OpenModelica Course Intermediate
Modelica.Fluid library description

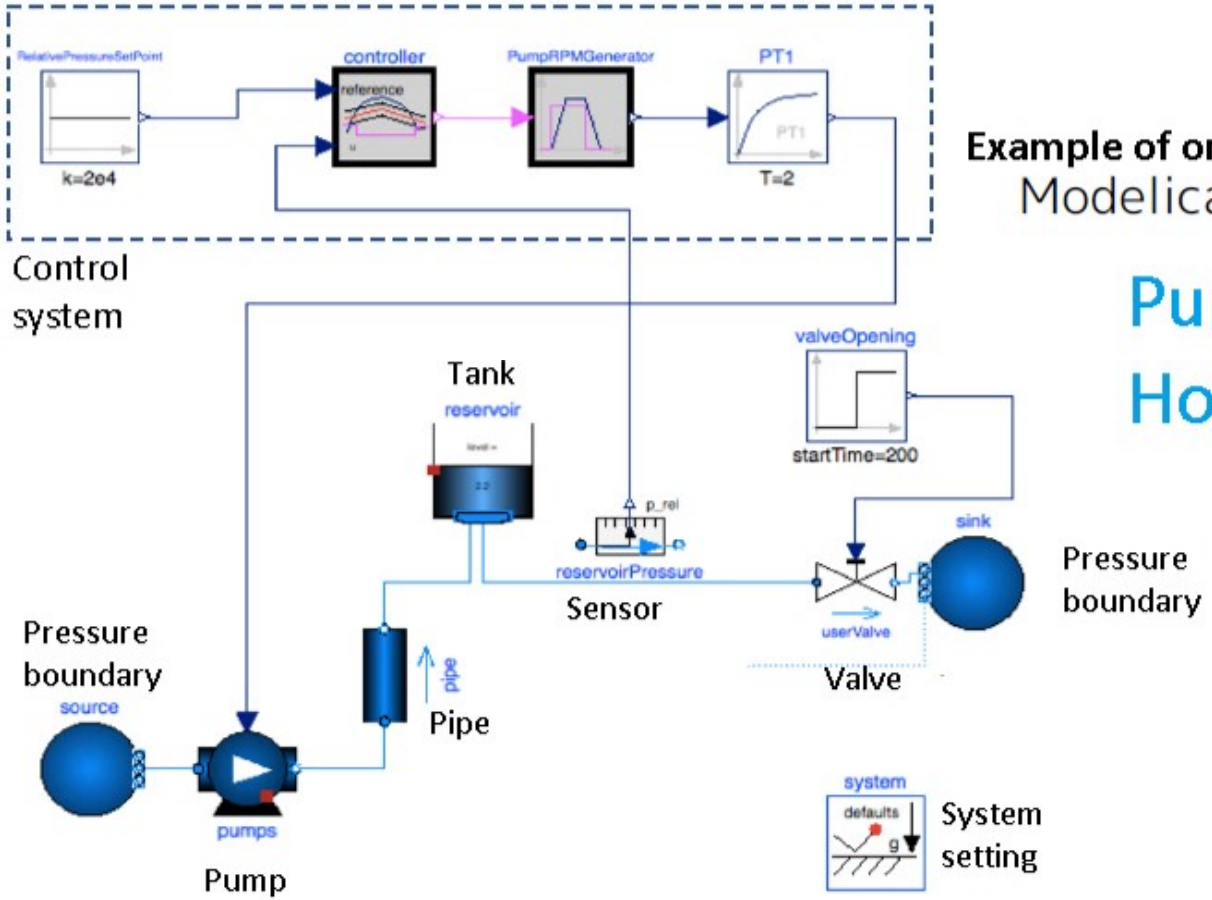
3. Modelica.Fluid library

December 7, 2017 Shu Tanaka (Amane Ryuken Co., Ltd.)

3.Modelica.Fluid library

Modelica.Fluid Library Goals

- **Providing interface and components for one-dimensional thermal fluid system models**
- **How Modelica implements fluid models**



Example of one-dimensional thermal fluid system model

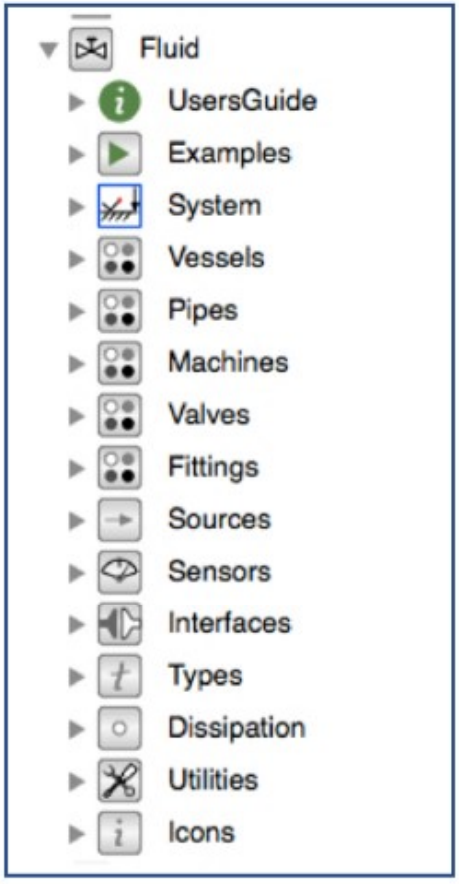
Modelica.Fluid.Examples.PumpingSystem

Pumping water into a tank

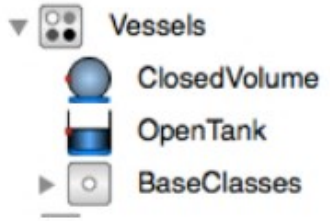
Hot water supply system

Modelica.Fluid library

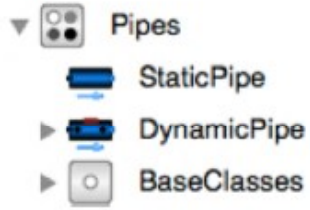
Modelica.Fluid Components



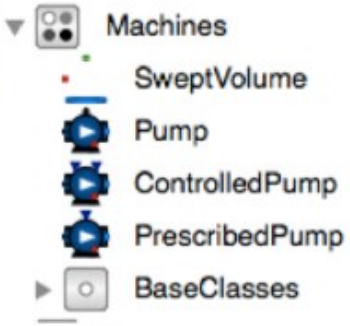
Vessels, tanks, etc.



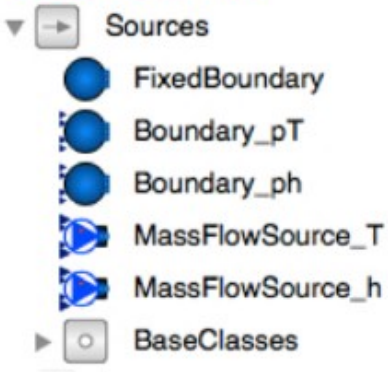
Pipes



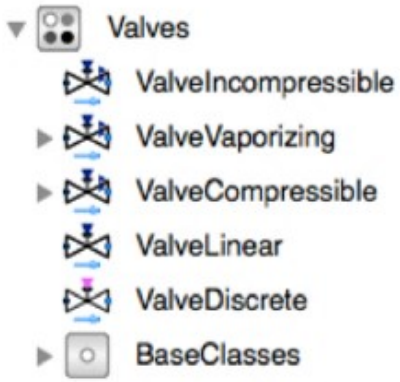
Pumps etc.



Flow rate, pressure regulation Boundaries

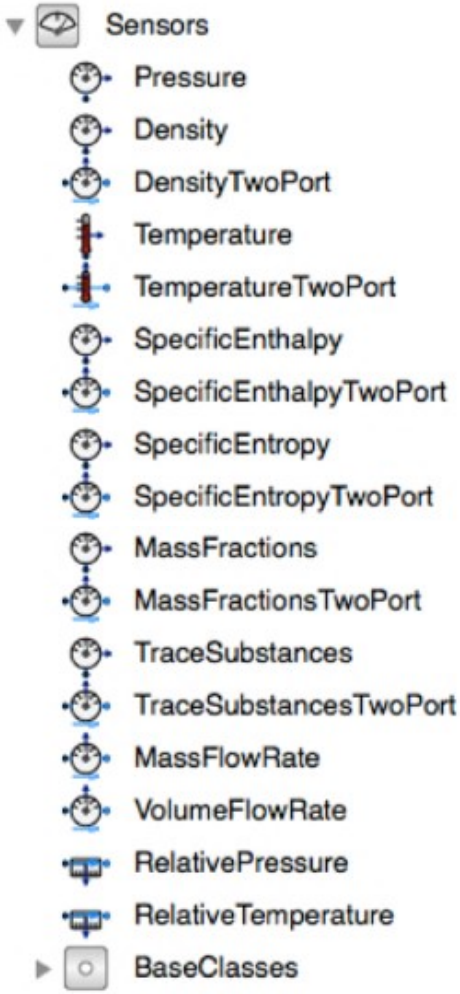


Valves



Bends, orifices, fittings, etc.

Sensors



Examples

The examples illustrate the basic concepts of the Modelica.Fluid library.

FluidExample1 FluidPort

- **WaterMix1** Mix hot and cold water
- **TraceSubstanceMix1** Mix water with different chlorine concentrations
- **GasMix1** Mix methane gas and air

FluidExample2 volume model

- **HotRoom1** Replace cold air in the room with warm air
- **HotRoom2** Split a room into two
- **WaterVolume1** Change the cold and hot water in the tank
- **WaterVolume2** Consider pressure loss at tank entrance

Examples

FlowExample3 flow model

- **StaticPipeTest1** Set up flow model and volume model
- **StaticPIpeTest2** Connect flow models directly

FluidExample4 DynamicPipe

- **WaterExchange1～4**

FluidExample5 VesselPortsData

- **WaterTank** Discharge hot water into a tank of water

FluidExample6 HeatTransfer モデル

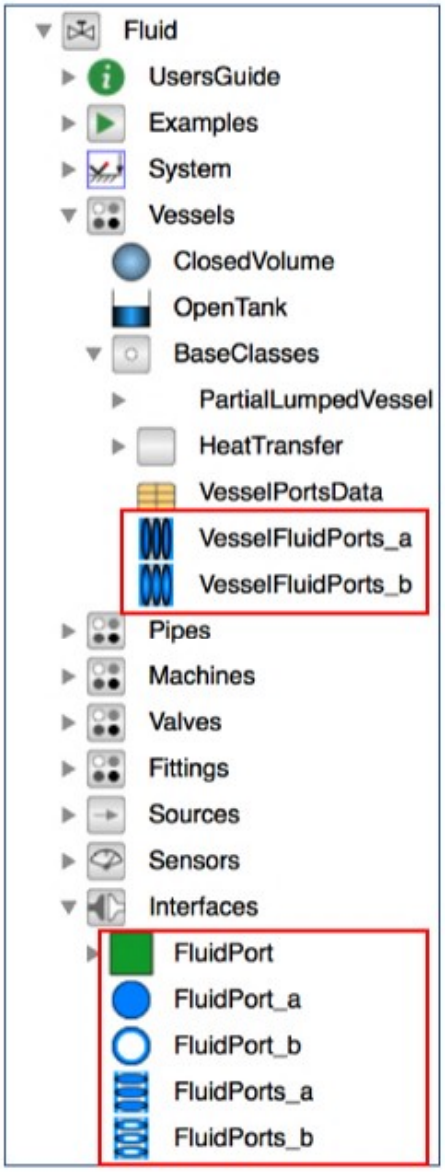
- **WaterHeating** Heating water in the tank to heat the pipe

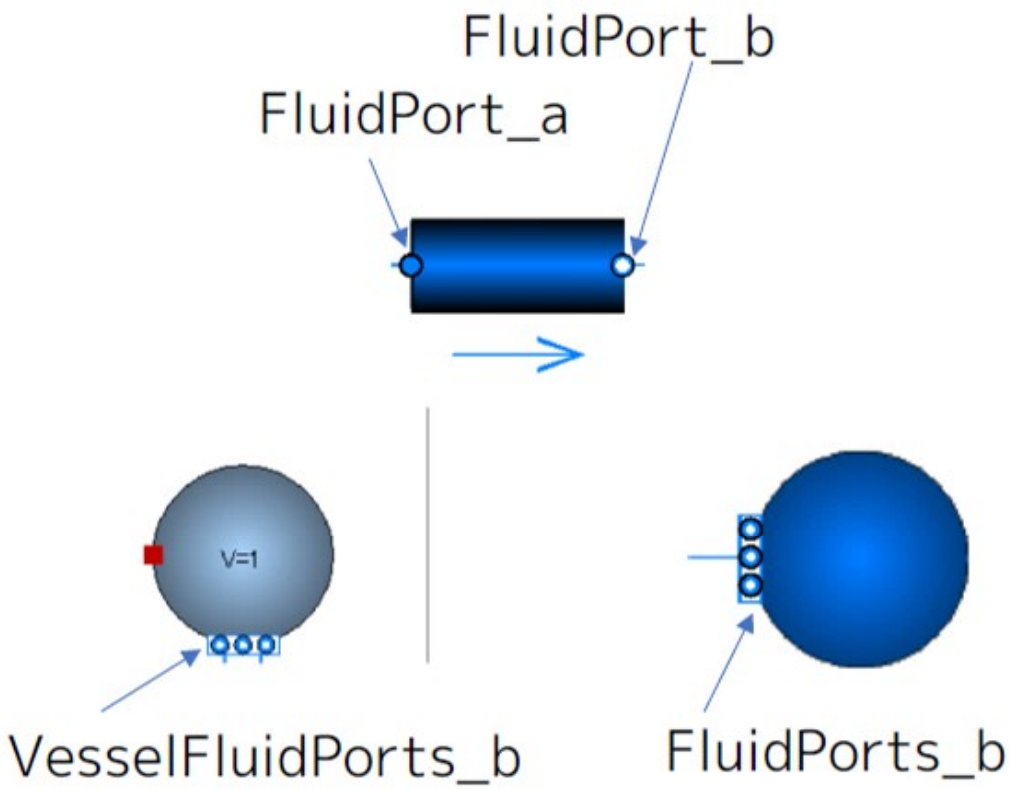
Modelica.Fluid.Examples.IncompressibleFluidNetwork

- Network model of incompressible fluid

FluidExample1

FluidPort: 1-dimensional fluid model of connector

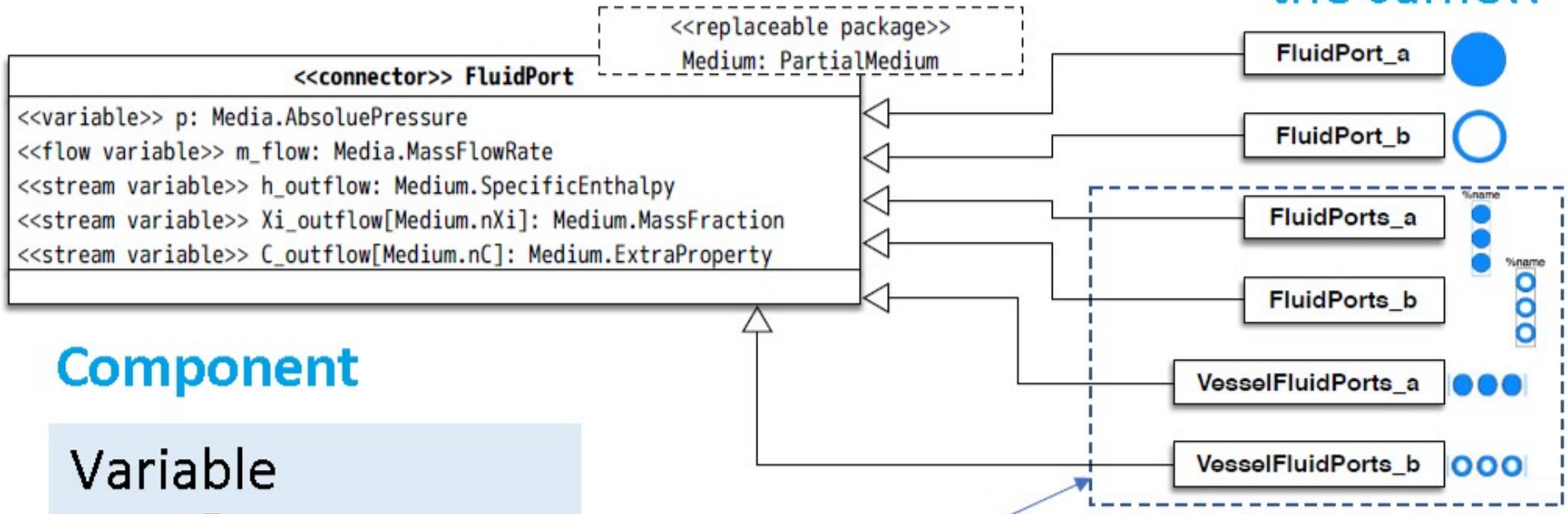




FluidPort

(1) FluidPort inheritance and components

Contents are the same!!



Component

Variable
- p: Pressure

Flow variable
- m_flow: Mass flow

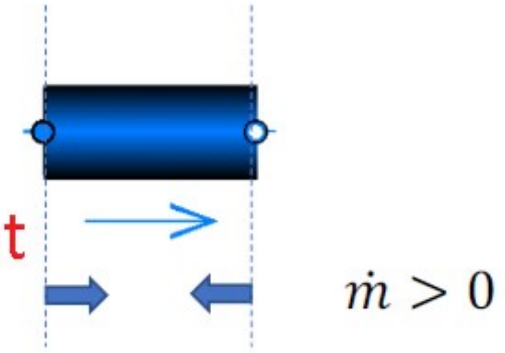
Stream variable
- h_outflow: Specific enthalpy
- Xi_outflow: Component mass fraction
- C_outflow: Concentration of additional substances (micro substances)

Use as a collection of multiple FluidPorts in a vector (array).

FluidPort

(2) Fluid mass flow direction

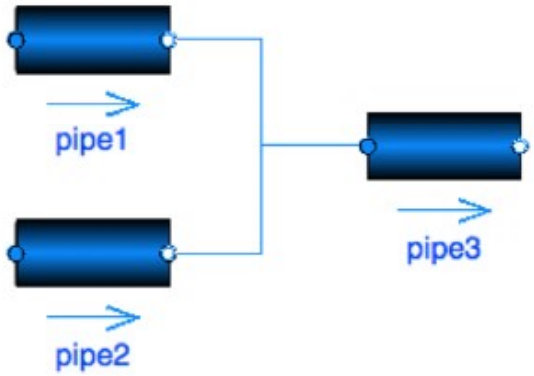
$\dot{m}$ = m_flow is positive from FluidPort to the inside of the component



$\dot{m} > 0$

(3) Equation by connecting FluidPort

Consider the case where pipes are connected as follows:



For multiple pipes/ports this could be simplified

p_i = pipe$i$.port_x.p
$\dot{m}_i$ = pipe$i$.port_x.m_flow
$h_{outflow_i}$ = pipe$i$. port_x.h_outflow

x=a or b
i = 1,2,3

Equation

```
connect(pipe1.port_b, pipe3.port_a);
connect(pipe2.port_b, pipe3.port_a);
```

FluidPort

The contents of the equation are as follows.

pipe1

p_1

$h_1 = h_{outflow_1}$

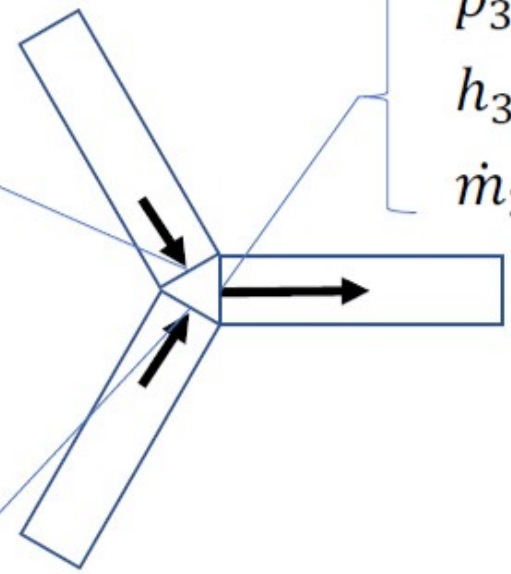
$\dot{m}_1 \leq 0$

pipe2

p_2

$h_2 = h_{outflow_2}$

$\dot{m}_2 \leq 0$



pipe3

p_3

$h_3 = inStream\left(h_{outflow_3}\right)$

$\dot{m}_3 > 0$

Variable p_i

$$p_1 = p_2 = p_3$$

Flow variable $\dot{m}_i$

$$\sum_i \dot{m}_i = \dot{m}_1 + \dot{m}_2 + \dot{m}_3 = 0$$

stream var. $h_{outflow}$, $\boldsymbol{Xi_{outflow}}$, $\boldsymbol{C_{outflow}}$

$h_{outflow_i}$ $\quad \dot{m}_i \leq 0$

$$inStream\left(h_{outflow_i}\right) \equiv \frac{\sum_{j \neq i} \max\left(-\dot{m}_j, 0\right) \cdot h_{outflow_j}}{\sum_{j \neq i} \max\left(-\dot{m}_j, 0\right)} \quad \dot{m}_i > 0$$

Built-in operator that calculates the value of the variable at inflow

FluidPort

(4) Handling stream variables inside components

port_h_outflow : Specific enthalpy
port_x.Xi_outfow : Mass fraction
port_a.C_outflow : Additional substance conc. (micro substance concentration)

Set the outflow value calculated inside the component.

A. When mass and energy cannot be stored inside the component

The amount that goes out of port_a
The amount coming from port_b,
The amount going out of port_b
Amount coming from port_a, ...

Snippet from StaticPipe source code:

```
equation
  ...
  0 = port_a.m_flow + port_b.m_flow;
  port_a.Xi_outflow = inStream(port_b.Xi_outflow);
  port_b.Xi_outflow = inStream(port_a.Xi_outflow);
  port_a.C_outflow = inStream(port_b.C_outflow);
  port_b.C_outflow = inStream(port_a.C_outflow);
  ...
  port_b.h_outflow = inStream(port_a.h_outflow) - system.g*height_ab;
  port_a.h_outflow = inStream(port_b.h_outflow) + system.g*height_ab;
  ...
end StaticPipe;
```

Enthalpy takes into account height differences at both ends of pipe

FluidPort

B. When storing mass or energy inside components

The advection term of the conservation equation is expressed as

$$\text{Enthalpy advection term} = \dot{m}_i \cdot actualStream(h_{outflow_i}) \ [\text{W}]$$

stream variable

$$actualStream(h_{outflow_i}) \equiv \begin{cases} inStream\left(h_{outflow_i}\right), & \dot{m}_i > 0 \\ h_{outflow_i} & \dot{m} \leq 0 \end{cases}$$

Built-in operator that calculates the values of variables that actually flow in and out depending on the flow direction

reference https://www.modelica.org/documents/ModelicaSpec32Revision1.pdf p.229-p.233

FluidPort

Excerpt from PartialLumpedVessel source code

```
ports[i].h_outflow  = medium.h;
ports[i].Xi_outflow = medium.Xi;
ports[i].C_outflow  = C;

ports_H_flow[i] = ports[i].m_flow * actualStream(ports[i].h_outflow)
"Enthalpy flow";
ports_E_flow[i] = ports[i].m_flow*(0.5*portVelocities[i]*portVelocities[i]
 + system.g*portsData_height[i])
"Flow of kinetic and potential energy";
ports_mXi_flow[i,:] = ports[i].m_flow * actualStream(ports[i].Xi_outflow)
"Component mass flow";
ports_mC_flow[i,:]  = ports[i].m_flow * actualStream(ports[i].C_outflow)
"Trace substance mass flow";
```

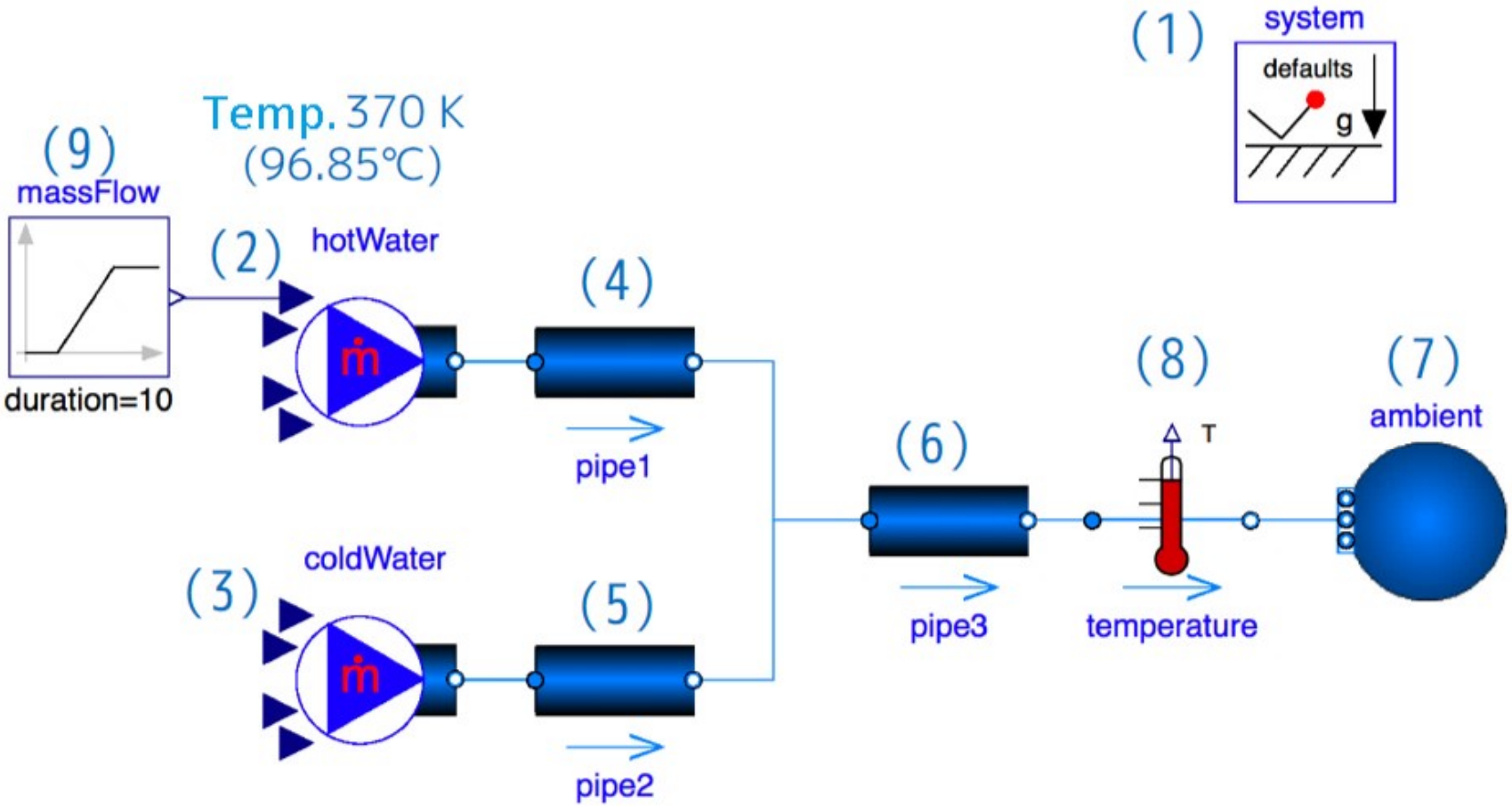
Set the value at the time of outflow calculated from the conservation formula inside the component.

Calculation of advection terms for conservative equations such as energy, component mass, and additional substances

WaterMix1

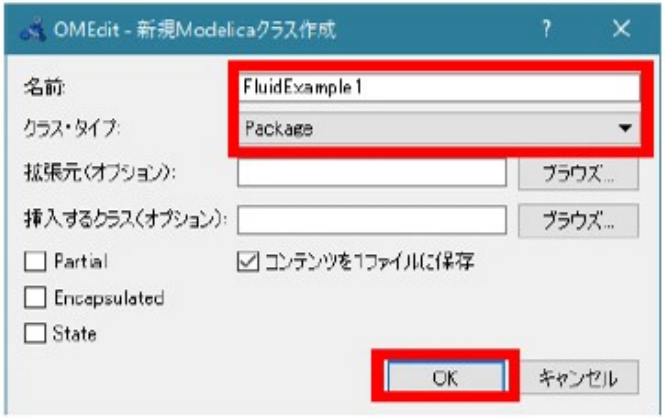
Mix hot and cold water

Check the function of FluidPort. Hot water is flowing in **pipe1** and cold water is flowing in **pipe2**. Make a model that mixes this with **pipe3**.



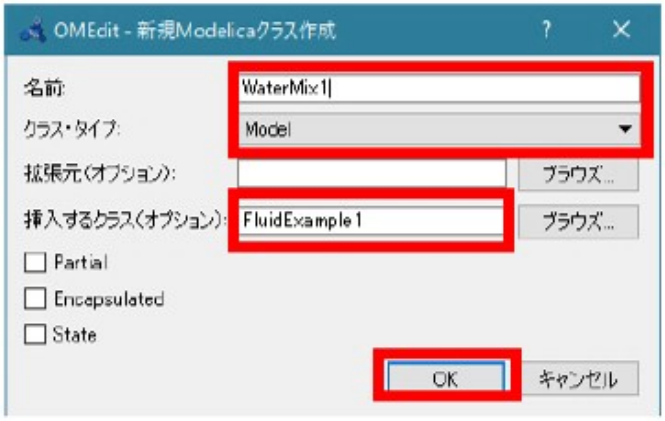
WaterMix1

① Create a package with File> New Modelica Class.



Name: FluidExample1
Class type: Package

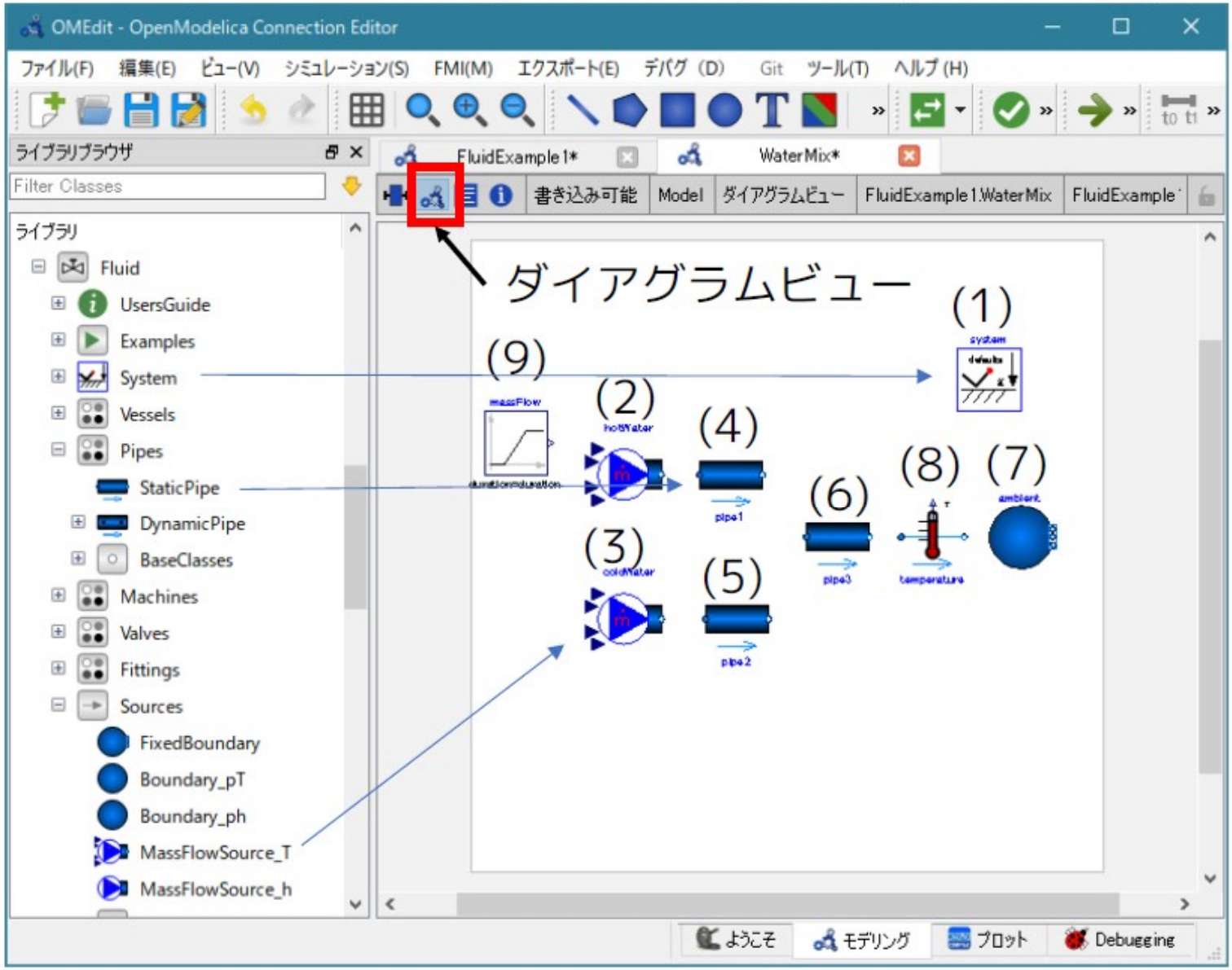
② Right-click FluidExample1 in the library browser
Select "Create a new Modelica class" and create a model.



Name: WaterMix1
Class type: Model
Inserted class: FluidExample1

WaterMix1

③ In the WaterMix1 diagram view, add the controls shown on the next slide. Drag and drop components (1) to (9).

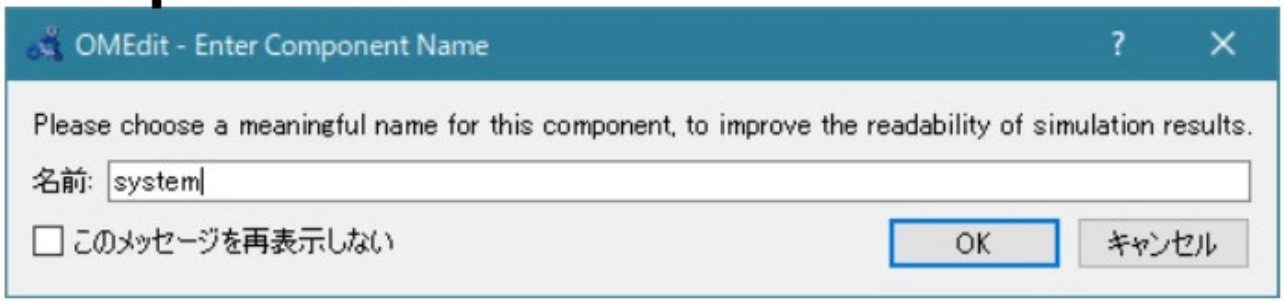


WaterMix1

Component name and class to place

Name	Class
(1)system:	Fluid.System
(2)hotWater:	Fluid.Sources.MassFlowSource_T
(3)coldWater:	Fluid.Sources.MassFlowSource_T
(4)pipe1:	Fluid.Pipes.StaticPipe
(5)pipe2:	Fluid.Pipes.StaticPipe
(6)pipe3:	Fluid.PIpes.StaticPipe
(7)ambient:	Fluid.Sources.FixedBoundary
(8)temperature:	Fluid.Sensors.TemperatureTwoPort
(9)massFlow:	Block.Sources.Ramp

Name the control that appears when you drop the component.

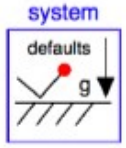


You can also change the name by right-clicking the component and selecting [Attribute].

WaterMix1

④ Right-click the component and select [Parameter] to set.

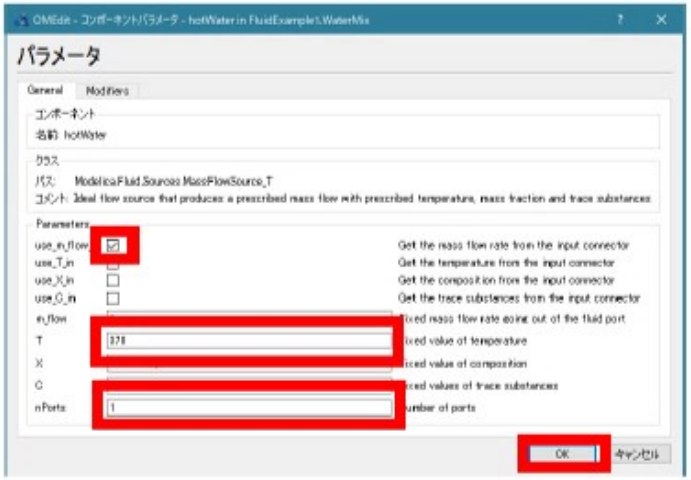
(1) Fluid.System

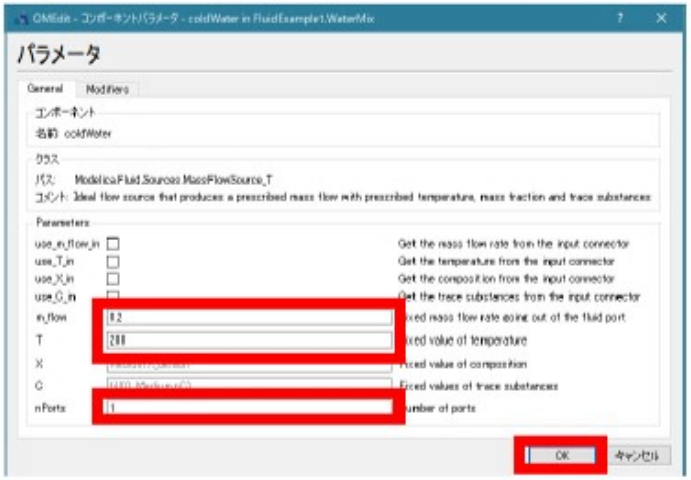


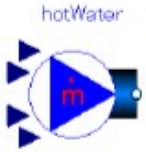
Be sure to place it in the fluid system model.

(1) system
- Leave the default

(2)(3) Fluid.Sources.MassFlowSources_T







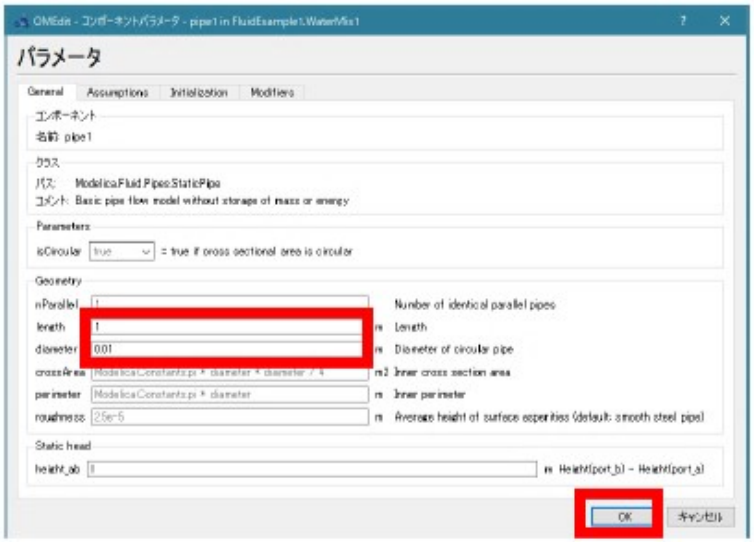
Specify the flow rate and temp.

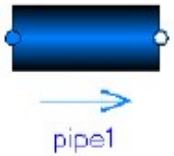
(2) hotWater
- Check use_m_flow_in
- T = 370 [K]
- nPorts = 1

(3) coldWater
- m_flow = 0.2 [kg/s]
- T = 280 [K]
- nPorts = 1

WaterMix1

(4)(5)(6) Fluid.Pipes.StaticPipe

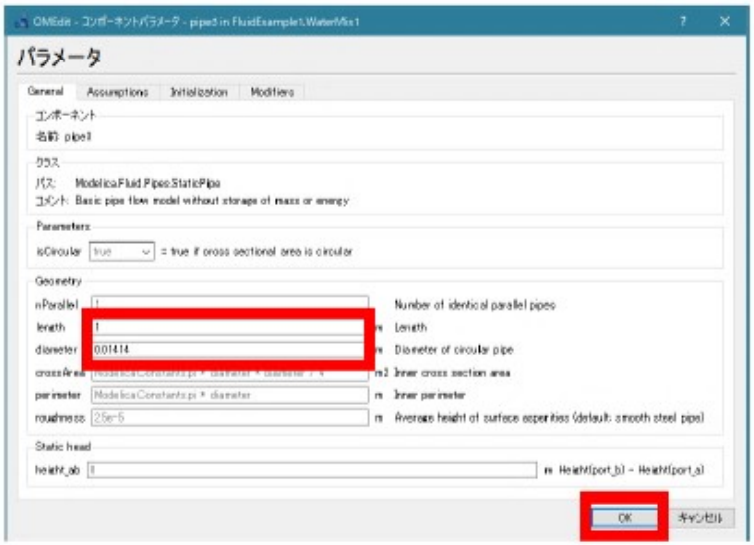




Set the length and inner diameter of the pipe that does not store mass or energy inside

(4) pipe1, (5) pipe2

- length = 1 [m]
- diameter = 0.01 [m]



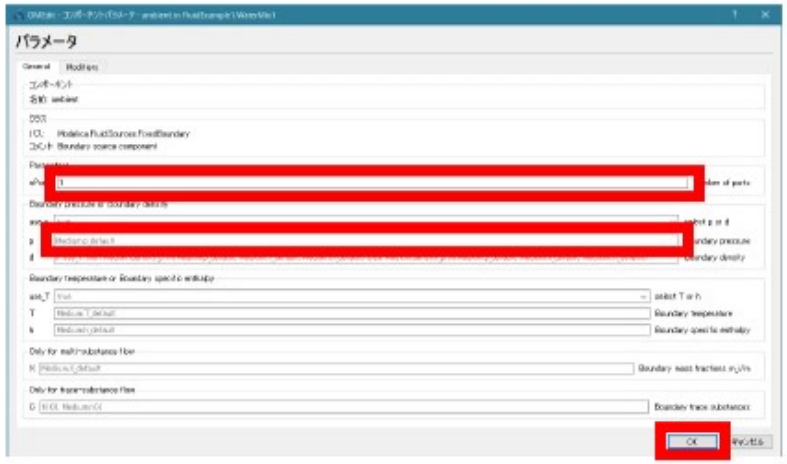
(6) pipe3

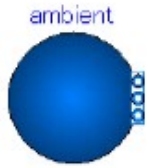
- length = 1 [m]
- diameter = 0.01414 [m]

Since water flows from pipe1 and pipe2 into pipe3, the sum of the cross-sectional areas of pipe1 and pipe2 was set to approximately the cross-sectional area of pipe3.

WaterMix1

(7) Fluid.Sources.FixedBoundary



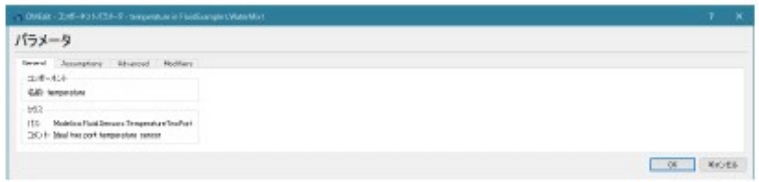


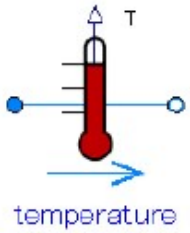
- Pressure or density
- Specify temperature or enthalpy

(7) ambient

- nPorts = 1
- p = 101325 [Pa]

(8) Fluid.Sensors.TemperatureTwoPort





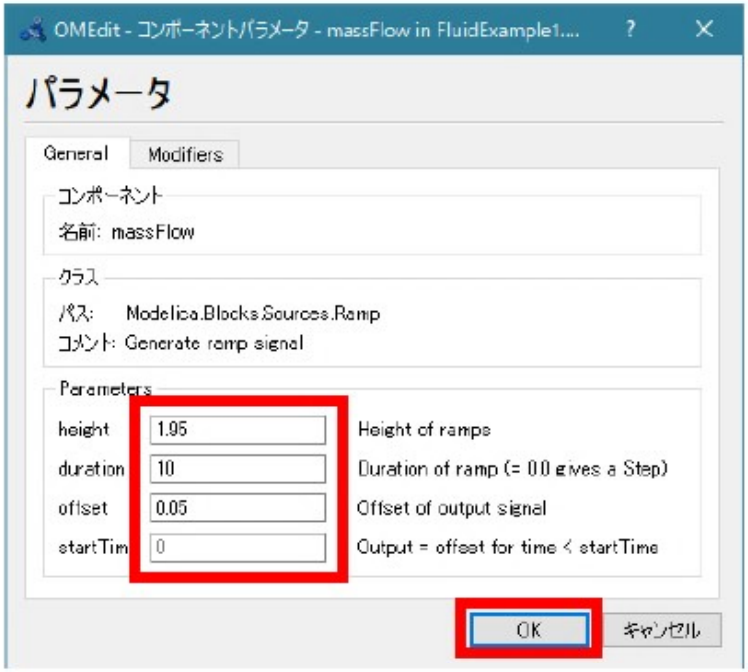
Temperature sensor

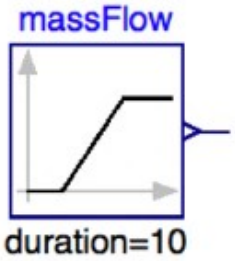
(8) temperature

- Default configuration

WaterMix1

(9) Blocks.Sources.Ramp



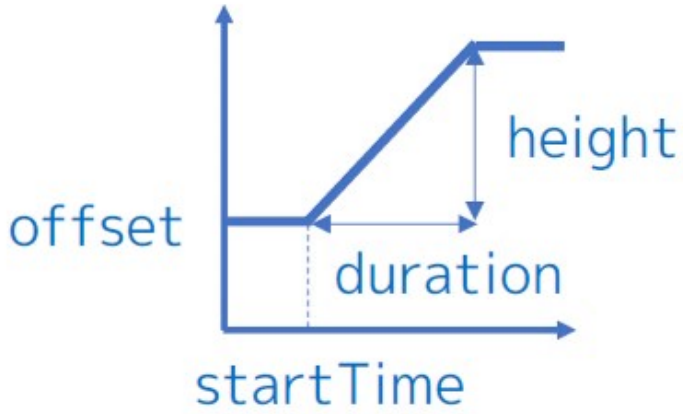


Signal source whose output changes like a ramp function

(9) massFlow

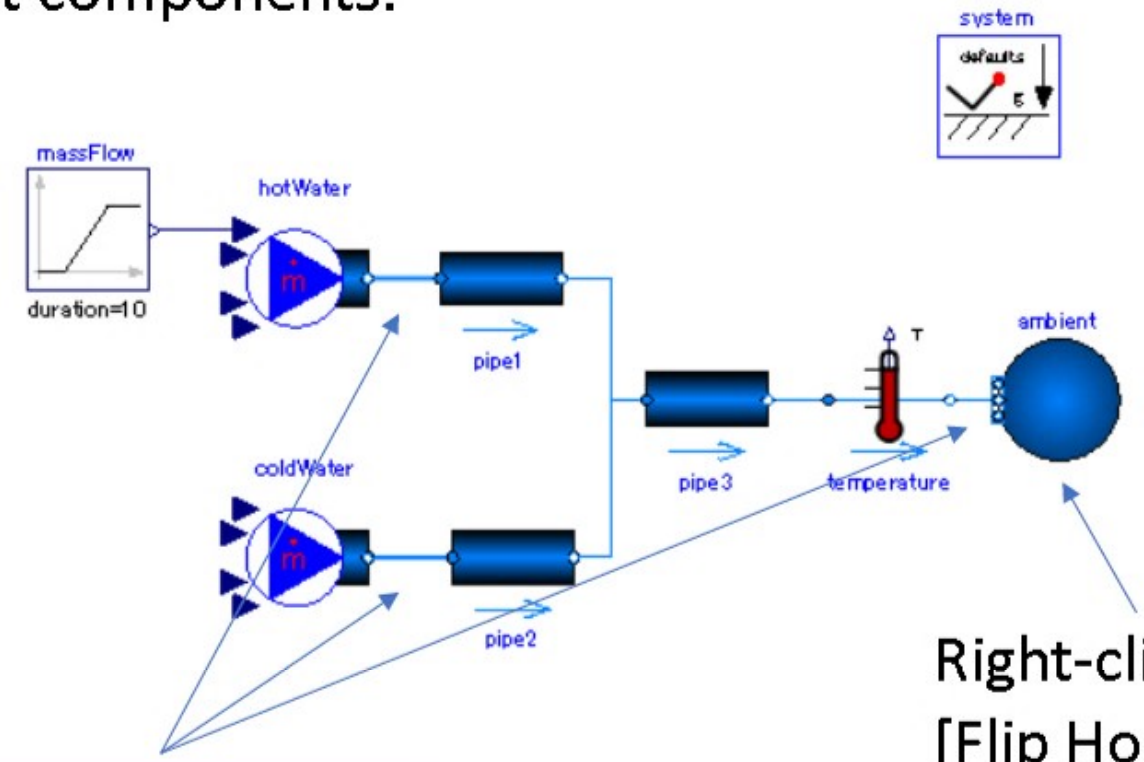
- height = 1.95
- duration = 10 [s]
- offset = 0.05
- startTime = 0 [s]

Change the mass flow rate of low concentration water from 0.05 kg / s to 2.00 kg / s in 10 seconds.



WaterMix1

⑤ Connect components.



Right-click and select [Flip Horizontal] for easy connection.

For hotWater, coldWater, and ambient, connectors fluidPort are set out. Select [1] (first element) in the dialog that appears when connecting.

Connect only one connection line to each element of the fluidPort array.

WaterMix1

⑥ Switch to the FluidExample1 text view and edit the code.

Edit the bold part.

Declare a physical package named Medium as an exchangeable local package.

Redeclare the components you use.

```
package FluidExample1
  import Modelica.Media;

model WaterMix1
  replaceable package Medium = Media.Water.StandardWater;                                    (1)
  inner Modelica.Fluid.System system annotation( ...);
  Modelica.Fluid.Sources.MassFlowSource_T hotWater(redeclare package Medium = Medium,        (2)
    T = 370, m_flow = 0.2, nPorts = 1, use_m_flow_in = true)  annotation( ...);
  Modelica.Fluid.Sources.MassFlowSource_T coldWater(redeclare package Medium = Medium,       (3)
    T = 280, m_flow = 0.2, nPorts = 1) annotation( ...);
  Modelica.Fluid.Pipes.StaticPipe pipe1(redeclare package Medium = Medium,                   (4)
    diameter = 0.01, length = 1)  annotation( ...);
  Modelica.Fluid.Pipes.StaticPipe pipe2(redeclare package Medium = Medium,                   (5)
    diameter = 0.01, length = 1)  annotation( ...);
  Modelica.Fluid.Pipes.StaticPipe pipe3(redeclare package Medium = Medium,                   (6)
    diameter = 0.01414, length = 1)  annotation( ...);
  Modelica.Fluid.Sources.FixedBoundary ambient(redeclare package Medium = Medium,            (7)
    nPorts = 1)  annotation( ...);
  Modelica.Fluid.Sensors.TemperatureTwoPort temperature(redeclare package Medium = Medium)   (8)
    annotation( ...);
  Modelica.Blocks.Sources.Ramp massFlow(duration = 10, height = 1.95, offset = 0.05,         (9)
    startTime = 0)  annotation( ...);
```

WaterMix1

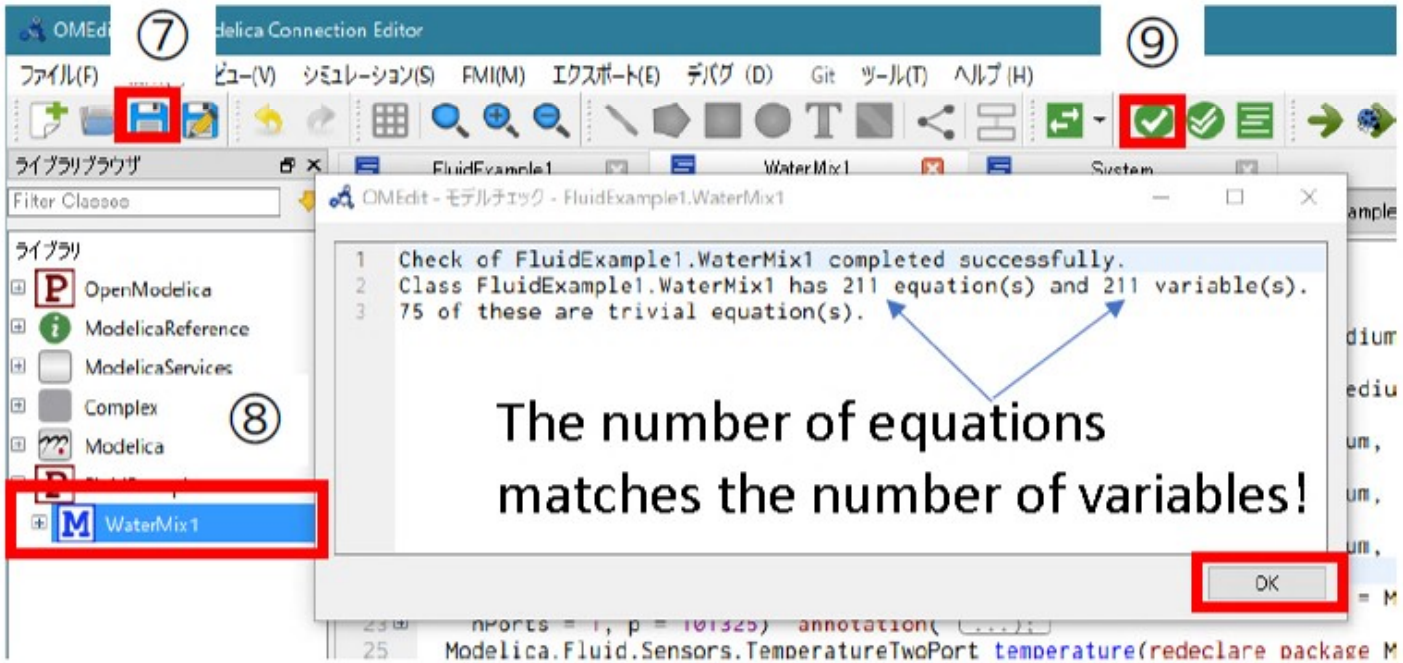
```
equation
    connect(massFlow.y, hotWater.m_flow_in) annotation( ...);
    connect(coldWater.ports[1], pipe2.port_a) annotation( ...);
    connect(hotWater.ports[1], pipe1.port_a) annotation( ...);
    connect(pipe2.port_b, pipe3.port_a) annotation( ...);
    connect(pipe1.port_b, pipe3.port_a) annotation( ...);
    connect(temperature.port_b, ambient.ports[1]) annotation( ...);
    connect(temperature.port_a, pipe3.port_b) annotation( ...);
end WaterMix1;

  annotation( ...);
end FluidExample1;
```

⑦ Save

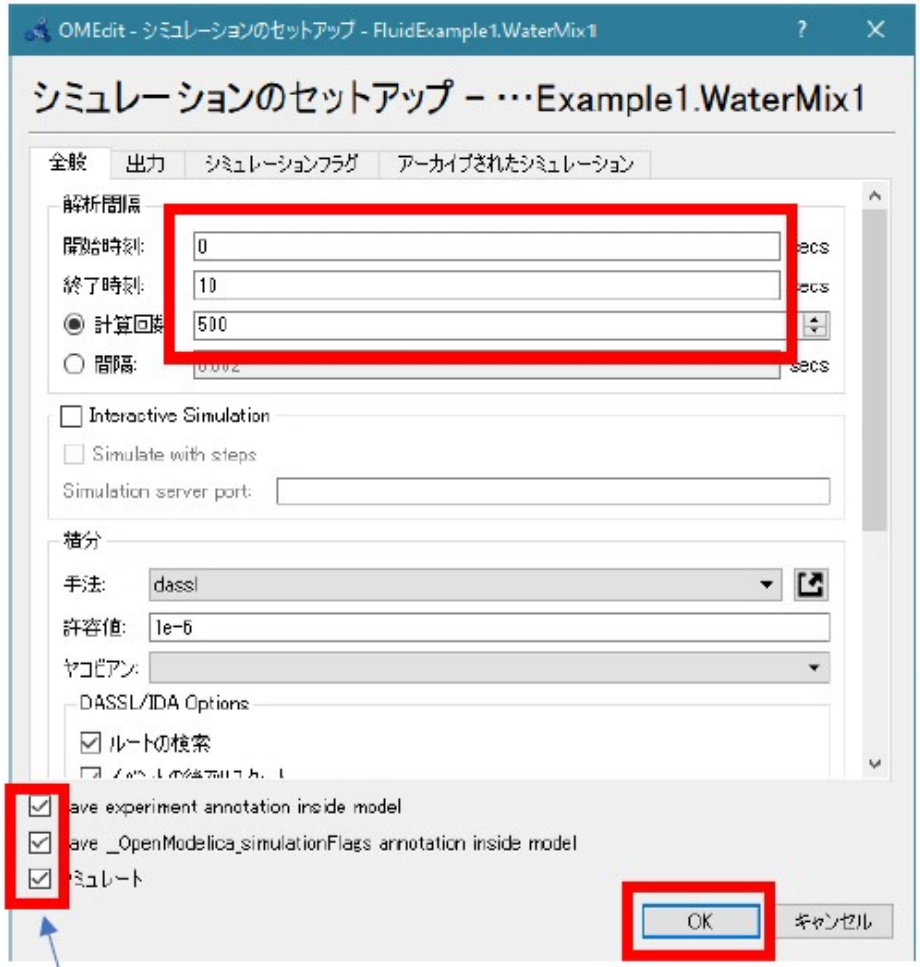
⑧Double-click WaterMix1 to activate it,

⑨ Click the [Model Check] button. Hopefully the screen on the right will be displayed.



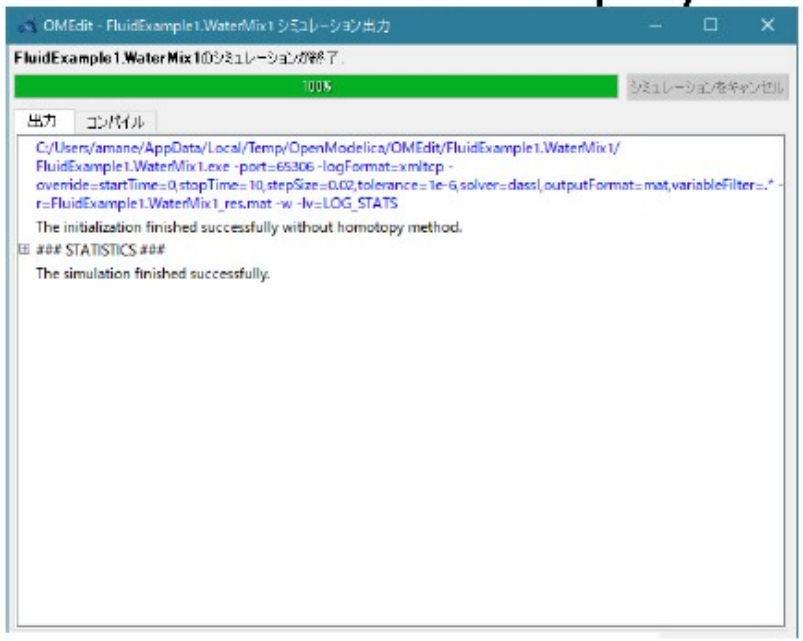
WaterMix1

⑩ Simulation> Simulation setup



- Start time 0 [sec]
- End time 10 [sec]
- Number of calculations 500

If the simulation is successful, this screen will be displayed.



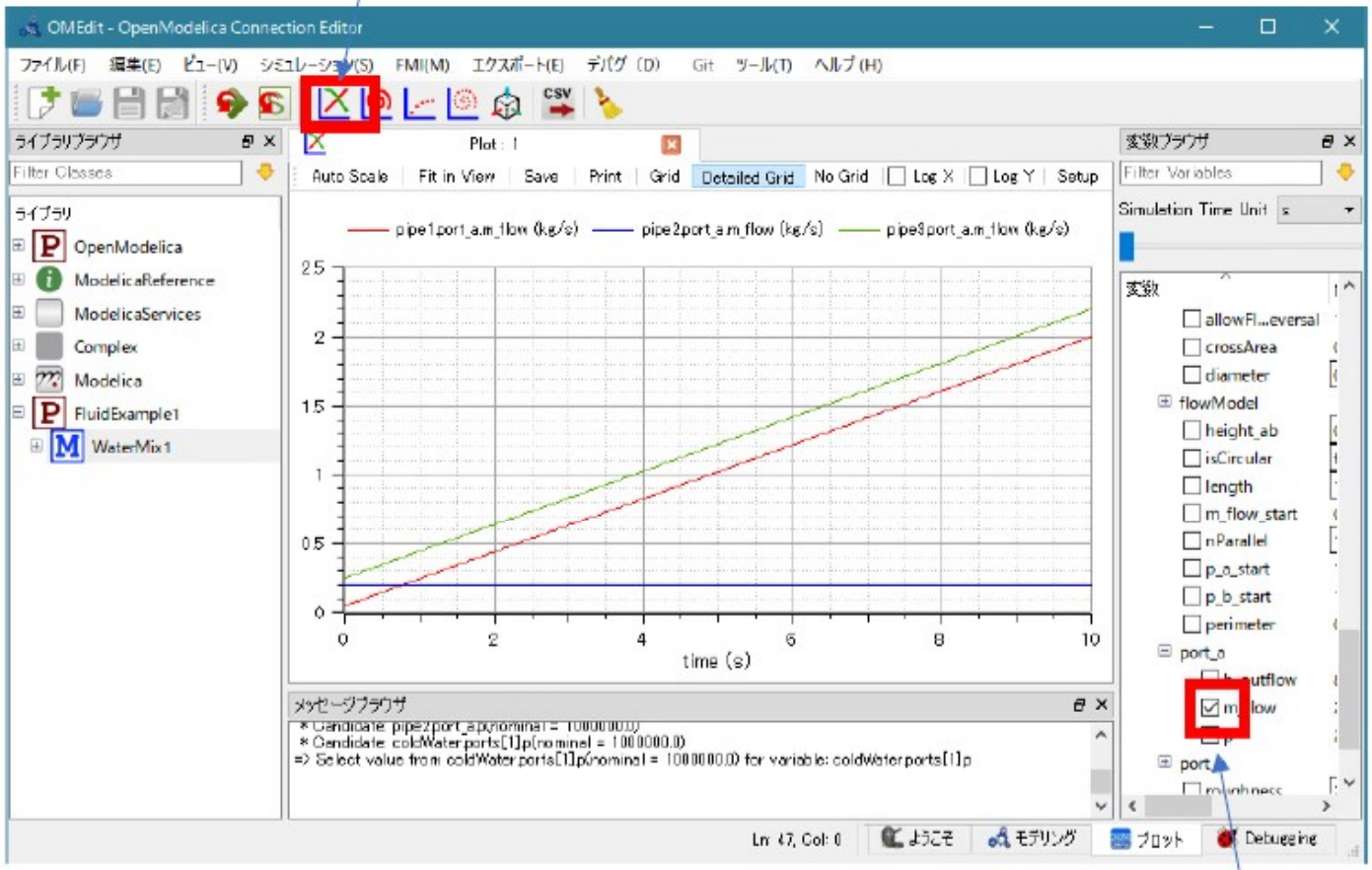
Checking this saves the calculation conditions in the model.

Checking the simulation starts the simulation with the [OK] button.

WaterMix1

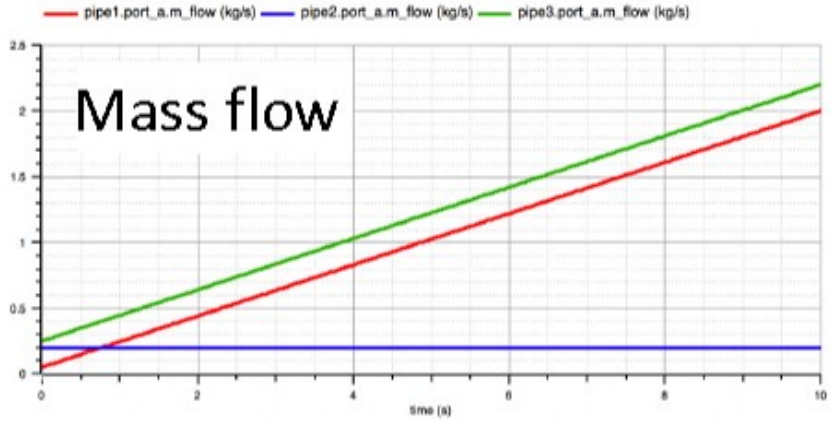
How to plot simulation results

① Create a new plot window.

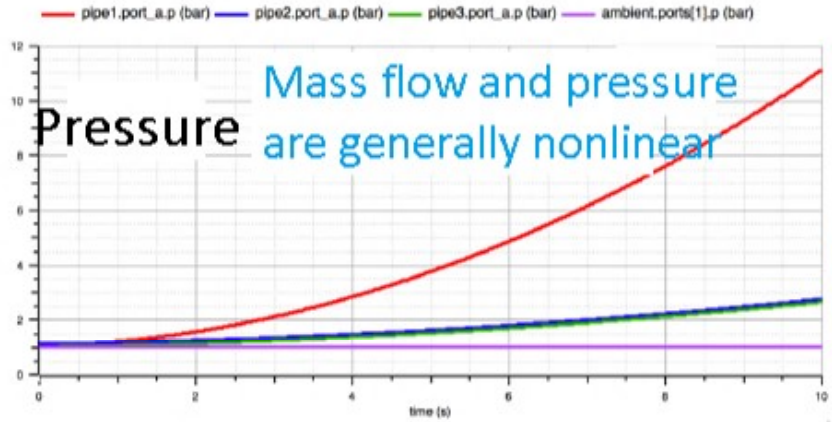


② Check variables to plot

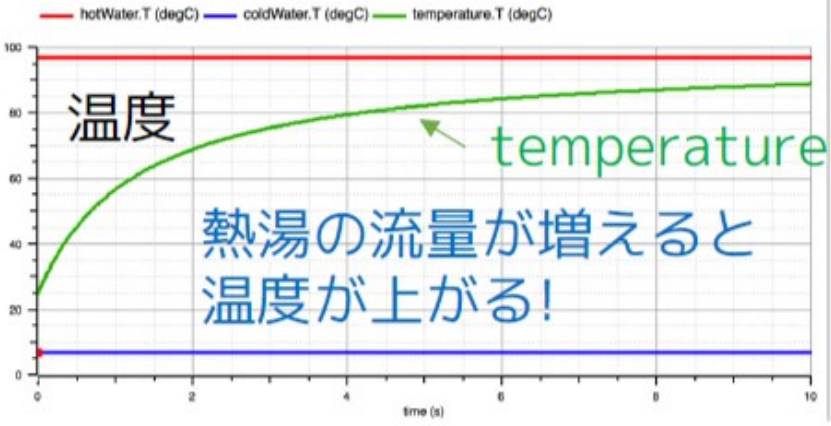
WaterMix1



pipe3
pipe1
pipe2



pipe1 large
pipe2 large
pipe3 large
atmosphere



hotWater
(Temperature sensor)
coldWate
r

Variable to plot

- pipe1.port_a.m_flow
- pipe2.port_a.m_flow
- pipe3.port_a.m_flow

- pipe1.port_a.p
- pipe2.port_a.p
- pipe3.port_a.p
- ambient.ports[1].p

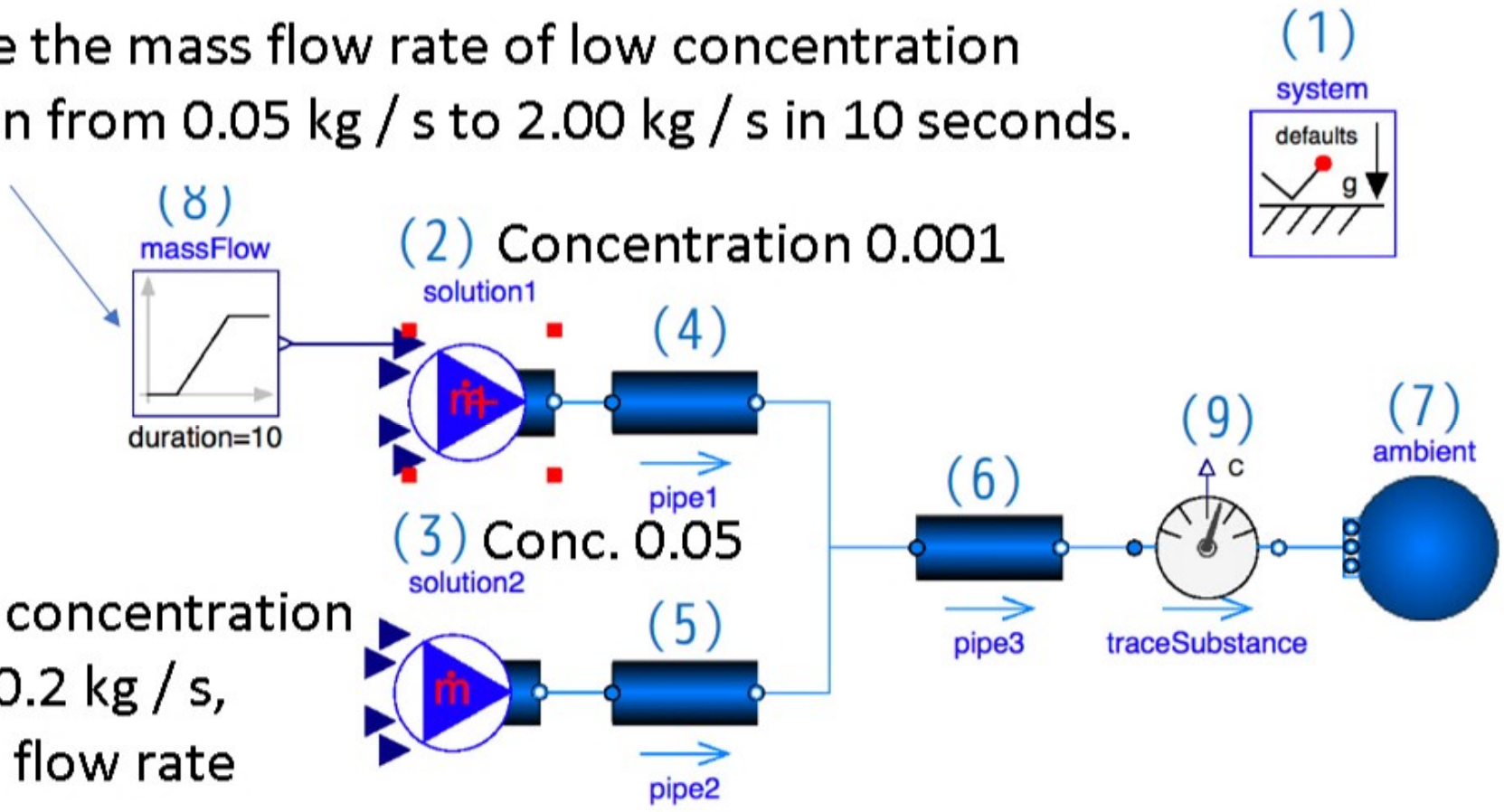
- hotWater.T
- coldWater.T
- temperature.T

TraceSubstanceMix1

Mix water with different chlorine concentrations

Chlorine is defined as extraProperties (additional substance), and high-concentration solution and low-concentration solution are combined at FluidPort.

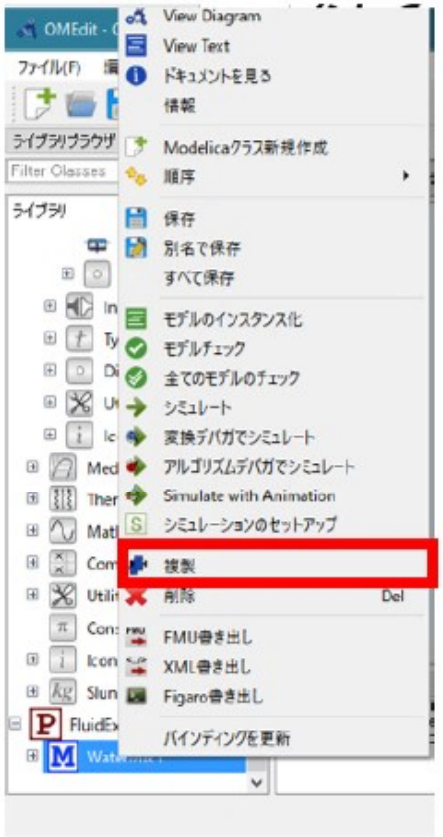
Change the mass flow rate of low concentration solution from 0.05 kg / s to 2.00 kg / s in 10 seconds.

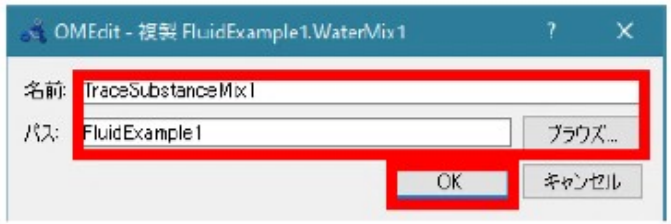


High concentration side 0.2 kg / s, fixed flow rate

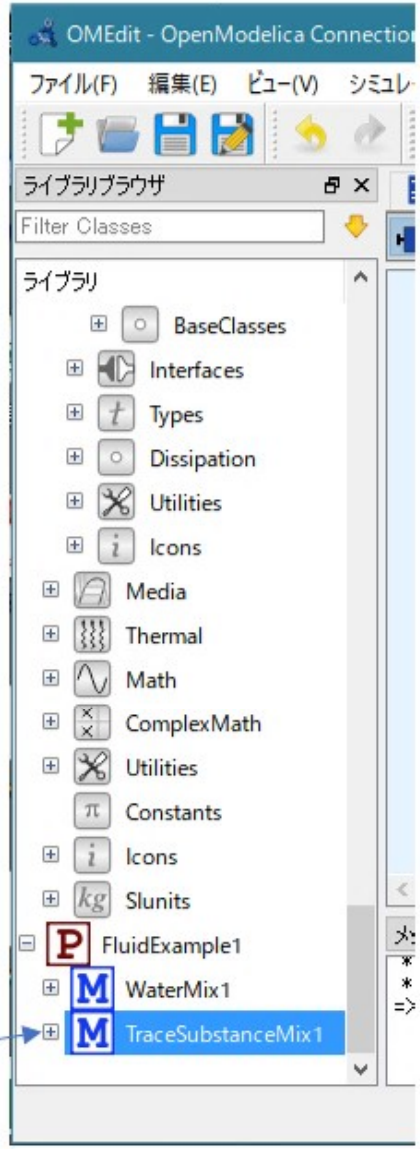
TraceSubstanceMix1

① Right-click WaterMix1 in the library browser and select Duplicate to create TraceSubstanceMix1.





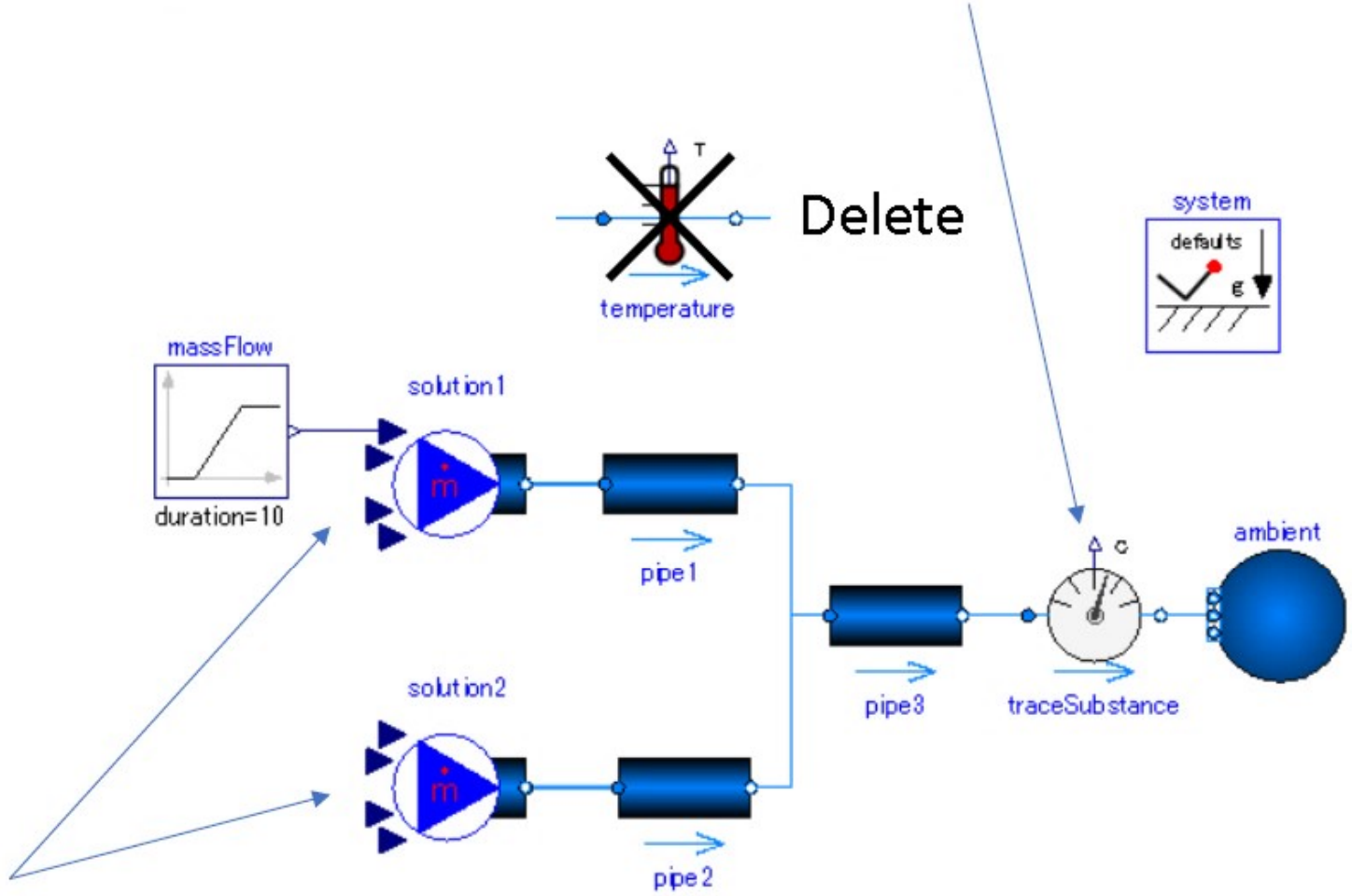
- Name TraceSubstanceMix1
- Path FluidExample1



TraceSubstanceMix1

TraceSubstanceMix1

② Right-click on temperature and select [Delete] to delete it. Instead, drag and drop **Fluid.Sensors.TraceSubstancesTwoPort** to connect.



③ Right-click hotWater and coldWater, select an attribute, and rename them to solution1 and solution2, respectively.

TraceSubstanceMix1

④

Edit the code

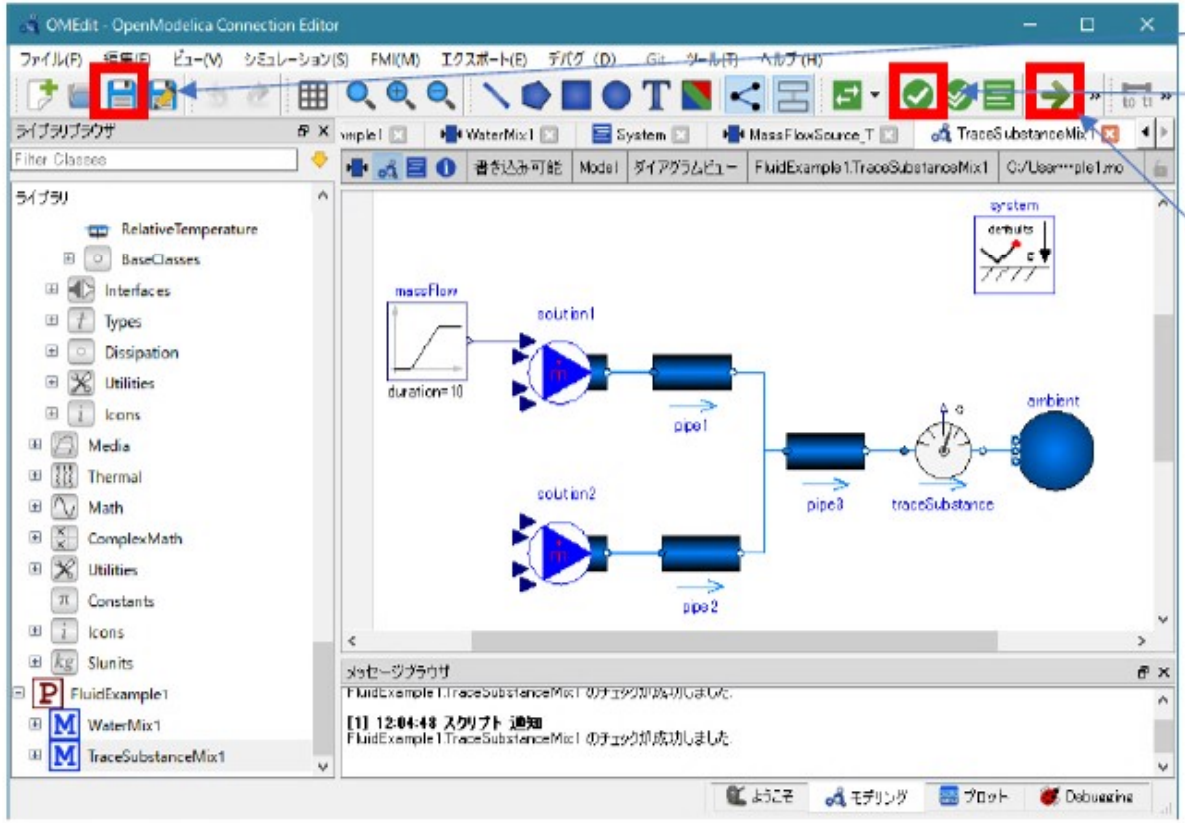
Setting extraProperties

```
model TraceSubstanceMix1
   replaceable package Medium = Media.Water.StandardWater(extraPropertiesNames={"chlorine"},
     C_nominal={0.005});
   inner Modelica.Fluid.System system annotation( ...);                                    (1)
   Modelica.Fluid.Sources.MassFlowSource_T solution1(redeclare package Medium = Medium,    (2)
     C = {0.001}, T = 293.15, nPorts = 1, use_m_flow_in = true) annotation( ...);
   Modelica.Fluid.Sources.MassFlowSource_T solution2(redeclare package Medium = Medium,    (3)
     C = {0.05}, T = 293.15, m_flow = 0.2, nPorts = 1) annotation( ...);
   Modelica.Fluid.Pipes.StaticPipe pipe1(redeclare package Medium = Medium,                (4)
     diameter = 0.01, length = 1) annotation( ...);
   Modelica.Fluid.Pipes.StaticPipe pipe2(redeclare package Medium = Medium,                (5)
     diameter = 0.01, length = 1) annotation( ...);
   Modelica.Fluid.Pipes.StaticPipe pipe3(redeclare package Medium = Medium,                (6)
     diameter = 0.01414, length = 1) annotation( ...);                                     (7)
   Modelica.Fluid.Sources.FixedBoundary ambient(redeclare package Medium = Medium, nPorts = 1)
     annotation( ...);
   Modelica.Blocks.Sources.Ramp massFlow(duration = 10, height = 1.95, offset = 0.05,      (8)
     startTime = 0) annotation( ...);
   Modelica.Fluid.Sensors.TraceSubstancesTwoPort traceSubstance(                           (9)
     redeclare package Medium = Medium, substanceName = "chlorine")  annotation( ...);
```

Setting the substance to be displayed by the concentration sensor

TraceSubstanceMix1

```
equation
    connect(solution1.ports[1], pipe1.port_a) annotation( ...);
    connect(massFlow.y, solution1.m_flow_in) annotation( ...);
    connect(traceSubstance.port_a, pipe3.port_b) annotation( ...);
    connect(ambient.ports[1], traceSubstance.port_b) annotation( ...);
    connect(solution2.ports[1], pipe2.port_a) annotation( ...);
    connect(pipe2.port_b, pipe3.port_a) annotation( ...);
    connect(pipe1.port_b, pipe3.port_a) annotation( ...);
end TraceSubstanceMix1;
```

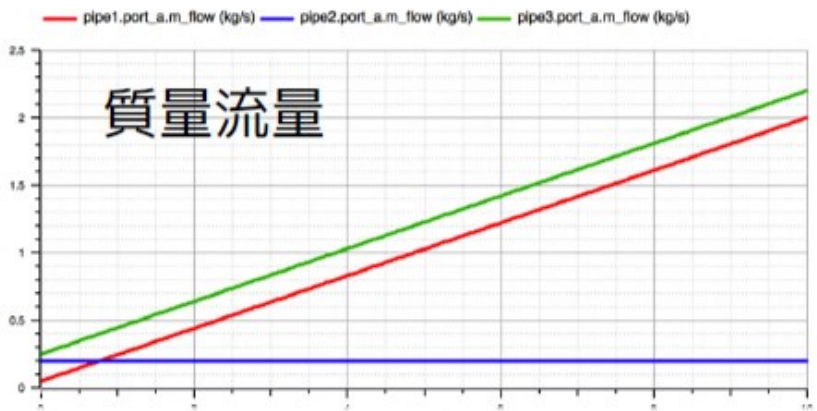


④ Save

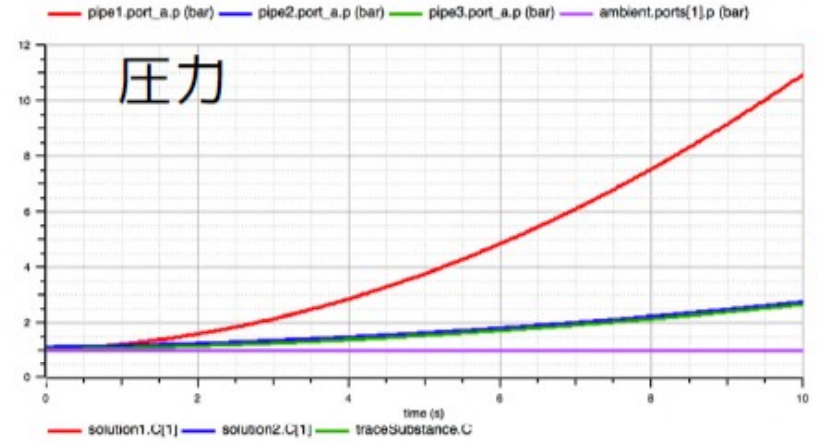
⑤Execute the check model

⑥ Run the simulation

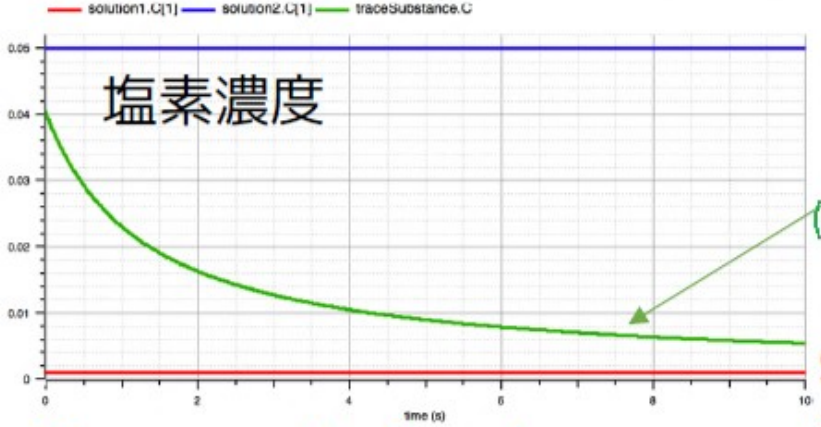
TraceSubstanceMix1 Simulation result



pipe3
pipe1
pipe2



pipe1
pipe2
pipe3
atm. pressure



solutions 2
traceSubstance
(Concentration sensor)
solutions 1

Variable to plots

- pipe1.port_a.m_flow
- pipe2.port_a.m_flow
- pipe3.port_a.m_flow

- pipe1.port_a.p
- pipe2.port_a.p
- pipe3.port_a.p
- ambient.ports[1].p

- solution1.C[1]
- solution2.C[1]
- traceSubstance.C

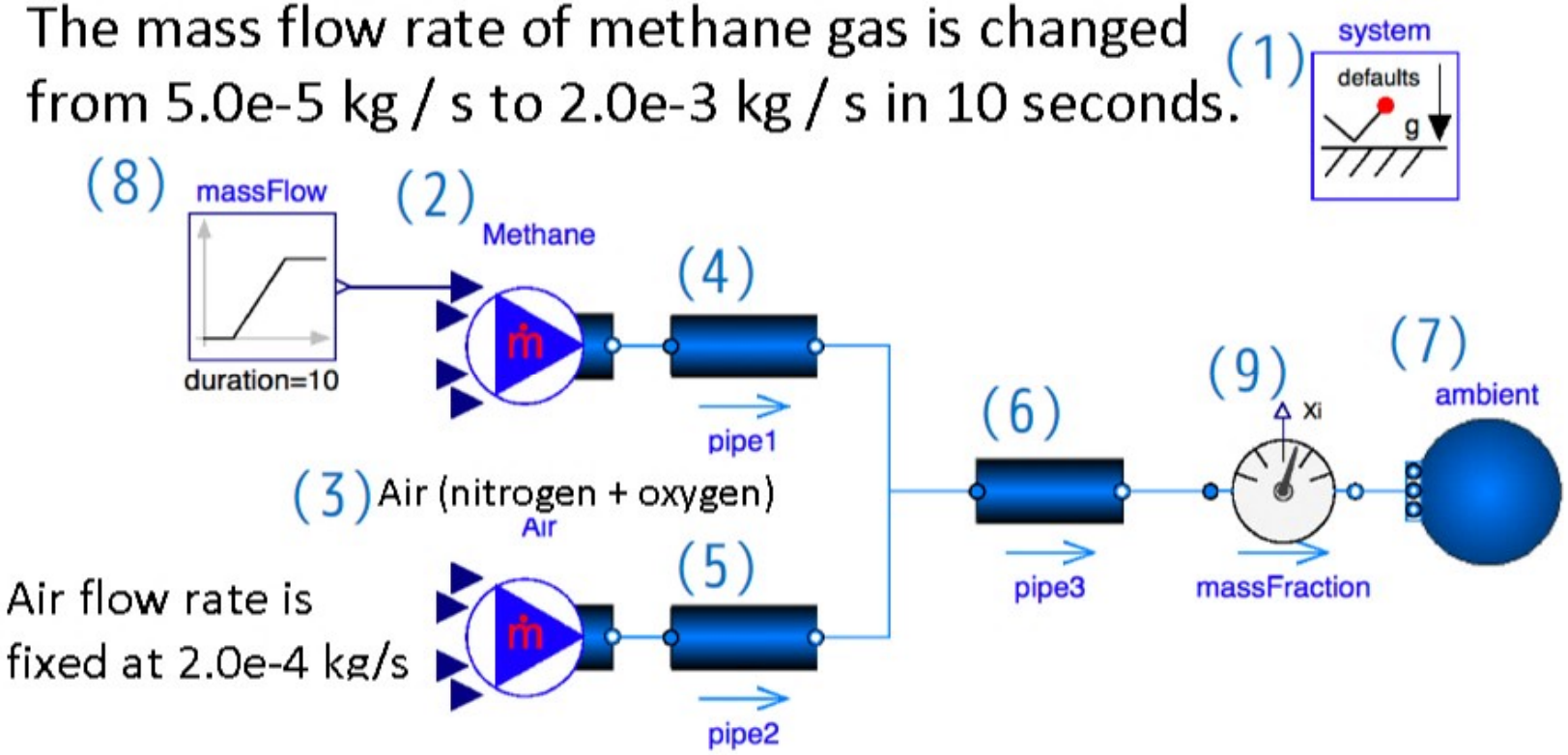
The concentration decreases as the flow rate of low concentration water increases!

GasMix1

Mix methane gas and air

Create a physical property model (Medium package) that can represent the mixed state of methane gas and air, and create a model in which methane gas and air are mixed at FluidPort.

The mass flow rate of methane gas is changed from 5.0e-5 kg / s to 2.0e-3 kg / s in 10 seconds.



GasMix1

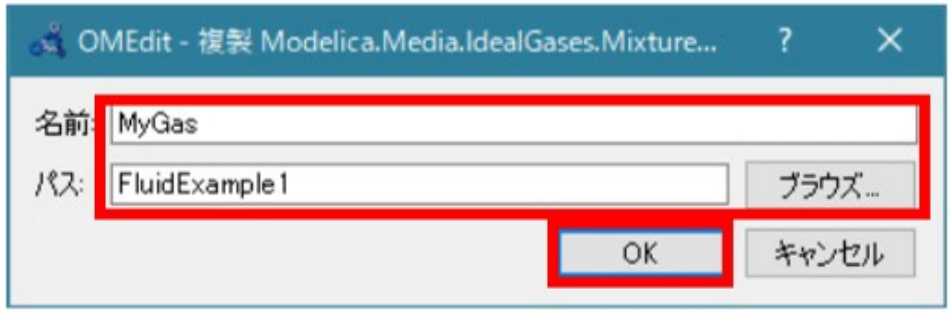
Procedure

1. Create MyGas, which represents a mixture of methane gas and air.
2. Modify the mass fraction sensor.
3. Create a system model.

GasMix1

1. Create MyGas representing a mixture of methane gas and air

① Right-click Modelica.Media.IdealGases.MixtureGases.CombustionAir, select Duplicate, and copy it to the path FluidExample1 as MyGas.



- Name: MyGas
- Path: FluidExample1

② Add the following import statement to the beginning of FluidExample1 to solve the problem of MyGas scope.

First part of FluidExample1

```
package FluidExample1
  import Modelica.Media;
  import Modelica.Media.IdealGases.Common;

  model WaterMix1
  ...
```

GasMix1

③ Edit MyGas as follows and change it to represent the mixed gas of methane gas and air.

MyGas (mixed gas of methane and air)

```
package MyGas "Nitrogen and Air"
   extends Common.MixtureGasNasa(mediumName = "MyGas",
   data = {Common.SingleGasesData.CH4, Common.SingleGasesData.N2, Common.SingleGasesData.O2},
       fluidConstants = {Common.FluidData.CH4, Common.FluidData.N2, Common.FluidData.O2},
       substanceNames = {"Methane", "Nitrogen", "Oxygen"},
       reference_X = {0.5, 0.4, 0.1});
    annotation( ...);
end MyGas;
```

The sum should be 1.0

GasMix1

2. Modification of mass fraction sensor

Fluid.Sensors.MassFractionsTwoPort (mass fraction sensor) has a bug, so copy it to FluidExample1 to create MassFractionsTwoPort1, modify it, and use it.

```
model MassFractionsTwoPort1 "Ideal two port sensor for mass fraction"
    extends Modelica.Fluid.Sensors.BaseClasses.PartialFlowSensor;
    extends Modelica.Icons.RotationalSensor;
    Modelica.Blocks.Interfaces.RealOutput Xi "Mass fraction in port medium"
annotation( ...);
    parameter String substanceName = "water" "Name of mass fraction";
  protected
    parameter Integer ind(fixed = false) "Index of species in vector of independent
mass fractions";
  initial algorithm
    ind := -1;
    for i in 1:Medium.nX loop
      if Modelica.Utilities.Strings.isEqual(Medium.substanceNames[i], substanceName)
then
        ind := i;
      end if;
```

Modify **Medium.nC** to **Medium.nX**

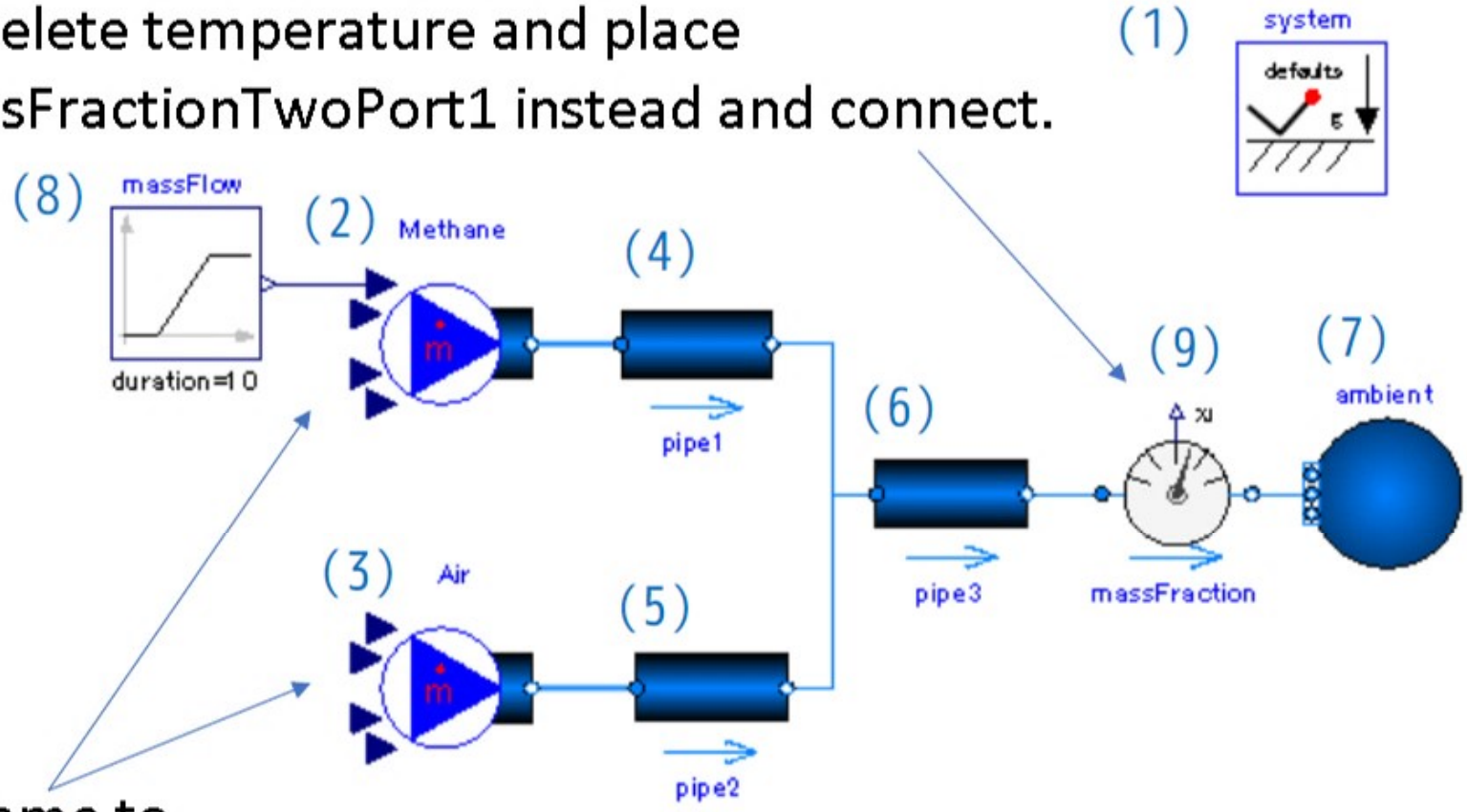
Abbreviated below

GasMix1

3. Creating a system model

① Create a model named GasMix1 by duplicating WaterMix

② Delete temperature and place MassFractionTwoPort1 instead and connect.



Rename to Methane and Air

GasMix1

④ Edit the GasMix1 code.

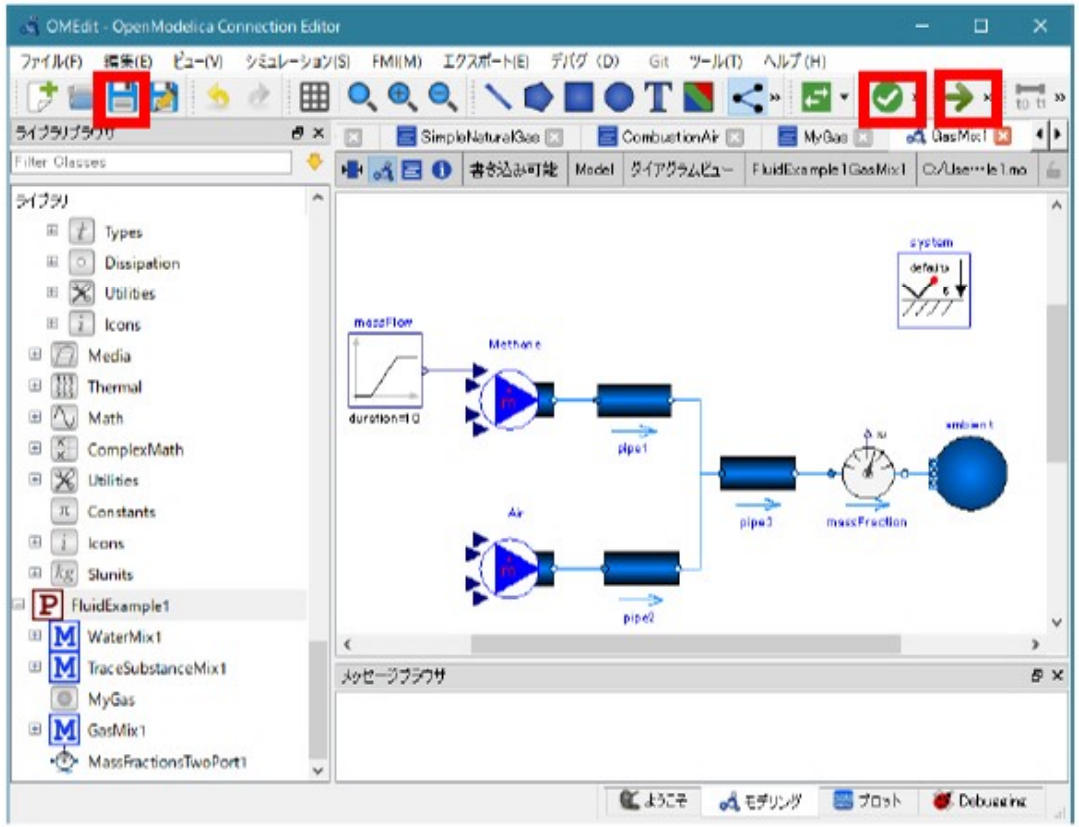
The air side is set so that the components are only nitrogen and oxygen.

The methane side is set so that the component is methane only.

```
model GasMix1
  replaceable package Medium = MyGas;
  inner Modelica.Fluid.System system annotation( ...);                                    (1)
  Modelica.Fluid.Sources.MassFlowSource_T Methane(redeclare package Medium = Medium,      (2)
    T = 293.15, X = {1.0, 0.0, 0.0}, nPorts = 1, use_m_flow_in = true) annotation( ...);
  Modelica.Fluid.Sources.MassFlowSource_T Air(redeclare package Medium = Medium,          (3)
    X = {0.0, 0.8, 0.2}, T = 293.15, m_flow = 2.0e-4, nPorts = 1) annotation( ...);
  Modelica.Fluid.Pipes.StaticPipe pipe1(redeclare package Medium = Medium,                (4)
    diameter = 0.01, length = 1) annotation( ...);
  Modelica.Fluid.Pipes.StaticPipe pipe2(redeclare package Medium = Medium,                (5)
    diameter = 0.01, length = 1) annotation( ...);
  Modelica.Fluid.Pipes.StaticPipe pipe3(redeclare package Medium = Medium,                (6)
    diameter = 0.01414, length = 1) annotation( ...);                                     (7)
  Modelica.Fluid.Sources.FixedBoundary ambient(redeclare package Medium = Medium, nPorts = 1)
    annotation( ...);
  Modelica.Blocks.Sources.Ramp massFlow(duration = 10, height = 1.95e-3, offset = 5.0e-5,  (8)
    startTime = 0) annotation( ...);
  FluidExample3.MassFractionsTwoPort1 massFraction(redeclare package Medium = Medium,     (9)
    substanceName = Medium.substanceNames[1]) annotation( ...);
```

GasMix1

```
equation
   connect(Air.ports[1], pipe2.port_a) annotation( ...);
   connect(massFraction.port_b, ambient.ports[1]) annotation( ...);
   connect(massFraction.port_a, pipe3.port_b) annotation( ...);
   connect(Methane.ports[1], pipe1.port_a) annotation( ...);
   connect(massFlow.y, Methane.m_flow_in) annotation( ...);
   connect(pipe2.port_b, pipe3.port_a) annotation( ...);
   connect(pipe1.port_b, pipe3.port_a) annotation( ...);
 end GasMix1;
```



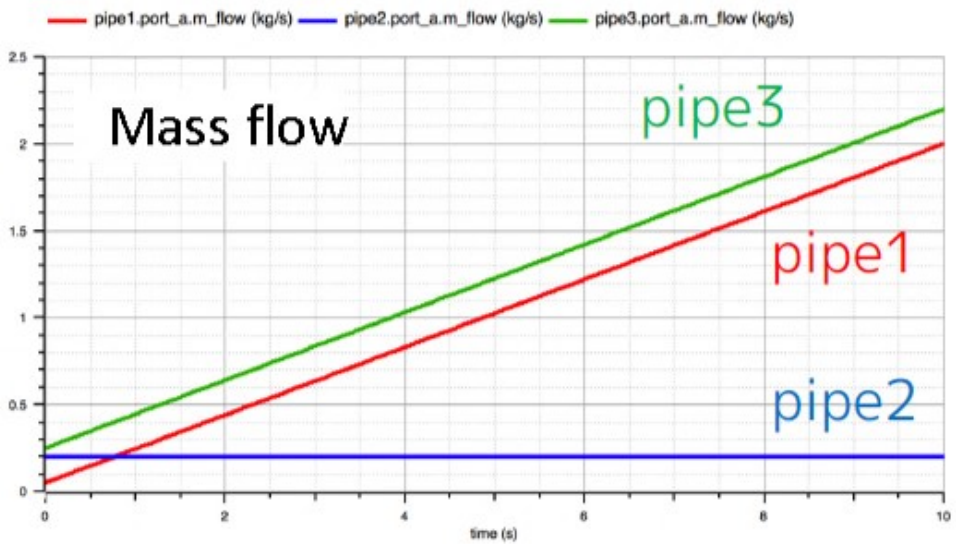
⑤ Save.

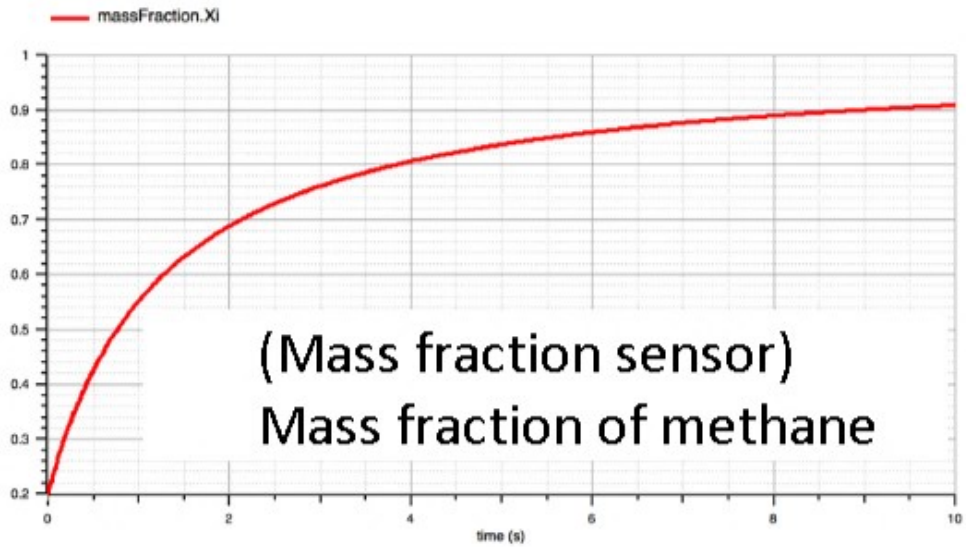
⑥ Execute the check model.

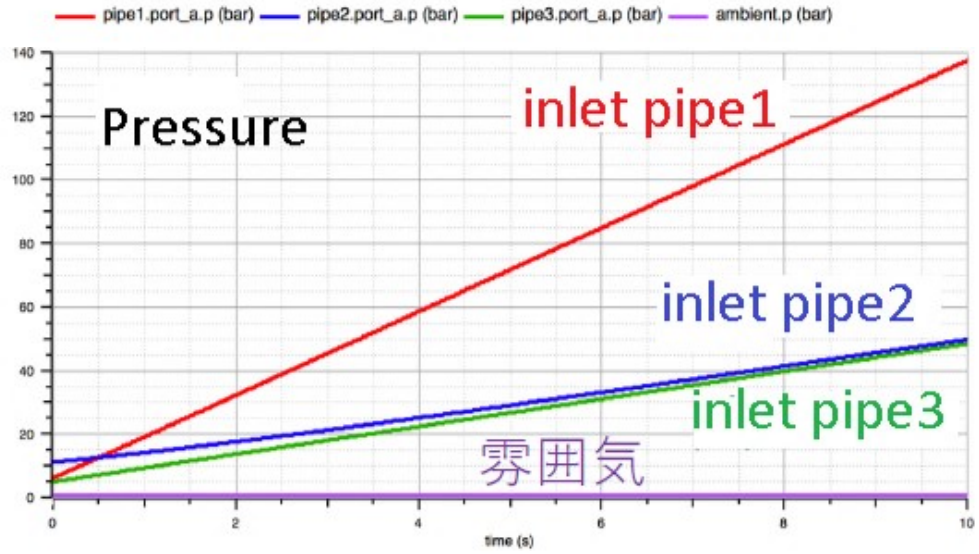
⑦ Run the simulation.

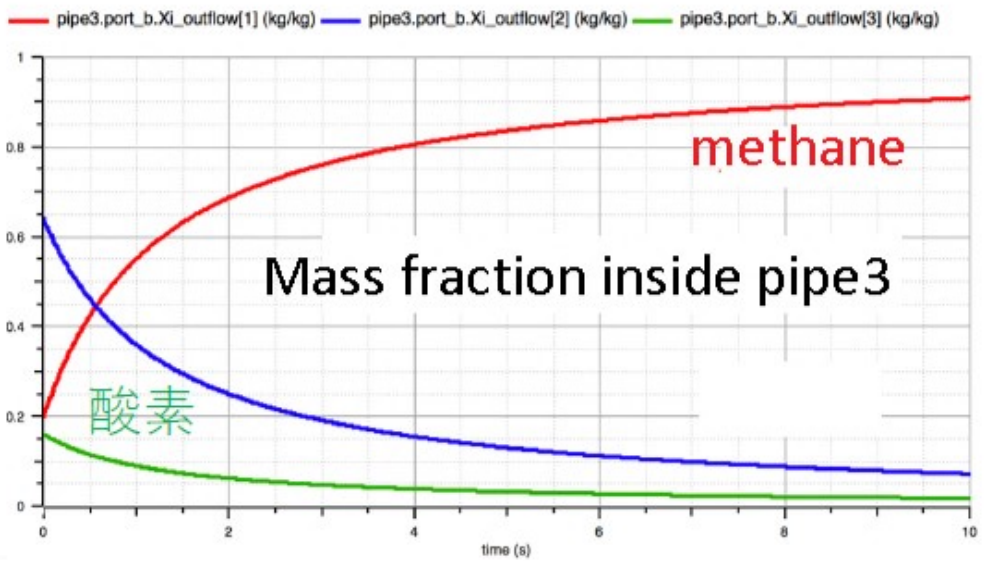
GasMix1

As the flow rate of methane increases, the mass fraction of methane increases!







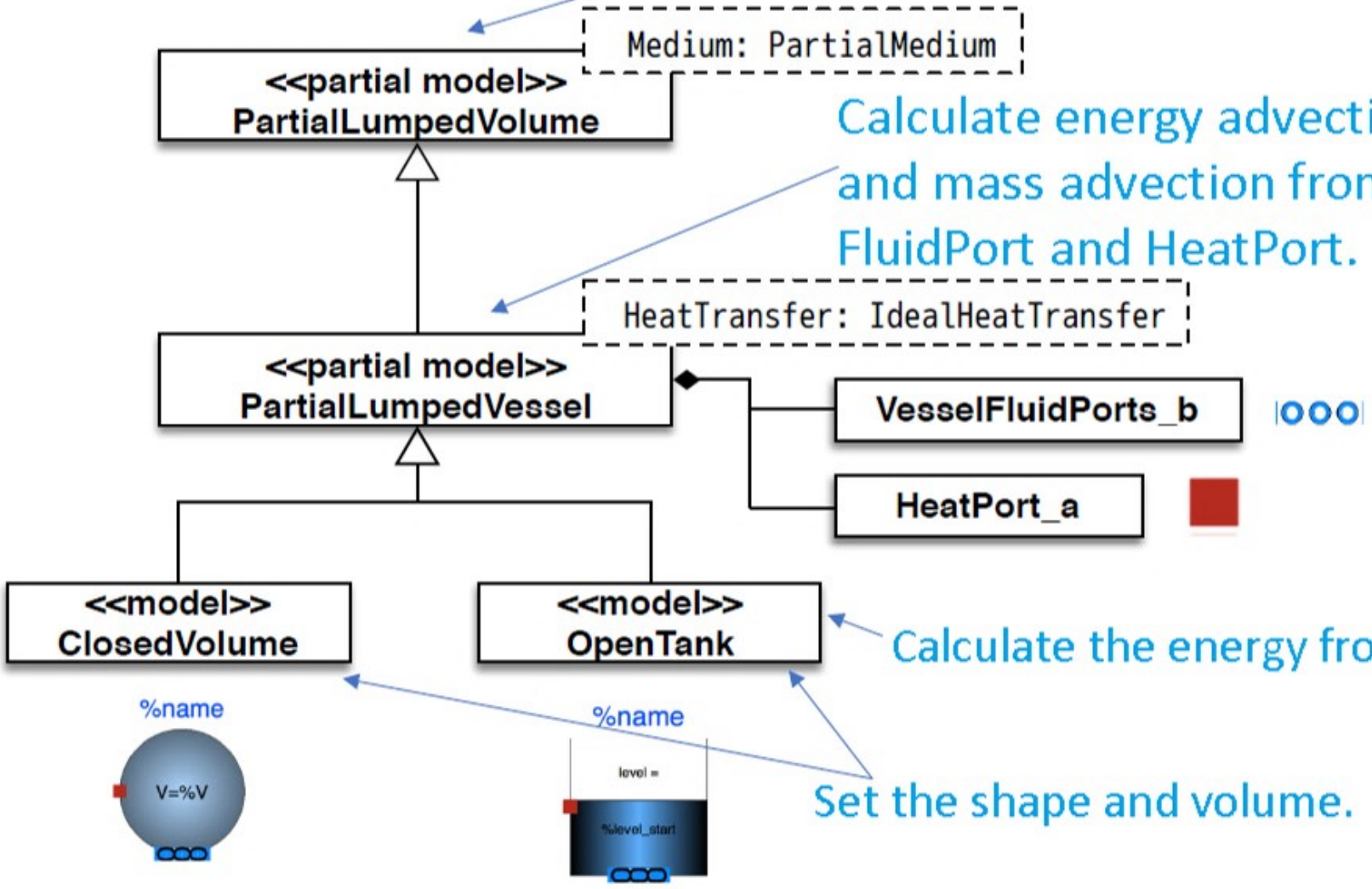


FluidExample2

Volume model

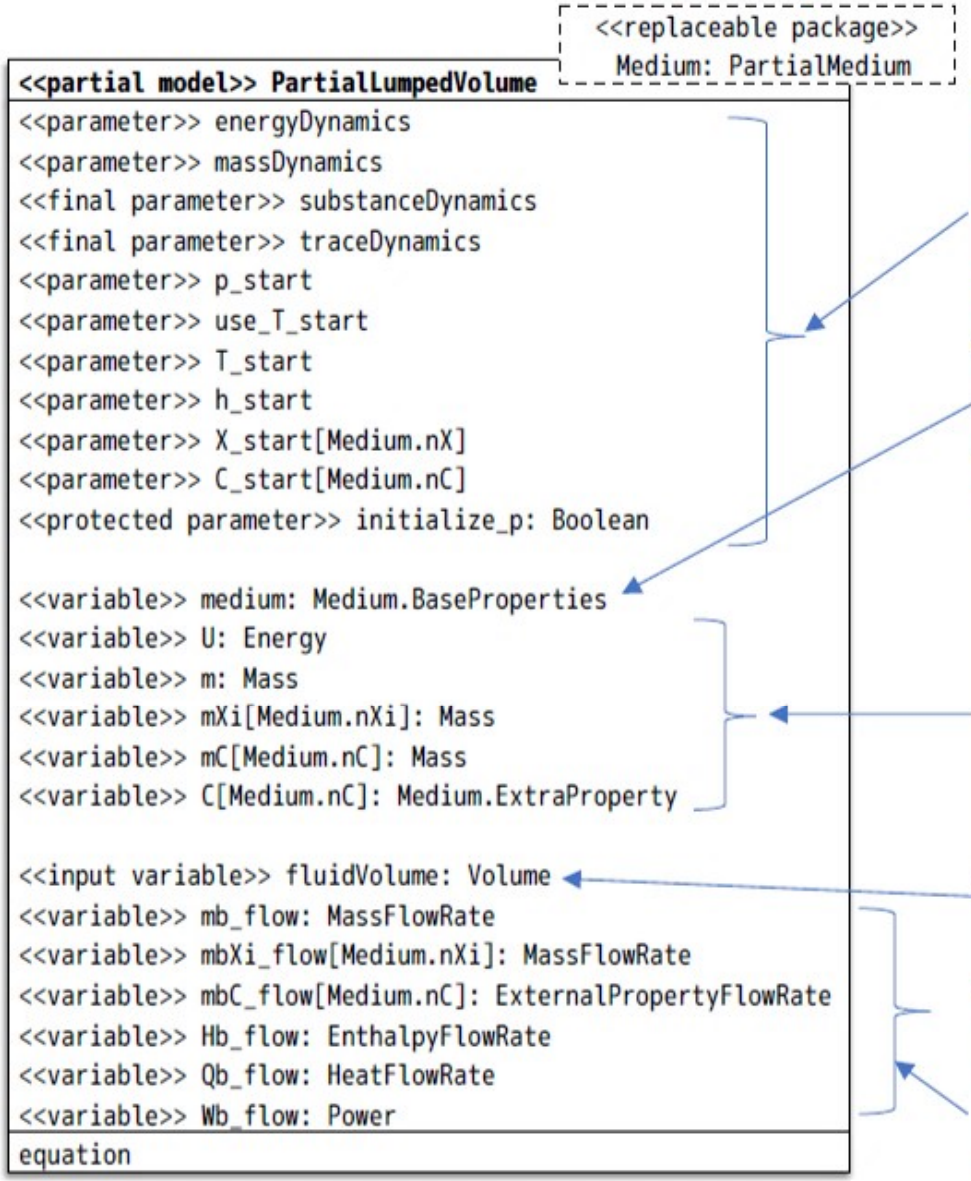
A model that stores the fluid inside by implementing the energy conservation formula and the mass conservation formula

Implement energy conservation and mass conservation formula.



volume model

Base model of PartialLumpedVolume model



Parameters to set steady / unsteady initial conditions

medium: BaseProperties model

State variables in the container

U: Energy

m: Mass

mX_i: Component mass

mC: Mass of micro-subst.

$fluidVolume$: Input variable volume

Enthalpy advection terms, mass advection terms, heat transfer terms, and work terms that define the equations in the inherited model

volume model

PartialLumpedVolume equation (1)

Switching equations and initial conditions with settings such as energyDynamics and massDynamics

```
equation
  assert(not (energyDynamics<>Dynamics.SteadyState and
massDynamics==Dynamics.SteadyState) or Medium.singleState,
         "Bad combination of dynamics options and Medium not conserving mass if
fluidVolume is fixed.");

  // Total quantities
  m = fluidVolume*medium.d;
  mXi = m*medium.Xi;
  U = m*medium.u;
  mC = m*C;

  // Energy and mass balances
  if energyDynamics == Dynamics.SteadyState then
    0 = Hb_flow + Qb_flow + Wb_flow;
  else
    der(U) = Hb_flow + Qb_flow + Wb_flow;
  end if;

  if massDynamics == Dynamics.SteadyState then
    0 = mb_flow;
  else
    der(m) = mb_flow;
  end if;
```

$m = fluidVolume \cdot \rho$ Total weight in container

$\boldsymbol{mX_i} = m\,\boldsymbol{X_i}$ Total mass for each component

$U = m\,u$ Total internal energy

$\boldsymbol{mC} = m\,\boldsymbol{C}$ Total mass of minute substance

Energy conservation equation

$$\begin{cases} 0 = Hb_{fow} + Qb_{flow} + Wb_{flow} & \text{Steady} \\ \dfrac{dU}{dt} = Hb_{flow} + Qb_{flow} + Wb_{flow} & \text{Unsteady} \end{cases}$$

Mass conservation equation

$$\begin{cases} 0 = mb_{flow} & \text{Steady} \\ \dfrac{dm}{dt} = mb_{flow} & \text{Unsteady} \end{cases}$$

PartialLumpedVolume equation (2)

```
if substanceDynamics == Dynamics.SteadyState then
  zeros(Medium.nXi) = mbXi_flow;
else
  der(mXi) = mbXi_flow;
end if;

if traceDynamics == Dynamics.SteadyState then
  zeros(Medium.nC)  = mbC_flow;
else
  der(mC_scaled) = mbC_flow./Medium.C_nominal;
end if;
mC = mC_scaled.*Medium.C_nominal;
```

Preservation formula for mass of component substances

$$\begin{cases} \mathbf{0} = \boldsymbol{mbXi_{flow}} & \text{Steady} \\ \dfrac{d\,\boldsymbol{mXi}}{dt} = \boldsymbol{mbXi_{flow}} & \text{Unsteady} \end{cases}$$

Mass conservation equation for normalized micromaterials

$$\begin{cases} \mathbf{0} = \boldsymbol{mbC_{flow}} & \text{Steady} \\ \dfrac{d\,\boldsymbol{mC_{scaled}}}{dt} = \dfrac{\boldsymbol{mbC_{flow}}}{Medium.C_{nominal}} & \text{Unsteady} \end{cases}$$

Normalization of fine substance concentration

$$\boldsymbol{mC} = \boldsymbol{mC_{scaled}} \cdot Medium.C_{nominal}$$

volume model

Initialization of PartialLumpedVolume (1) (some comments are omitted)

```
initial equation
  // initialization of balances
  if energyDynamics == Dynamics.FixedInitial then
    /* ... */
    if Medium.ThermoStates == IndependentVariables.ph or
       Medium.ThermoStates == IndependentVariables.phX then
       medium.h = h_start;
    else
       medium.T = T_start;
    end if;
  elseif energyDynamics == Dynamics.SteadyStateInitial then
    /* ... */
    if Medium.ThermoStates == IndependentVariables.ph or
       Medium.ThermoStates == IndependentVariables.phX then
       der(medium.h) = 0;
    else
       der(medium.T) = 0;
    end if;
  end if;
```

$$h = h_{start} \text{ or } T = T_{start}$$

$$\frac{dh}{dt} = 0 \text{ or } \frac{\mathrm{d}T}{dt} = 0$$

volume model

Initializing PartialLumpedVolume (2)

```
if massDynamics == Dynamics.FixedInitial then
   if initialize_p then
     medium.p = p_start;
   end if;
elseif massDynamics == Dynamics.SteadyStateInitial then
   if initialize_p then
     der(medium.p) = 0;
   end if;
end if;

if substanceDynamics == Dynamics.FixedInitial then
  medium.Xi = X_start[1:Medium.nXi];
elseif substanceDynamics == Dynamics.SteadyStateInitial then
  der(medium.Xi) = zeros(Medium.nXi);
end if;

if traceDynamics == Dynamics.FixedInitial then
  mC_scaled = m*C_start[1:Medium.nC]./Medium.C_nominal;
elseif traceDynamics == Dynamics.SteadyStateInitial then
  der(mC_scaled) = zeros(Medium.nC);
end if;

annotation ( ...);
end PartialLumpedVolume;
```

$$p = p_{start}$$

$$\frac{dp}{dt} = 0$$

$$\boldsymbol{X_i = X_{start}}$$

$$\frac{\boldsymbol{dX_i}}{\boldsymbol{dt}} = 0$$

$$\boldsymbol{mC_{scaled}} = m\,\frac{\boldsymbol{C_{start}}}{C_{nominal}}$$

$$\frac{d\,\boldsymbol{mC_{scaled}}}{dt} = \mathbf{0}$$

volume model

PartialLumpedVessel

i: FluidPort number
j: number of component sub.
k: number of additional subs.

Mass advection term

$$mb_{flow} = \sum_i m\_flow[i] \ , i = 1, \dots, nPorts$$

$$mbXi_{flow}[\mathrm{j}] = \sum_i \left\{ m_{flow[i]} \cdot actualStream\left(Xi_{outflow}[i,j]\right) \right\}, j = 1, \dots, nX_i$$

$$mbC_{flow}[\mathrm{k}] = \sum_i \left\{ m_{flow[i]} \cdot actualStream\left(C_{outflow}[i,k]\right) \right\} \ , k = 1, \dots, nC$$

Energy advection term

$$Hb_{flow} = \sum_i \left\{ m_{flow[i]} \cdot \ actualStream\left(h_{outflow_i}\right) \right\}$$

Enthalpy advection from FluidPort

$$+ \sum_i \left\{ m_{flow}[i] \cdot (0.5 \ v[i]^2 + gh[i]) \right\}$$

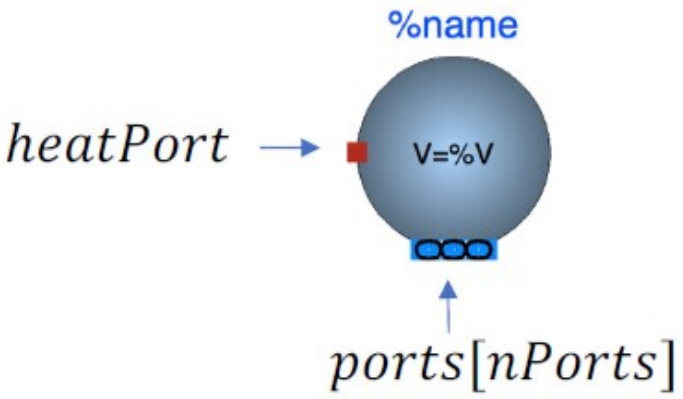
Kinetic energy and potential energy

$$Qb_{flow} = heatTransfer.Q\_flows[1]$$

Heat transfer from HeatPort

volume model

ClosedVolume Fixed size container with entrance



Shape parameters

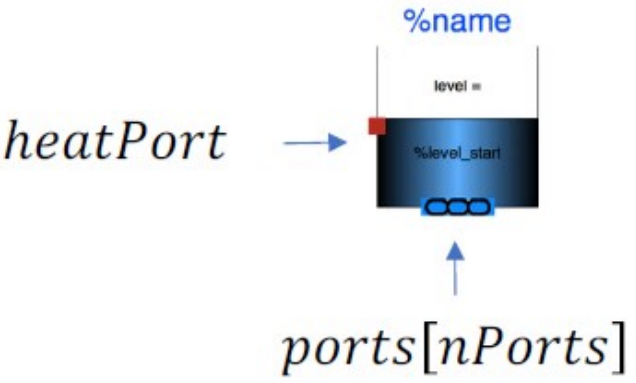
- V [m3]: Volume

Volume $V = \frac{4}{3}\pi r^3$

Heat transfer area $A = 4\pi r^2$

fluidVolume = V

OpenTank A simple tank with an entrance



Shape parameters

- $height$ [m]: Tank height
- $crossArea$ [m2]: Tank cross section
- $level\_start = 0.5 * height$ [m]: Initial liquid level
- $p_{ambient} = system.p_{ambient}$ [Pa]: Atm. pressure
- $T_{ambient} = system.T_{ambient}$ [K]: Ambient temp.

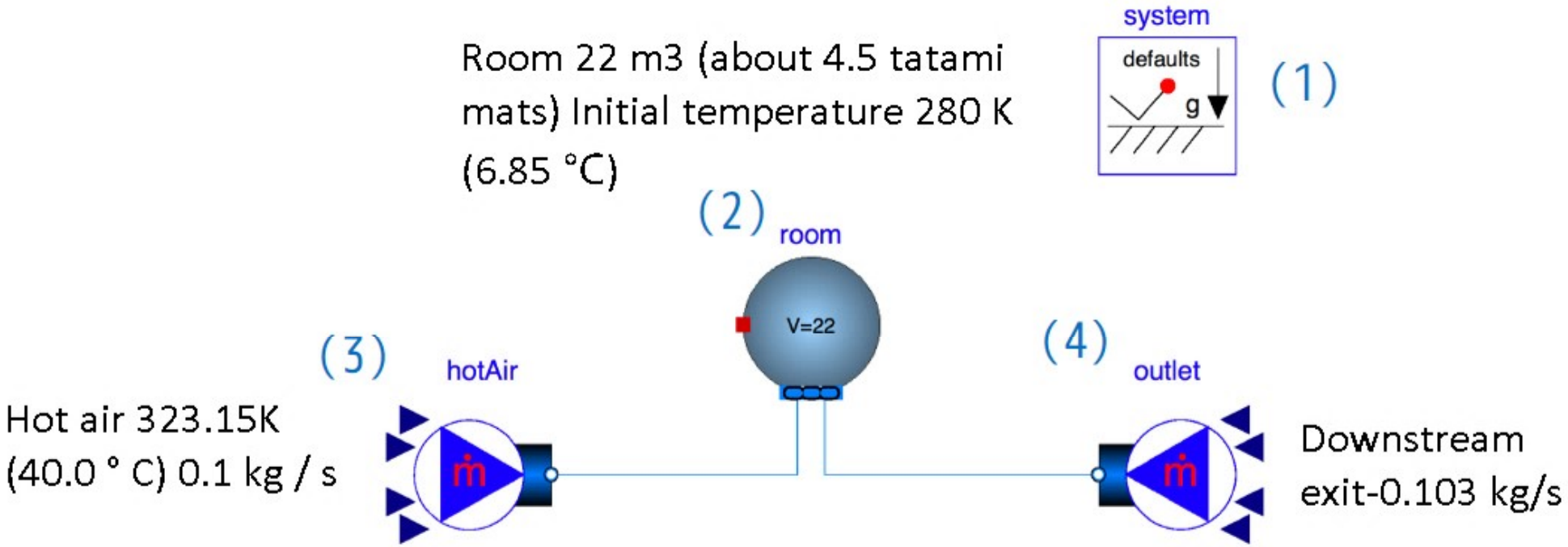
fluidVolume = V
V = crossArea*level

- The liquid level is released to the atmosphere of the fixed pressure ***p ambient;***
- There is no inflow or outflow from the top to the atmosphere, and the simulation stops when the liquid level exceeds the tank height

HotRoom1

Replace cold air in the room with warm air

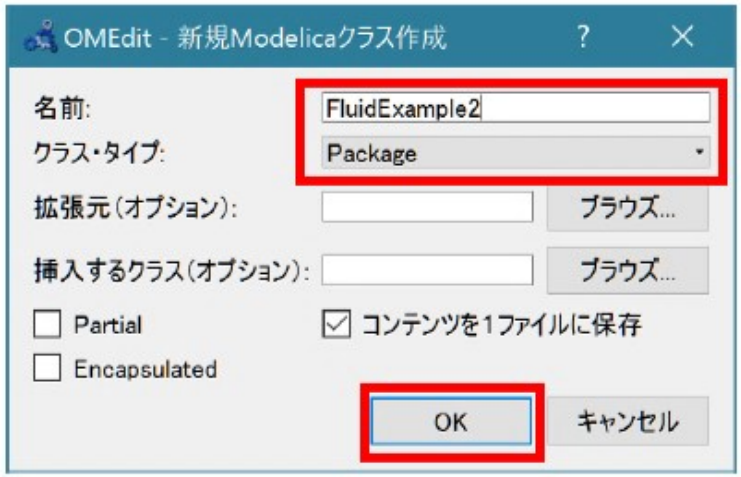
A fixed volume room is filled with cold air (6.85 °C). Blow warm air (40°C, 0.1 kg/s) from the upstream side. The temperature of the room gradually rises due to heat advection. On the downstream side, air is sucked in at 0.103 kg / s.



To specify the initial value of the thermodynamic state of the indoor air, set energyDynamics and massDynamics to FixedInitial, and set T_start = 280 [K] and P_start = 101325 [Pa]. This model cannot be used with incompressible fluids.

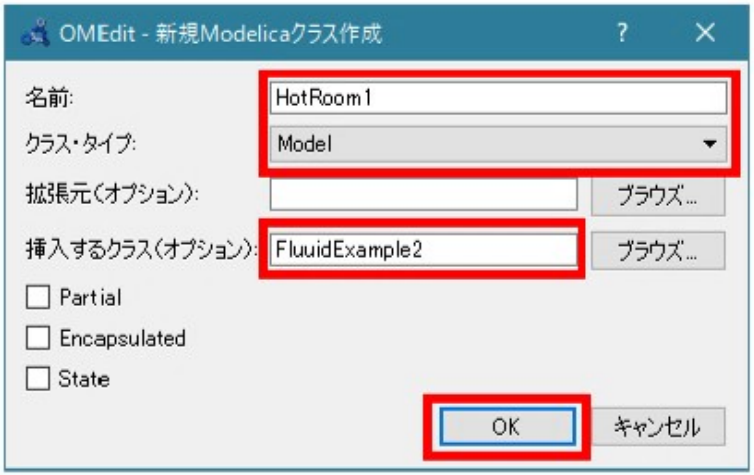
HotRoom1

① Create a package with File> New Modelica Class



Name: FluidExample2
Class type: Package

② Right-click FluidExample2 in the library browser and select "Create New Modelica Class" to create a model



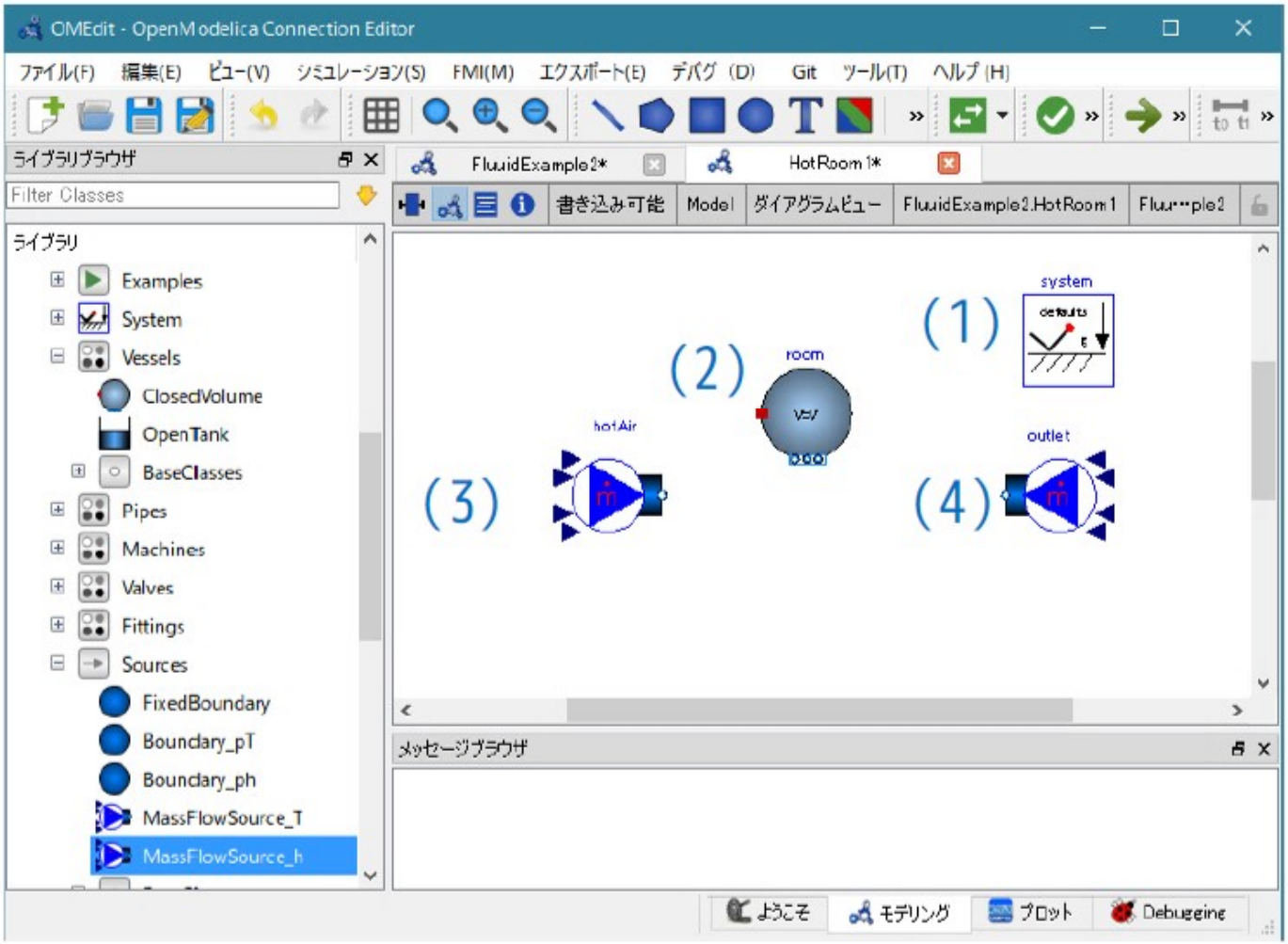
Name: HotRoom1
Class type: Model
Inserted class: FluidExample2

HotRoom1

③ Add an import statement to the text view of FluieExample2

```
package FluidExample2
  import Modelica.Media;
```

④ Place the component in the VolumeTest1 diagram view

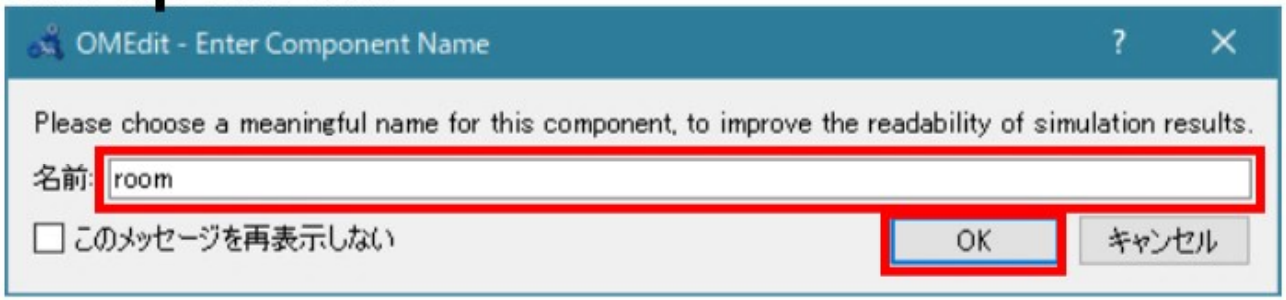


HotRoom1

Component name and class to place

(1)system: Fluid.System
(2)room: Fluid.Vessels.ClosedVolume
(3)hotAir: Fluid.Sources.MassFlowSource_T
(4)outlet: Fluid.Sources.MassFlowSource_T

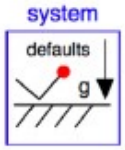
Put the name in the dialog that appears when you drop the component



HotRoom1

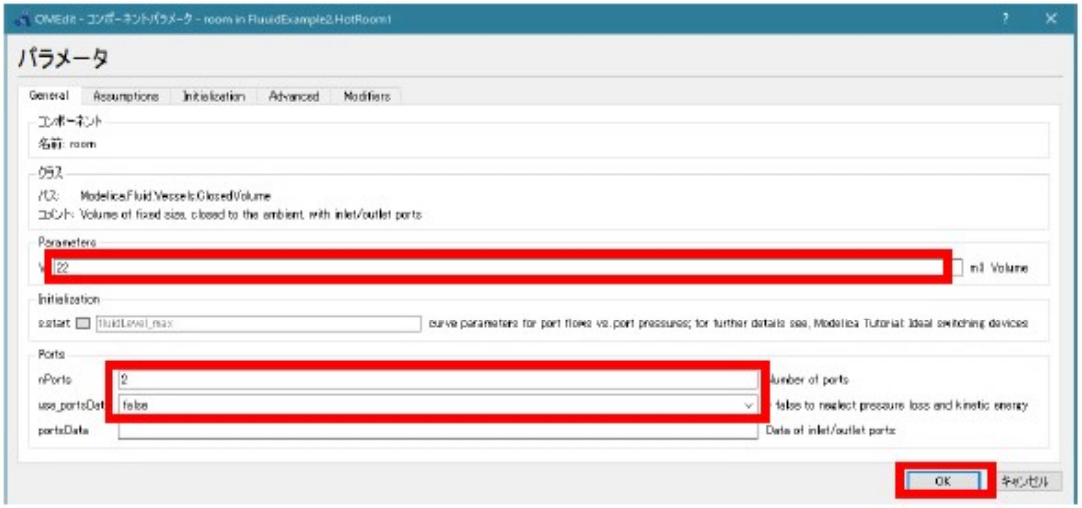
⑤ Right-click the component, select [Parameter] and enter

(1) Fluid.System



Be sure to place them in the fluid system model.

(2) Fluid.Vessels.ClosedVolume



(1) system

- Leave the default

Fixed volume container

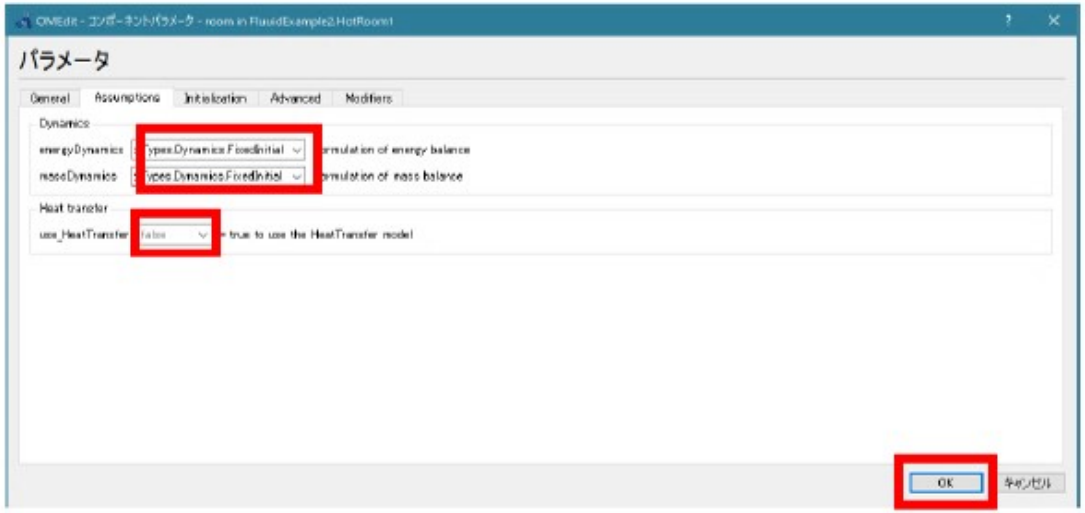
(2) room

[General]

- V = 22 [m3]
- nPorts = 2
- use_portsData = false

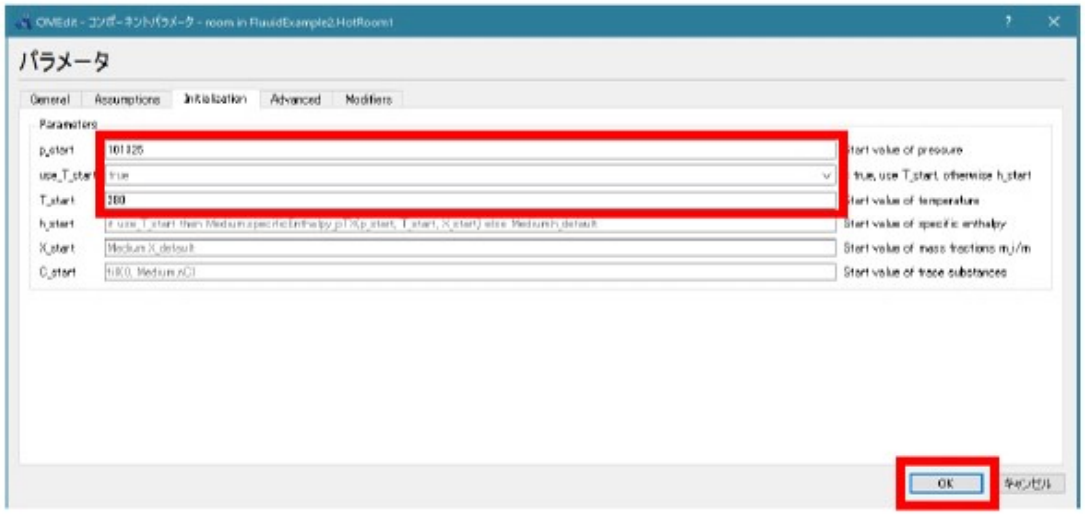
- Set nPorts = 2 to provide two FluidPorts representing entrances and exits.
- Use_portsData = false. As a result, the pressure inside the container and the pressure in the FluidPort become equal.

HotRoom1



[Assumptions]

- energyDynamics = FixedInitial
- massDynamics = FixedInitial
- use_HeatTransfer=false

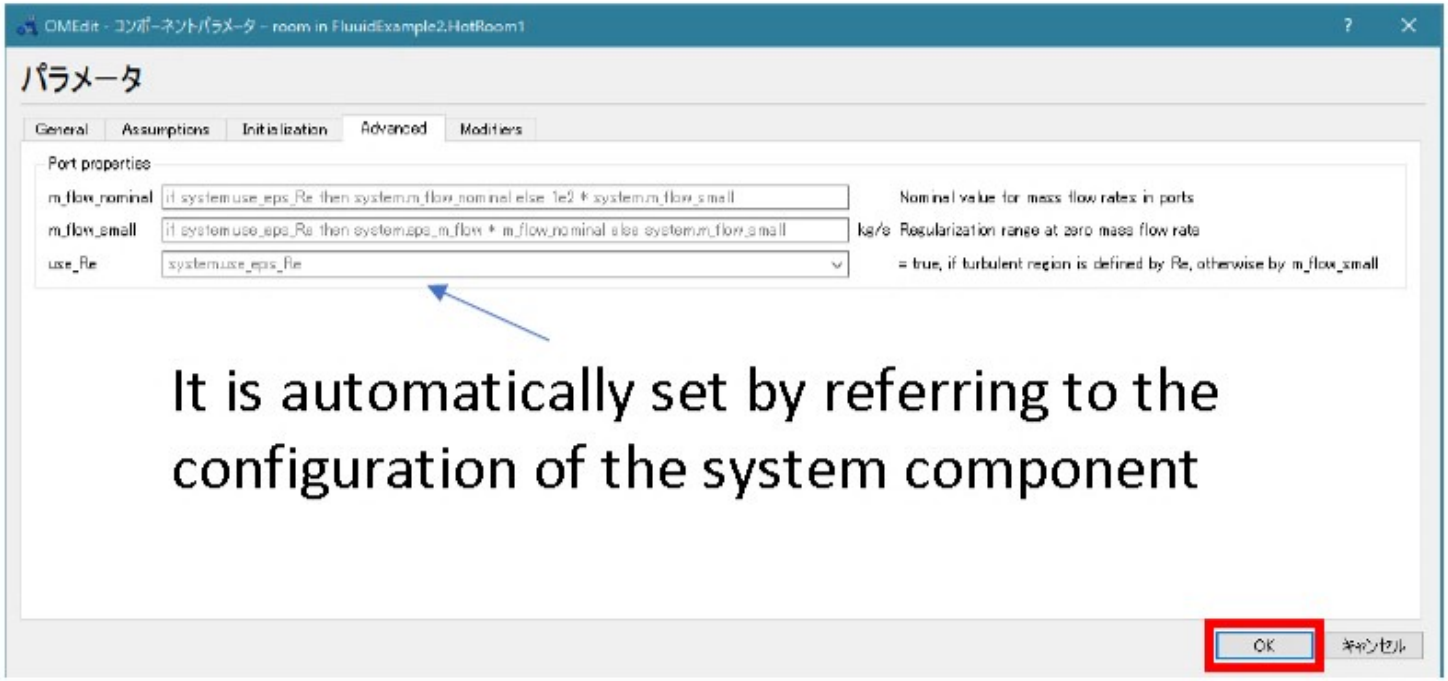


[Initialization]

- p_start = 101325 [Pa]
- use_T_start = true
- T_start = 280 [K]

Set energyDynamics and massDynamics to FixedInitial to specify the initial thermodynamic state of indoor air by pressure and temperature or enthalpy

HotRoom1



[Advanced]

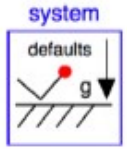
- Do not set

- If Re (Reynolds number) is small and goes out of the turbulence region, perform regularization (optimization of flow)
- The range where the flow rate is smaller is treated as zero flow

The parameters on this tab define the range for performing regularization and the range for handling as zero flow.

HotRoom1

System component settings (default values)



Dynamics

$allowFlowReversal = true$
$energyDynamics = DynamicFreeInitial$
$massDynamics = energyDynamics$
$substanceDynamics = massDynamics$
$traceDynamics = massDynamics$
$momentumDynamics = SteadyState$
$m\_flow\_start = 0,$
$p_{start} = p_{ambient},$
$T_{start} = T_{ambient}$

For many components, default values of parameters such as Dynamics and Regularization are set by referring to the system component.

General

$p_{ambient} = 101325$
$T_{ambient} = 293.15$
$g = Modelida.Constants.g\_n$

Regularization

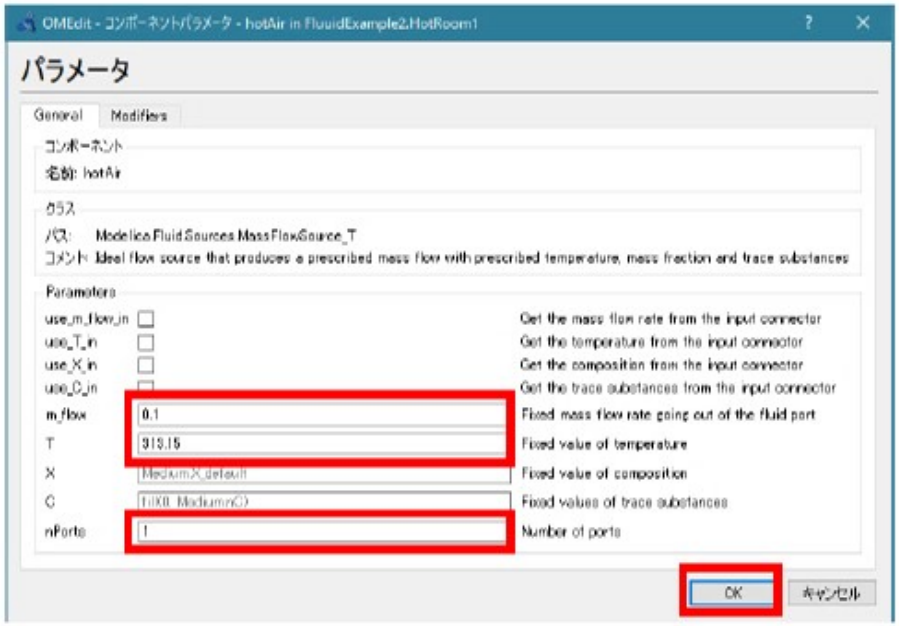
$use\_eps\_Re = false$

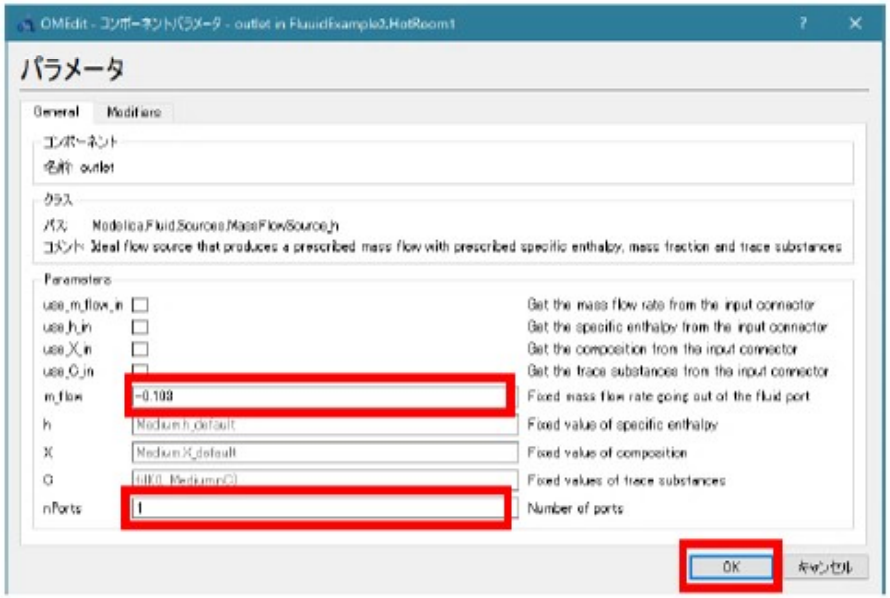
$$m\_flow\_nominal = \begin{cases} 1, & use\_eps\_Re = true \\ 10^2 \times m\_flow\_small, & use\_eps\_Re = false \end{cases}$$

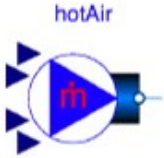
$dp\_small = 1$ [Pa]
$m\_flow\_small = 10^{-2}$ [kg/s]

HotRoom1

(3)(4) Fluid.Sources.MassFlowSource_T







Specify the flow rate and temperature

(3) hotAir

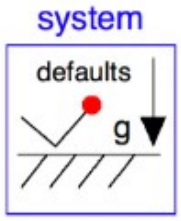
- m_flow = 0.1 [kg/s]
- T = 313.15 [K]
- nPorts = 1

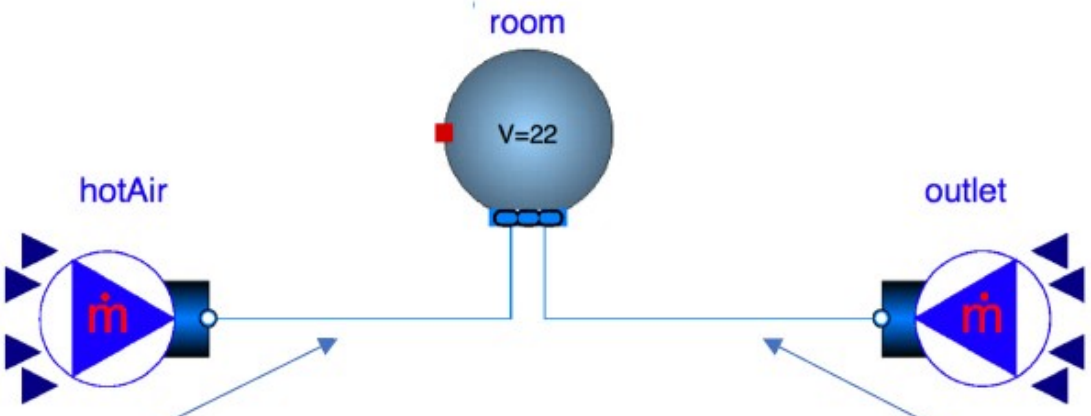
(4) outlet

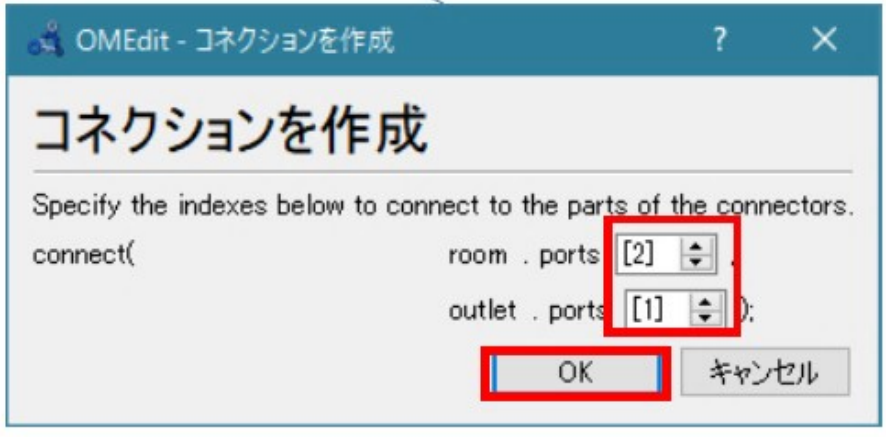
- m_flow = -0.103 [kg/s]
- nPorts = 1

HotRoom1

⑥ Connect components. FluidPort is arrayed
So pay attention to the port number.








```
connect(hotAir.ports[1], room.ports[1]) annotation( ...);
connect(room.ports[2], outlet.ports[1]) annotation( ...);
```

HotRoom1

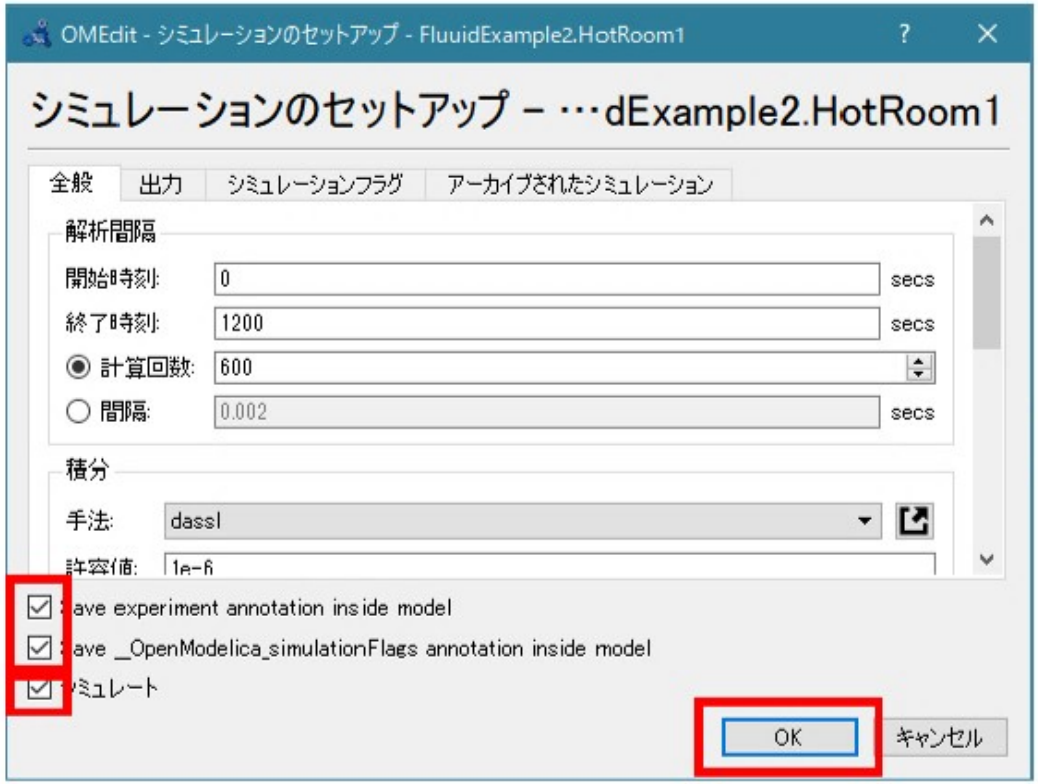
⑦ Switch to text view and edit the source code.

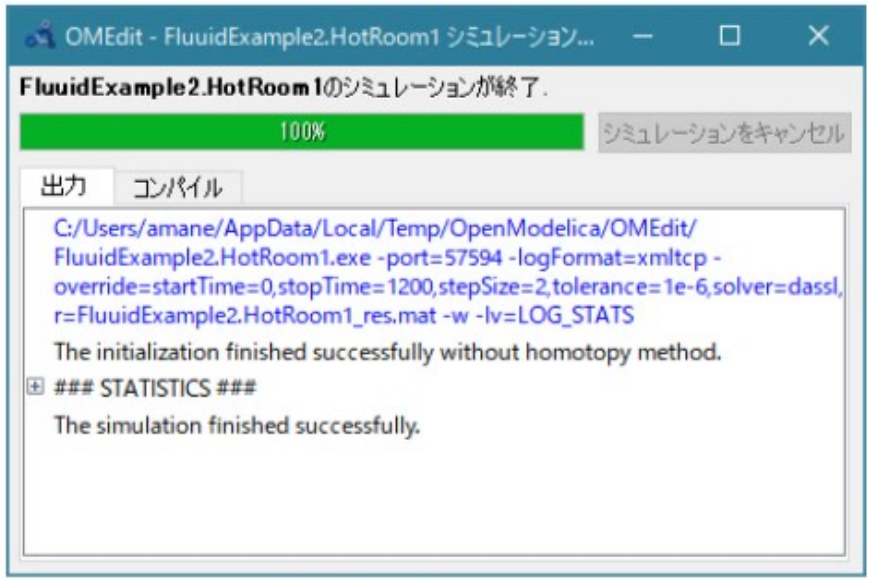
```
model HotRoom1
  replaceable package Medium = Media.Air.DryAirNasa;
  inner Modelica.Fluid.System system annotation( ...);                    (1)
  Modelica.Fluid.Vessels.ClosedVolume room(redeclare package Medium = Medium,  (2)
    T_start = 280, V = 22,
    energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    massDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    nPorts = 2, p_start = 101325, use_portsData = false) annotation( ...);
  Modelica.Fluid.Sources.MassFlowSource_T hotAir(redeclare package Medium = Medium,  (3)
    T = 313.15, m_flow = 0.1, nPorts = 1) annotation( ...);
  Modelica.Fluid.Sources.MassFlowSource_T outlet(redeclare package Medium = Medium,  (4)
    m_flow = -0.103, nPorts = 1) annotation( ...);
equation
  connect(hotAir.ports[1], room.ports[1]) annotation( ...);
  connect(room.ports[2], outlet.ports[1]) annotation( ...);
  annotation( ...);
end HotRoom1;
```

⑧Save and perform model check

HotRoom1

⑦ Set up the simulation and execute the simulation





Hopefully this screan will be displayed

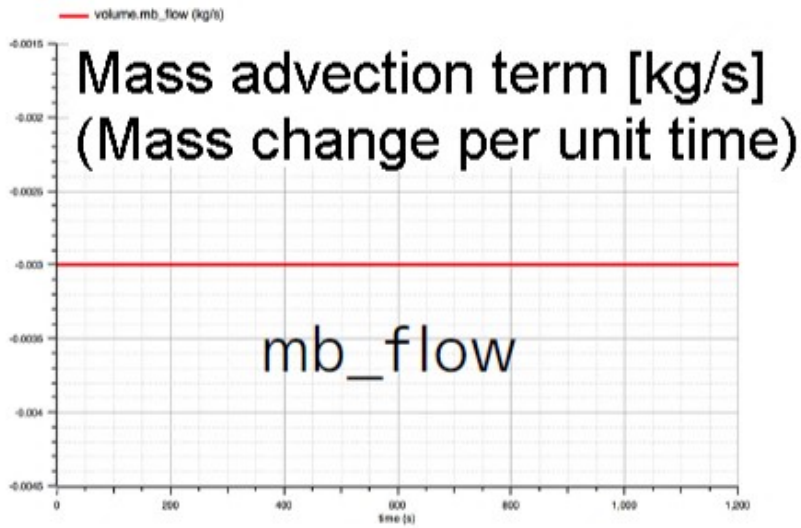
- Start time 0 [sec]
- End time 1200 [sec]
- Number of calculations 600

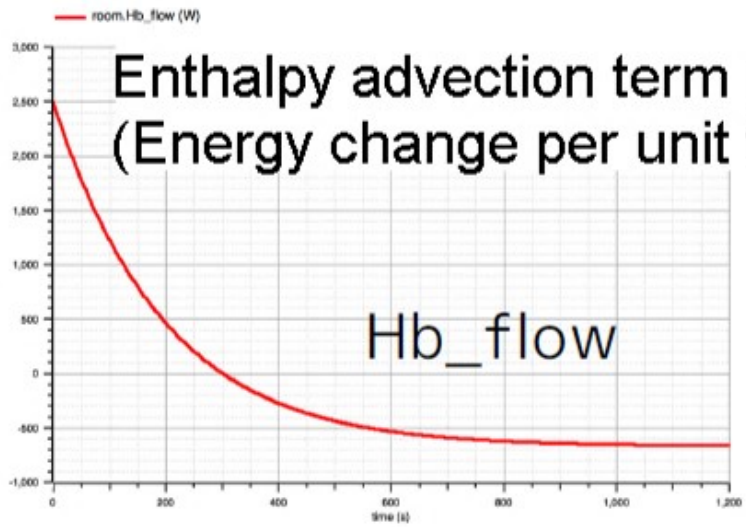
Check [Simulate] and click [OK] to execute the simulation

HotRoom1

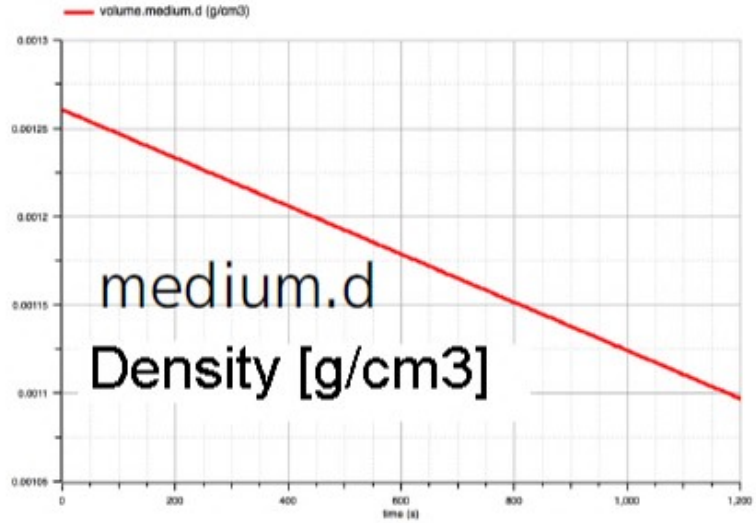
Simulation result

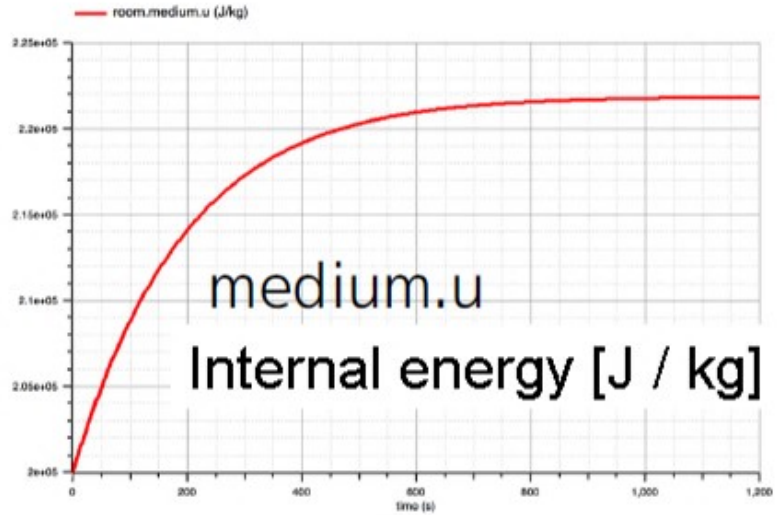
Mass and enthalpy (heat) flow in and out of the FluidPort





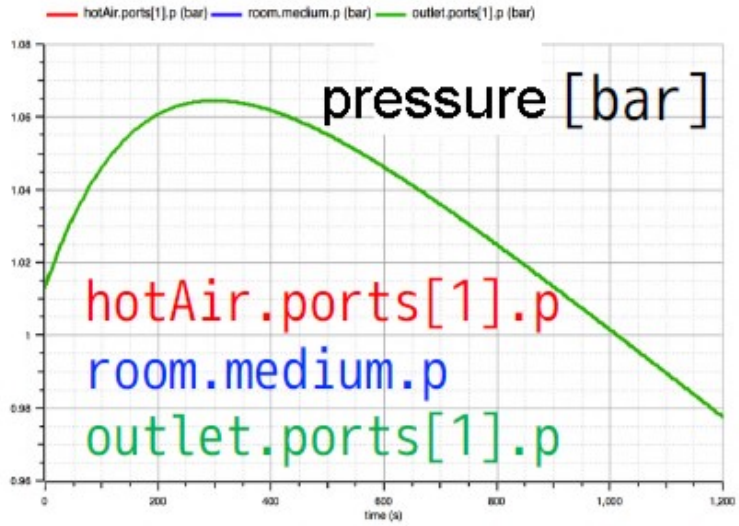
The indoor air density and internal energy change

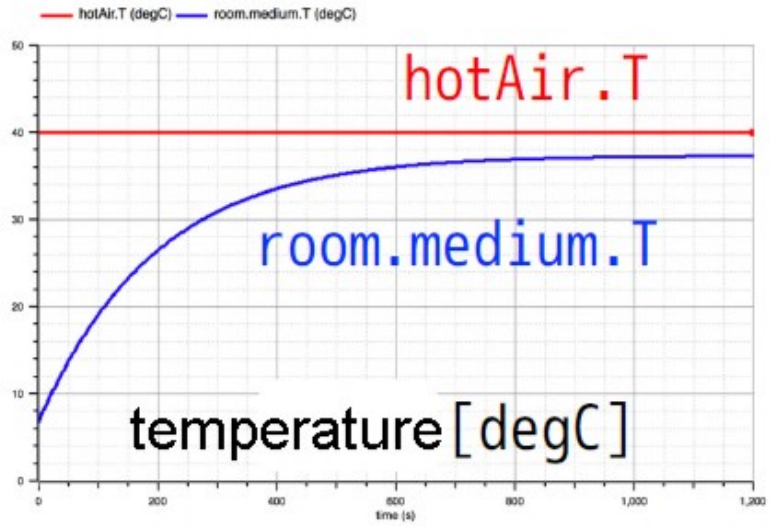




HotRoom1

The pressure, temperature, and specific enthalpy in the room change according to the thermodynamic relational expression.



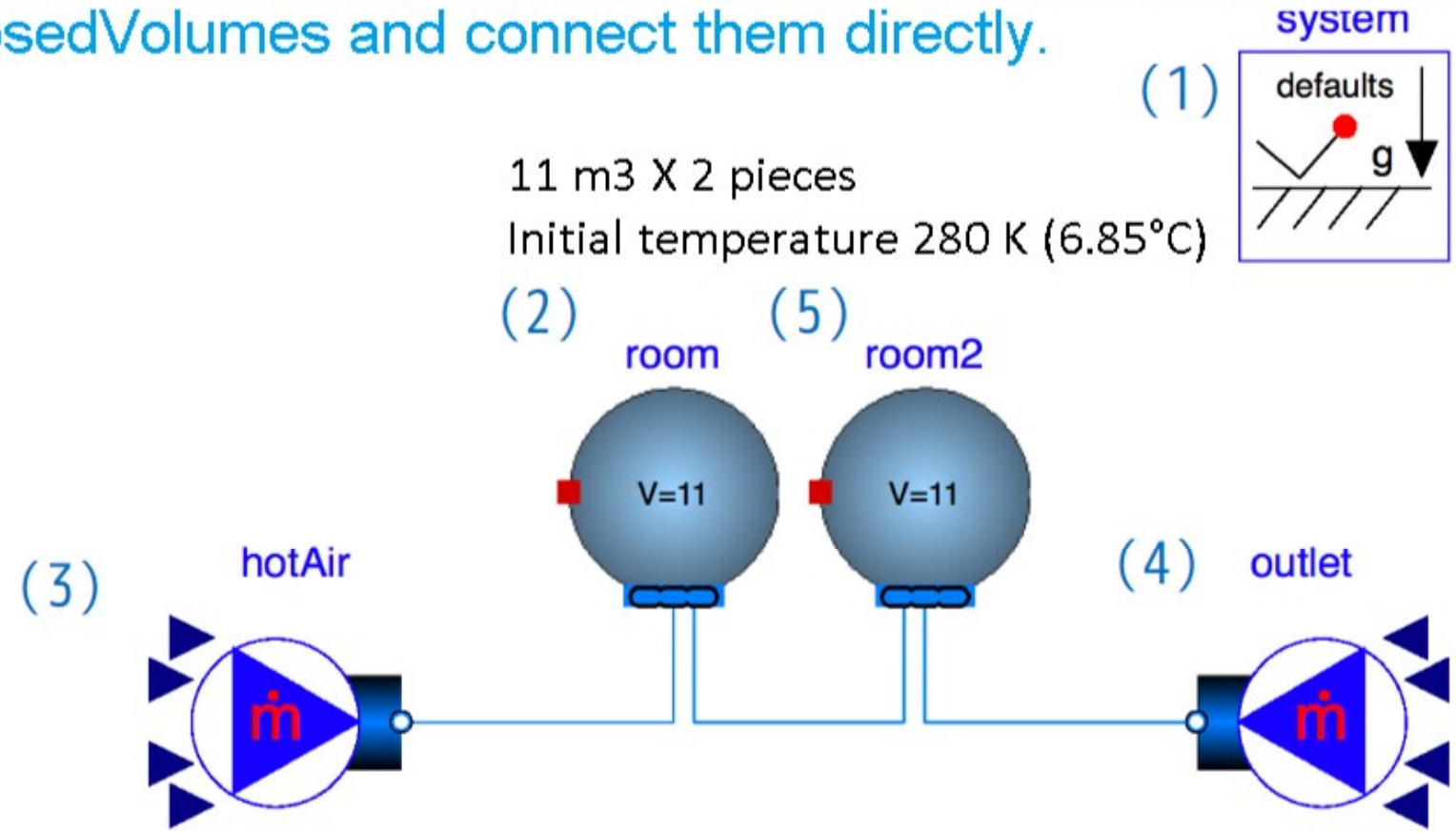


- The volume model is a differential equation model that calculates the time evolution of mass (or density) and internal energy in a vessel
- Since the relational expression of thermodynamic variables is given by the Baseproperties model, it can be considered as a differential algebraic equation model related to other combinations of thermodynamic variables, for example, pressure and temperature

HotRoom2

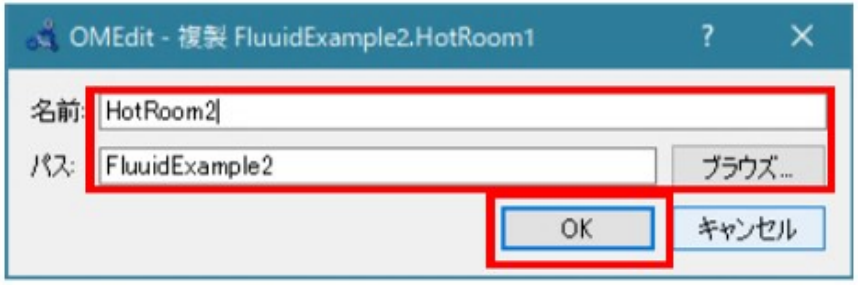
Split a room into two

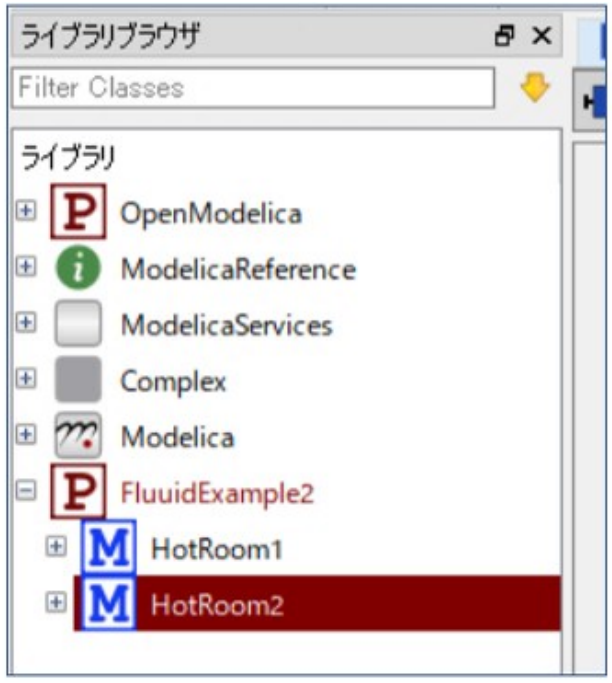
Divide the room represented by one ClosedVolume into two ClosedVolumes and connect them directly.



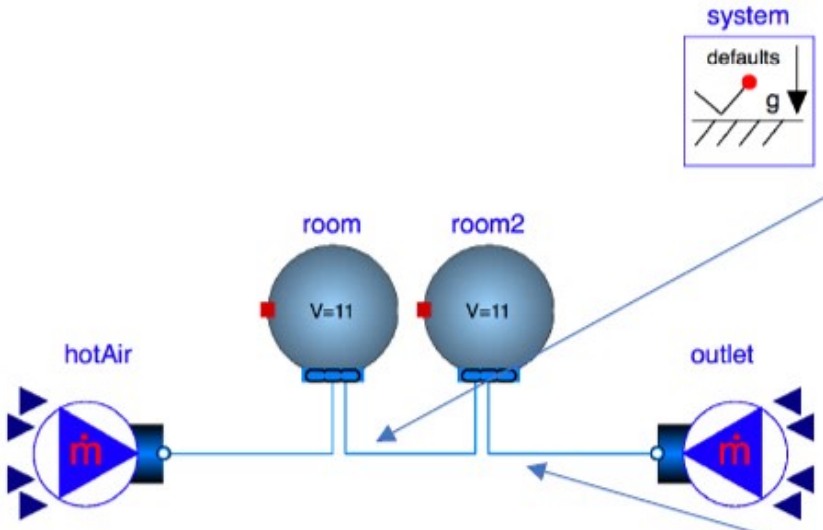
HotRoom2

② Right-click HotRoom1 in the library browser, select "Duplicate", and select the path. Create HotRoom2 in FluidExample2.

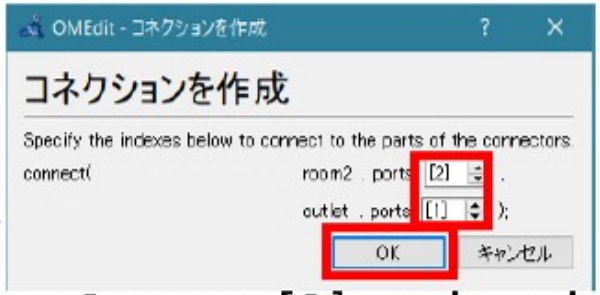




③ Switch to the diagram view, right-click the room and select Duplicate to create a copy room2. Connect as follows.



room.ports [2] and room2.ports [1]

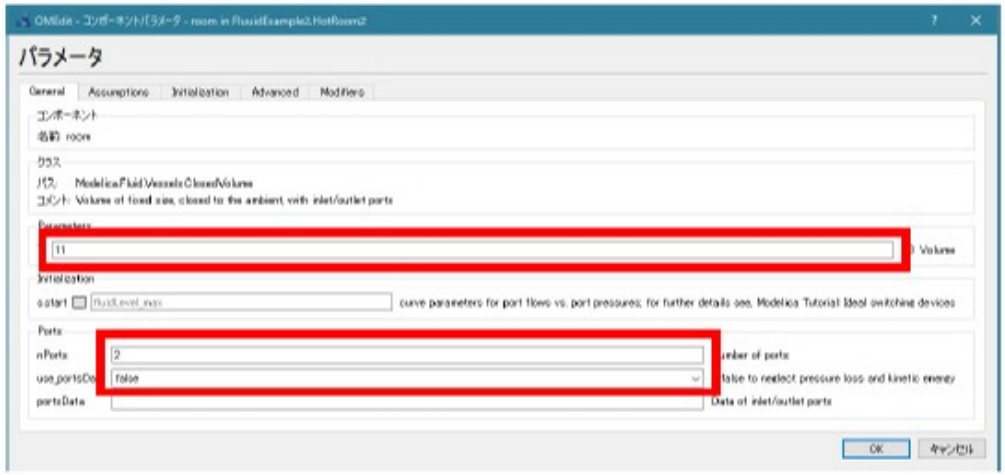


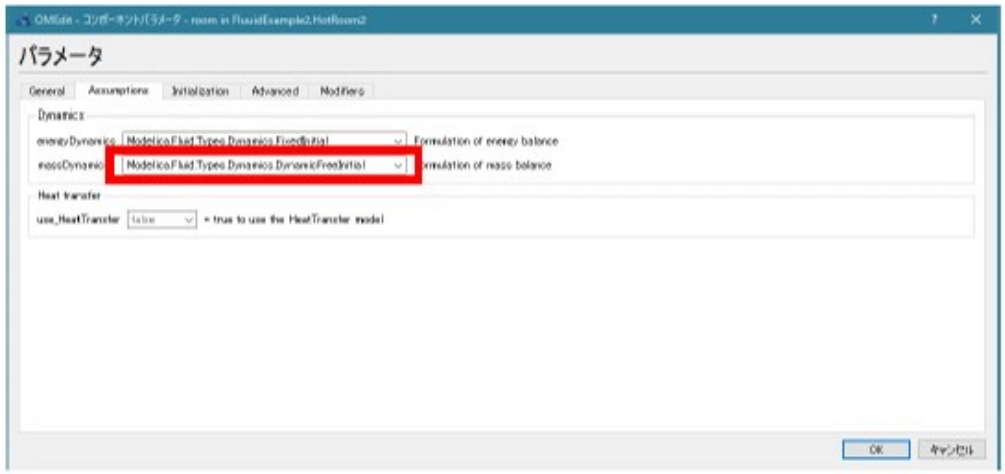
room2.ports [2] and outlet.ports [1]

HotRoom2

③ Reduce the volume of room and room2 by half. Set room's massDynamics to FreeInitial.

Fluid.Vessels.ClosedVolume





(2) room, (5) room2
[General]

- V = 11 [m3]
- nPorts = 2
- use_portsData = false

(2) room
[Assumption]
massDynamics
= DynamicFreeInitial

Due to the direct connection, the pressures in room and room2 are equal. If both massDynamics are set to FixedInitial and the initial value of the pressure is fixed, it will be excessively set, so one is set to FreeInitial.

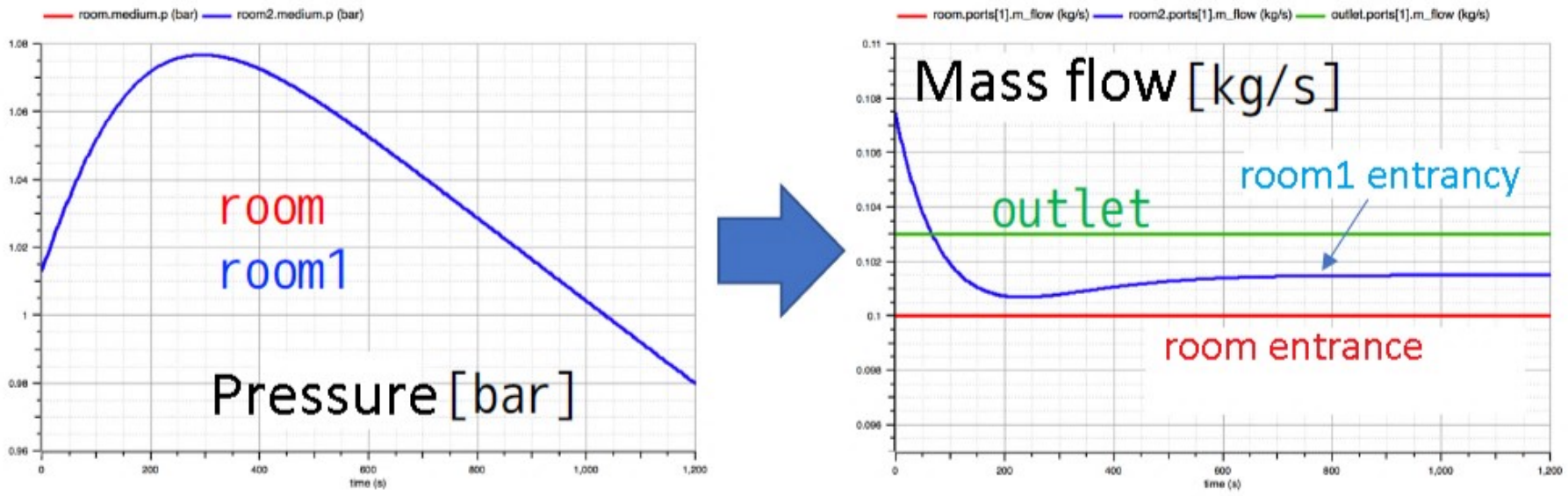
HotRoom2

④ Edit the room2 part of the source code in the text view.

```
Modelica.Fluid.Vessels.ClosedVolume room2(redeclare package Medium = Medium,
  T_start = 280, V = 11,
  energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
  massDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
  nPorts = 2, p_start = 101325, use_portsData = false) annotation( ...);
```

HotRoom2

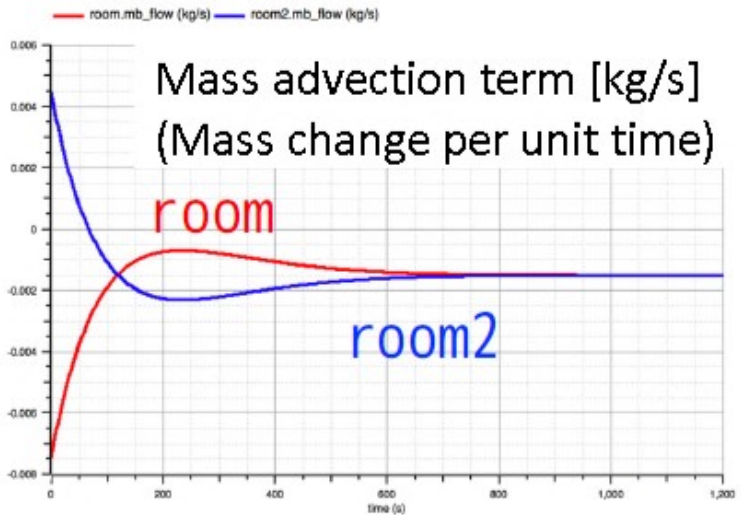
Simulation result

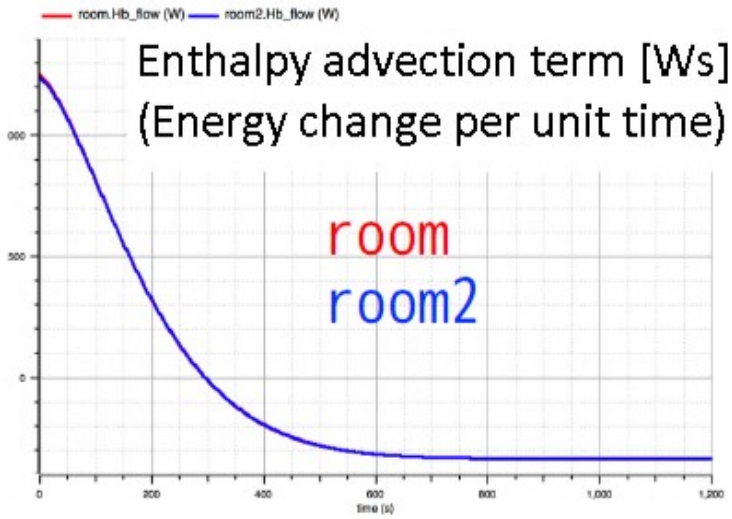


- When the volume models are directly connected, the pressures of the two volumes do not change independently, but match.
- The equations are high index differential algebraic equations with respect to pressure.
- The mass flow between volume models is determined under the constraint that the pressures match

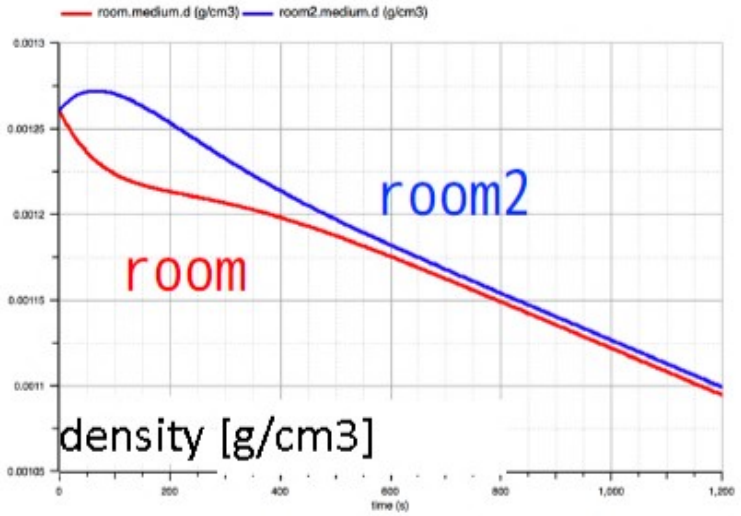
HotRoom2

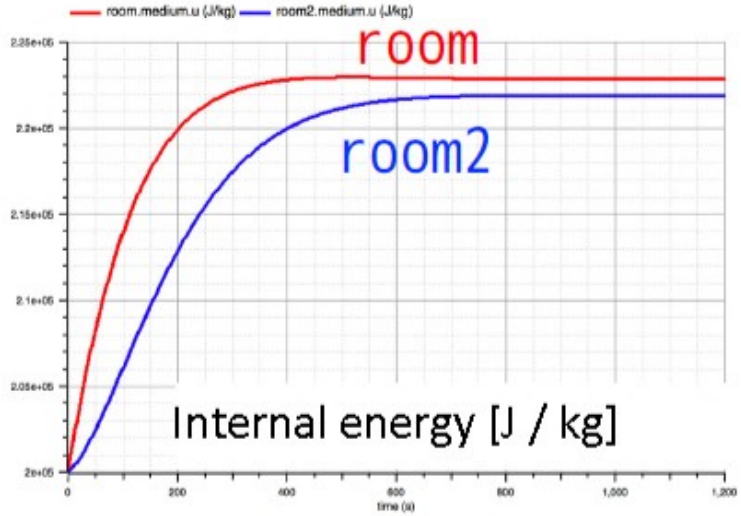
Mass and enthalpy (heat) flow in and out of the FluidPort.





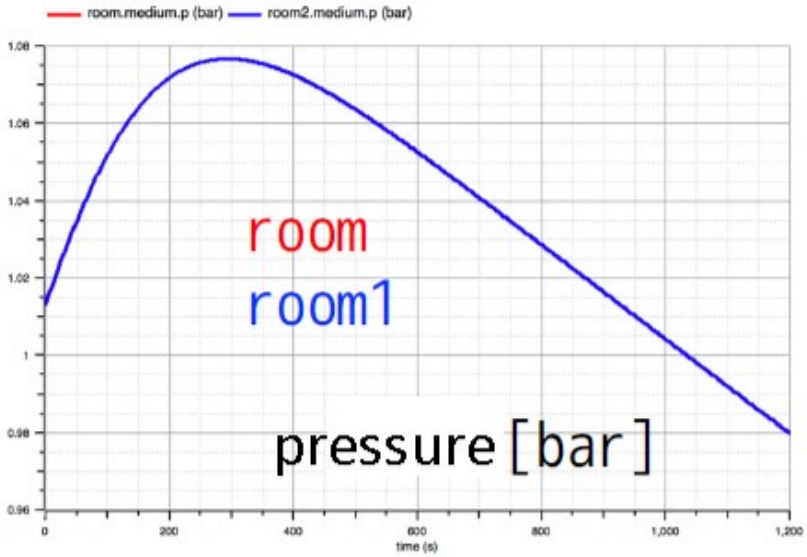
The density and internal energy in the container change

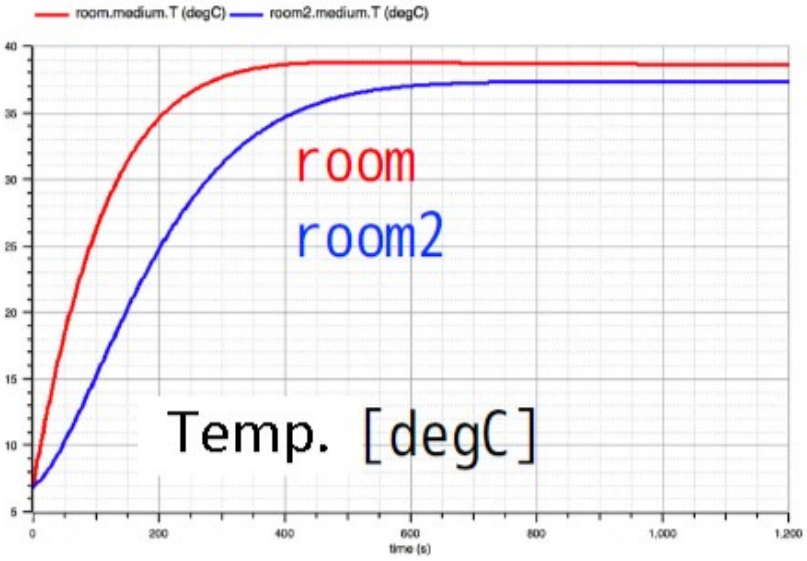




HotRoom2

Pressure and temperature also change according to thermodynamic relations

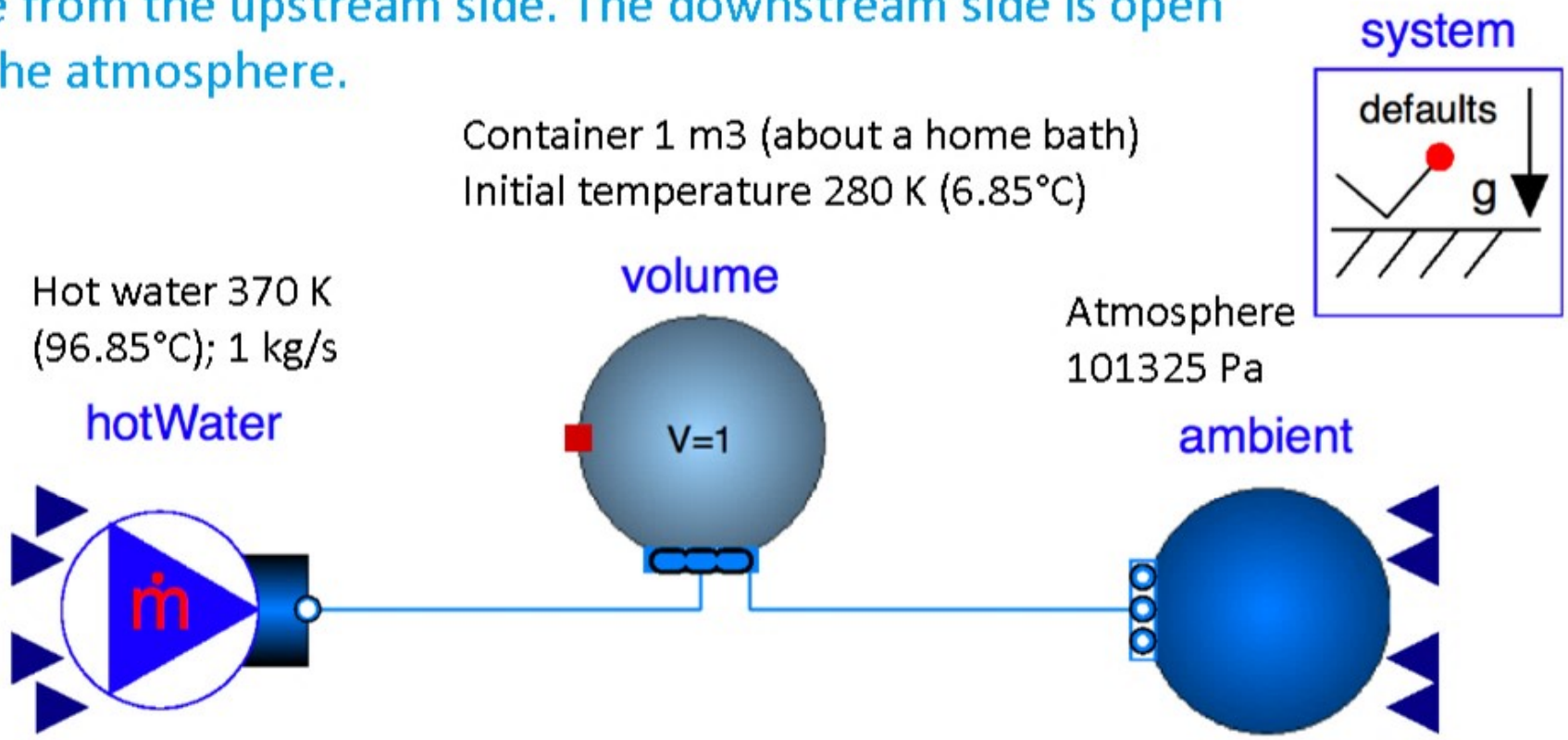




WaterVolume1

Replace cold water in container with hot water

Cold water is contained in a 1 m3 container. Pour hot water at a constant flow rate from the upstream side. The downstream side is open to the atmosphere.



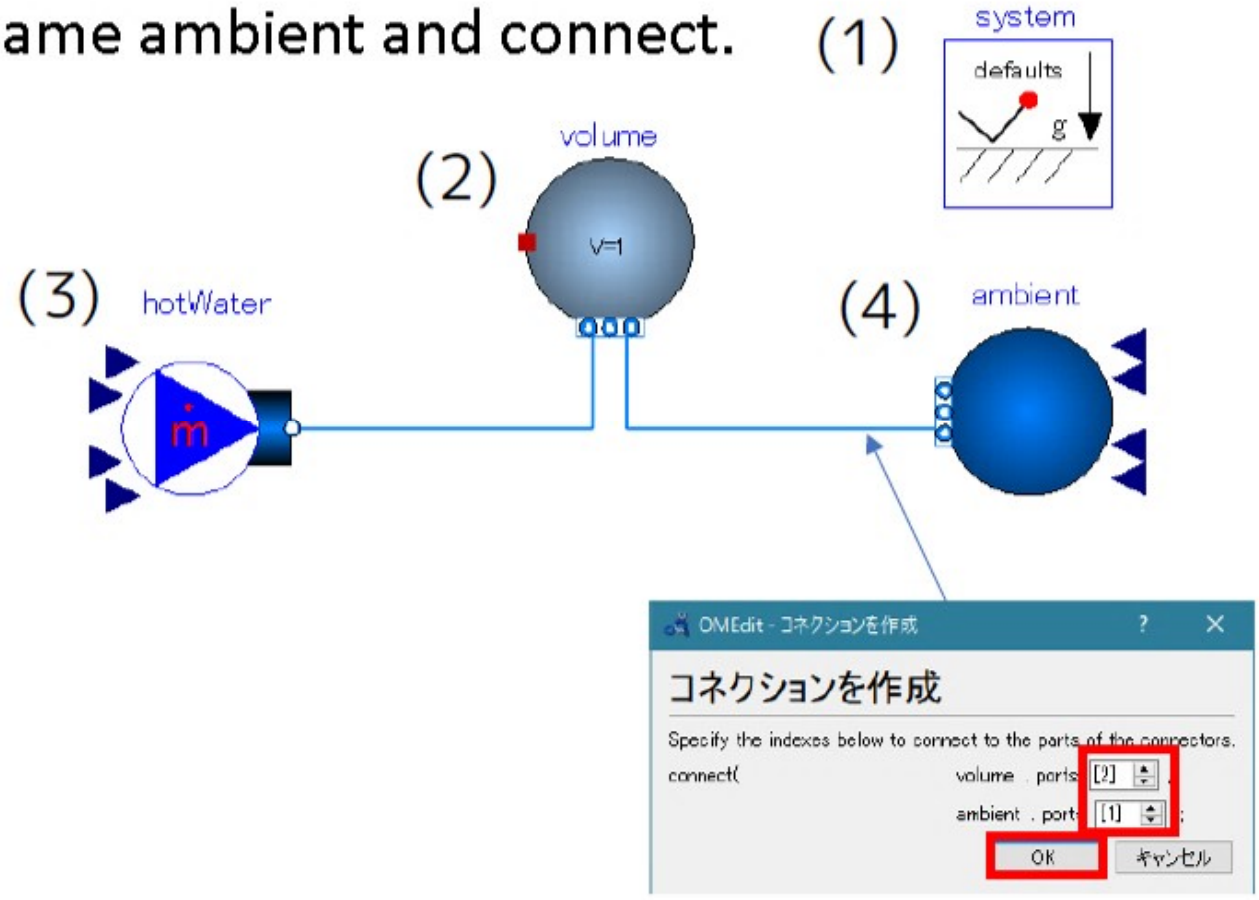
In this case, the pressure of the water in the container is determined by the atmospheric pressure. Incompressible fluid can be used.

WaterVolume1

① Right-click HotRoom1 and select Duplicate. Create a copy on the path FluidExample2 with the name WaterVolume1.

② Rename hotAir to hotWater and room to volume.

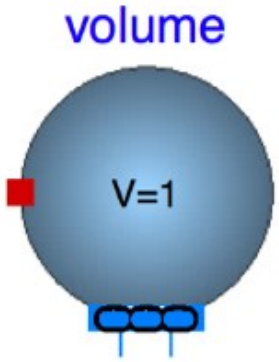
③ Delete the outlet, and instead place Fluid.Sources.Boundary_pT under the name ambient and connect.



WaterVolume1

④ Set the parameters

Fluid.Vessels.ClosedVolume



(2) volume

[General]

- V = 1 [m3]
- nPorts = 2
- use_portsData = false

[Assumptions]

- energyDynamics = FixedInitial
- MassDynamics = FreeInitial
- use_HeatTransfer = false

[Initialization]

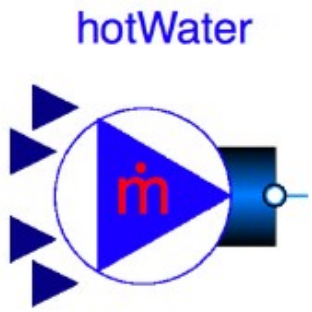
- use_T_start = true
- T_start = 280

WaterVolume1

Fluid.Sources.MassFlowSource_T

(3) hotWater

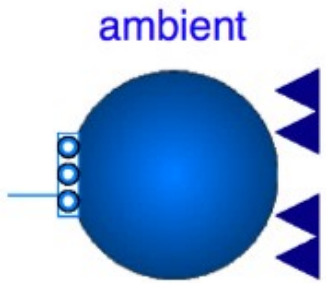
- m_flow = 1 [kg/s]
- T = 370 [K]
- nPorts = 1



Fluid.Sources.Boundary_pT

(4) ambient

- p = 101325 [Pa]
- nPorts = 1



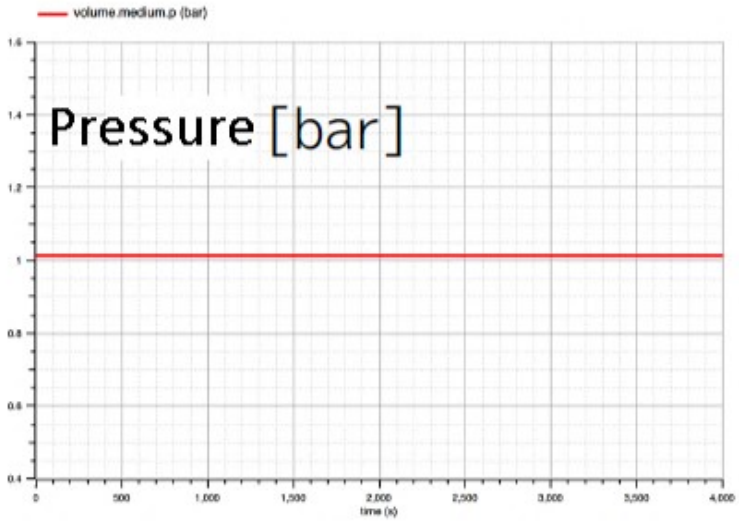
WaterVolume1

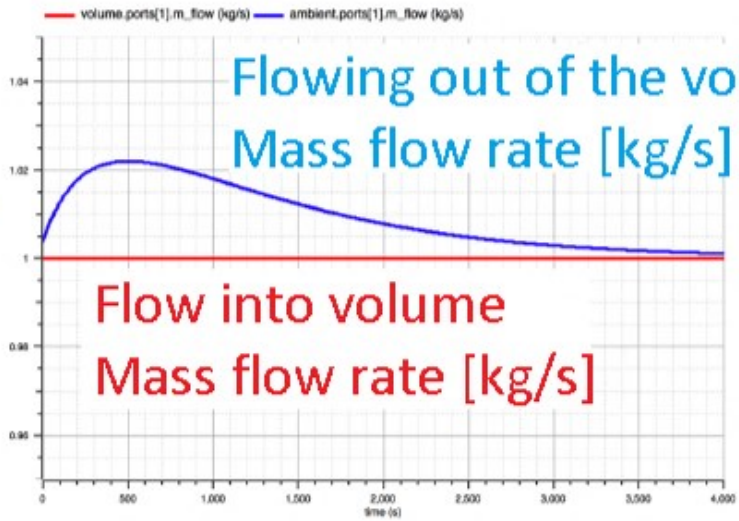
⑤ Edit the source code in text view

```
model WaterVolume1
  replaceable package Medium = Media.Water.StandardWater;
  //replaceable package Medium = Media.Water.ConstantPropertyLiquidWater;
  inner Modelica.Fluid.System system annotation( ...);
  Modelica.Fluid.Vessels.ClosedVolume volume(redeclare package Medium = Medium,
    T_start = 280, V = 1,
    energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    massDynamics = Modelica.Fluid.Types.Dynamics.DynamicFreeInitial,
    nPorts = 2, p_start = 101325, use_portsData = false) annotation( ...);
  Modelica.Fluid.Sources.MassFlowSource_T hotWater(redeclare package Medium = Medium,
    T = 370, m_flow = 1, nPorts = 1, use_T_in = false) annotation( ...);
  Modelica.Fluid.Sources.Boundary_pT ambient(redeclare package Medium = Medium,
    nPorts = 1, p = 101325) annotation( ...);
equation
  connect(hotWater.ports[1], volume.ports[1]) annotation( ...);
  connect(volume.ports[2], ambient.ports[1]) annotation( ...);
  annotation( ...);
end WaterVolume1;
```

WaterVolume1

Simulation result



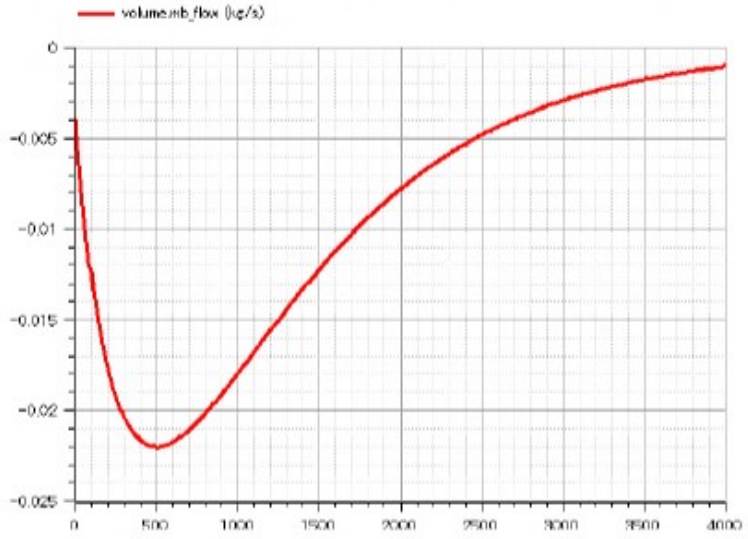


The mass flow out of the volume is determined under the constraint that the pressure of the volume matches the pressure of the ambient.

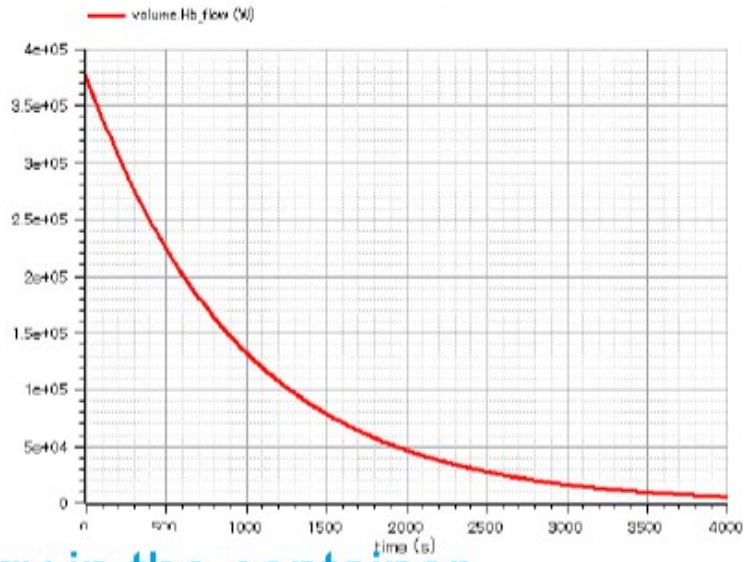
WaterVolume1

Mass and enthalpy (heat) flow in and out of the FluidPort

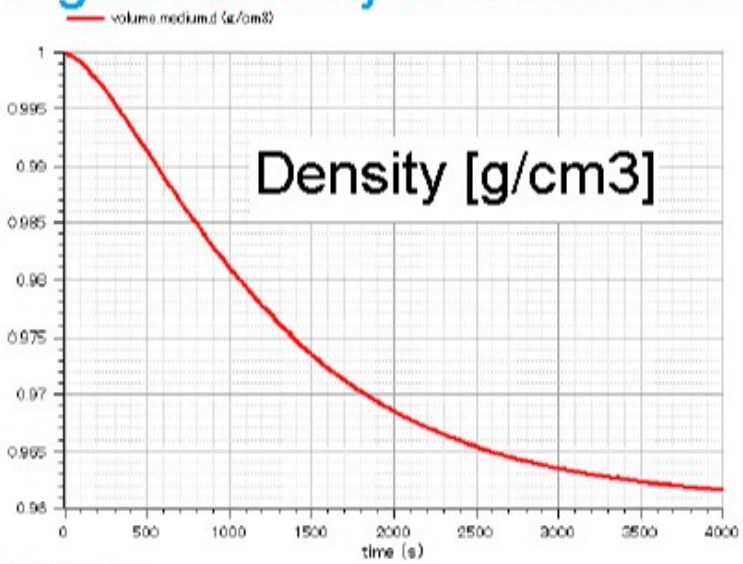
Mass advection term [kg/s]
(Mass change per unit time)

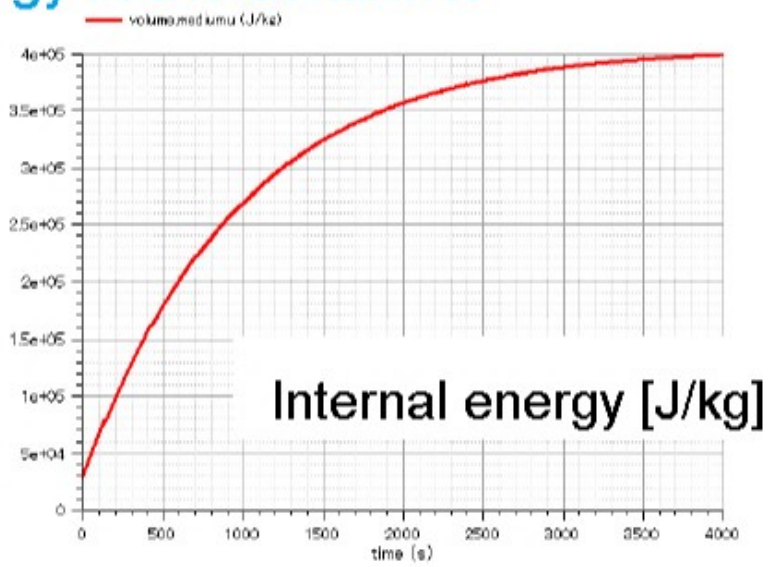


Enthalpy advection term [Ws]
(Energy change per unit time)



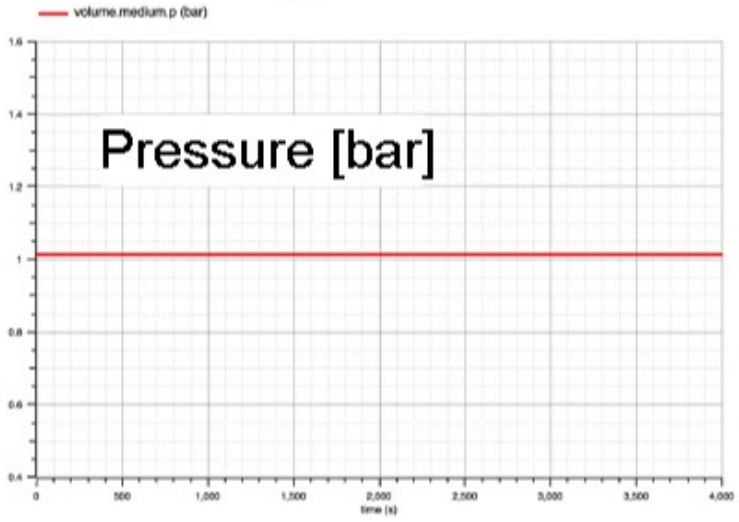
Change of density and internal energy in the container

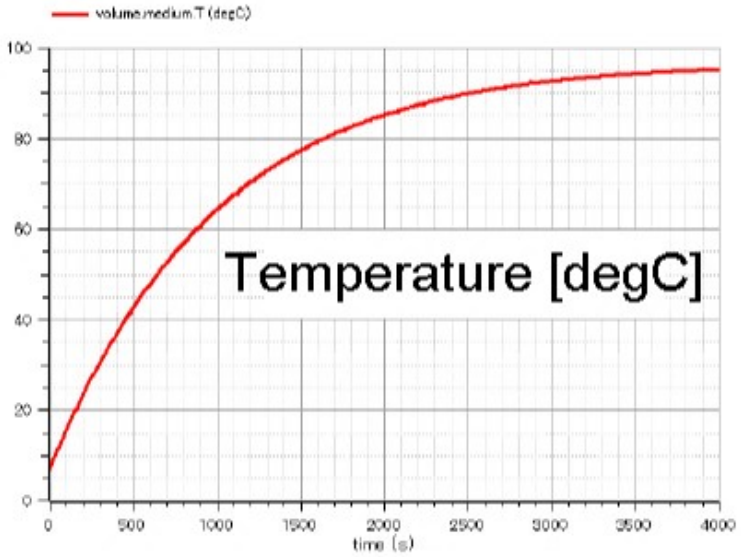




WaterVolume1

According to the thermodynamic law, the pressure and temperature in the room also change.

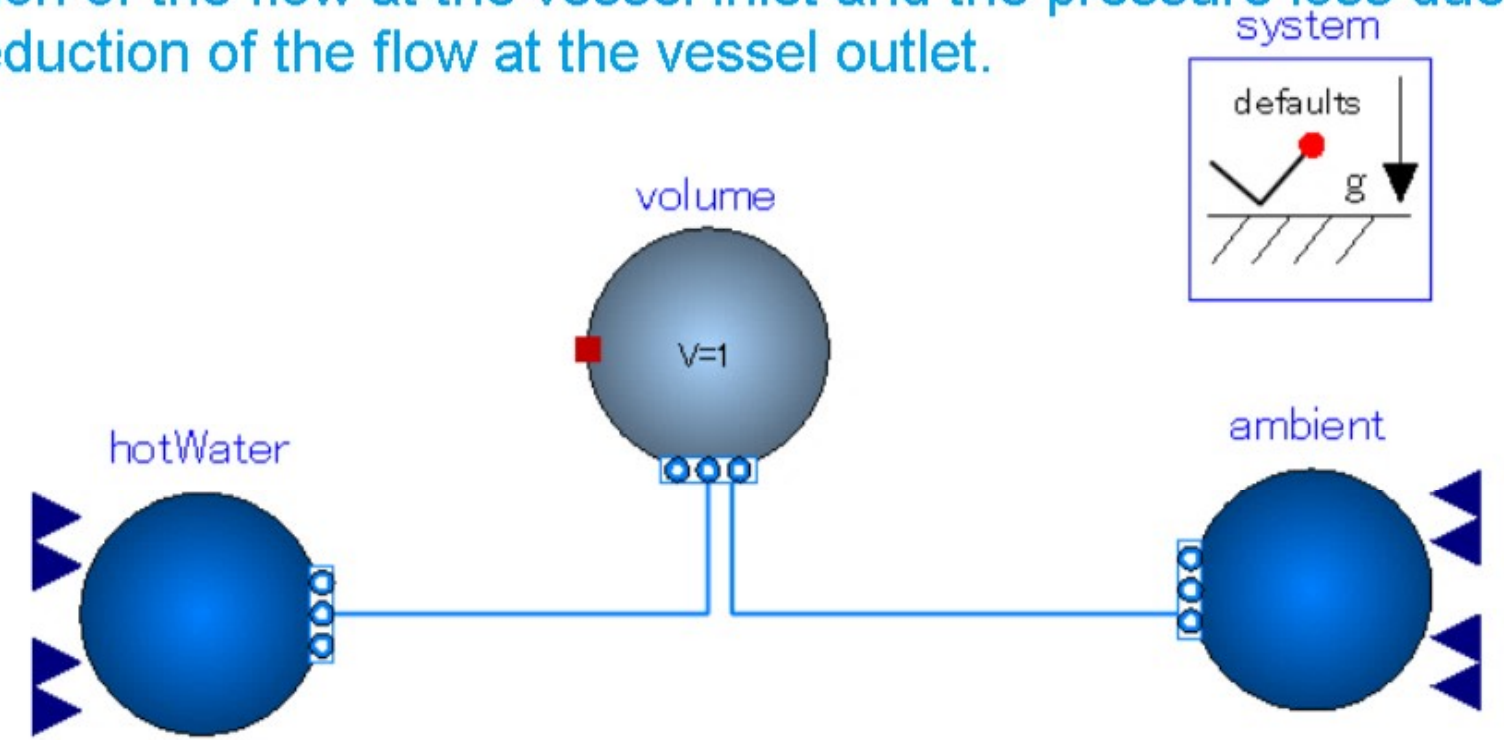




WaterVolume2

Consider the pressure loss at the container entrance and exit

A model is defined that defines the pressures upstream and downstream of the vessel and takes into account the pressure loss due to the rapid expansion of the flow at the vessel inlet and the pressure loss due to the rapid reduction of the flow at the vessel outlet.



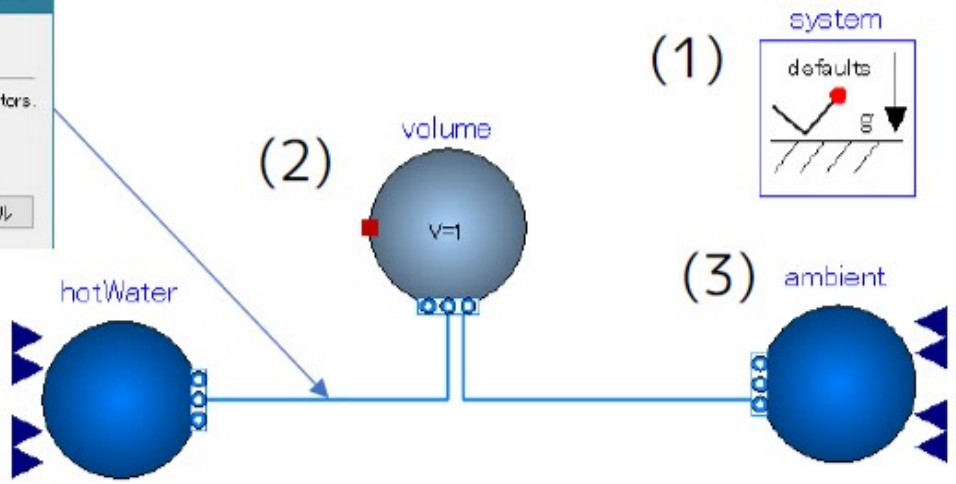
WaterValume2

① Right-click on WaterVolume1 and select Duplicate, and create a copy on FluidExample2 with the name WaterVolume2.

② Switch to the text view and add the following import statement to the beginning of WaterVolume2.

```
model WaterVolume2
  import Modelica.Fluid.Vessels.BaseClasses.VesselPortsData;
  replaceable package Medium = Media.Water.StandardWater;
```

③ Switch to the diagram, delete hotWater, and place Fluid.Sources.Boundary_pT as hotWater instead and connect.



Fluid.Sources.Boundary_pT

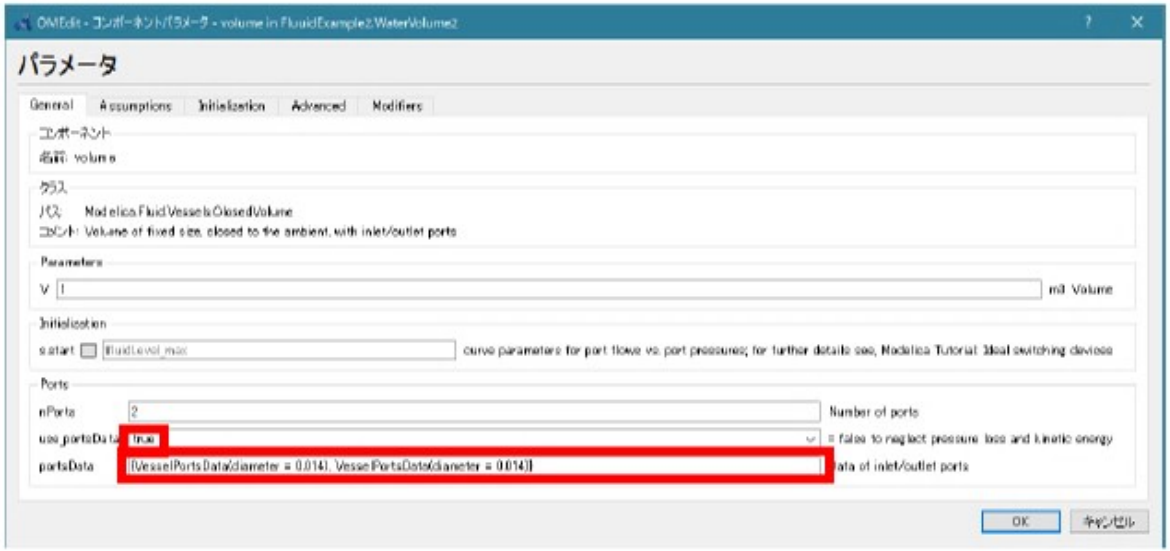
(4) hotWater

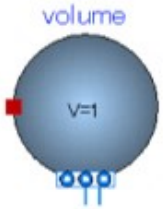
- p = 131325 [Pa]
- T = 370 [K]
- nPorts = 1

WaterVolume2

④ Set the component parameters

(2) Fluid.Vessels.ClosedVolume





Use portsData to calculate the pressure loss and set the inlet diameter.

[General]

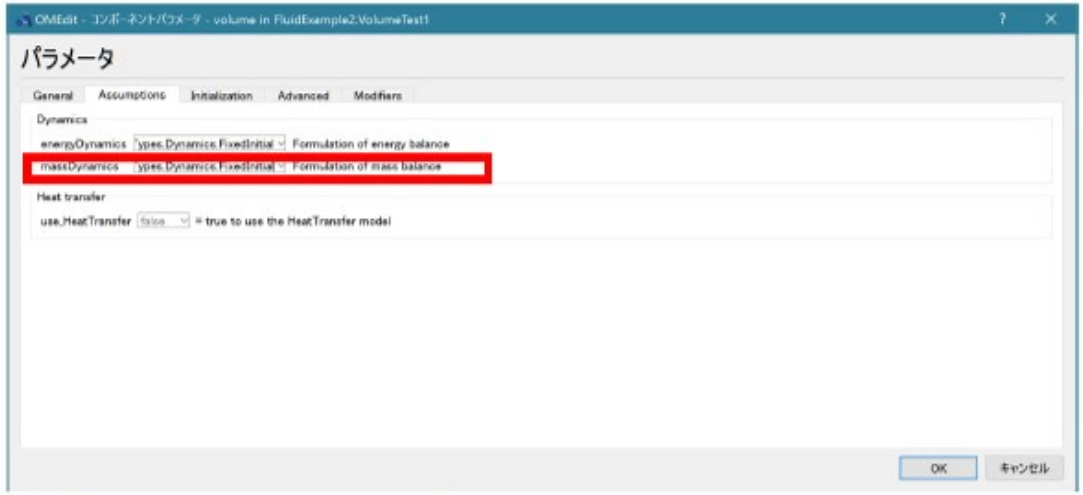
- V=1
- nPorts = 2
- **use_portsData = true**
- **portsData**
  = {VesselPortsData(diameter = 0.014), VesselPortsData(diameter = 0.014)}

1.4 cm diameter inlet
(Approximately a water supply pipe)

Do not enter "="

The portsData text box behaves strangely and its settings may be invalid, so it is convenient to copy and paste it into Notepad to keep the edits.

WaterVolume2



[Assumptions]

- energyDynamics = Dynamics.FixedInitial
- massDynamics = **Dynamics.SteadyStateInitial**
- use_HeatTransfer = false

WaterVolume2

⑤ Switch to text view and edit the source code.

volume

```
Modelica.Fluid.Vessels.ClosedVolume volume(redeclare package Medium = Medium,
   T_start = 280, V = 1, fluidLevel = 1.0,
   energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
   massDynamics = Mod elica.Fluid.Types.Dynamics.DynamicFreeInitial, nPorts = 2,
   portsData = {VesselPortsData(diameter = 0.014), VesselPortsData(diameter = 0.014)},
   use_portsData = true) annotation( ...);
```

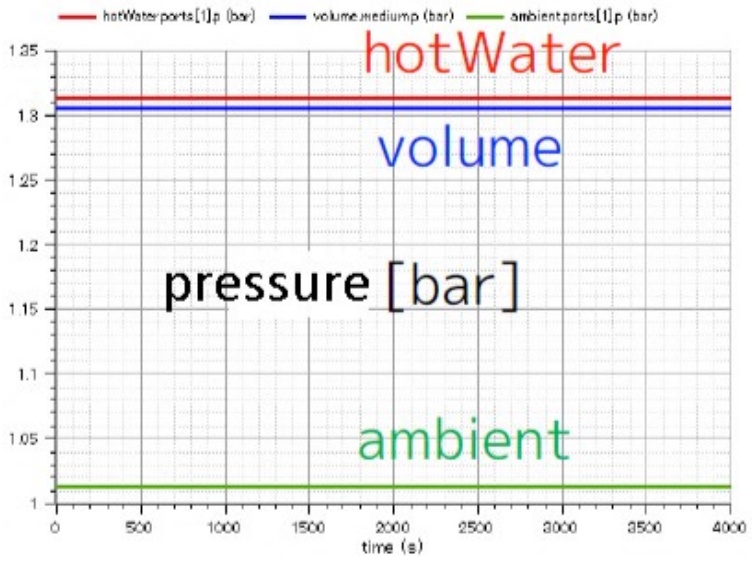
PartialLumpedVessel has fluidLevel> **fluidLevel> 0.2 * diamter**
If you do not set a value such that ports_penetration [i] <1, it seems that the pressure loss coefficient is not evaluated correctly.

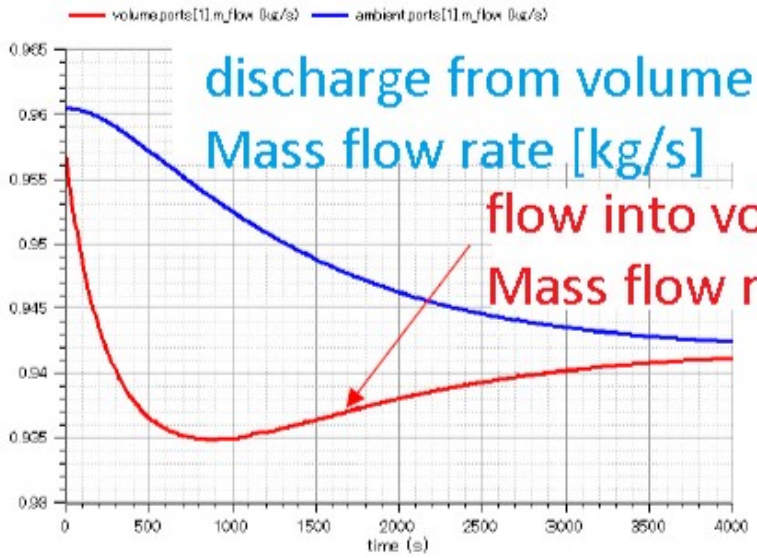
hotWater

```
Modelica.Fluid.Sources.Boundary_pT hotWater(redeclare package Medium = Medium,
   T = 370, nPorts = 1, p = 131325) annotation( ...);
```

⑥ Save, model check, etc., and execute simulation.

WaterVolume2



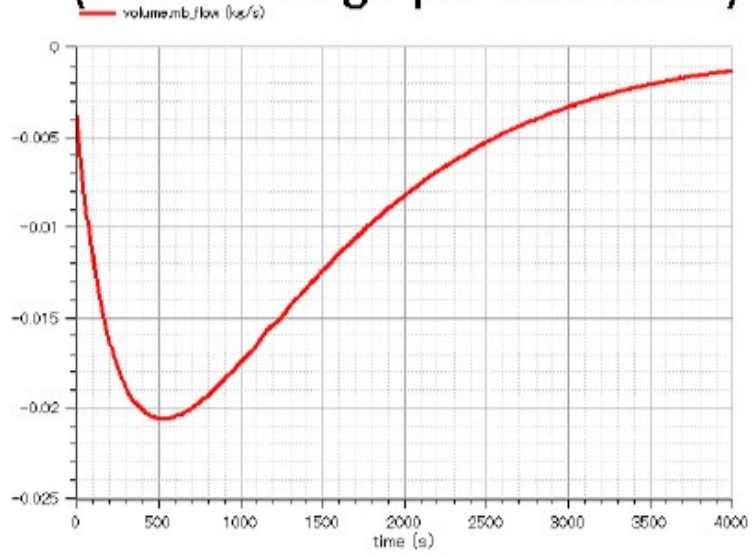


- The pressure drop at the inlet and outlet of the vessel is expressed as a nonlinear equation for flow rate. (See Appendix 2)
- The vessel pressure and mass flow can be obtained by simultaneous use of this nonlinear equation.

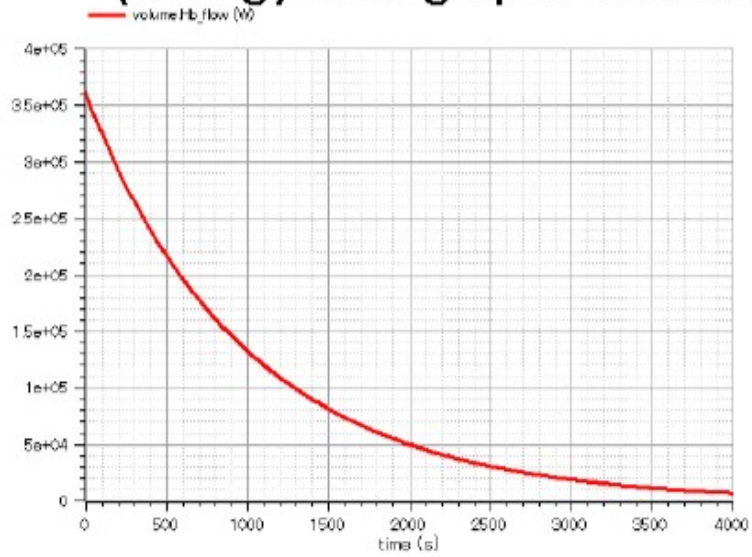
WaterVolume2

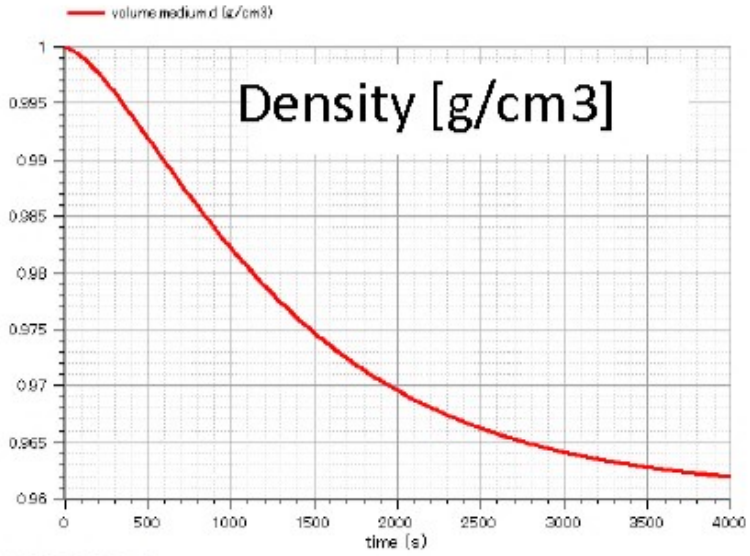
Mass and enthalpy (heat) flow in and out of the FluidPort.

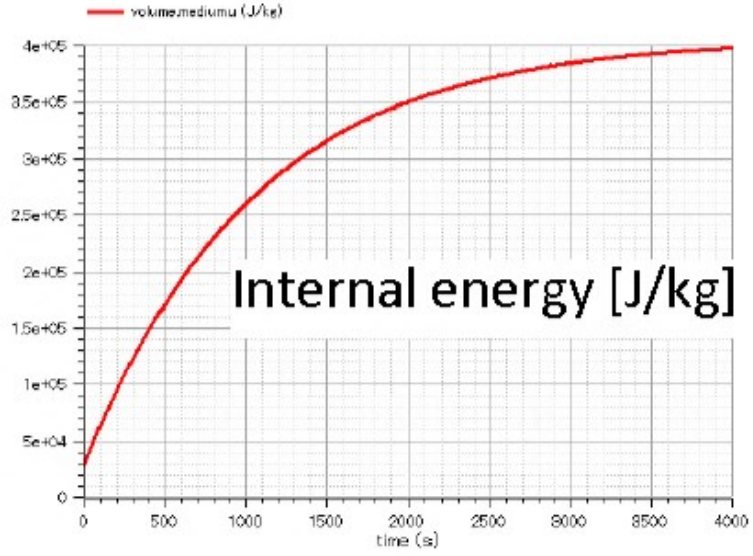
Mass advection term [kg/s]
(Mass change per unit time)



Enthalpy advection term [Ws]
(Energy change per unit time)

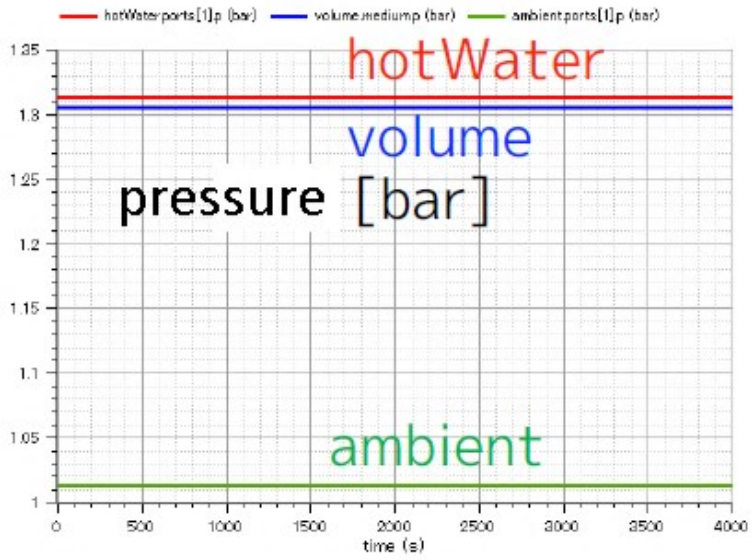


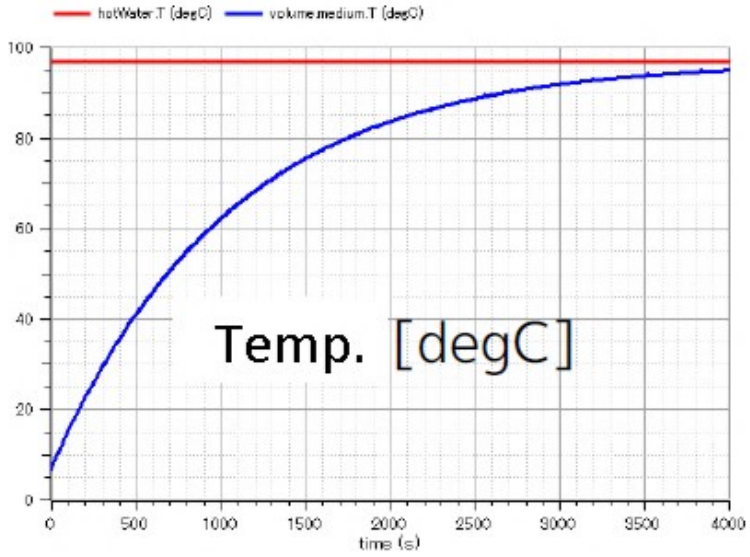




WaterVolume2

Pressure and temperature also change

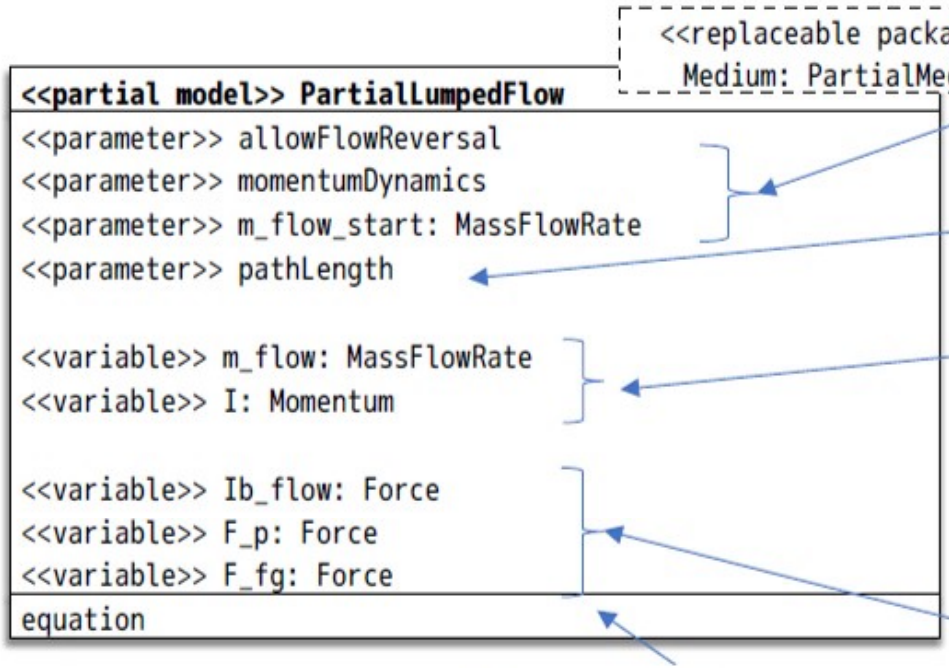




fluidExample3

Model that implements the flow model momentum equation

PartialLumpedFlow Base model of one flow model



Parameters to set steady / unsteady, initial conditions

The flow path length

State variables m_{fow}, I

Variable that defines the equation in the inherited model

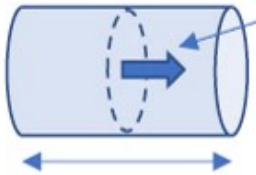
Ib_{flow} [N]: Momentum advection term

F_p [N]: Force due to pressure

F_{fg} [N]: Friction and gravity

Momentum conservation type

$I = m_{flow} \cdot pathLength$ [kg m/s]: Momentum



m_{flow} [kg/s]
: Mass flow

$pathLength$ [m]: length

Flow model

PartialLumpedFlow equation

```
equation
  // Total quantities
  I = m_flow*pathLength;

  // Momentum balances
  if momentumDynamics == Types.Dynamics.SteadyState then
    0 = Ib_flow - F_p - F_fg;
  else
    der(I) = Ib_flow - F_p - F_fg;
  end if;

initial equation
  if momentumDynamics == Types.Dynamics.FixedInitial then
    m_flow = m_flow_start;
  elseif momentumDynamics == Types.Dynamics.SteadyStateInitial then
    der(m_flow) = 0;
  end if;

  annotation ( ...);
end PartialLumpedFlow;
```

Relationship between momentum and mass flow

$$I = m_{flow} \cdot pathLength$$

Momentum balance

$$0 = Ib_{flow} - F_p - F_{fg} \quad \text{Steady}$$

$$\frac{dI}{dt} = Ib_{flow} - F_p - F_{fg} \quad \text{Unsteady}$$

Initial condition

m_{flow}=m_fow_start or $\frac{dm_{flow}}{dt} = 0$

Flow model

Base model of multiple flow models
PartialLumpedFlow Implementing momentum conservation formula with multiple elements

Inheritance relationship of flow model

Medium: PartialMedium

Introduction of physical property model

<<partial model>>
PartialDistributedFlow

One-dimensional staggered mesh shape definition, advection term, pressure term definition

Laminar flow model

<<partial model>>
PartialStaggeredFlowModel

<<model>>
NominalLaminarPipeFlow

Introduction of wall friction model

<<redeclare>> Quadratic turbulence model
WallFriction→LaminarAndQuadraticTurbulent

WallFriction: Detailed

<<partial model>>
PartialGenericPipeFlow

<<model>>
NominalTurbulentPipeFlow

A model that connects a laminar flow model and a quadratic turbulence model

WallFriction→LaminarAndQuadraticTurbulent

<<model>>
TurbulentPipeFlow

Definition of viscous term and external force term (gravity)

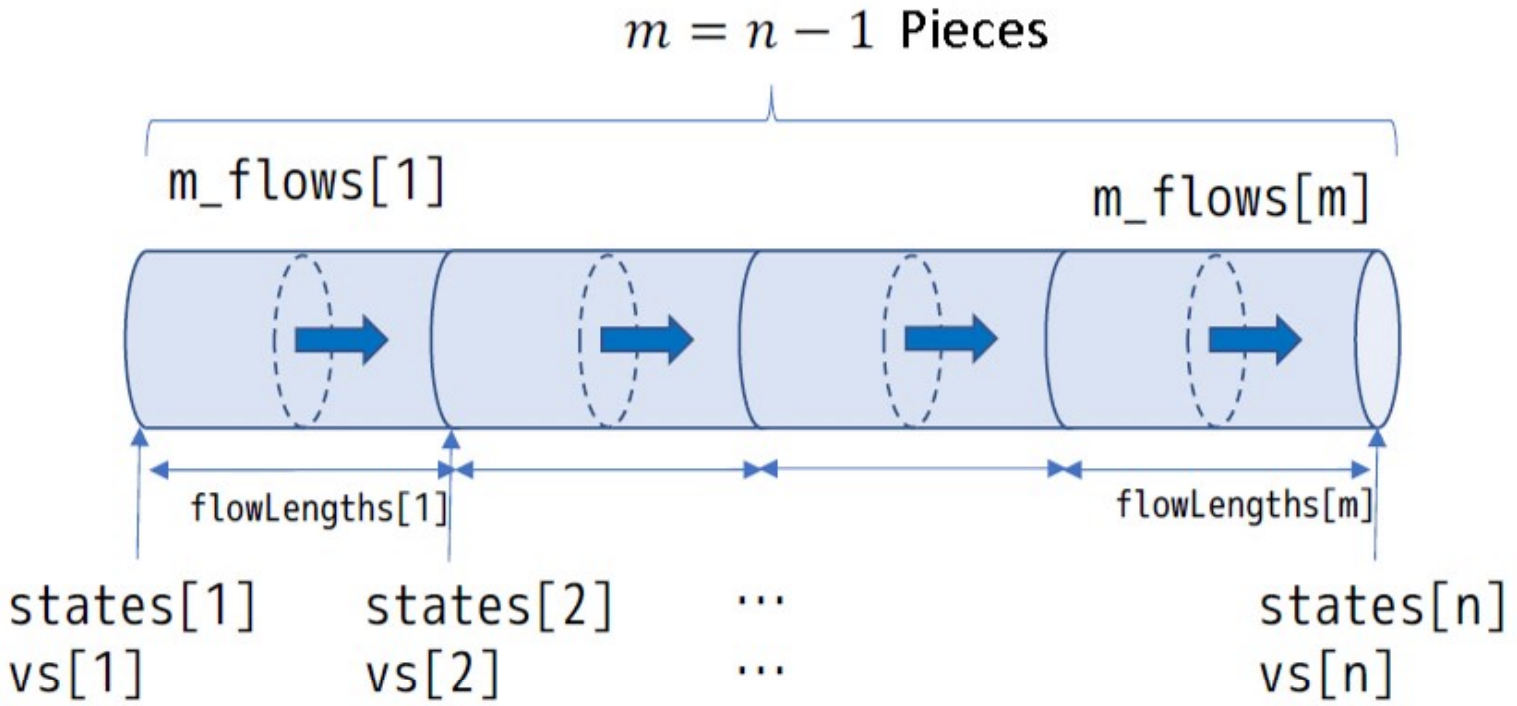
WallFriction→Detailed Detailed turbulence model

<<model>>
DetailedPipeFlow

Non-linear equations describing the relationship between pressure loss due to wall friction and gravity and mass flow

Flow model

PartialStageredFlowModel



Boundary condition

`states[i]`: Thermodynamic state variable (ThermodynamicState record)

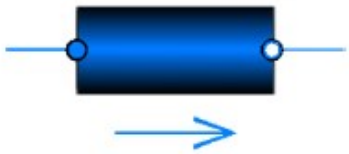
`vs[i]`: Flow velocity

State variables

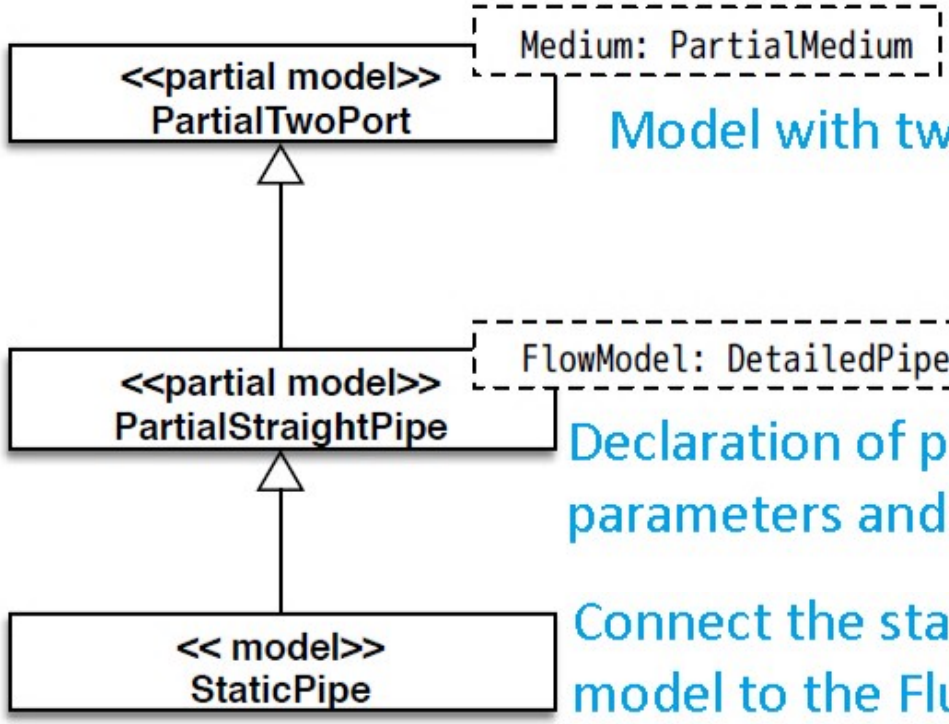
ls[i] = m_flows[i]*flowLengths[i]

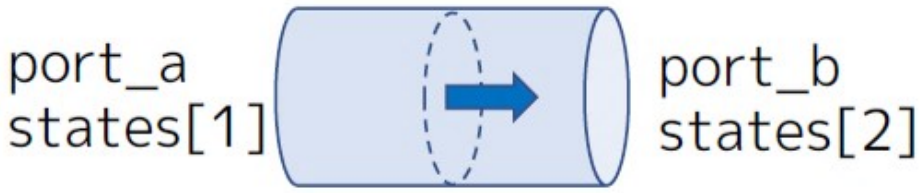
Flow model

StaticPipe



- massDynamics is fixed to steadyState to represent the steady state.
- Does not have a function to store fluid inside.





A model for finding a steady flow between two thermodynamic states [1] and states [2]

StaticPipeTest1

Place flow model and volume model

volume model

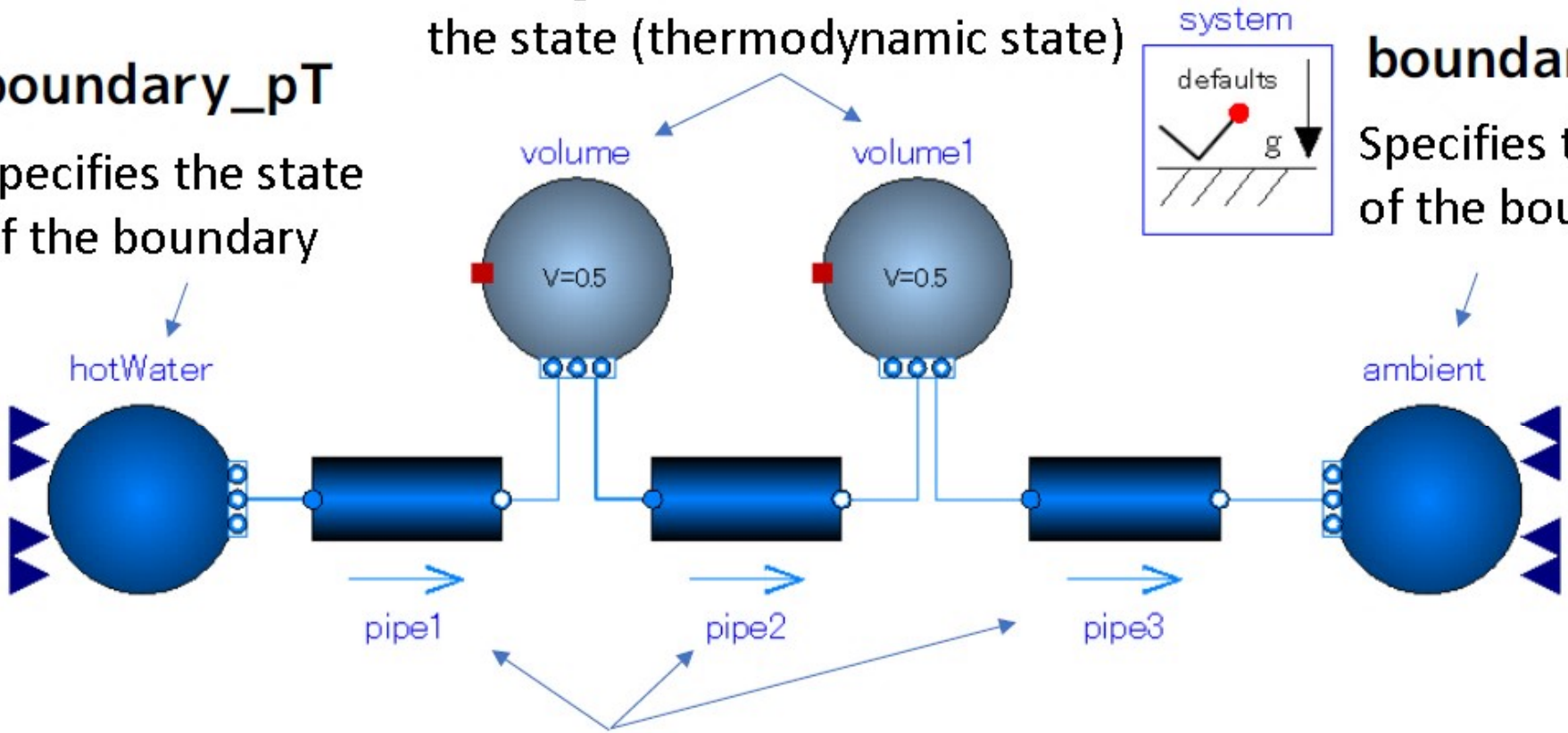
Seeking the time evolution of
the state (thermodynamic state)

boundary_pT

Specifies the state
of the boundary

boundary_pT

Specifies the state
of the boundary



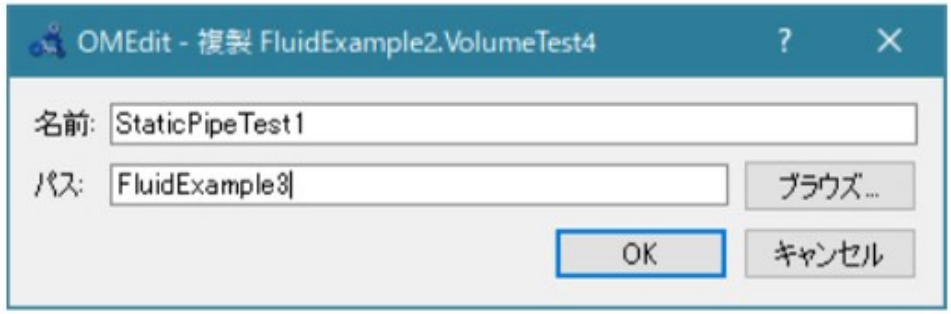
flow model

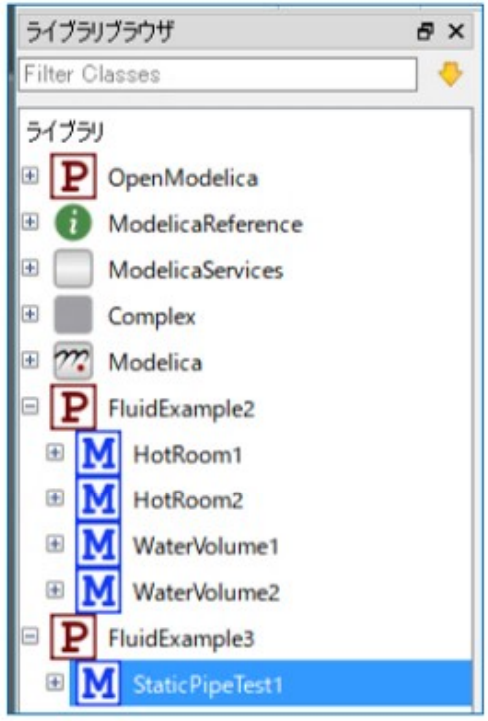
Find the mass flow between the two states

StaticPipeTest1

① Create a package named FluidExample3.

② Right-click WaterVolume2 in FluidExample2, select Duplicate, and copy it to FluidExample3 as StaticPipeTest1.



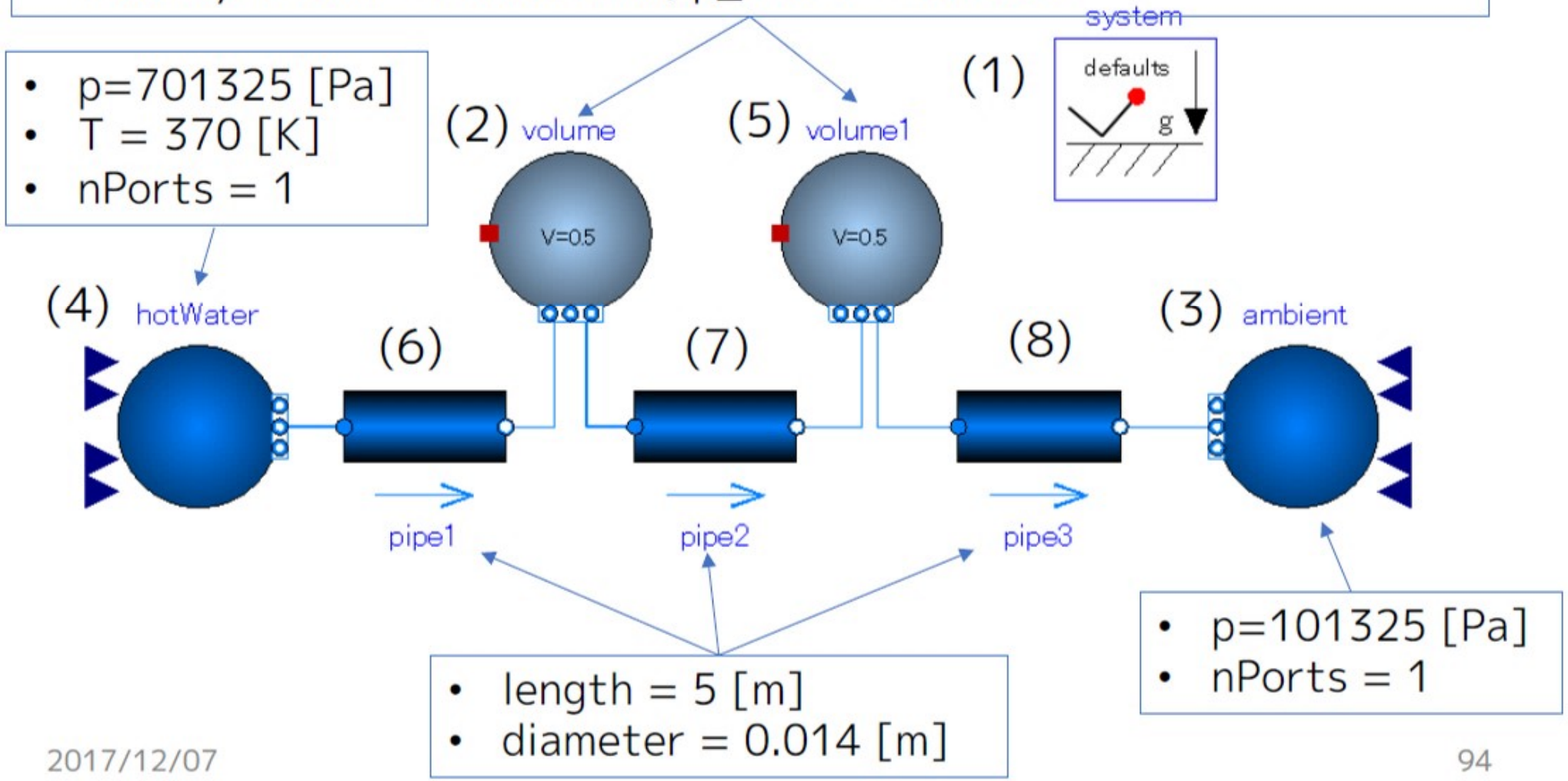


③ Create volume1 by duplicating volume.

④ As shown in the next slide, place three Fluid.Pipes.StaticPIpe and connect them.

⑤ Set the parameters.

- V=0.5 [m3]
- nPorts = 2
- use_portsData = false
- No portsData
- energyDynamics = FixedInitial, use_T_start = true, T_start = 280
- massDynamics = FixedInitial, p_start = 101325



- p=701325 [Pa]
- T = 370 [K]
- nPorts = 1

- p=101325 [Pa]
- nPorts = 1

- length = 5 [m]
- diameter = 0.014 [m]

StaticPipeTest1

⑥ Switch to text view and insert an import statement at the beginning of FluidExample3. Set the newly placed physical properties package for volume1 and pipe1 to pipe3.

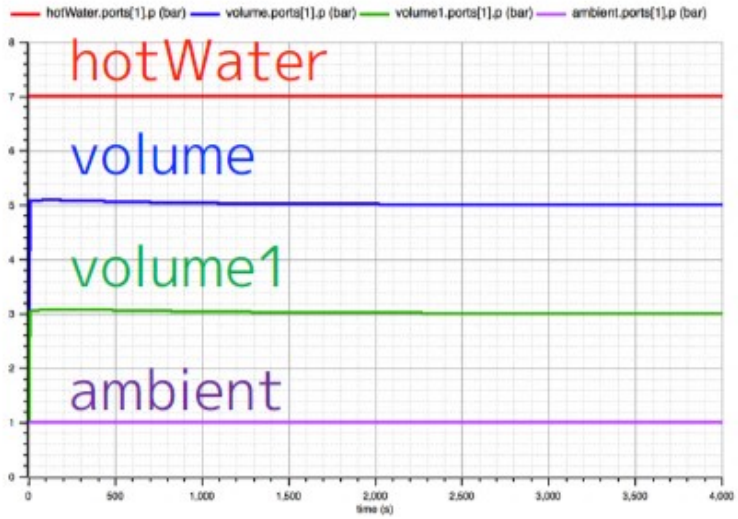
```
package FluidExample3
  import Modelica.Media;
  model StaticPipeTest1
```

```
Modelica.Fluid.Vessels.ClosedVolume volume1(redeclare package Medium = Medium,
  T_start = 280, V = 0.5,
  energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial, fluidLevel = 1.0,
  massDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
  nPorts = 2, p_start = 101325, use_portsData = false) annotation( ...);
```

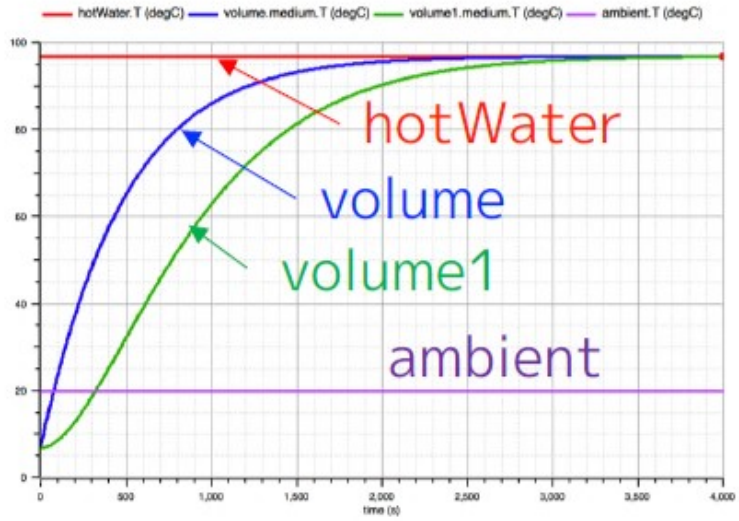
```
Modelica.Fluid.Pipes.StaticPipe pipe1(redeclare package Medium = Medium,
  diameter = 0.014, length = 5)  annotation( ...);
Modelica.Fluid.Pipes.StaticPipe pipe2(redeclare package Medium = Medium,
  diameter = 0.014, length = 5)  annotation( ...);
Modelica.Fluid.Pipes.StaticPipe pipe3(redeclare package Medium = Medium,
  diameter = 0.014, length = 5)  annotation( ...);
```

⑦ Save and model check and execute simulation

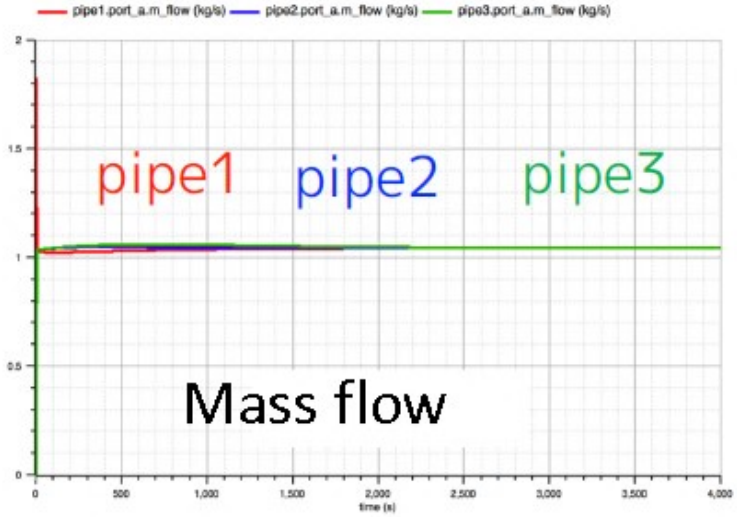
StaticPipeTest1

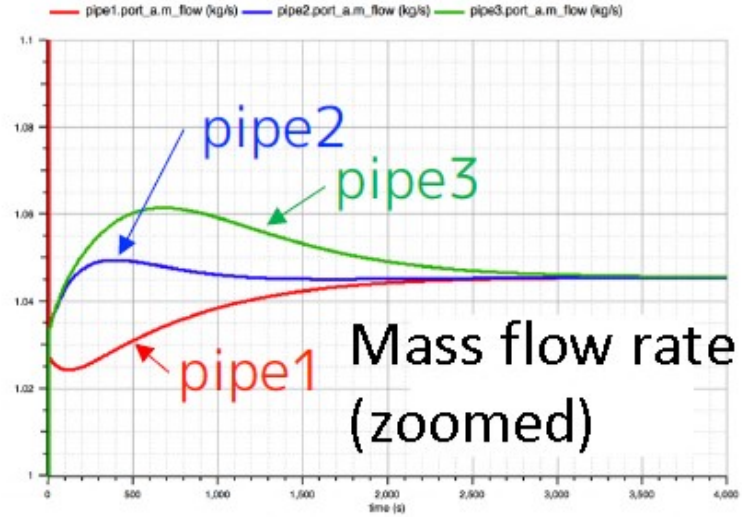


pressure



temperature



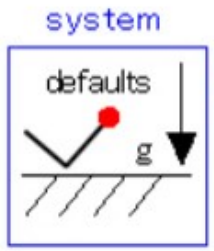


StaticPipeTest2

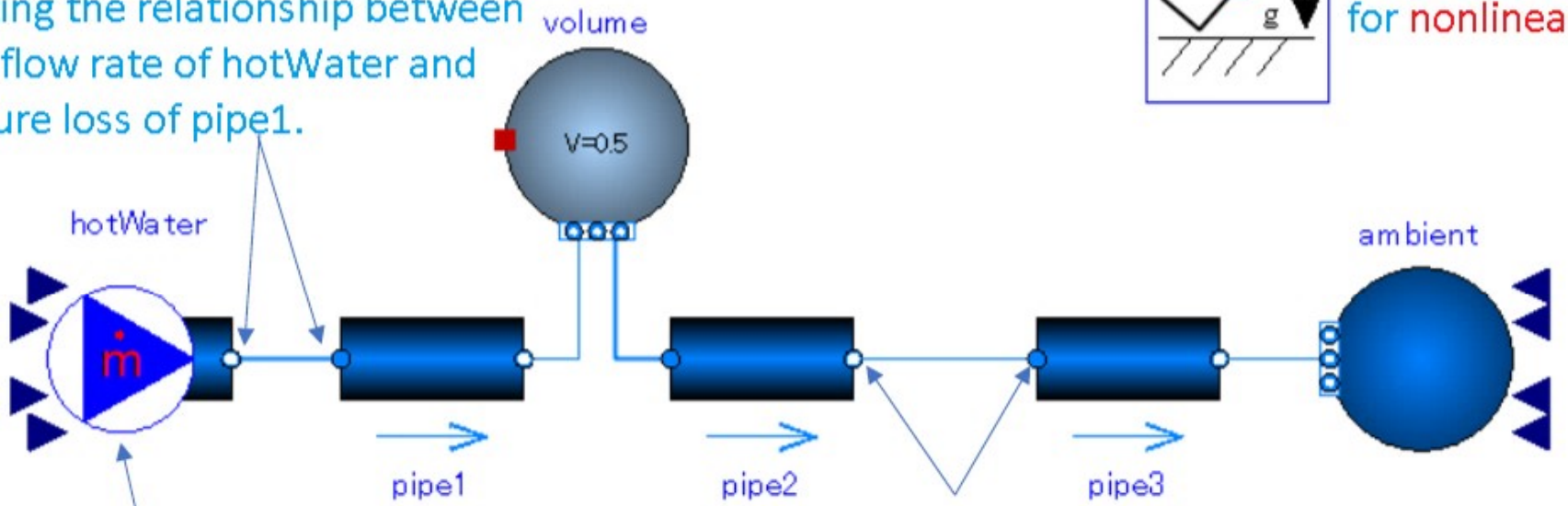
Connect flow models directly

- Delete volume1 and connect pipe2 and pipe3 directly.
- Replace hotWater from Boundary_pT with MassFlowSource_T.

The pressure of the FluidPort is determined by a nonlinear equation representing the relationship between the mass flow rate of hotWater and the pressure loss of pipe1.



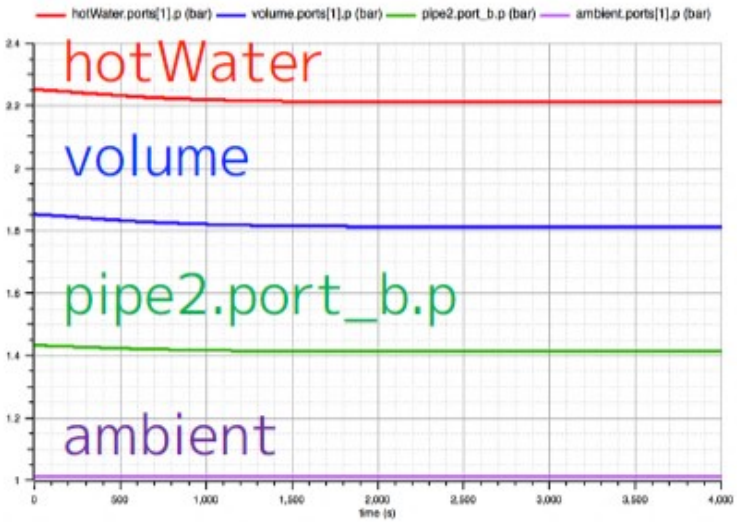
See Appendix 1 for nonlinear equations



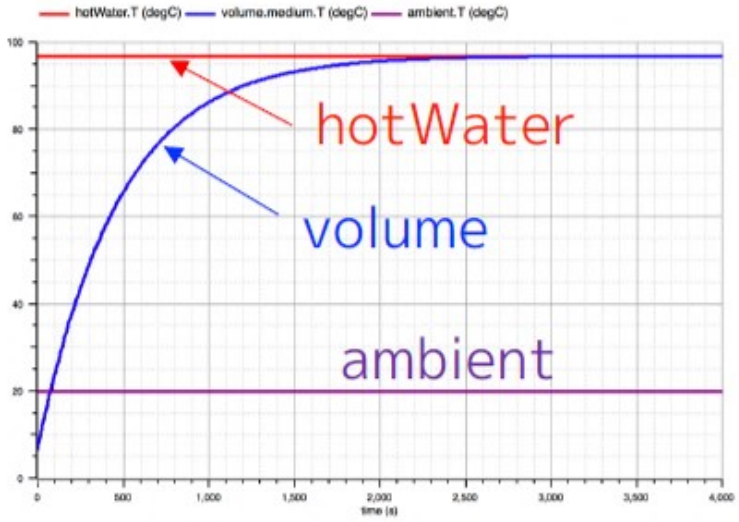
m_flow = 1.045 [kg/s]
T = 370 [K]
nPorts = 1

The pressure and flow rate of the FluidPort are determined by simultaneous nonlinear equations representing the relationship between the mass flow rate and the pressure loss of pipe1 and pipe2.

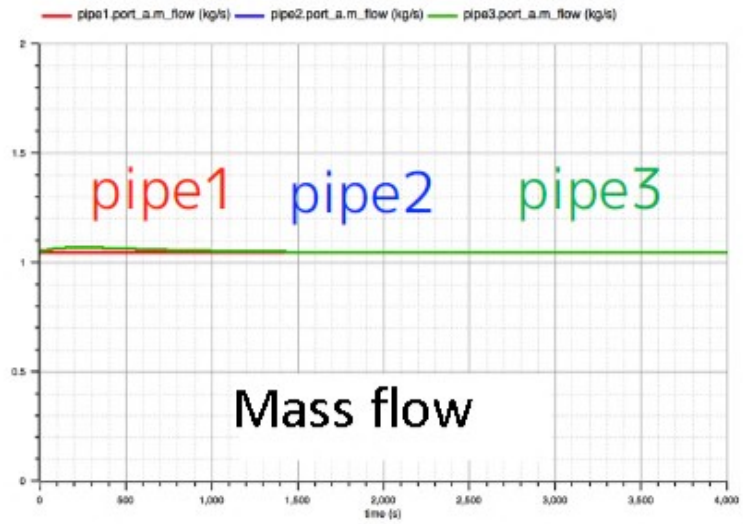
StaticPipeTest2

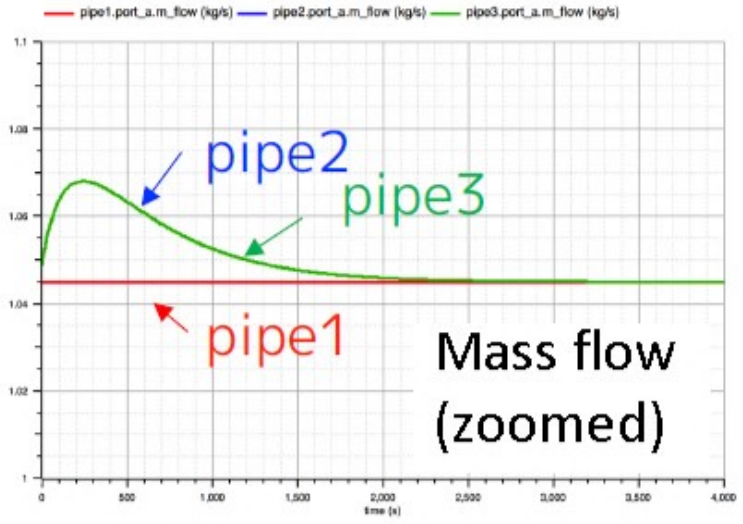


Pressure



Temperature



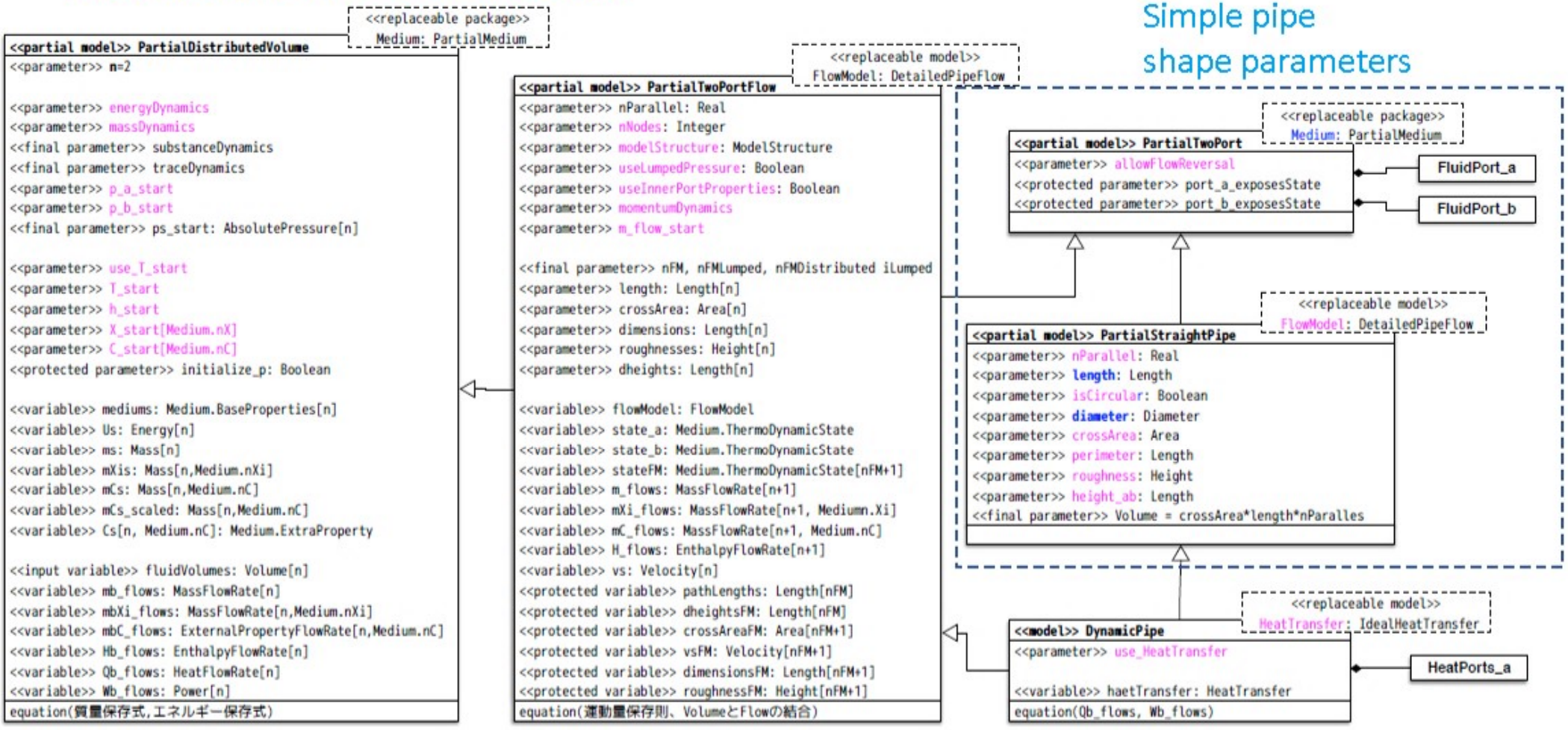


FluidExample4

DinamicPipe

Required input parameters
Selection input parameter

Inheritance relationship of DynaicPipe



Multiple Volume models (mass conservation, energy conservation)

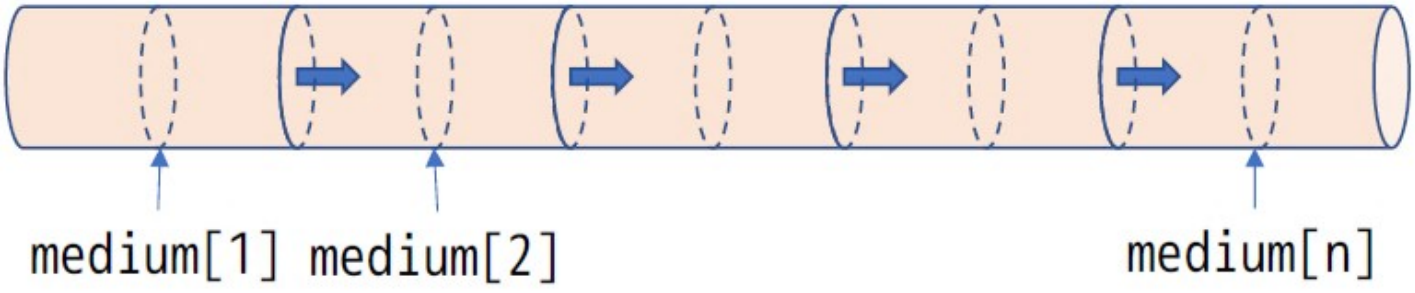
Model structure of Volume model and Flow model (Momentum, mass conservation formula implementation)

Correlation between pipe shape parameters and model parameters
Addition of heat transfer model

DynamicPipe

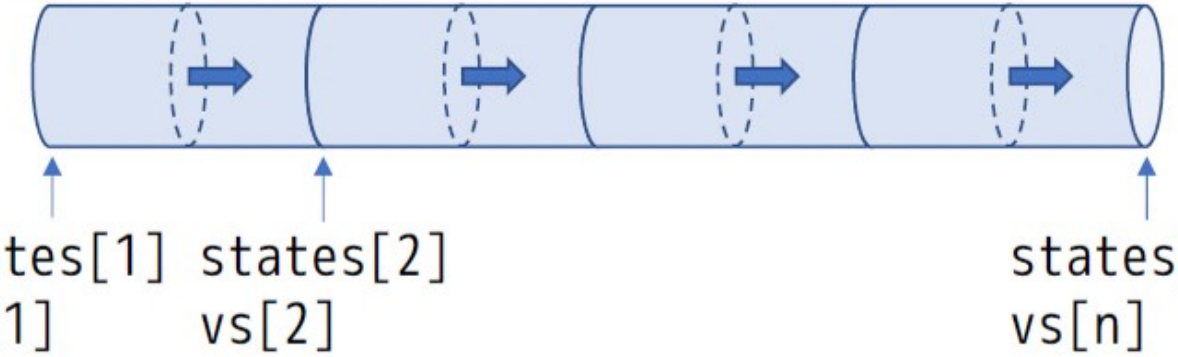
PartialDistributedVolume

Volume model that satisfies the law of conservation of mass and energy



PartialStaggeredFlowModel

Flow model implementing the
momentum conservation law



Number of models in
partialTwoPortFlow

PartialTwoPortFlow

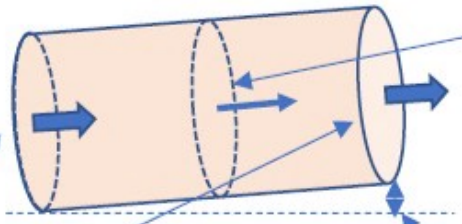
Arrange the volume model and the flow model alternately.

DynamicPipe

Volume model

State variables

`Us[n], ms[n], mXis[n,Xi], mCs[n,nC]`



`mediums[n]`
`vs[n]`

Boundary condition

`m_flows[n+1]`
`mXi_flows[n+1,nXi]`
`mC_flows[n+1,nC]`
`H_flows[n+1]`

Parameters

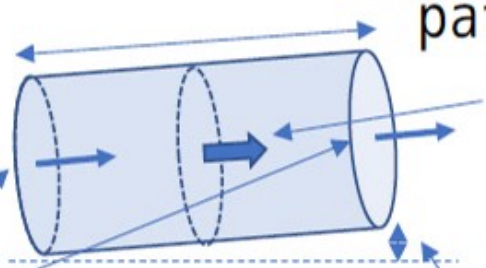
dheights [n]: Height difference
lengths [n]: length
crossAreas [n]: cross section
dimensions [n]: representative length
roughnesses [n]: Surface roughness

DynamicPipe

Flow model

State variables

```
Is[i] = m_flows[i]*pathLengths[i]
```



Boundary condition

statesFM [nFM +1]
vsFM [nFM +1]: Velocity

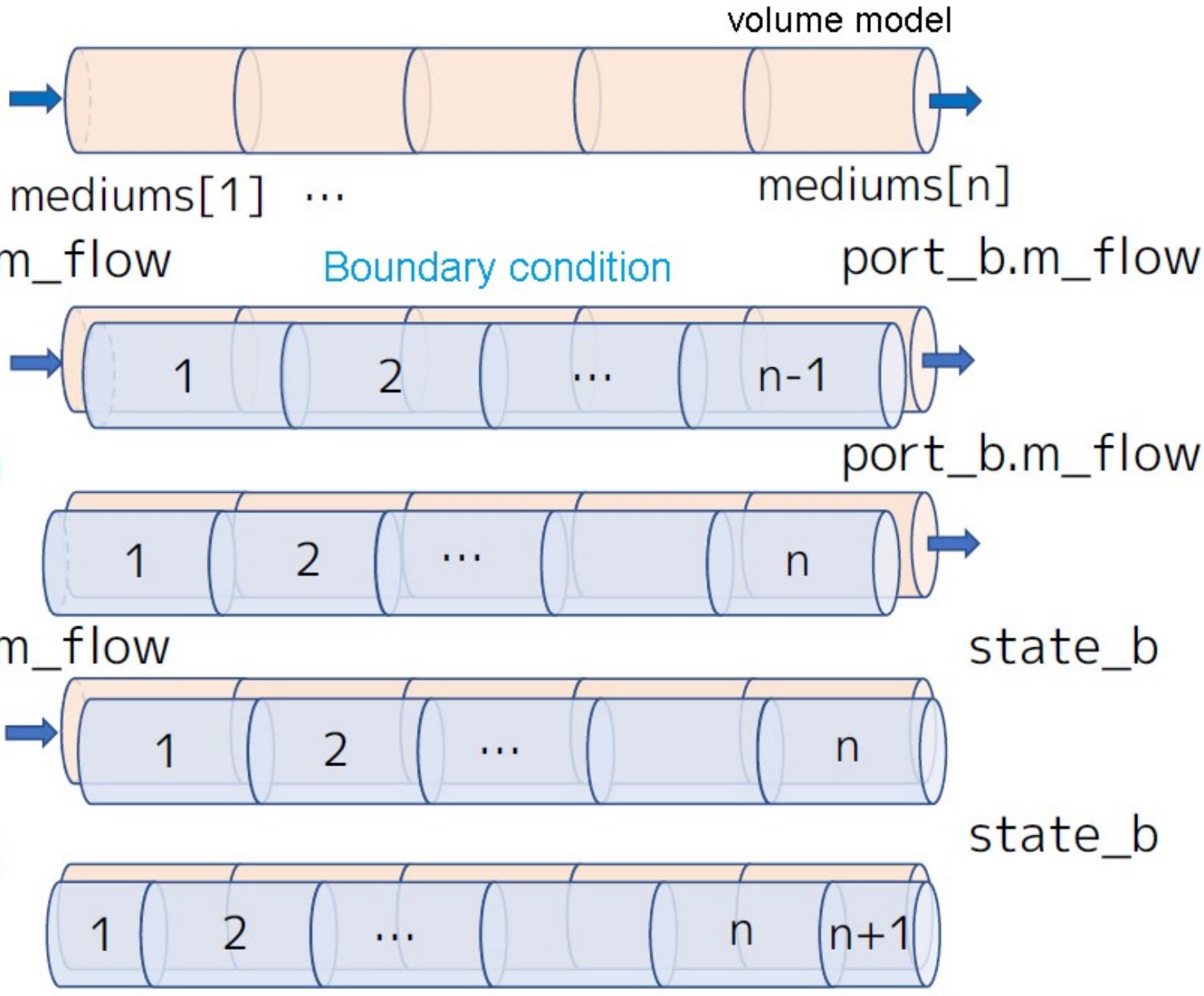
Parameters

dheightsFM [nFM]: Height difference
crossAreasFM [nFM +1]: Cross section
dimensionsFM [nFM +1]: representative length
roughnessFM [nFM +1]: Surface roughness

DynamicPipe

modelStructure

n = nNodes



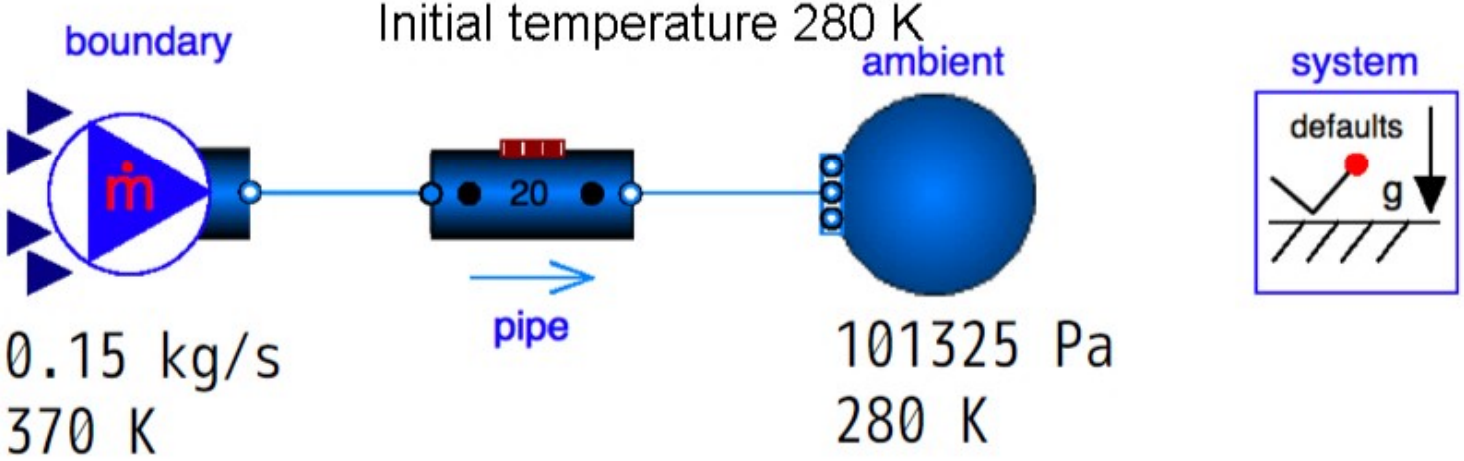
WaterExchange1～4

Pour hot water into a pipe containing cold water

Hot water (370 K (96.85 °C)) is poured into a pipe containing cold water (initial temperature 280 K (6.85 °C)) having an inner diameter of 0.015 m and a length of 20 m. Pipe is represented by DaynamicPipe. The upstream mass flow rate is set to 0.15 kg/s so that the flow velocity in the pipe is about 1.0 m/s.

WaterExchange1 20 Division
WaterExchange2 100 Division

Pipe
0.015 m inside diameter
Length 20 m
Initial temperature 280 K



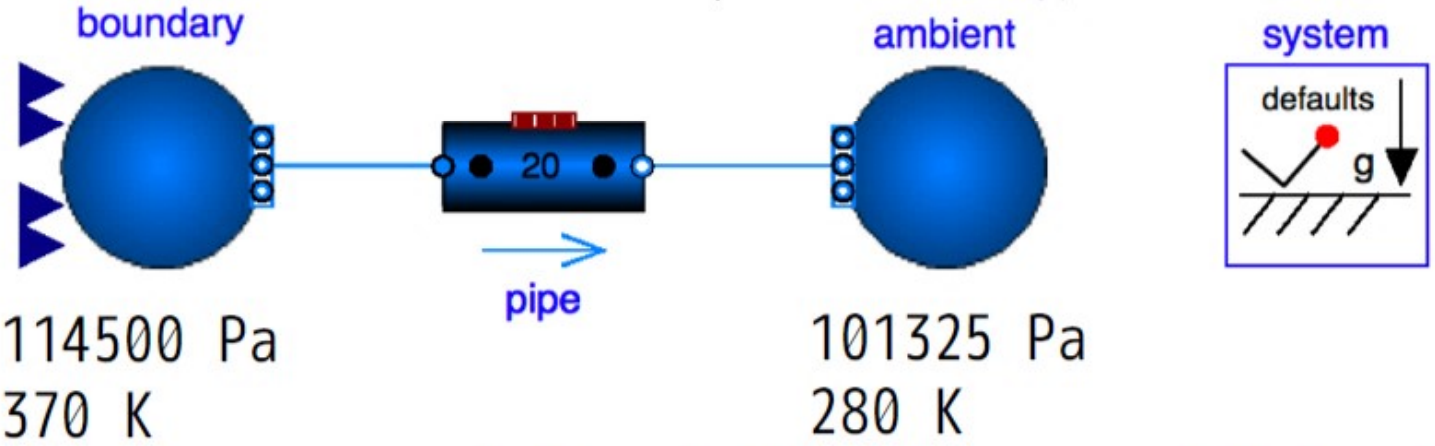
The upstream end is the flow boundary, so the volume model is inside the pipe, and the downstream end is the flow model inside the pipe because the thermodynamic state is fixed. Therefore, modelStructure selects av_b.

WaterExchange1～4

Adjust the upstream pressure to 114500 Pa so that the flow velocity in the pipe is about 1.0 m / s.

WaterExchange3 20 Division
WaterExchange4 100 Division

Pipe
0.015 m inside diameter
Length 20 m
Initial temperature 280 K



The temperature and pressure at both ends of the pipe are specified, and the thermodynamic state is established. Select a_v_b with states at both ends as boundary conditions.

WaterExchange1～4

```
model WaterExchange1
  replaceable package Medium = Media.Water.StandardWater;
  inner Modelica.Fluid.System system annotation( ...);
  Modelica.Fluid.Pipes.DynamicPipe pipe(redeclare package Medium = Medium,
    T_start = 280, diameter = 0.015,
    energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    length = 20, m_flow_start = 0.15,
    massDynamics = Modelica.Fluid.Types.Dynamics.DynamicFreeInitial,
    modelStructure = Modelica.Fluid.Types.ModelStructure.av_b,
    momentumDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    nNodes = 20, use_T_start = true) annotation( ...);
  Modelica.Fluid.Sources.FixedBoundary ambient(redeclare package Medium = Medium,
    T = 280, nPorts = 1, p = 101325) annotation( ...);
  Modelica.Fluid.Sources.MassFlowSource_T boundary(redeclare package Medium = Medium,
    T = 370, m_flow = 0.15, nPorts = 1) annotation( ...);
equation
  connect(boundary.ports[1], pipe.port_a) annotation( ...);
  connect(pipe.port_b, ambient.ports[1]) annotation( ...);
  annotation( ...);
end WaterExchange1;
```

- Start Time = 0
- Stop Time = 60
- Number of Intervals = 150
- Non Linear Solver = hybrid

The distance flowing in one time step should be less than 1/2 of pathLength.

WaterExchange1～4

```
model WaterExchange2
  replaceable package Medium = Media.Water.StandardWater;
  inner Modelica.Fluid.System system annotation( ...);
  Modelica.Fluid.Pipes.DynamicPipe pipe(redeclare package Medium = Medium,
    T_start = 280, diameter = 0.015,
    energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    length = 20, m_flow_start = 0.15,
    massDynamics = Modelica.Fluid.Types.Dynamics.DynamicFreeInitial,
    modelStructure = Modelica.Fluid.Types.ModelStructure.av_b,
    momentumDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    nNodes = 100, use_T_start = true) annotation( ...);
  Modelica.Fluid.Sources.FixedBoundary ambient(redeclare package Medium = Medium,
    T = 280, nPorts = 1, p = 101325) annotation( ...);
  Modelica.Fluid.Sources.MassFlowSource_T boundary(redeclare package Medium = Medium,
    T = 370, m_flow = 0.15, nPorts = 1) annotation( ...);
equation
  connect(boundary.ports[1], pipe.port_a) annotation( ...);
  connect(pipe.port_b, ambient.ports[1]) annotation( ...);
  annotation( ...);
end WaterExchange2;
```

- Start Time = 0
- Stop Time = 60
- Number of Intervals = 750
- Non Linear Solver = hybrid

WaterExchange1～4

```
model WaterExchange3
  replaceable package Medium = Media.Water.StandardWater;
  inner Modelica.Fluid.System system annotation( ...);
  Modelica.Fluid.Sources.Boundary_pT boundary(redeclare package Medium = Medium,
    T = 370, nPorts = 1, p = 114500) annotation( ...);
  Modelica.Fluid.Pipes.DynamicPipe pipe(redeclare package Medium = Medium,
    T_start = 280, diameter = 0.015,
    energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    length = 20, m_flow_start = 0.1,
    massDynamics = Modelica.Fluid.Types.Dynamics.DynamicFreeInitial,
    modelStructure = Modelica.Fluid.Types.ModelStructure.a_v_b,
    momentumDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    nNodes = 20, use_T_start = true) annotation( ...);
  Modelica.Fluid.Sources.FixedBoundary ambient(redeclare package Medium = Medium,
    T = 280, nPorts = 1, p = 101325) annotation( ...);
equation
  connect(pipe.port_b, ambient.ports[1]) annotation( ...);
  connect(boundary.ports[1], pipe.port_a) annotation( ...);
  annotation( ...);
end WaterExchange3;
```

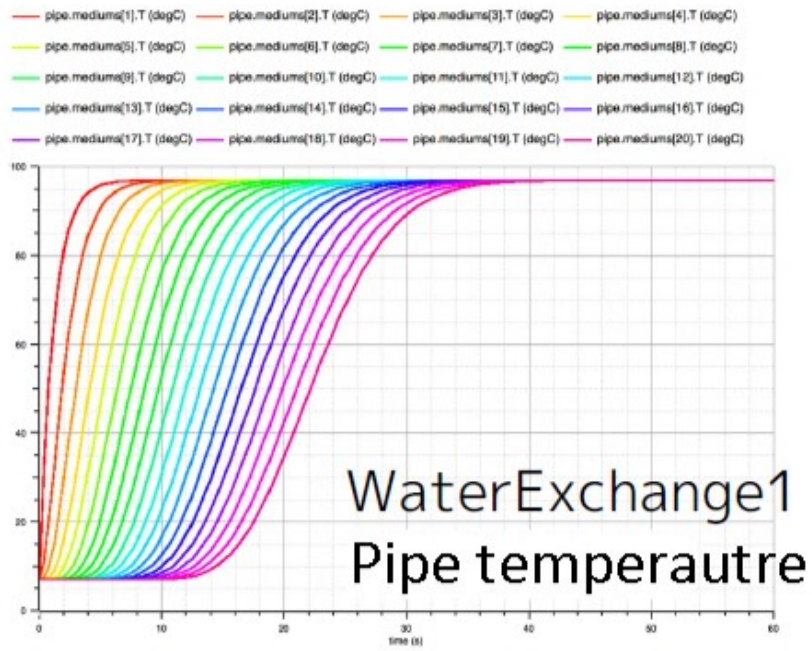
- Start Time = 0
- Stop Time = 60
- Number of Intervals = 150
- Non Linear Solver = hybrid

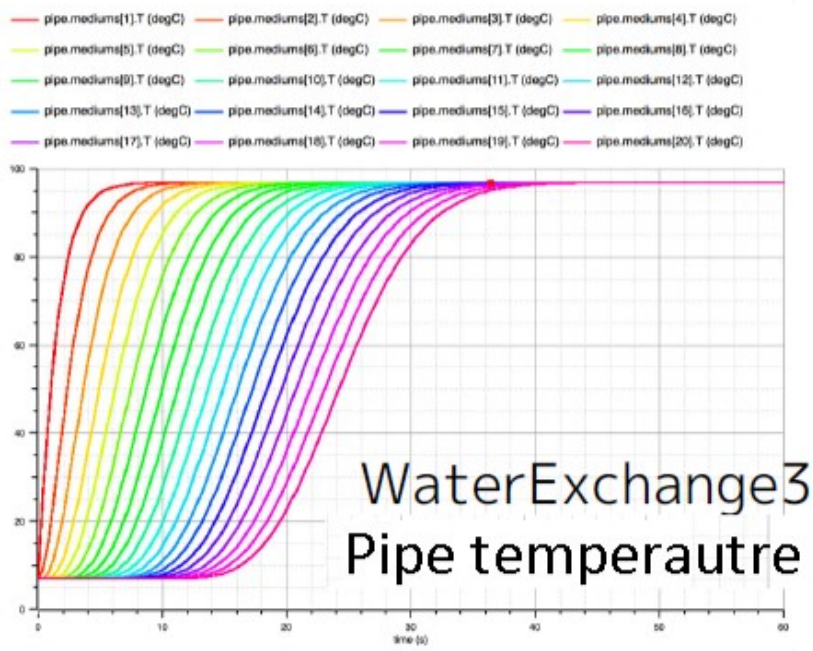
WaterExchange1～4

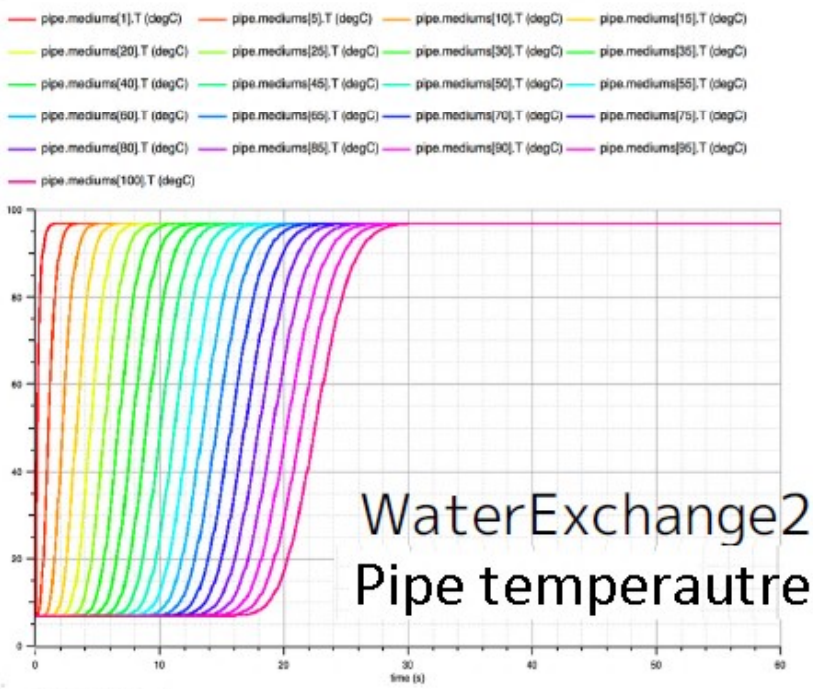
```
model WaterExchange4
  replaceable package Medium = Media.Water.StandardWater;
  inner Modelica.Fluid.System system annotation( ...);
  Modelica.Fluid.Sources.Boundary_pT boundary(redeclare package Medium = Medium,
    T = 370, nPorts = 1, p = 114500) annotation( ...);
  Modelica.Fluid.Pipes.DynamicPipe pipe(redeclare package Medium = Medium,
    T_start = 280, diameter = 0.015,
    energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    length = 20, m_flow_start = 0.1,
    massDynamics = Modelica.Fluid.Types.Dynamics.DynamicFreeInitial,
    modelStructure = Modelica.Fluid.Types.ModelStructure.a_v_b,
    momentumDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    nNodes = 100, use_T_start = true) annotation( ...);
  Modelica.Fluid.Sources.FixedBoundary ambient(redeclare package Medium = Medium,
    T = 280, nPorts = 1, p = 101325) annotation( ...);
equation
  connect(pipe.port_b, ambient.ports[1]) annotation( ...);
  connect(boundary.ports[1], pipe.port_a) annotation( ...);
  annotation( ...);
end WaterExchange4;
```

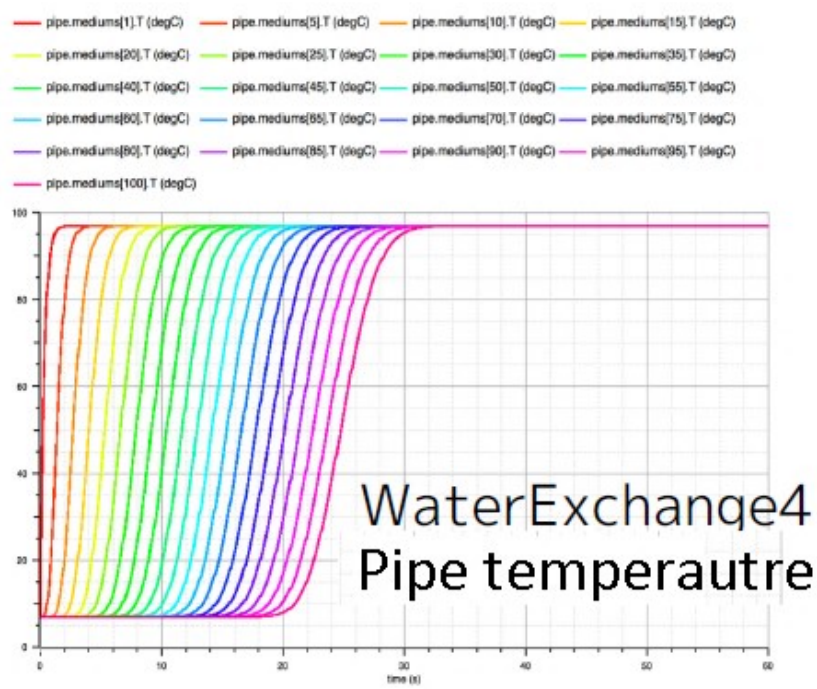
- Start Time = 0
- Stop Time = 60
- Number of Intervals = 750
- Non Linear Solver = hybrid

WaterExchange1～4







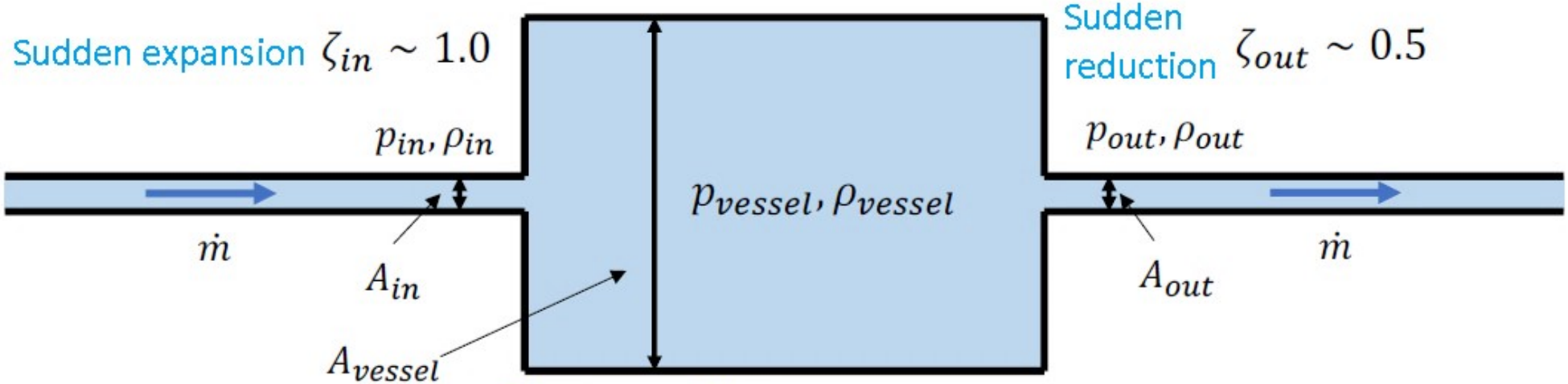


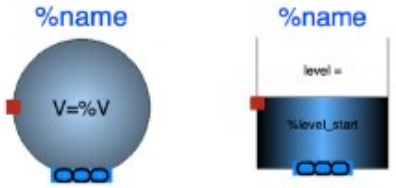
FluidExample5

VesselPortsData FluidPort characteristic data

<<record>> Modelica.Fluid.Vessels.BaseClasses.VesselPortsData
<<parameter>> **diameter**: SI.Diameter
<<parameter>> **height**=0: SI.Height
<<parameter>> **zeta_out**(min=0)=0.5: Real
<<parameter>> **zeta_in**(min=0)=1.04: Real

$diameter$: Inner diameter [m]
$height$: Height from bottom [m]
ζ_{out} : Pressure loss coefficient at outflow
ζ_{in} : Pressure loss coefficient at inflow

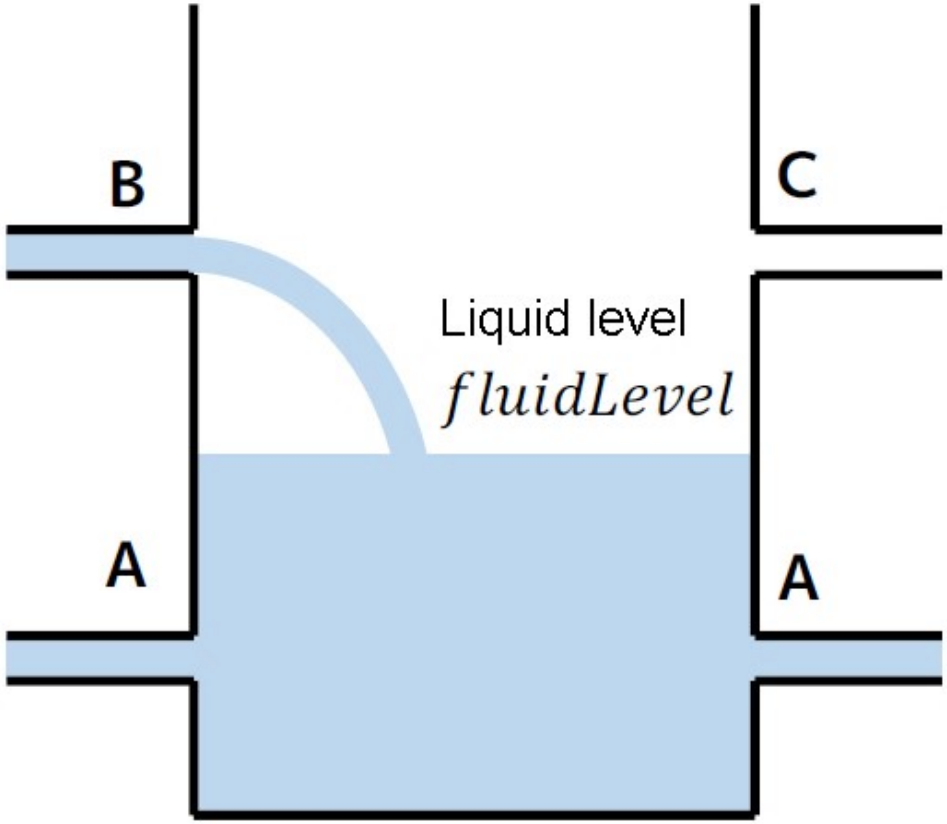




ClosedVolume OpenTank

- Valid when use_portsData = true
- height is invalid for ClosedVolume

VesselPortsData



A. regularFlow

Condition $fluidLevel \geq height_i$

if use_portsData = true

$p_i = p_{port}$ (See Appendix 2)

if use_portsData = false

$p_i = p_{vessel_i}$

$s_i = fluidLevel - height_i$

s_i = s[i]: State variable representing inflow

When use_portsData = false, $height_i = 0$, so it becomes regularFlow.

VesselPortsData

B. inFlow

Conditions $not\ regularFlow\ and\ (s_i > 0\ \ or\ height_i \geq fluidLevel_{max})$

Processing $p_i = p_{vessel_i}$

$s_i = \dot{m}_i$

C. noFlow

Conditions and other cases

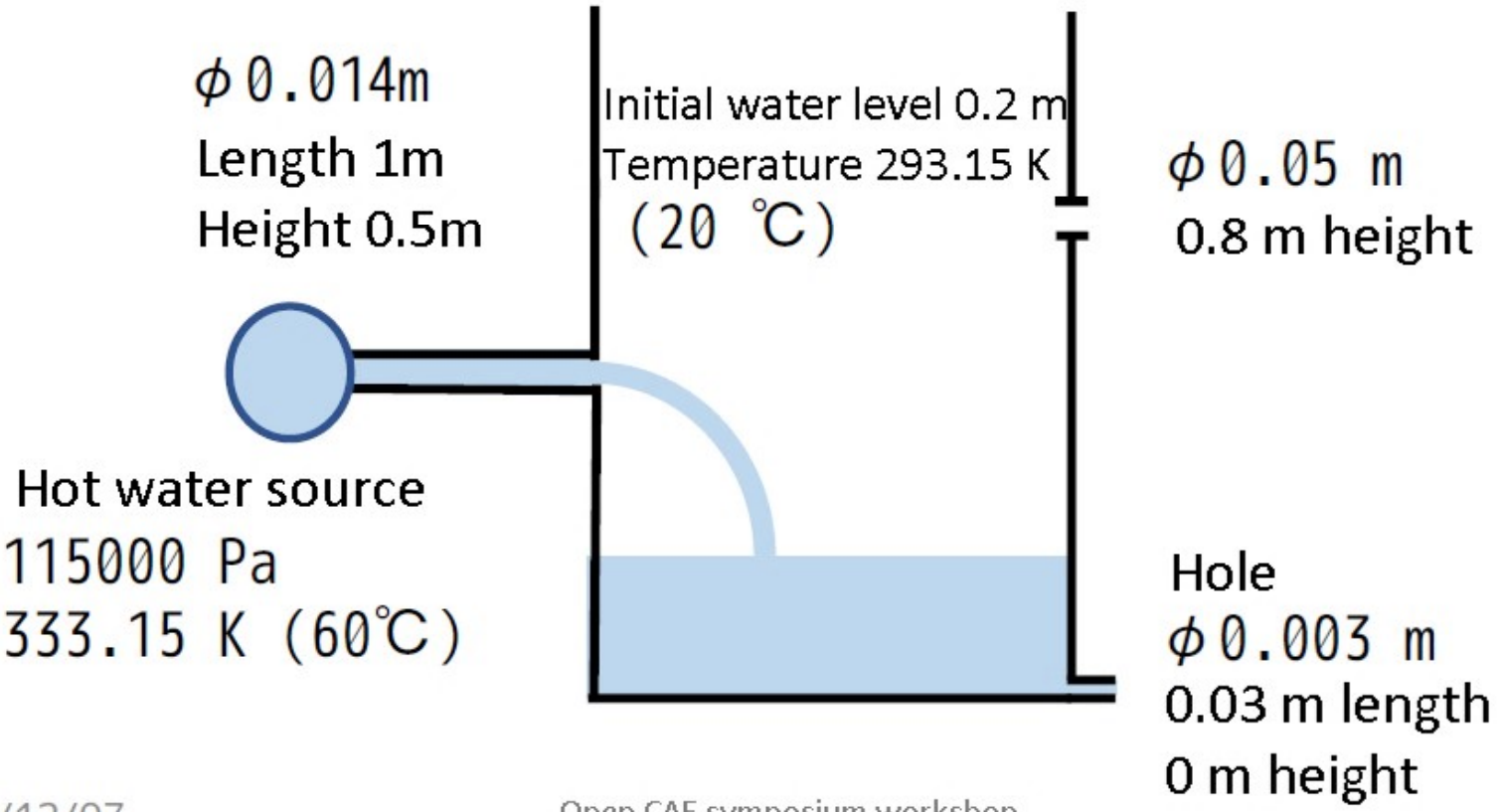
Processing $\dot{m}_i = 0$

$$s_i = \frac{p_i - p_{vessel_i}}{p_{default}} \cdot (height_i - fluidLevel)$$

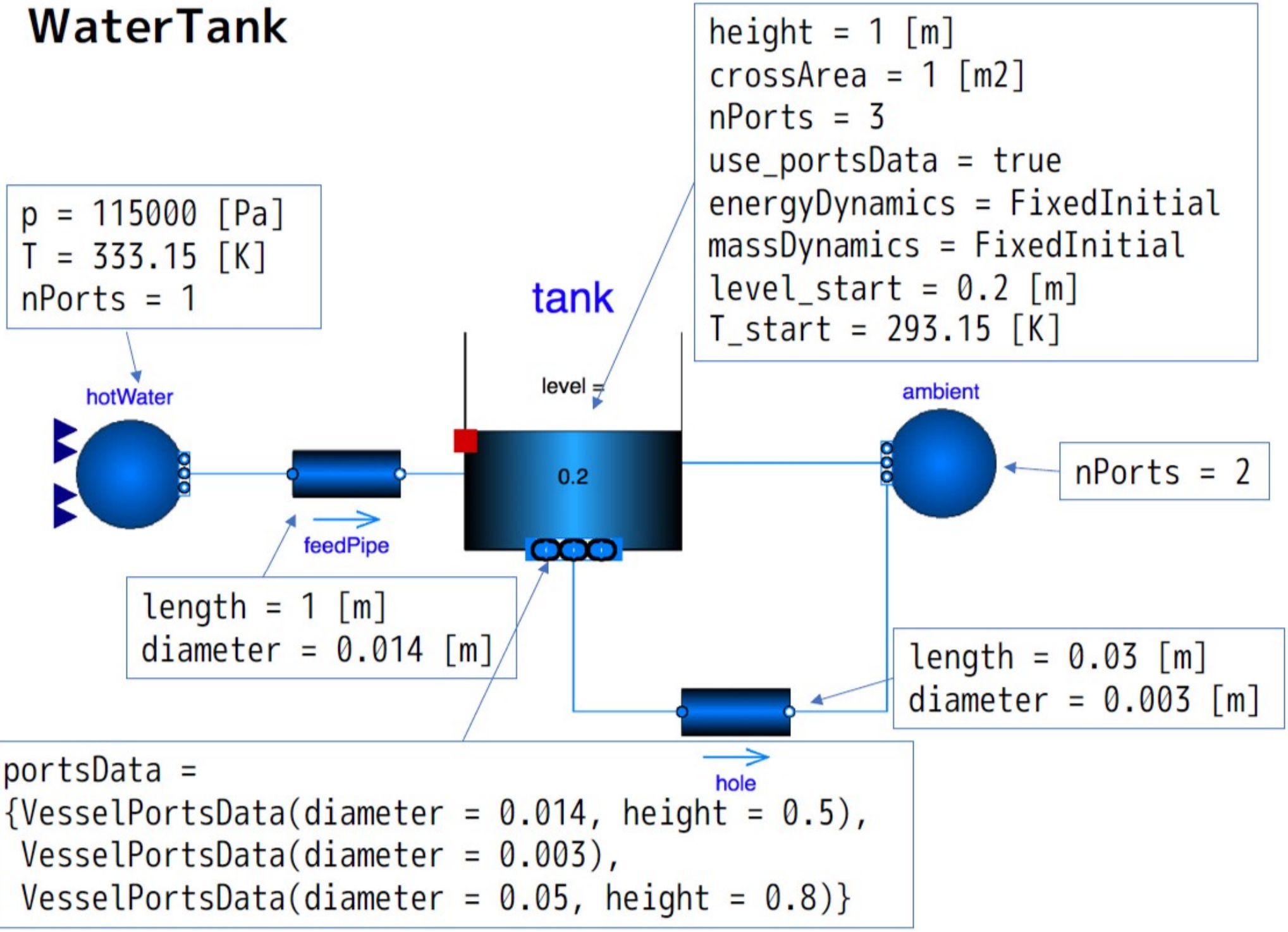
WaterTank

Pour hot water in a tank containing water

- The tank contains water at a temperature of 20 ° C and a water level of 0.2 m.
- There is a water inlet at a height of 50 cm and hot water of 60 ° C is poured.
- There is a hole 3 mm in diameter and 3 cm in length at the bottom of the tank.
- There is a drain hole of 5 cm in diameter and 0.8 m in height on the wall of the tank.
- The water supply port is a 14 mm diameter, 1 m long pipe connected to a hot water source with a temperature of 60 ° C and a pressure of 115 kPa.



WaterTank


height = 1 [m]
crossArea = 1 [m2]
nPorts = 3
use_portsData = true
energyDynamics = FixedInitial
massDynamics = FixedInitial
level_start = 0.2 [m]
T_start = 293.15 [K]
p = 115000 [Pa]
T = 333.15 [K]
nPorts = 1
tank
level =
hotWater
ambient
0.2
nPorts = 2
feedPipe
length = 1 [m]
diameter = 0.014 [m]
length = 0.03 [m]
diameter = 0.003 [m]
hole
portsData =
{VesselPortsData(diameter = 0.014, height = 0.5),
VesselPortsData(diameter = 0.003),
VesselPortsData(diameter = 0.05, height = 0.8)}

WaterTank

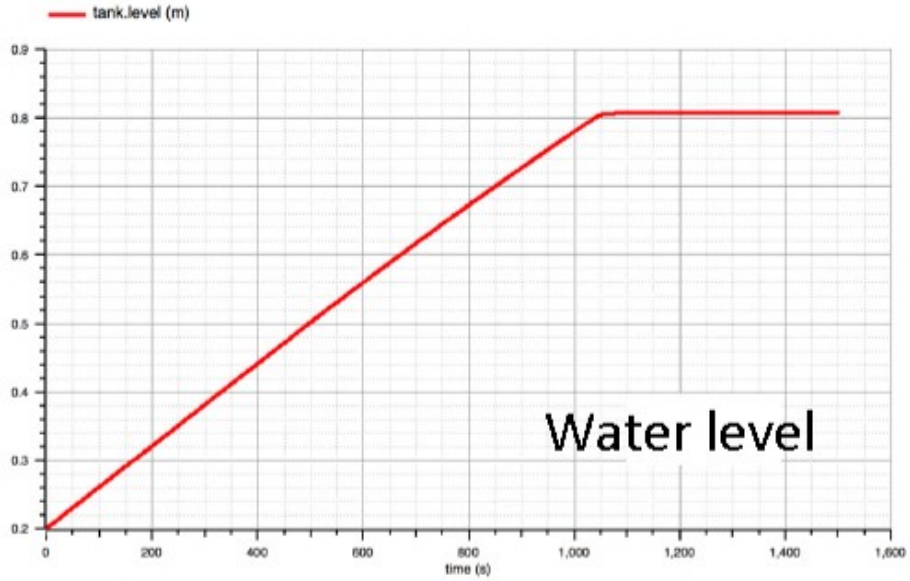
```
model WaterTank
  replaceable package Medium = Modelica.Media.Water.StandardWater;
  import Modelica.Fluid.Vessels.BaseClasses.VesselPortsData;
  Modelica.Fluid.Vessels.OpenTank tank(redeclare package Medium = Medium,
    T_start = 293.15,crossArea = 1,
    energyDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
    height = 1, level_start = 0.2,
    massDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial, nPorts = 3,
    portsData
     = {VesselPortsData(diameter = 0.014, height = 0.5),
        VesselPortsData(diameter = 0.003),
        VesselPortsData(diameter = 0.05, height = 0.8)})
    annotation( ...);
  Modelica.Fluid.Sources.Boundary_pT hotWater(redeclare package Medium = Medium,
    T = 333.15, nPorts = 1, p = 115000)
    annotation( ...);
  Modelica.Fluid.Pipes.StaticPipe pipe(redeclare package Medium = Medium,
    diameter = 0.014, length = 1)  annotation( ...);
  Modelica.Fluid.Sources.FixedBoundary ambient(redeclare package Medium = Medium,
    nPorts = 2)  annotation( ...);
  Modelica.Fluid.Pipes.StaticPipe hole(redeclare package Medium = Medium,
    diameter = 0.003, length = 0.03)  annotation( ...);
  inner Modelica.Fluid.System. system annotation( ...);
```

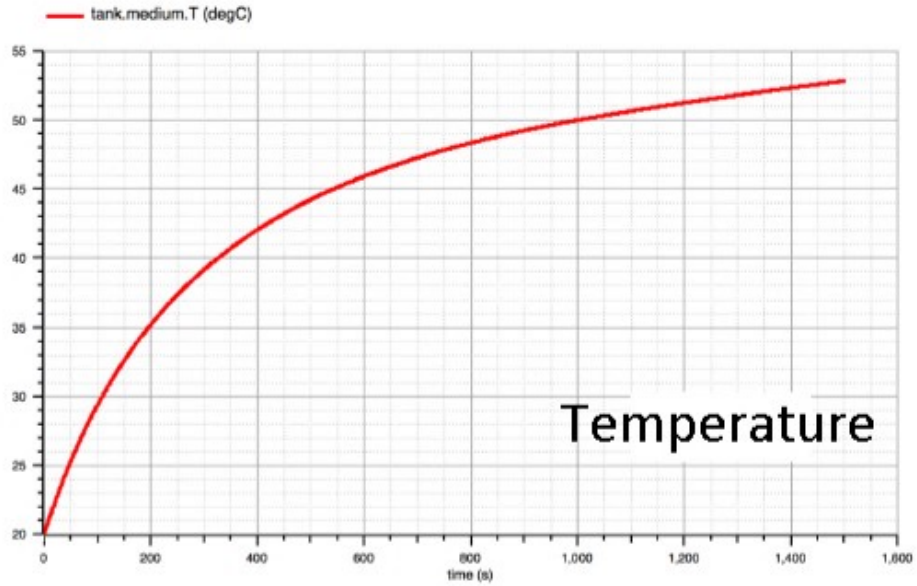
WaterTank

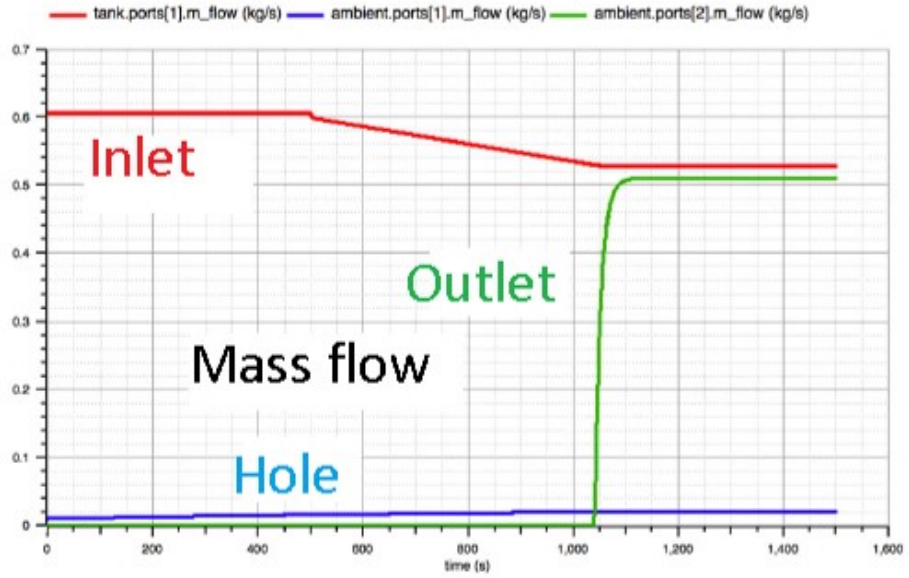
```
equation
  connect(tank.ports[3], ambient.ports[2]) annotation( ...);
  connect(hole.port_b, ambient.ports[1]) annotation( ...);
  connect(hole.port_a, tank.ports[2]) annotation( ...);
  connect(pipe.port_b, tank.ports[1]) annotation( ...);
  connect(pipe.port_a, hotWater.ports[1]) annotation( ...);
  annotation(
end WaterTank;
```

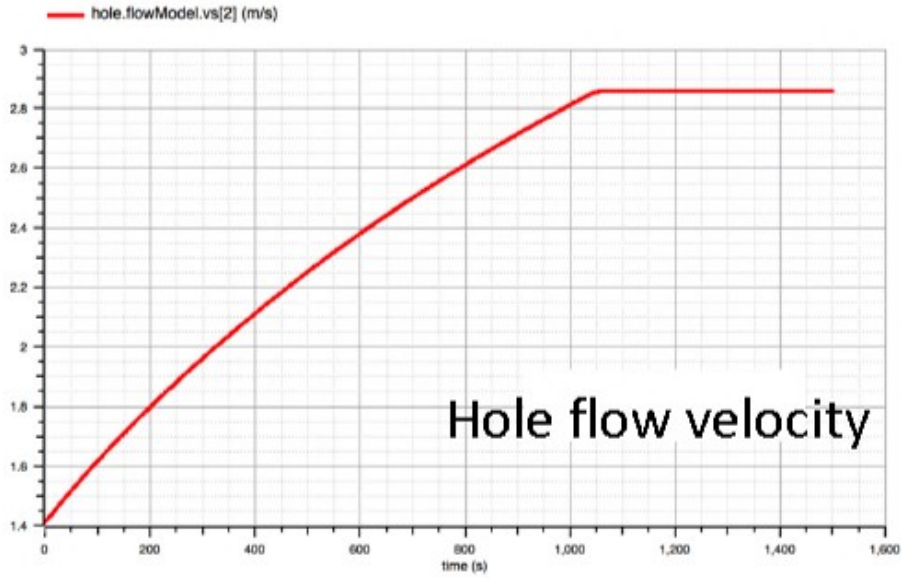
WaterTank

Simulation result







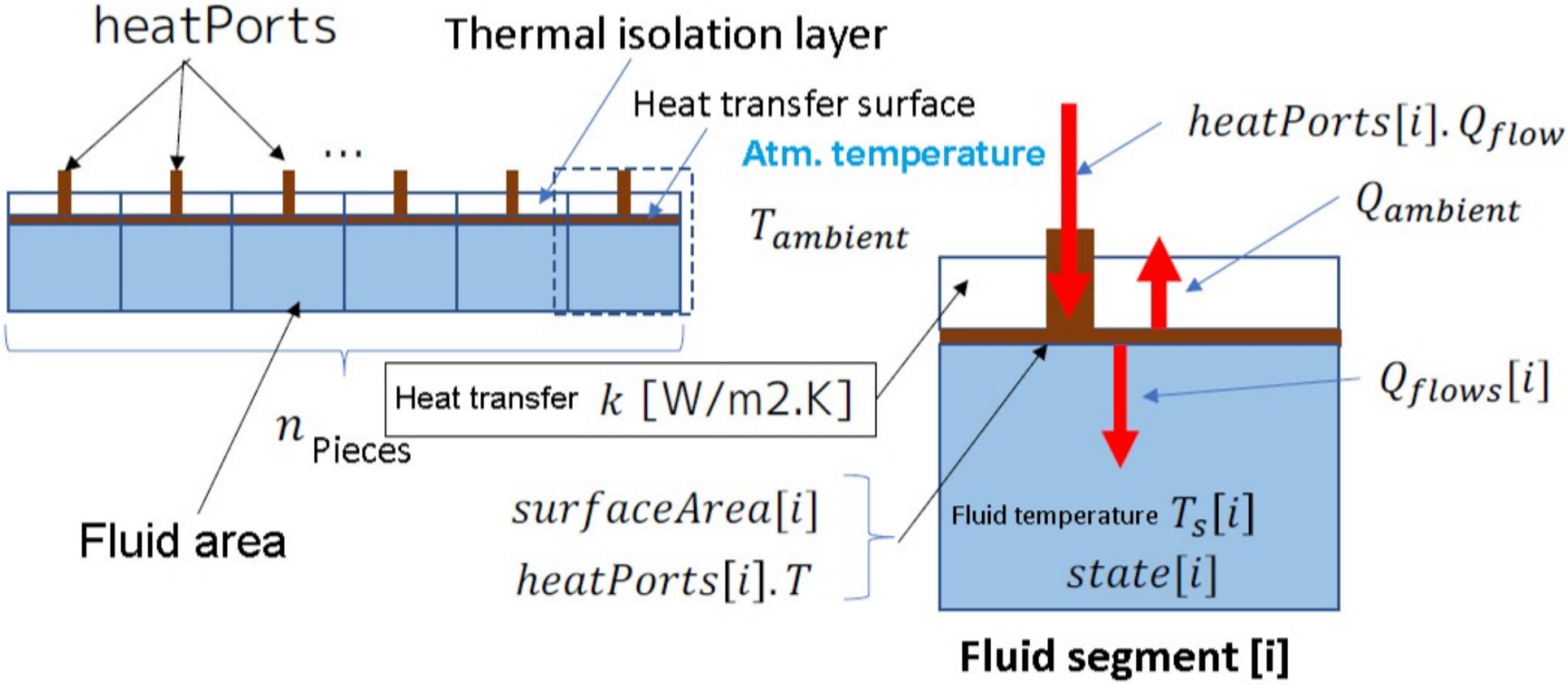


FluidExample6

HeatTransfer model

PartialHeatTransfer

- Base model of heat transfer
- Divide the fluid region, heat transfer surface, and heat insulation layer into n Segments and calculate the amount of heat supplied to the fluid.



HeatTransfer model

in the case of $use\_k = true$

$$Q_{flows}[i] = heatPorts[i].Q_{flow} - Q_{ambient}$$
$$= heatPorts[i].Q_{flow}$$
$$+ k \cdot surfaceAreas[i] \cdot (T_{ambient} - heatPorts[i].T)$$

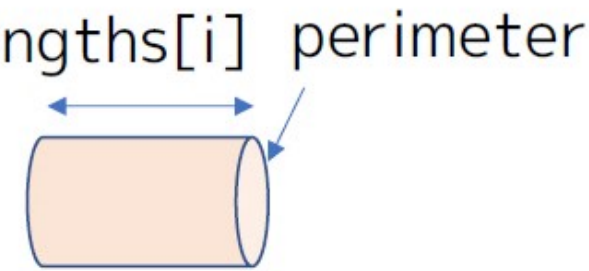
$use\_k = false$ No heat flow to atmosphere

$$Q_{flows}[i] = heatPorts[i].Q_{flow}$$

IdealHeatTransfer

$heatPorts[i].T = T_s[i]$ **Heat transfer surface temperature = Fluid temperature**

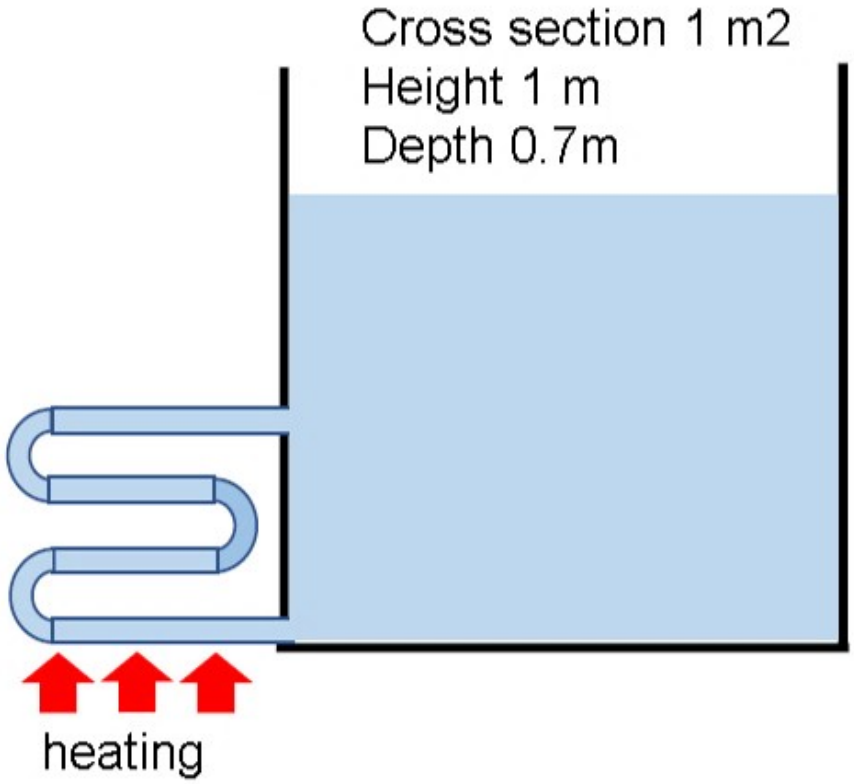
DynamicPipe



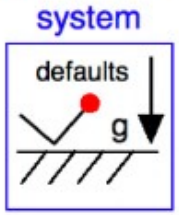
surfaceAreas=perimeter*lengths

WaterHeating

Heat the pipe to heat the water in the tank



WaterHeating




```
Q_flow = 20000.0 * pipe.dxs [W]
```

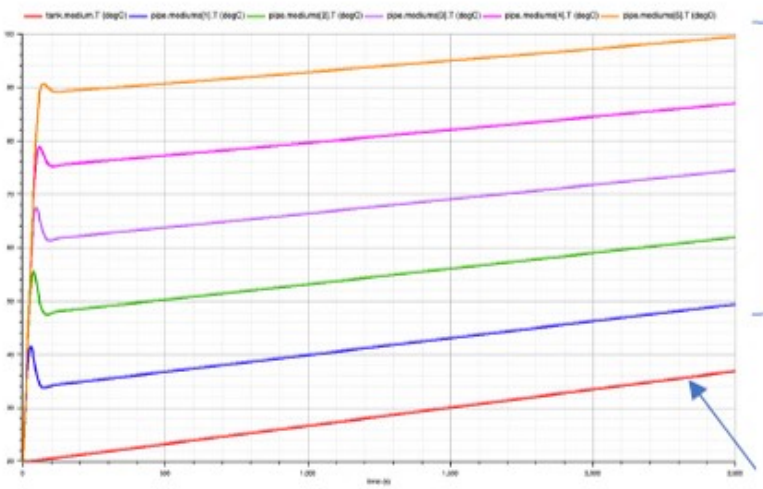
```
length = 5 [m]
diameter = 0.03 [m]
height_ab = 0.3
energyDynamics = FixedInitial
massDynamics = FixedInitial
momentumDynamics = FixedInitial
use_HeatTransfer = true
T_start = 293.15 [K]
m_flow_start = 0.02 [kg/s]
nNodes = 5
modelStructure = a_v_b
```

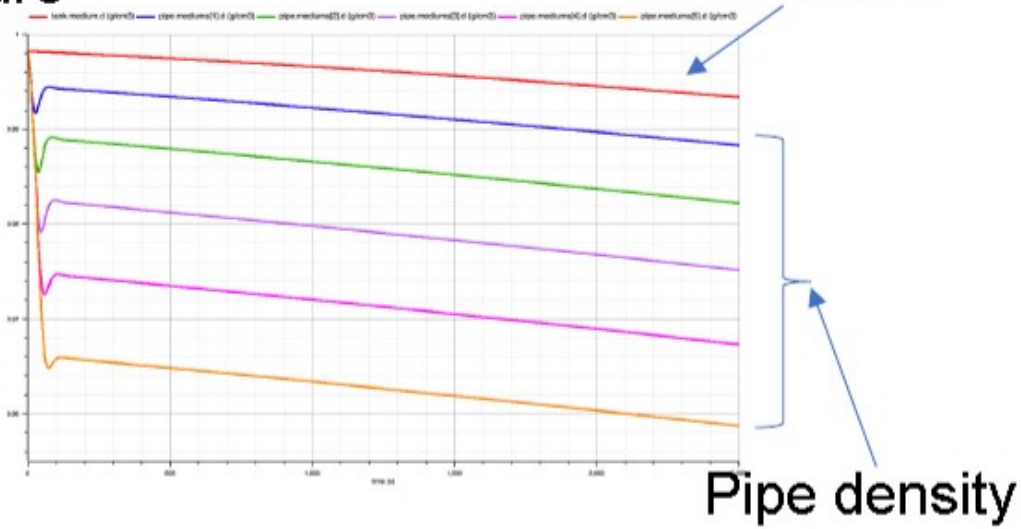
```
height = 1 [m]
crossArea = 1 [m]
nPorts = 2
use_portsData = true
energyDynamics = FixedInitial
massDynamics = FixedInitial
level_start = 0.7 [m]
use_T_start = true
T_start = 293.15 [K]
```

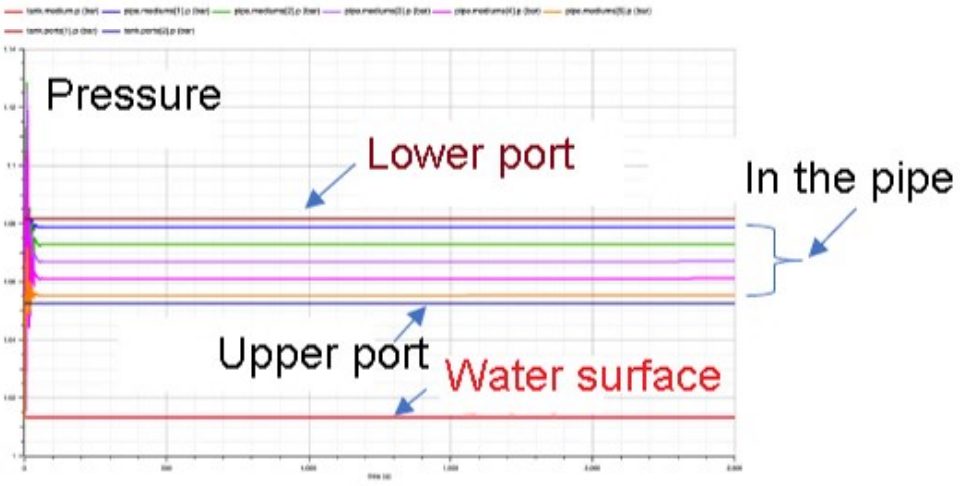
```
portsData
={VesselPortsData(diameter = 0.03),
VesselPortsData(diameter = 0.03, height = 0.3)}
```

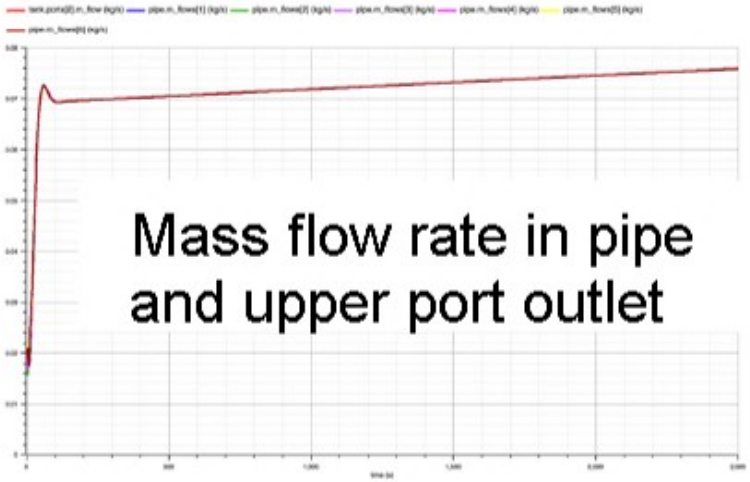
WaterHeating

Simulation result









IncompressibleFluidNetwork

Copy the model from Modelica.Fluid.Examples

Copy Modelica.Fluid.Examples.IncompressibleFluidNetwork into a user created package.

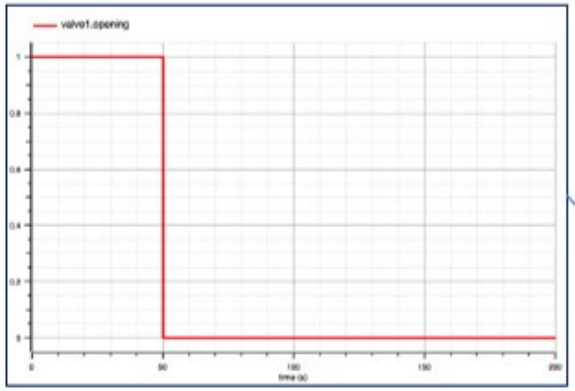
If the simulation cannot be performed due to the scope problem, describe the import statement so that the components to be used can be referenced.

```
package MyPackage
  import Modelica.Fluid.Types;
  import Modelica.Fluid.Pipes;
  import Modelica.Fluid.Valves;
  import Modelica.Fluid.Sources;
  import Modelica.Thermal;

  model IncompressibleFluidNetwork "Multi-way connections of pipes and incompressible
medium model"
...
```

IncompressibleFluidNetwork

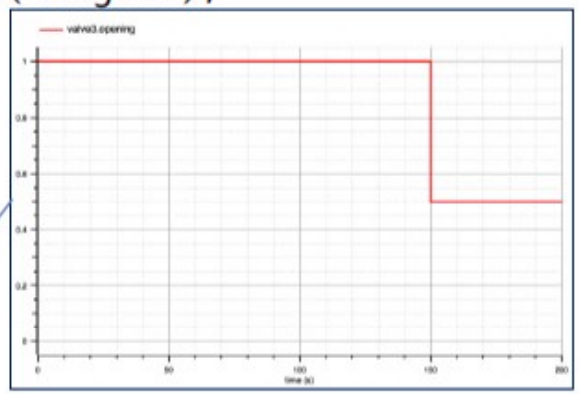
Network model of incompressible fluid consisting of 2 pressure sources, 11 pipes and 3 valves



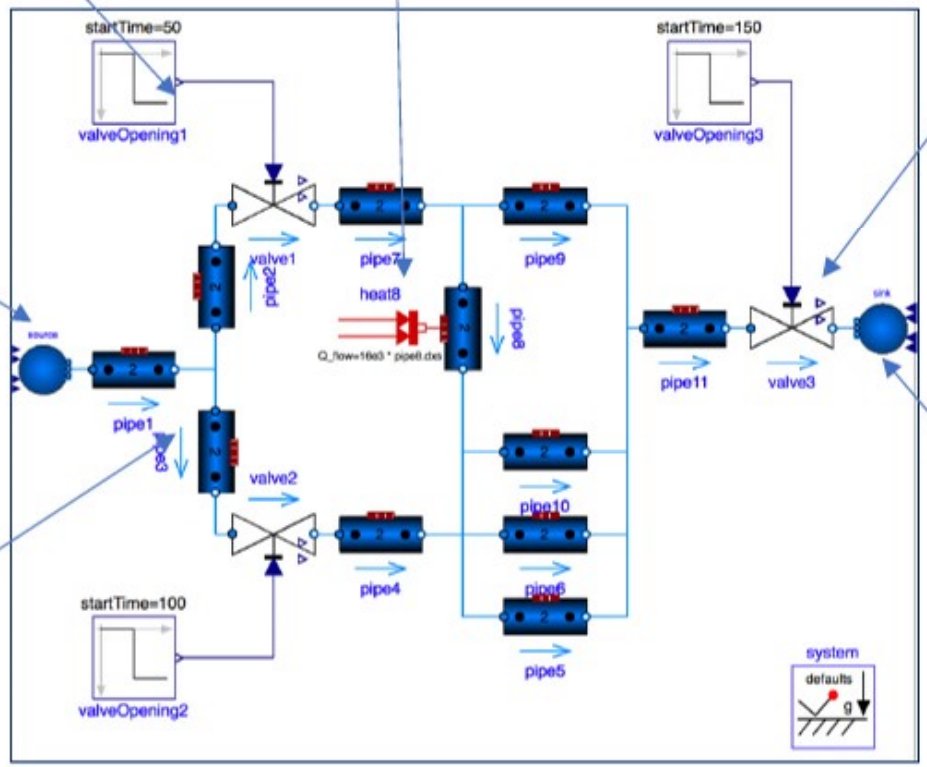
heat8
- 16e3*pipe8.dxs [W]

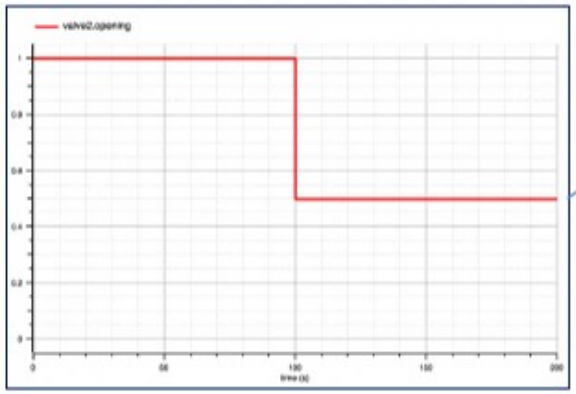
Working fluid
- Glycol47

```
final parameter Real[n] dxs = lengths/sum(lengths);
```



source
- 5.0e5 [Pa]
- 300 [K]

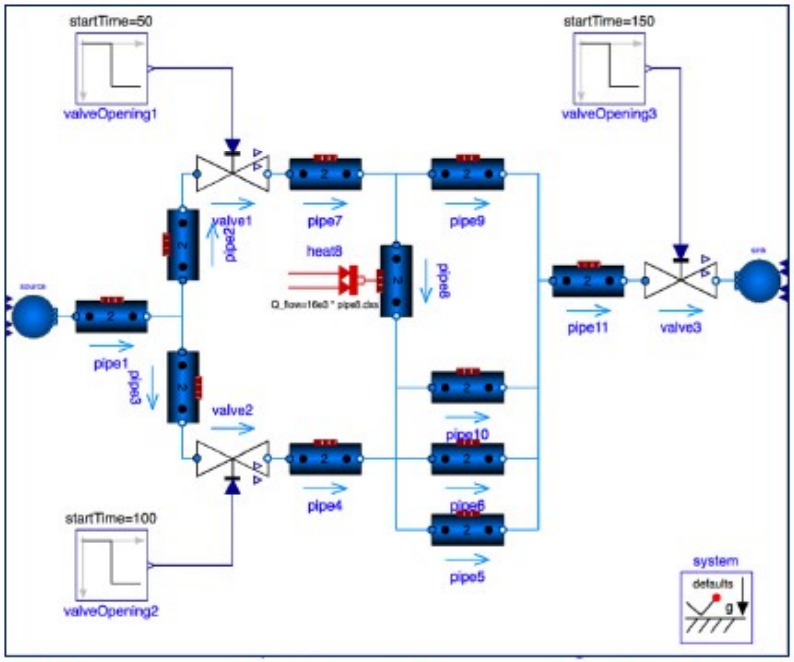




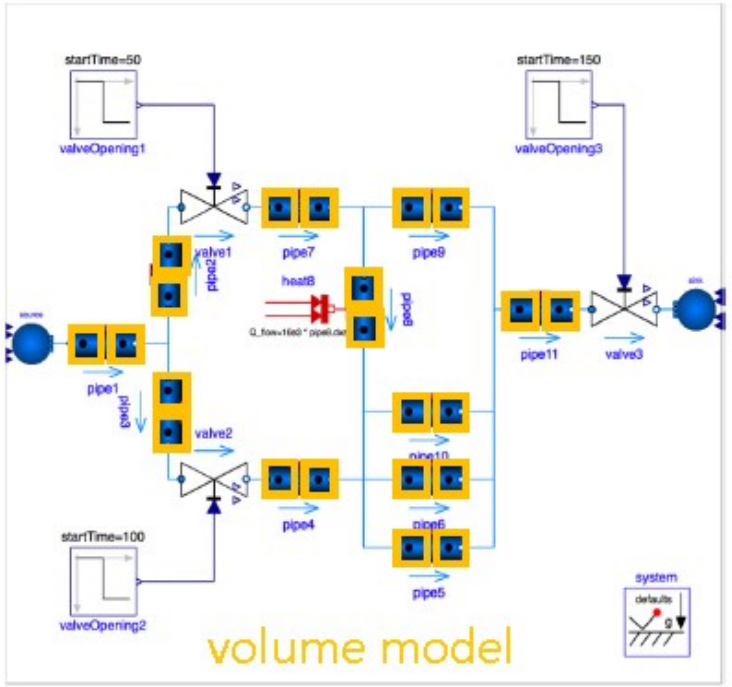
sink
- 1.0e5 [Pa]
- 300 [K]

IncompressibleFluidNetwork

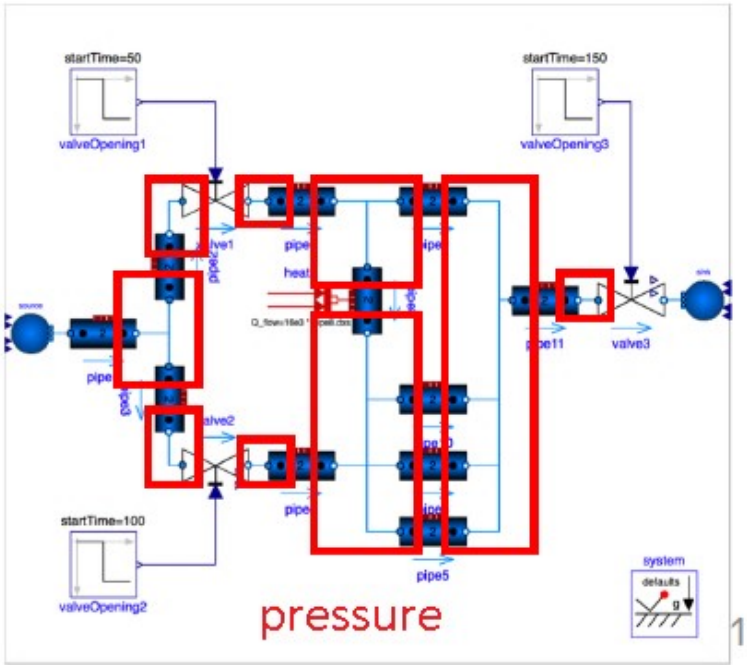
- The 11 DynamicPipes have nNodes = 2, and there are a total of 22 volume models (yellow squares on the right).
- Since the ModelStructure of DynamicPipe is av_vb, the volume model is directly connected between pipes, so the pressures are equal and there are only 9 states (red square in the lower right).



flow model (one for each pipe)

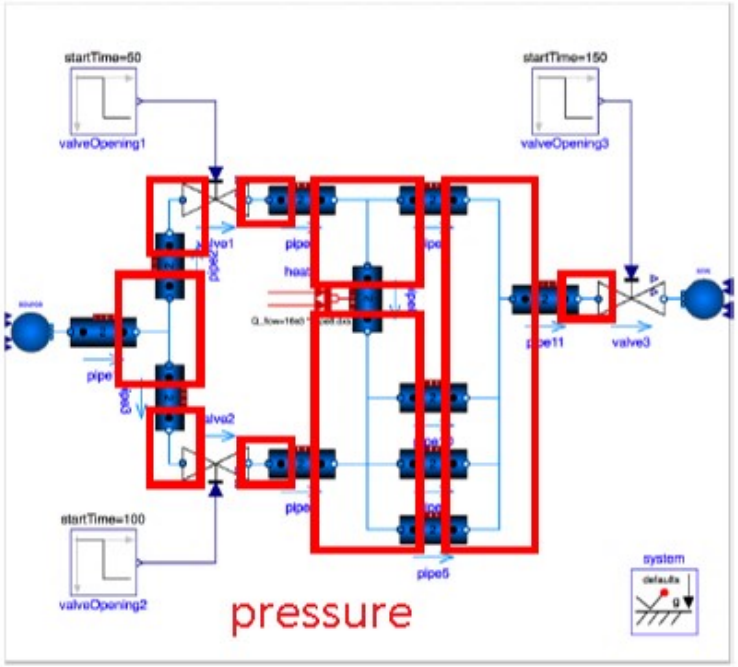


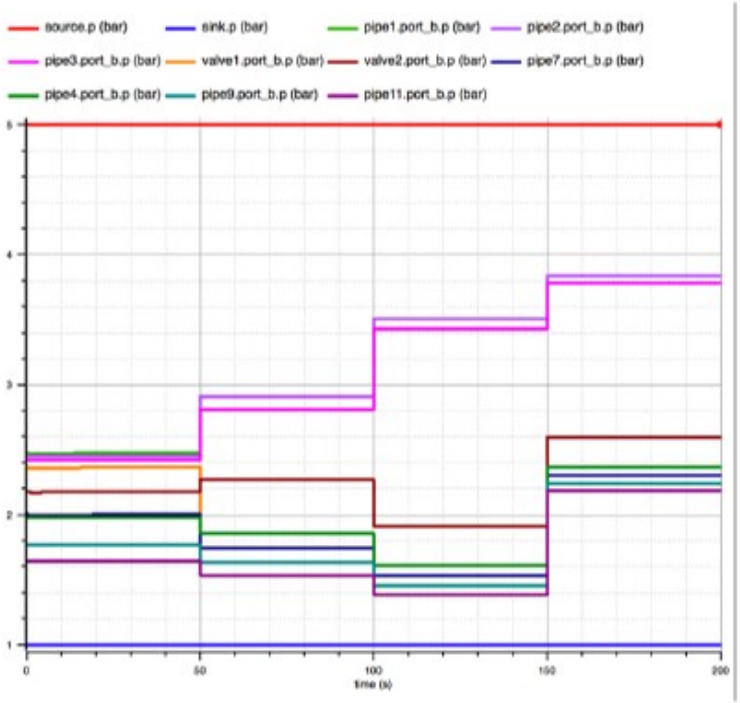
volume model

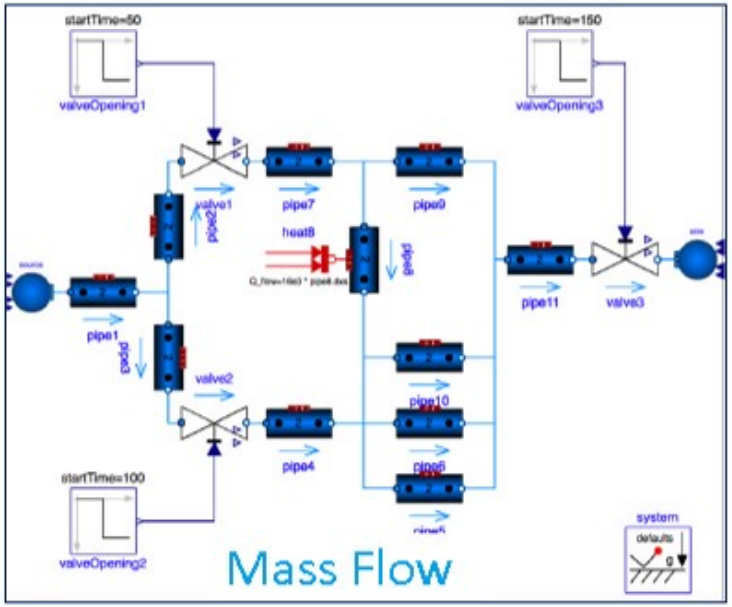


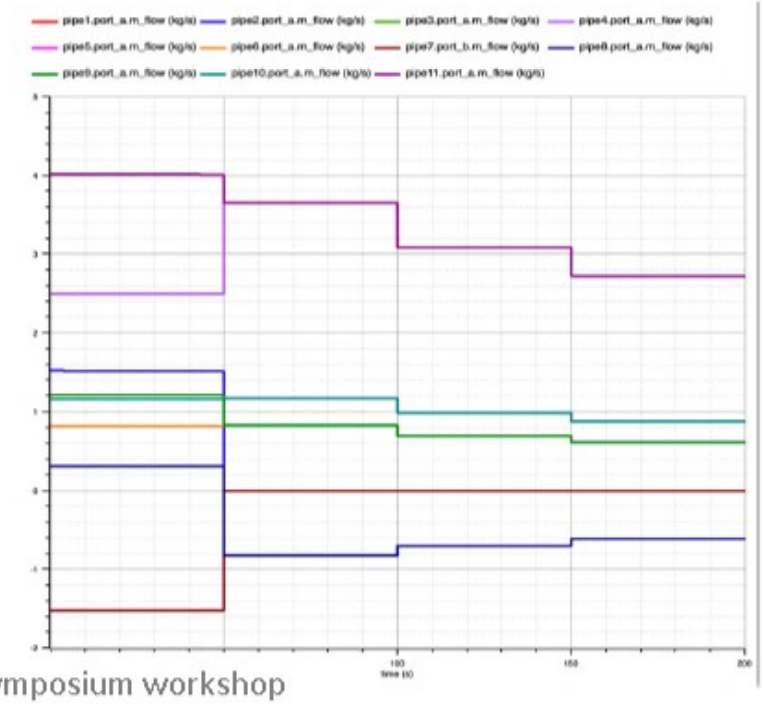
pressure

IncompressibleFluidNetwork

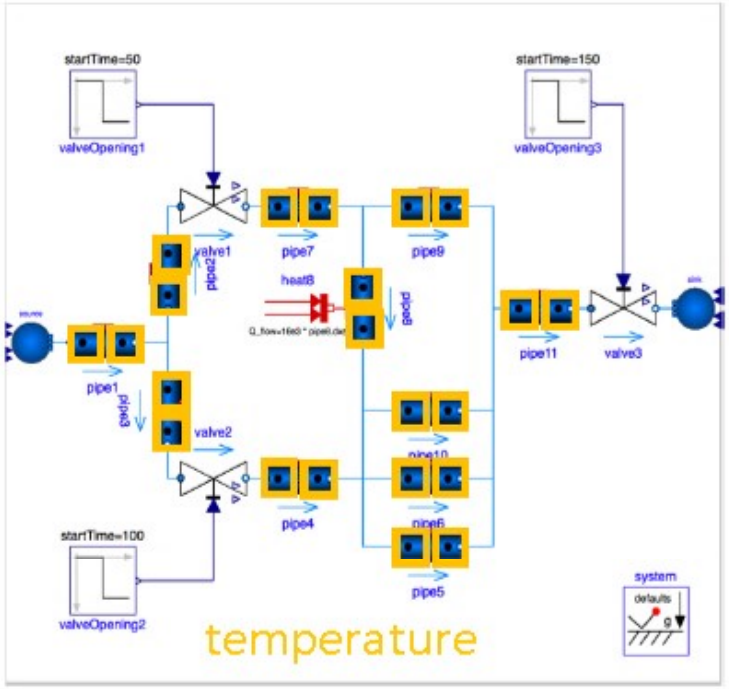


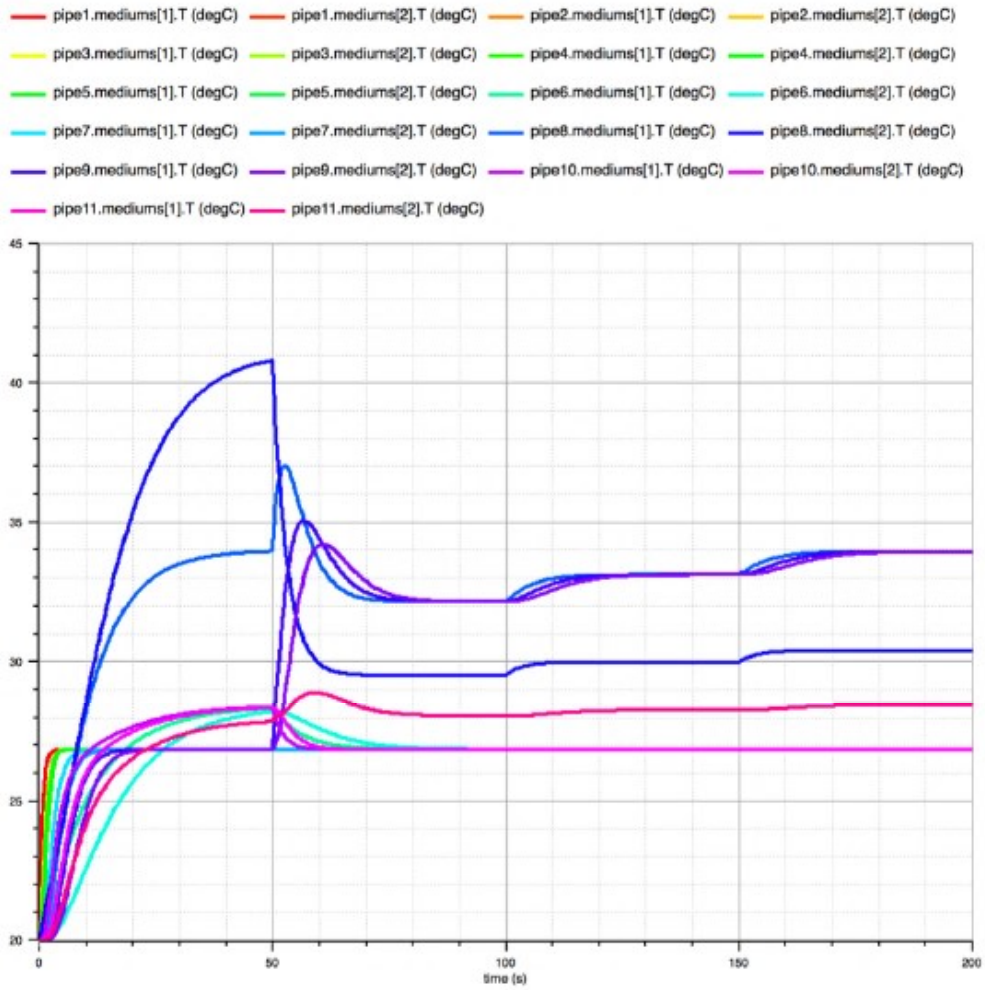






IncompressibleFluidNetwork





Conclusion

The concept and usage of the following contents of the Modelica.Fluid library are described using examples.

- FluidPort
- Volume model
- Flow model
- DynamicPipe
- VesselPortsData
- HeatTransfer model

- The purpose of this document is introducing Media and Fluid Libraries in the Modelica Standard Library (MSL). This document uses libraries, software, figures, and documents included in MSL and those modifications. Licenses and copyrights of those are written in next page.
- Copyright and License of this document are written in the last page.

Modelica Standard Library License

https://github.com/modelica/ModelicaStandardLibrary/blob/master/LICENSE