionalities to Ptolemy II that allow coupling Modelica, EnergyPlus, MATLAB and Simulink to Ptolemy II for data exchange during the simulation. Interfaces to actual building control systems will be added in the future to enable use of models during the operation of the building.

The here described `Buildings` library is based on the `Modelica.Fluid` library 1.0 that uses the new concept of stream variables [8]. The `Modelica.Fluid` library provides a set of component models for one-dimensional thermo-fluid flow in networks of pipes. It demonstrates how to implement fluid flow component models that may have flow friction, heat and mass transfer. The library demonstrates how to deal with difficult design issues such as connector design, handling of flow reversal and initialization of states in a computationally efficient way. While many classes of this library can be used for our application domain, we provide in the `Buildings` library classes that extend and augment models from `Modelica.Fluid` where applicable, using the same modeling approach as `Modelica.Fluid`. Our library implements classes that are specifically needed for energy and control analysis at the whole building system level, as opposed to the development of individual equipment such as a refrigeration engine. Since our applications typically involve annual simulations, we generally do not model two-phase flow and refrigerant distribution in vapor compression cycles such as in [11, 10], although our library can be coupled to more detailed models.

When designing building energy systems, decisions that significantly affect building performance are typically done in the early design stage prior to specific equipment selection and prior to sizing the duct and piping networks. To enable assessing the performance of building energy systems when such detailed information is not yet available, many models in the `Buildings` library are implemented using dimensional analysis. Using dimensional analysis allows computation of a component's performance over a range of operating conditions based on a user-specified nominal operating point and its nominal performance, which is then scaled to different operating points based on laws of physics. Equations (4) and (7) are examples of such models.

Other developments for building energy system simulations in Modelica include the `ATPlus` library [9, 7]. We chose however to base our library on `Modelica.Fluid` as it allows modeling fluids with multiple compositions and trace substances. The first is important for humidity control in buildings while the second is needed to assess indoor air quality, for example, in variable air volume flow systems.

In Section 2, we will discuss the architecture of the `Buildings` library. Section 2.1.3 discusses the main partial models, and Sections 2.1.1 to 2.1.11 discuss the main packages and their models. In Section 3, we present an illustrative example in which we modeled a hydronic heating system. For other examples that include controls design, we refer to [12].

# 2 Library for building energy and control systems

Version 0.6.0 of the Buildings library consists of 73 models and blocks, and 26 functions that are public and non-partial.

## 2.1 Packages of the `Buildings` Library

The `Buildings` library is organized into the packages shown in Fig. 1. Components in these packages augment components from the Modelica Standard Library and from the `Modelica.Fluid` library. Most packages contain a package called `Examples`, which contains example applications that illustrate the typical use of components in the parent directories and that are used to conduct unit tests.

### 2.1.1 Package `Controls`

The package `Controls` contains blocks that are typically needed to implement controllers of building energy systems. For example, the package `Controls.Continuous` contains a block of a composite controller where a hysteresis block can switch equipment on/off, a timer allows for the locking out of equipment for a minimum time, and a PID controller computes the actuator signal when the controller is in the on state. Such a controller can for example be used to control a modulating boiler. The package `Controls.SetPoints` includes blocks for gain scheduling and

```
Buildings.Controls.Continuous
                  .Discrete
                  .SetPoints
Buildings.Fluids.Actuators.Dampers
                          .Motors
                          .Valves
                .Boilers
                .Chillers
                .Delays
                .FixedResistances
                .HeatExchangers
                .HeatExchangers.CoolingTowers
                               .Radiators
                .Interfaces
                .MassExchangers
                .MixingVolumes
                .Movers
                .Sensors
                .Sources
                .Storage
                .Utilities
          .HeatTransfer
          .Media
          .Utilities.Diagnostics
                    .IO
                    .Math
                    .Psychrometrics
                    .Reports
```

Figure 1: Package structure of the `Buildings` library. Only the major packages are shown.

time schedules. For example, there is a gain scheduler that can be used to compute in a hydronic heating system the set point of the supply water temperature as a function of the outside temperature. There is also a block for an occupancy schedule that has, as one of its outputs, the time until the next occupancy. This output can, for example, be used to start ventilating the building prior to occupancy to remove volatile organic compounds that may have accumulated when the ventilation was switched off.

### 2.1.2 Package `Fluid`

The `Fluid` package contains component models for thermo-fluid flow systems. The level of modeling detail is comparable with the models of the `Modelica.Fluid` library, and most models in `Buildings.Fluid` extend models from `Modelica.Fluid` to form components that are typically needed when modeling building energy systems.

The pressure drop calculation of most resistance models in our library is implemented in the partial model `BaseClasses.FlowModels.BasicFlowModel`. This model computes the relation

$$\dot{m} = \mathrm{sign}(\Delta p)\, k \sqrt{|\Delta p|}, \tag{1}$$

where the flow coefficient $k$ is assigned by models that extend `PartialResistance`. The model is used to model pressure drop in valves, pipes and mechanical equipment. The implementation is realized using the function `Modelica.Fluid.Utilities.regRoot2`, which regularizes the equation near the origin. Our implementation uses mass flow rate instead of volume flow rate. This has been done to avoid the influence of density, and hence temperature, on the pressure drop calculation. In pressure drop calculations for piping and duct networks in buildings, the uncertainty in the pressure drop calculation is typically larger than the error introduced when assuming a constant density. Note that our implementation still allows the use of a fluid model with variable density in order to model, for example, pressure differences due to a stack effect which can be important for high rise buildings or for naturally ventilated buildings.

At low flow rate, equation (1) is regularized to model laminar flow and to avoid numerical problems as its derivative is unbounded. For undisturbed flow in a pipe, the flow transition between laminar and turbulent typically occurs for Reynolds numbers in the range of 1500 to 4000, but turbulence may occur at much smaller Reynolds numbers due to flow mixers, diverters or bends. Also, in early design of buildings, the piping or ducting diameters are typically unknown. Thus, in our models, a user can specify at what fraction of the nominal flow rate equation (1) is regularized. Alternatively, the transition region can be specified by entering parameter values for the hydraulic diameter and the critical Reynolds number where turbulence occurs. If a user requires more detailed flow friction models, then models from `Modelica.Fluid` can be used in conjunction with the `Buildings` library.

Next, we will discuss the package `Fluids.Interfaces` which provides partial classes that are used by most models in the `Fluids` package. After this discussion, we will present the other packages.

### 2.1.3 Package `Fluids.Interfaces`

Similarly to `Modelica.Fluid.Interfaces`, there is a package `Buildings.Fluids.Interfaces` that defines the partial models for components

that exchange heat or mass with one or two fluid streams. Such partial models are implemented for components with two and with four fluid ports. For models with two fluid ports, i.e., models that have one fluid stream, the top-level model is `PartialStaticTwoPortInterface`, which extends `Modelica.Fluid.Interfaces.PartialTwoPort` to add equations for the states at the ports and variables for the mass flow rate and pressure drop. This partial model is extended by two models: `PartialStaticTwoPortHeatMassTransfer` and `PartialDynamicTwoPortTransformer`. The first adds equations for the enthalpy, species, trace substances and pressure drop between its ports. It also introduces the variables `Q_flow` and `mXi_flow[Medium.nXi]` that need to be assigned by models that extend this partial model. These variables can for example be used to implement a steady-state model for a heater and humidifier. The second partial model, `PartialDynamicTwoPortTransformer`, adds a pressure drop element and an instantaneously mixed volume to the base class. This partial model is used to implement dynamic components that add heat or species to the fluid. Clearly, it would have been convenient to use `PartialDynamicTwoPortTransformer` also for steady-state models by configuring the volume model in such a way that energy and mass balance are steady-state. Such a configuration has been tested, but it led to a larger equation system, it required using the computationally less efficient function `actualStream()` to compute the enthalpy flow rate at the ports of the volume, and it led in test models to divisions by zero for zero mass flow rate. For all of the above models, there are also similar versions with four fluid ports that are used as partial models for implementing models that exchange heat or mass between two fluid streams.

#### 2.1.4 Package `Fluids.Actuators`

The package `Fluids.Actuators` contains models of valves and air dampers, as well as of motors that can be used in conjunction with the actuators. Actuator models are based on a flow coefficient $\phi$, defined as the flow rate at the current actuator position $y \in [0, 1]$ divided by the flow rate at $y = 1$, i.e.,

$$\phi = \frac{C_v(y)}{C_v(y=1)}. \tag{2}$$

This flow ratio is proportional to $k$ in (1). For the two-way valve model with linear opening characteristics, `TwoWayLinear`, the flow ratio is $\phi = l + y(1-l)$, where $0 \leq l \ll 1$ is the valve leakage. Fig. 2 shows $\phi$ as a function of the valve opening $y$ for a linear valve, an equal percentage valve and a quick opening valve. For better display, untypical values of $l = 0.05$ and a rangeability for the equal percentage valve of 10 have been selected. These two-way valve models are also used to construct three way valve models with linear characteristics, or with a combination of equal percentage characteristics in the main flow path and linear characteristics in the bypass, to allow the modeling of valves that are typically used in hydronic heating systems.

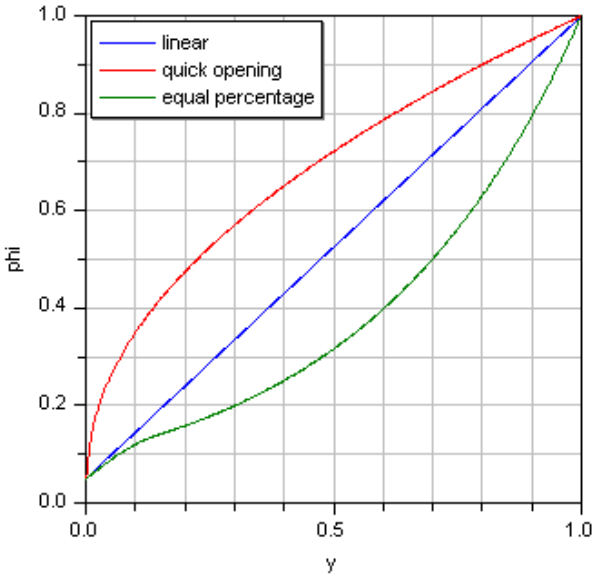



Figure 2: Flow characteristic $\phi(y)$ for valves.

Besides valves, there are also air damper models with exponential opening characteristics and models for terminal boxes of variable air volume flow systems.

There is also a model of a motor with hysteresis and finite actuation speed that can be used with the valve or the damper models. If the current actuator position $y$ is below (or above) the input signal $u$ by an amount bigger than a hysteresis $\delta$, then the position $y$ is increased (decreased) using a finite speed until it reaches $u$.

#### 2.1.5 Package `Fluids.Boilers`

The package `Fluids.Boilers` contains a boiler model that can either be configured as a dynamic model or as a steady-state model. The heat transferred from the combustion to the the water, $\dot{Q}$, is computed as

$$\dot{Q} = \dot{Q}_0 \frac{\eta(y,T)}{\eta(1,T_0)} y, \tag{3}$$

where $\dot{Q}_0$ is the nominal heating capacity, $y \in [0,1]$ is a control signal (typically, $y \in 0 \cup [0.3,\ 1]$ for a modulating boiler) and $\eta : [0,1] \times \Re \rightarrow \Re$ is the furnace efficiency based on the load fraction $y$ and the furnace temperature $T$. The nominal heating capacity $\dot{Q}_0$ and its associated temperature $T_0$ are parameters. The model allows users to select different functional forms for $\eta(\cdot,\cdot)$. The heat $\dot{Q}$ is added to a volume from `Modelica.Fluid` (to model the fluid's heat balance) which is connected to a solid heat storage element from the Standard Modelica Library (to model heat stored in the metal). The model also exposes a heat port of a heat conductor to allow modeling heat losses to the ambient environment.

### 2.1.6 Package `Fluids.Chillers`

This package contains a model of a chiller whose coefficient of performance (COP) is proportional to the Carnot efficiency. Parameters are either the Carnot effectiveness $\eta_{c,0}$ at the nominal conditions, or the COP and the evaporator and condenser temperatures at the nominal conditions, $\mathrm{COP}_0$, $T_{e,0}$ and $T_{c,0}$. In the latter case, the Carnot effectiveness is computed as

$$\eta_{c,0} = \mathrm{COP}_0 \frac{T_{c,0} - T_{e,0}}{T_{e,0}}. \tag{4a}$$

A user can specify what temperatures should be used as the evaporator (or condenser) temperature. The available options are the temperatures of the fluid volume, of `port_a`, of `port_b`, or of the average temperature of `port_a` and `port_b`. The chiller COP is computed as the product

$$\mathrm{COP} = \eta_{c,0} \frac{T_e}{T_c - T_e} \eta_{pl}(y), \tag{4b}$$

where $\eta_{pl}(\cdot)$ is a polynomial in the control signal $y$ that can be used to take into account a change in COP at part load conditions. The electrical power consumption of the compressor is $P = P_0 y$, where $P_0$ is a parameter for the nominal compressor power. The heat extracted from the evaporator is $\dot{Q}_e = \mathrm{COP}\, P$, and the heat transferred to the condenser is $\dot{Q}_c = \dot{Q}_e + P$. The condenser and evaporator are modeled using volumes from `Modelica.Fluid` and can therefore be modeled dynamic or at steady-state.

### 2.1.7 Package `Fluids.HeatExchangers`

This package contains steady-state and dynamic heat exchanger models, some of which compute condensation of water vapor that may occur at a cooling coil. Simple models include a model with prescribed heat input into the medium, i.e., the transferred heat is $\dot{Q} = \dot{Q}_0 u$, where the maximum heat transfer $\dot{Q}_0$ is a parameter and $u \in [0,\ 1]$ is an input signal.

There is also a constant effectiveness heat exchanger in which

$$\dot{Q} = \varepsilon \min(|\dot{C}_1|, |\dot{C}_2|)\, \Delta T_{in}, \tag{5}$$

where $\varepsilon \in (0,1)$ is a parameter, $\dot{C}_1$ is the heat capacity flow of the stream 1, and $\Delta T_{in}$ is the temperature difference of the two inlet temperatures.

If a more detailed dynamic model is required, then finite volume models of a dynamic cooling coil, optionally with water vapor condensation, can be used. In these models, each pipe is discretized along its flow path. Pipes can be arranged in parallel to form a register. There are headers to redirect the flow between registers, or between pipes inside a register. The registers are exposed to the air stream that flows from one register to another. The most basic element of the coil models consists of two fluid volumes, one for the air stream and one for the water stream, that are connected to a thermal storage model for the metal pipe (see Fig. 3).

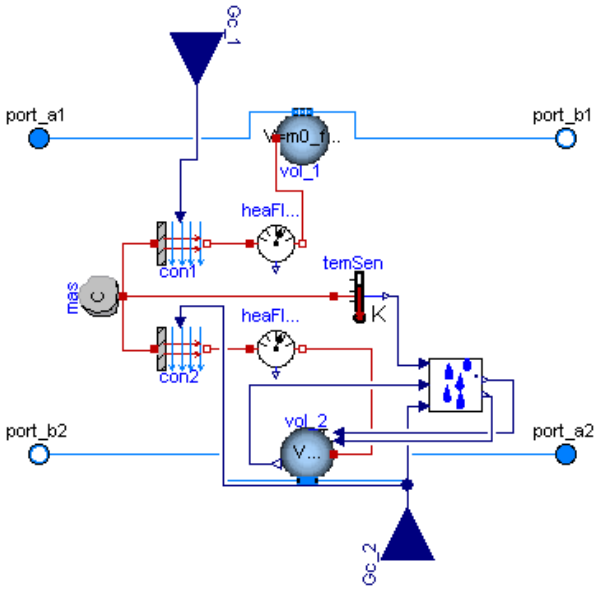



Figure 3: Basic element for a dynamic cooling coil with condensation. Models with index 1 and 2 refer to the water and air side, respectively.

The sensible convective heat transfer coefficient can be constant, or it can be a function of mass flow rate and temperature. The convective mass transfer coefficient is calculated based on similarity laws between heat and mass transfer. Using the Lewis number, which is defined as the ratio between the heat and

mass diffusion coefficients, the ratio between convection heat transfer coefficient $h$ (in $\mathrm{W}/(\mathrm{m}^2\,\mathrm{K})$) and mass transfer coefficient $h_m$ (in $\mathrm{m}/\mathrm{s}$) is obtained as

$$\frac{h}{h_m} = \rho\, c_p\, Le^{(1-n)}, \tag{6a}$$

where $\rho$ is the mass density, $c_p$ is the specific heat capacity of the bulk medium and $n$ is a coefficient from the boundary layer analysis, which is typically $n = 1/3$. From this equation, the water vapor mass flow rate $\dot{m}_w$ (in $\mathrm{kg}/\mathrm{s}$) is obtained as

$$\dot{m}_w = \frac{G_c}{c_p\, Le^{(1-n)}}\,(X_s - X_\infty), \tag{6b}$$

where $G_c = hA$ is the sensible heat conductivity (in $\mathrm{W}/\mathrm{K}$) and $X_s$ and $X_\infty$ are the water vapor mass fractions in the boundary layer and in the bulk of the medium. In this model, $X_s$ is the saturation water vapor mass fraction corresponding to the surface temperature of the pipe.

The package `Fluids.HeatExchangers.CoolingTowers` contains models of cooling towers. The air inlet temperature is obtained from an input signal which can be set to the dry bulb or wet bulb temperature. There is a simple model in which the water outlet temperature minus the air inlet temperature is constant. There is also a more detailed model in which the water outlet temperature is computed based on the performance curve of a York cooling tower. This model can operate either with forced flow when the fan is operating or in free convection mode.

The package `Fluids.HeatExchangers.Radiators` contains a radiator model for hydronic space heating systems. The model can be configured as a steady-state or a dynamic model, and it can be discretized into finite volumes. If $n \geq 1$ denotes the number of discretization volumes, then the convective and radiative heat transfer is

$$\dot{Q}_c = (1-f_r)\,\dot{Q}_0\,\frac{1}{n}\sum_{i=1}^{n}\left(\frac{\Delta T_{c,i}}{\Delta T_0}\right)^n, \tag{7a}$$

$$\dot{Q}_r = f_r\,\dot{Q}_0\,\frac{1}{n}\sum_{i=1}^{n}\left(\frac{\Delta T_{r,i}}{\Delta T_0}\right)^n, \tag{7b}$$

where $f_r$ is the fraction of radiative heat transfer, $\dot{Q}_0$ and $\Delta T_0$ are the nominal heating capacity and temperature difference, and $\Delta T_{c,i}$ and $\Delta T_{r,i}$ are the convective and radiative temperature differences of the $i$-th volume. The parameters $f_r$, $\dot{Q}_0$ and $\Delta T_0$ are typically available from product catalogs.

### 2.1.8 Package `Fluids.MassExchangers`

This package contains two models for mass exchangers. The model `HumidifierPrescribed` adds water to an air stream in the amount of $\dot{m}_w = y\,\dot{m}_{w,0}$, where $\dot{m}_{w,0}$ is a parameter and $y \in [0,\ 1]$ is an input.

The model `ConstantEffectiveness` transfers heat and moisture in the amount of

$$\dot{Q} = \varepsilon_s \min(|\dot{C}_1|, |\dot{C}_2|)\,\Delta T_{in}, \tag{8a}$$

$$\dot{m} = \varepsilon_l \min(|\dot{m}_1|, |\dot{m}_2|)\,\Delta X_{in}, \tag{8b}$$

where $\varepsilon_s$ and $\varepsilon_l$ are the sensible and latent effectiveness, $\dot{C}_1$ is the heat capacity flow rate of medium one, $\Delta T_{in}$ is the temperature difference over the inlet streams, $\dot{m}_1$ is the mass flow rate of stream one and $\Delta X_{in}$ is the difference in water vapor mass fraction across the two inlet streams.

### 2.1.9 Package `Fluids.Storage`

This package contains models of thermal energy storage tanks.

The model `Stratified` uses a user-specified number of fluid volumes that are connected in series to model a stratified storage tank. Heat conduction is modeled between the volumes through the fluid, and between the volumes and the ambient environment through the tank enclosure. A heat port that is connected to the fluid volume can be used to add a temperature sensor or to inject heat into the fluid when modeling a heat exchanger. The tank also contains a model that emulates buoyancy if there is a temperature inversion in the tank.

The model `StratifiedEnhanced` extends the above model to add equations that reduce the numerical dissipation that is introduced when fluid flows from one volume to another. Such numerical dissipation is typical for upwind discretization schemes. To reduce the numerical dissipation, we implemented a model that is based on the one described by Wischhusen [14].

### 2.1.10 Package `Media`

This package contains media models that can be used in addition to the models from `Modelica.Media`. Some of the media models in this package are based on simplified state equations and property equations that lead generally to a faster and more robust simulation compared to the models of `Modelica.Media`. For example, there are models that are similar to `Modelica.Media.Air.MoistAir`, but our implementations in `Buildings.Media.PerfectGases` are based on the

Table 1: Benchmark tests for medium models.

| system model | Medium model | init. | | | simulation | | | computing time in [s] |
|---|---|---|---|---|---|---|---|---|
| | | l | nl | J | l | nl | J | |
| 1 | `Modelica.Media.Air.MoistAir` | 0 | 25 | 0 | 0 | 1 | 26 | 20 |
| 1 | `Buildings.Media.PerfectGases.MoistAir` | 0 | 25 | 0 | 0 | 1 | 26 | 13.5 |
| 1 | `Buildings.Media.GasesPTDecoupled.MoistAir` | 0 | 7 | 0 | 0 | 1 | 26 | 8.6 |
| 2 | `Modelica.Media.Air.SimpleAir` | 0 | 31 | 0 | 0 | 1 | 0 | * |
| 2 | `Buildings.Media.GasesPTDecoupled.SimpleAir` | 0 | 21 | 0 | 0 | 0 | 26 | 0.55 |

* Failed to solve initialization problem.

thermally perfect medium assumption [2], i.e., the specific heat capacity is constant and internal energy and enthalpy are functions of temperature only.

There is also a package `Buildings.Media.GasesPTDecoupled` that contains medium models that do *not* follow the ideal gas law. Instead, the medium model is based on the equation $\rho/\rho_{stp} = p/p_{stp}$, where $p$ denotes pressure and $\rho$ denotes mass density, and $p_{stp} = 101325\,\mathrm{Pa}$ and $\rho_{stp} = 1.2\,\mathrm{kg/m^3}$ for air. Hence, in an isobar heat exchange, density remains constant, which generally leads to smaller equation systems. Table 1 shows benchmark problems conducted with Dymola 7.1 using models that are part of the example models of the `Buildings` library. The system model 1 is a dynamic cooling coil with feedback control and valve motor with hysteresis (model `WetCoilDiscretizedPControl`). The system model 2 is an open-loop simulation of a variable air volume flow system that serves 6 rooms (model `MITScalable`). In Tab. 1, the first three columns are the statistics for the initialization problem, the next three columns are for the time integration, and the last column is the computing time. The columns with header "l" or "nl" denote the largest dimension of a linear or nonlinear system of equations, and the columns with header "J" denote the number of numerical Jacobians. For the system model 1, decoupling pressure and temperature reduced computing time by more than a factor of two. For the system model 2, Dymola was only able to solve the initialization problem with the simplified medium model.

#### 2.1.11 Package `Utilities`

The package `Utilities.Diagnostics` contains blocks with assert statements that can for example be used to check that setpoints are within expected ranges, which can be helpful to automate some of the debugging during the development of large models.

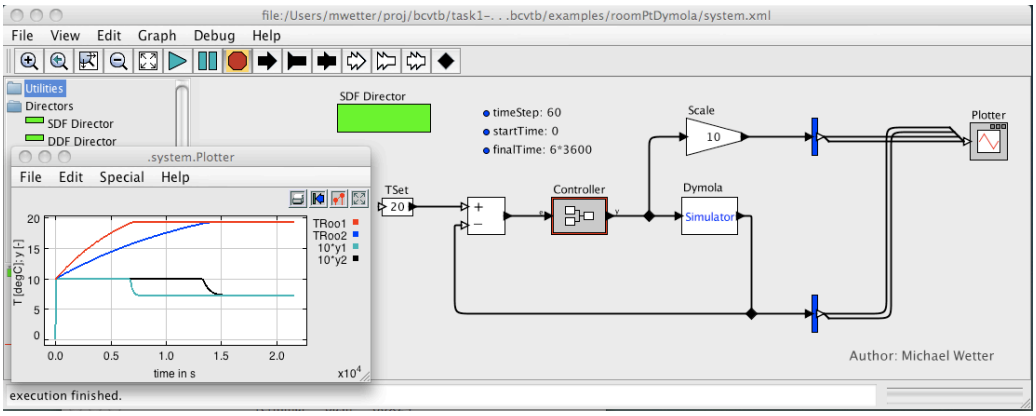
The package `Utilities.IO` contains blocks for input and output. Its subpackage `Utilities.IO.BCVTB` contains a block that allows linking Modelica to the Building Controls Virtual Test Bed (BCVTB). At the start of the simulation, this block establishes a socket connection using the Berkeley Software Distribution socket (BSD socket) implementation, and then exchanges data with the BCVTB during the simulation. The BCVTB interface allows

1. co-simulation between Modelica and the EnergyPlus whole building energy simulation program, MATLAB and/or Simulink,
2. the use of Ptolemy II to link heterogeneous models of computation with Modelica models, and
3. linking Modelica with building control systems through the controls interface that will be added to the BCVTB.

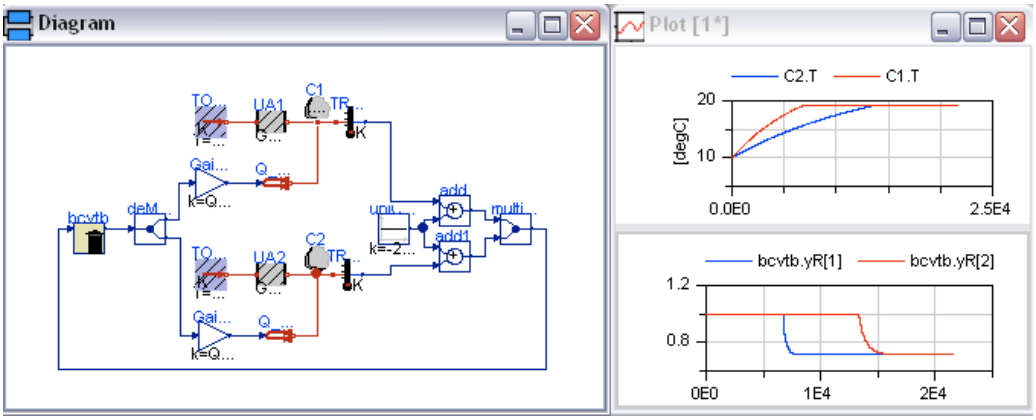
Fig. 4 shows a simple model that was used for testing the BCVTB interface. Fig. 4(a) shows the implementation in Ptolemy II, which models the controls. The block labeled "Dymola" sends control signals to Dymola and receives measured temperatures from Dymola. Fig. 4(b) shows the implementation of the heat transfer model in Dymola. The model on the very left sends the measured temperatures to Ptolemy II and receives the control signal from Ptolemy II. The data exchange is done through BSD sockets and hence can be across a network.

The package `Utilities.Math` contains functions and blocks that augment the mathematical functions of the Modelica Standard Library. Examples are once continuously differentiable approximations to the functions $y = \min(x_1, x_2)$, $y = \exp(|x|)$ and $y = |x|^n$, for $n > 0$, that are used by various models in the `Buildings` library.

There is also a package `Utilities.Psychrometrics` with psychrometric functions and a package `Utilities.Reports` with blocks for reporting output values to files.




(a) Ptolemy model that implements the models for the controls.




(b) Modelica model that implements the models for the heat transfer.

Figure 4: Illustration of the BCVTB interface for co-simulation using Ptolemy II and Modelica.

# 3 Applications

We will now present an example application of a hydronic heating system which is part of our library (see model `Buildings.Examples.HydronicHeating`). Applications in which the library has been used for controls design can be found in [12].

Fig. 5 shows the Modelica model of the hydronic heating system. On the lower left quadrant of the figure is the heating plant, with a dynamic model of a boiler and a circulation pump. The boiler loop is connected to a stratified thermal energy storage tank. Based on the temperatures of the top and bottom segment of the tank, a finite state machine switches the boiler on and off. The vertical lines in the middle of the figure are the water supply and return pipes to the rooms. The supply water temperature set point is scheduled based on the outside air temperature. An equal-percentage three-way valve mixes the water. The distribution pump has a controller that adjusts the pump revolution in order to maintain a constant pressure difference across the pump. On the upper right quadrant are models of two rooms. Each room has a thermostatic radiator valve with equal-percentage opening characteristics. The radiator models are dynamic and exchange radiative heat with the room enclosures and convective heat with the room air. The room air is modelled using a thermal capacity model and a thermal conductor (to model ventilation heat losses) from the Modelica Standard Library, and a finite difference model for transient heat conduction through the walls. There is also an occupancy schedule for each room, which is used to model heat gains from occupants, lights and equipment.

The total system model contained 353 simulation components that led to a differential algebraic equation system with 2278 scalar equations and 29 state variables. A simulation of two days took about 12 seconds on a desktop computer.

Fig. 6 shows the time trajectories for the second day of simulation. The top figure shows the room temperatures. The middle figure shows the valve positions (red and blue are radiator valves and black is the three-way valve). The bottom figure shows the supply

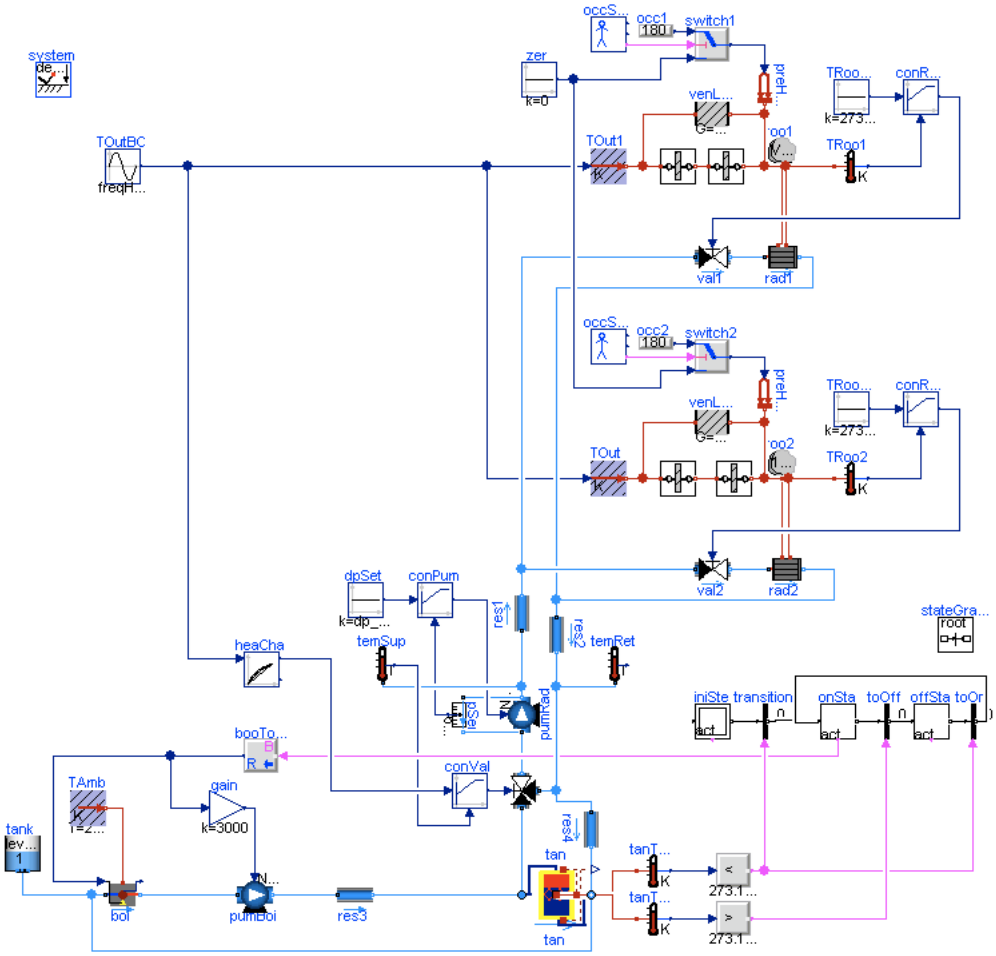



Figure 5: Model of the hydronic heating system with thermal energy storage.

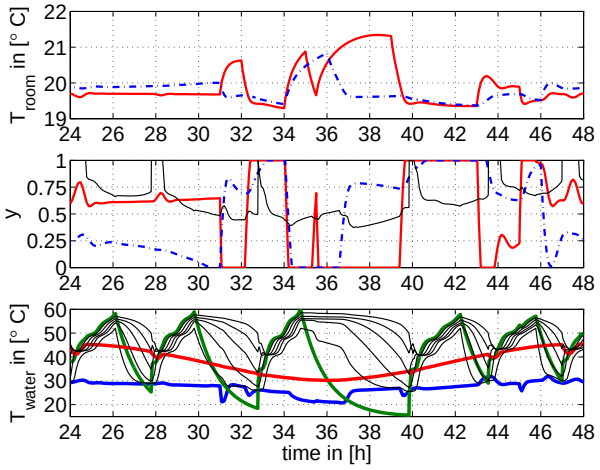



Figure 6: Time trajectories for the second day of simulation of the hydronic system model.

water temperature after the three-way valve (red), the return water temperature from the two rooms (blue), the boiler temperature (green) and the temperatures of the five volumes used to discretize the thermal storage tank (black).

The room temperature set point is well maintained. The peaks in room air temperature are due to internal heat gains and cause the thermostatic radiator valves to close as expected.

# 4 Conclusions

The `Modelica.Fluid` package provided a rich set of basic models that we could either use directly or adapt to implement a library for building energy systems. The flexibility of Modelica has been shown to enable analysis of building energy and control systems [12] that are outside the capability of traditional building energy simulation programs.

However, numerically solving for initial conditions and time trajectories of such systems can sometimes pose difficulties for solvers. We believe that further advances in model formulation and in solution algorithms are needed to make equation-based modeling

of building energy and control systems available to a larger audience who is not trained in addressing numerical problems. In the meantime, however, Modelica is a valuable tool for researchers who can debug numerically challenging problems, as it allows flexible system modeling and the addition of new models to existing libraries with less effort compared to causal programming languages.

## 5 Acknowledgments

This research was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Office of Building Technologies of the U.S. Department of Energy, under Contract No. DE-AC02-05CH11231.

## References

[1] Christopher Brooks, Edward A. Lee, Xiaojun Liu, Steve Neuendorffer, Yang Zhao, and Haiyang Zheng. Ptolemy II – heterogeneous concurrent modeling and design in Java. Technical Report No. UCB/EECS-2007-7, University of California at Berkeley, Berkeley, CA, January 2007.

[2] William B. Brower. *A primer in fluid mechanics: dynamics of flows in one space dimension*. CRC Press, Boca Raton, FL, USA, 1999.

[3] Drury B. Crawley, Linda K. Lawrie, Curtis O. Pedersen, Richard J. Liesen, Daniel E. Fisher, Richard K. Strand, Russell D. Taylor, Frederick C. Winkelmann, W. F. Buhl, A. E. Erdem, and Y. J. Huang. EnergyPlus, a new-generation building energy simulation program. In *Proc. of Renewable and Advanced Energy Systems for the 21st Century*, Lahaina, Maui, Hawaii, April 1999.

[4] Building Technologies Program – planned program activities for 2008-2012. Technical report, U.S. Department of Energy, Energy Efficiency and Renewable Energy, 2008.

[5] Buildings Energy Data Book. Technical Report DOE/EE-0325, U.S. Department of Energy, Washington, D.C., September 2008.

[6] J. Eker, J. W. Janneck, E. A. Lee, J. Liu, J. Ludwig, S. Neuendorffer, S. Sachs, and Y. Xiong. Taming heterogeneity – the Ptolemy approach. *Proc. IEEE*, 91(1):127–144, January 2003.

[7] Felix Felgner, Rolf Merz, and Lothar Litz. Modular modelling of thermal building behaviour using modelica. *Mathematical and Computer Modelling of Dynamical Systems*, 12(1):35–49, 2006.

[8] Rüdiger Franke, Francesco Casella, Martin Otter, Katrin Proelss, Michael Sielemann, and Michael Wetter. Standardization of thermo-fluid modeling in Modelica.Fluid. In Francesco Casella, editor, *Proc. of the 7-th International Modelica Conference*, Como, Italy, September 2009. Modelica Association.

[9] Rolf Mathias Merz. *Objektorientierte Modellierung thermischen Gebäudeverhaltens*. PhD thesis, Universität Kaiserslautern, September 2002.

[10] Christoph C. Richter. *Proposal of New Object-Oriented Equation-Based Model Libraries for Thermodynamic Systems*. PhD thesis, Technischen Universität Carolo-Wilhelmina zu Braunschweig, Germany, January 2008.

[11] Hubertus Tummescheit, Jonas Eborn, and Katrin Prölss. AirConditioning - a Modelica library for dynamic simulation of AC systems. In Gerhard Schmitz, editor, *Proceedings of the 4th Modelica conference*, pages 185–192, Hamburg, Germany, March 2005. Modelica Association and Hamburg University of Technology.

[12] Michael Wetter. Modelica-based modeling and simulation to support research and development in building energy and control systems. *Journal of Building Performance Simulation*, 2(2):143–161, June 2009.

[13] Michael Wetter and Philip Haves. A modular building controls virtual test bed for the integration of heterogeneous systems. In *Proc. of SimBuild*, Berkeley, CA, August 2008. IBPSA-USA.

[14] Stefan Wischhusen. An enhanced discretization method for storage tank models within energy systems. In Christian Kral and Anton Haumer, editors, *Proc. of the 5-th International Modelica Conference*, volume 1, pages 243–248, Vienna, Austria, September 2006. Modelica Association and Arsenal Research.