ISRN LUTMDN/TMHP-16/5369-SE

ISSN 0282-1990

Dynamic Simulation of District Heating Networks in Dymola

Rickard Hägg

Thesis for the degree of Master of Science in Engineering
Department of Energy Sciences
Faculty of Engineering | Lund University

Dynamic Simulation of District Heating Networks in Dymola

Rickard Hägg

June 2016, Lund

This degree project for the degree of Master of Science in Engineering has been conducted at the Department of Energy Sciences, Faculty of Engineering, Lund University, and at Modelon AB in Lund.

Supervisor at the Department of Energy Sciences was Senior Lecturer Patrick Lauenburg
Supervisor at Modelon AB was Stéphane Velut and Per-Ola Larsson
Examiner at Lund University was Assistant Professor Kerstin Sernhed.

Thesis for the Degree of Master of Science in Engineering

ISRN LUTMDN/TMHP-16/5369-SE

ISSN 0282-1990

Department of Energy Sciences
Faculty of Engineering, Lund University
Box 118, 221 00 Lund
Sweden

www.energy.lth.se

Abstract

To simulate district heating networks, in this project the simulation environment Dymola was used. The goal was to run simulation of district heating network and at the same time have good network representation to easily change between different models and also change between doing simulation and optimization. Different models had to be created in Dymola and tested to ensure they work accordingly. The method was to first implement new models for consumers, pipes and producers. The new models were first tested on a component level and later used to simulate larger networks.

There are existing models in the Modelica standard library that could be used to simulate networks although they are not suitable to simulate larger networks. The major part of the thesis was the implementation of a new pipe model. The pipe model in the Modelica standard library is based on the finite volume method and this is not suitable to use in large networks, since the method is unnecessary complex. The new pipe model is based on the spatialDistribution operator from Modelica standard library. The spatialDistribution operator is based on a plug-flow approach which is less complex. The simulations showed that the new pipe model had the wanted characteristics with a reduced complexity to make it possible to simulate larger networks.

A heat loss implementation was introduced for the new pipe. The implementation was simulated and compared to data from an existing district heating network. The results from the simulation showed that the simulated and measured data differed very little.

Pressure driven networks with loops, branches, several producers was possible to be simulated using the new components. The simulation time for a complex network with 100 consumers was below two minutes. At the current state 100 consumers is the highest amount of consumer models a simulation can handle, but improvements can be made.

The results showed that the new implemented models were able to simulate larger district heating networks that included both pressure and heat loss. There is also room to add more complexity into the models.

Acknowledgements

I would like to thank Gerald Schweiger introducing me to the idea of the thesis and all the assistance. Also thanks to Modelon AB for the resources and work space needed to complete the thesis. A special thanks to my supervisors at Modelon, Stéphane Velut and Per-Ola Larsson for their knowledge and aid throughout the project. Thanks to Patrick Lauenburg as well for sharing his expertise about district heating.

Nomenclature

Aggregation:	A method to reduce the complexity of a district heating network.
MSL:	Modelica standard library
SpatialDistribution:	Operator used to calculate the delay and heat loss in the pipe model.
Split-ratio	Ratio used to select how much of the pipe actual volume is handled by the spatialDistribution operator and the volume model.

C_p	Heat capacity
Q	Heat flow rate
$\dot{m}$	Mass flow
T	Delay
λ	Thermal conductivity
P	density

T_{supply}	Temperature in the supply pipe
T_{return}	Temperature in the return pipe

Contents

1. Introduction

To design the future district heating networks there is a need for dynamic modeling tools to optimize and simulate complex systems. This thesis is part of a larger project, which is about doing optimization on different aspect for district heating networks and use Model predictive controller (MPC). Performing optimization on district heating networks first requires the network to be simulated to get initial values, the values are then used to initiate the optimization. Simulation is then redone to ensure that the result from optimization is the actual optimal solution. The values from a simulation can also be used to aggregate networks. In order to start an optimization some initial values has to be set, the initial values comes from the simulation results.

The thesis focuses on the dynamic simulation and to create models for the district heating network. The goal is to get as accurate result as possible compared to reality but still be able to run simulations under a reasonable timespan. The models are created in the Modelica language, where the user can select what is of importance to simulate. The current simulation tools often use simplifications such as a fixed return temperature. However using Modelica models the user can choose what complexity is desired.

Larger district heating networks includes many different components and a need to simplify the networks might arise. This method is called aggregation and the basics is included in the thesis to get an overall view of the method.

2. Theory

2.1 District Heating

The purpose of district heating is to generate or utilize heated water and distribute the hot water to consumers at a cost lower than if it would be produced locally. The system can be divided into three subsystem; production, distribution and consumers. The heat sources in the production system can be different and include combined heat and power (CHP), waste heat from industries, heat pumps, geothermal or simple combustions of different kind of fuels. The heated water from the production system is transported through insulated pipes, these pipes are generally buried underground to the consumers. Once the water has been cooled down at the consumer the water is returned to the power plant through a parallel network of return pipes.

2.1.1 Consumer

Consumer buildings can be defined as buildings with heat demands such as offices, houses or public buildings. The heat demand can arise from a various heat loads such as domestic hot water, space heating or ground heating. These different loads have different characteristics; space heating for example is related to the outdoor temperature. For domestic hot water the temperature of the incoming cold water is almost constant throughout the year, hence this load is not as weather dependent (Frederiksen & Werner, 2013).

Hydraulic separation through a heat exchanger is used in the majority of consumer buildings using a heat exchanger. There are various reasons why this is often preferred. One reason is that if the water contains dissolved oxygen, which is normal in drinking water and non-treated water, then the oxygen will cause corrosion in the pipes, thus the water in district heating network is often chemical treated. Having hydraulic separation also usually reduce both the water and energy consumption (Frederiksen & Werner, 2013). The hydraulic separation is done through one or several heat exchangers with counter-current flow to maximize the efficiency of the exchanger. To control the flow inside the heat exchanger a valve with a control unit is used. The main advantages for the customer to use district heating as a heat source is the comfortable and reliable heat delivery, lower investment cost and less floor space for the customer's own heating equipment (Frederiksen & Werner, 2013).

2.1.2 Producer

The production unit in district heating networks can be heat stations, combined heat and power plants (CHP), heat pumps, geothermal, but it can also be waste heat from industries. The objective for the producer unit is to heat up the cold water from the return pipe to make sure it can be used in the consumer's substations. The producer can be connected to the electrical grid for example with a

heat pump, and use the surplus to store it in the district heating network. This process is fairly rare in the current situation.

2.2 Programming Languages

Programming language is a computer language designed to give instructions to a computer. These languages can be used to express algorithms or create programs. Programming languages are specialized in certain fields.

2.2.1 Modelica

Modelica is an open source, equation based programming language used to model complex systems for example mechanical, thermal, hydraulic or electric power. The language is developed by the Modelica Association which also develop the free Modelica Standard Library (MSL).

Equations in Modelica is acausal, meaning that the equations are not assignments, which is typical in most programming languages. Instead it describes equality. To give a simple example, the equation y= x+2 can be studied. A BasicExample-model is created in Dymola to demonstrate how an acausal program works. The equations behind this simple model can be seen in *F*igure *2.1*. The model calculates the value of *y.* To begin with, the model declares the variables *x* and *y* and below equation-line the equalities are described. In the majority of programming languages the value of y has to be declared before *x* can be calculated but in Modelica the declaration of *x* can be done even further down in the code and the program is still capable to solve the equation. Modelica can also solve *x* from this equation if *y* is known, something that most programming language cannot do.

```
model BasicExample
 Real x;
 Real y;

equation
  y = x + 2;
  x = 3;

end BasicExample;
```

Figure 2.1. A basic model example to show non-causality

A model can have different parameters that are modifiable, these parameters can be changed without the need to rewrite the model. These parameters are more than real numbers, they have physical quantities. It is possible to add SI-units related to the parameters. This makes it possible for Modelica to perform a unit-check or to do declarations more easily.

2.2.2 Python and NetworkX

Python is a general purpose programming language that is used in many applications. NetworkX is a Python package that is used to create and manipulate networks, such as doing aggregation or change between optimization and simulation models. For this project Python and NetworkX is used to aggregate network, represent networks and translate an existing network into Modelica code.

2.3 Simulation Tool

Dymola is a modeling and simulation environment for complex systems using the Modelica language. It is used for many applications such as aerospace, energy, robotics and other industries. In Dymola it's possible to simulate dynamic behavior and the interactions between different systems which enables the user to build integrated models that depict reality (Dymola, 2015).

To transfer information between Dymola models connections are done through ports. The ports have specific variables that has to be calculated or given for every model. These variables can be quantities such as mass flow into components, specific enthalpy of the incoming medium, pressure or other variables.

2.4 Numerical challenges

Nonlinear system often appears in nature and are difficult to solve mathematically and are generally approximated by linear equations and solved iteratively by Newton's algorithm using the linear equations. Compared to a linear system a nonlinear system is demanding for the computer to solve and should be avoided if possible.

When Dymola can't compute the derivative of an expression a numerical Jacobian is created. The Jacobian is used by the solver to solve the nonlinear systems. The matrix is defined as following:

$$J = \begin{pmatrix} \frac{df_1}{dx_1} & \cdots & \frac{df_1}{dn} \\ \vdots & \ddots & \vdots \\ \frac{df_m}{dx_1} & \cdots & \frac{df_m}{dx_n} \end{pmatrix}$$

These matrices are computationally expensive to compute and should be eliminated if possible. This is possible by providing Dymola with the expression for the derivate of the function within the model that give rise to the Jacobian matrix. Another possibility is to reduce the number of nonlinear system and then there is no need for a Jacobian (Claytex Services Limited, 2011).

2.3 Aggregation

District heating networks generally have a large amount of pipes, producers and consumers. This makes it unsuitable to simulate a large network in a dynamic simulation. A need emerges to aggregate this large network into a smaller network to be able to run the simulations in a small enough timescale. In order to reduce the complexity of district heating networks two separate methods have been developed. These methods are often called the Danish- and German-method. Using these method there is a possibility to aggregate the model down to a system with lower number of components but still retain a high accuracy of the simulation. Both methods are defined for steady state conditions but shown that they can still be used with good accuracy in dynamic contexts (Larsen, Bøhm, & Wigbels, A comparison of aggregated models for simulation, 2003).

2.3.1 Difference between the aggregation methods

Both the German and the Danish method have similar methods to aggregate but with a few important key differences. The German and the Danish methods conserve different kinds of quantitates but both methods keep the total water volume, heat load, mass flow and delays for the whole network after aggregation. However the Danish method does not consider any pressure drop but preserves the heat loss from both supply and return pipe. The German method can be used to calculate pressure drops however the heat loss from the pipes is not conserved. The German method has the ability to aggregate loops and several producers. (Larsen, Bøhm, & Wigbels, A comparison of aggregated models for simulation, 2003). In optimization it is important to reduce the number of component to as few as possible and the chosen method used for aggregation is based on the layout of the network. Simulation can handle more components and the need to aggregate loops and producers is not as great and the method chosen is more depending on what is important for the simulation results. In case the pressure is of importance then the German method should be used and if heat loss is of more importance then the Danish method should be considered.

Aggregation of branches

The basic method of aggregate branches into a line structure is illustrated in Figure 2.2. This method is used for both the Danish method and the German method, however the conservation of different quantities are depending on the method (as discussed in the previous chapter). This is done to be able to do an aggregation of serial consumers.

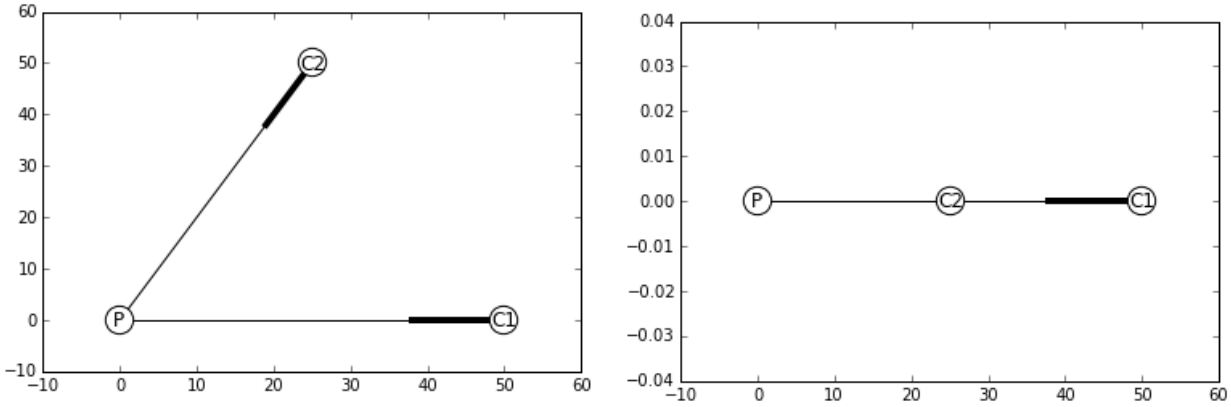

Figure 2.2. The basic principle of aggregating a branch into a serial. The left diagram is a simple network created in NetworkX with two consumers before aggregation and the right one is the aggregated network.

Aggregation of serial consumers

Aggregation of consumers connected in series differs for the chosen method. The Danish method locates four consumer connected in series and aggregate the two middle ones into a new consumer as illustrated by Figure 2.3 with updated information about the load and parameters of the consumers and pipes. This process can be done iteratively together with branching to reduce the number of consumer and pipes to the desired amount.

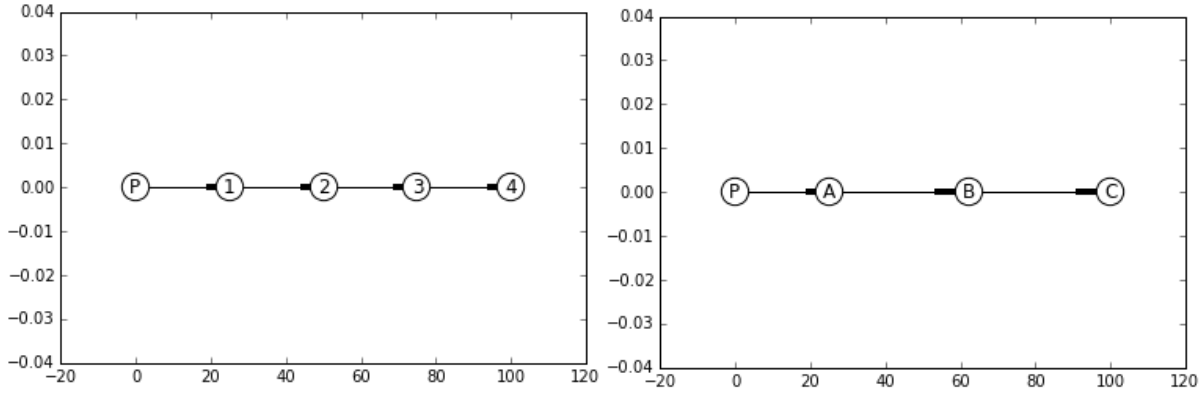

Figure 2.3. The Danish method of aggregating a number of serial consumers into a fewer number of consumers. A producer (heat plant) and four consumers in a serial as can be seen to left, are turned into three consumers.

The German method however locates three consumers and aggregate the middle one into the other two consumers. A picture of the principle can be seen in Figure 2.4. Same as for the Danish method the German method can together with branching reduce the number of consumers until the wanted number of consumers have been reached.

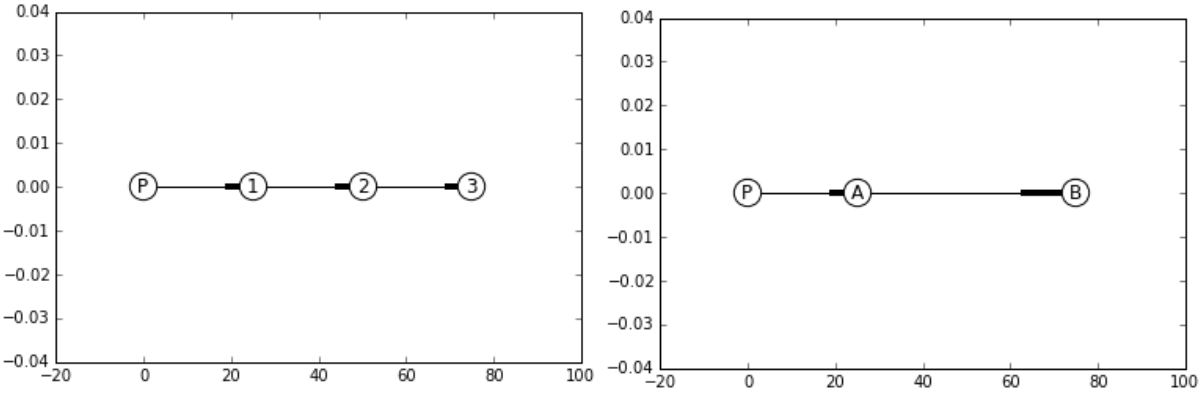

Figure 2.4. The German method have a different approach and aggregates a serial of three consumers and divide the middle consumer load and mass flow into the consumer next to it.

3. Method

The first step is to take a closer look on what models currently are used to simulate district heating networks and improve the existing models or create new models to simulate networks. Simulations have to be done on these updated or new models to verify they maintain the wanted characteristics for the specific component but still keep the complexity at a reasonable level and at the same time ensure it is possible to simulate larger district heating networks.

After creating these new components they have been tested in simulations of larger network. To ensure that the models also is able to simulate larger networks and not only work on component level. It is also of importance to study how big of a network the new models are able to simulate to get an understanding of how complex network can be simulated and if the complexity has to be reduced.

3.1 Component models

The mass flow of the water in district heating networks is driven by differential pressure of the network. Pressure boundary is an existing component that is used to set the pressure at different points of the network. From the pressure differences and a pressure loss model, Dymola is able to calculate the mass flow corresponding to the pressure difference. These pressure boundaries can have fixed values for the pressure or change them during a simulation. The pressure boundaries are used to pressure the system and has the same function as the pump in a real world case. The pressure boundaries are also used to give the temperature of the outgoing medium if the flow is going out of the boundary.

Another possibility is using mass flow sources that use a given mass flow at various locations of the network. From the mass flows Dymola compute the pressure needed to get the given mass flow. The disadvantage to use mass flow source is that instead of having a pressure driven system where the valves controls the flow the flow is given by the mass flow sources. Using mass flow sources is also often more time consuming for the user to use since there is a need to sum up the mass flow of each consumer correlated to the load profiles and the heat flow rate of the heat exchanger.

Components are created to be able to integrate with both Modelon Base library and the Modelica Standard library. There is no difference between the models created for this study except the fact that they have different ports and the difference of components taken from the different libraries.

At the start of a simulation the nodes has to be given initial values for dynamic states, such as pressure and temperature. If the initialization is done poorly there is a probability that there would be large transients of mass flow inside the network. To avoid this, all components of a larger network have the same fixed initial values for the pressure and temperature. The pressure is slowly increased in one of the boundaries to get the desired differential pressure.

In modeling and mathematics division by zero should be avoided. In many models used for simulation, division by mass flow occurs. For example the travel time of the water through a pipe is given by:

$$\tau = \frac{(r^2 \cdot \pi \cdot L \cdot \rho)}{\dot{m}}$$

This equation would break the model if the mass flow of the water is zero or close to zero. To avoid this problem a leakage is introduced to the components. The leakage is a fixed mass flow when the

flow goes towards zero. This leakage however is contained to that model and is not transferred to the next model. Meaning if the mass flow was zero into the model it is still zero on the other end.

3.1.1 Pipe models

Most implementations to calculate the flow through a pipe are based on discretization of the pipe into smaller segments and by solving the partial differential equations for each segment. This method is called the finite volume. In district heating the important quantities are the mass flow, pressure and temperature and the finite volume method is an unnecessary complex method to get accurate results for said quantities. To simulate large scale district heating networks a pipe based on finite volume would lead to time consuming computations. The pipe in MSL is based on this method and in order to simulate larger district heating networks a new pipe-model would have to to be implemented.

Flow resistance

The mass flow inside a pipe is dependent on the pressure difference at any segment and can be described for one dimension by coupled partial differential equations 3.1 and equation 3.2

$$\frac{dq}{dt} + \frac{A}{\rho}\frac{dp}{dx} + f(q) = 0 \qquad (3.1)$$

$$\frac{dp}{dt} + \frac{c^2\rho}{A}\frac{dq}{dx} = 0 \qquad (3.2)$$

Where p is the pressure, ρ is the medium density, the speed of sound is c and q is the flow rate. Here $f(q)$ is called the friction factor and describes the pressure losses per unit of length and depends of the viscosity of the medium and in what flow regime the pipe is in (Velut & Tummescheit, 2011).

To calculate the mass flow for a pipe for a certain differential pressure a flow resistance function is used from the Modelica standard library. The function is based on the equations 3.1 and 3.2 and it is already regularized for when the mass flow is zero or close to zero.

As discussed in chapter 3.1 pressure boundaries are often used, meaning that the mass flow is calculated depending on the differential pressure.

Heat front delay and heat loss

Calculating the temperature for one dimension in a district heating pipe is often a good approximation. The energy balance for the pipe in one dimension can be described by the partial differential equation 3.3. In equation 3.3 $C_p$is the heat capacity, Q is the heat flow rate, $\dot{m}$ is the mass flow, ρ is the density, r is the radius of the pipe and T is the temperature.

$$\frac{dT}{dt} + \frac{\dot{m}}{\pi r^2 \rho}\frac{dT}{dx}\frac{1}{\pi r^2 C_p} q\big(T(x)\big) = 0 \qquad (3.3)$$

The heat loss to the surroundings is given by $q\big(T(x)\big)$ and depends on the ambient temperature and is often described as a linear function that can be seen in equation 3.4.

$$q\big(T(x)\big) = kS(T(x) - T_{ambient}) \qquad (3.4)$$

Here *k* is heat conductivity of the insulation of the pipe and the surrounding ground. S is the shape factor for that pipe

If the mass flow is positive in equation 3.3 it can be solved and the temperature out of the pipe is given by 3.5, where τ is the time the water spend inside the pipe and τ_p is given by equation 3.6.

$$T(L,t) = T_{ambient} + (T_{in}(t-\tau) - T_{ambient})e^{-\frac{\tau}{\tau_p}} \qquad (3.5)$$

$$\tau_p = \frac{c_p \pi \rho R^2}{kS} \qquad (3.6)$$

Using equation 3.5, 3.6 and a specific T_{in} it´s possible to calculate the temperature in the pipe depending on what the mass flow is inside the pipe and the pipe characteristics (Velut & Tummescheit, 2011). Using a shape factor it is possible to take the thermal transmission coefficient into account for the thermal insulation. Using the insulation thickness instead τ_p is given by equation 3.7. Here R_2 is the radius of the insulation and λ is the thermal transmission coefficient.

$$\tau_p = \frac{c_p \pi \rho R^2}{2\lambda\pi \log {\frac{R}{R_2}}\Big/_R} \qquad (3.7)$$

The heat loss does not happen instantaneously inside the pipe and there is also a travel time for the water inside the pipe. One way to keep track on the time the water spent inside the pipes and reduce

the number of state variables is to use the build-in operator spatialDistribution in Modelica. The operator keeps track on the spatial distribution with a suitable interpolation, sampling and shifting of the stored distribution (Modelica, 2012).

The operator has the following call sequence in Dymola:

(out0, out1) = spatialDistribution (in0, in1, x, Boolean, initialPoints = {0.0, 1.0}, initialValues = {0.0, 0.0});

Here in0 and in1 is the transported quantity, x is the primitive function of the velocity of the quantity inside the pipe. The Boolean is a true or false statement whether the quantity goes in the intended direction or reversed. Initial points and values is used to set a start value inside the operator for said quantity.

For the new pipe model two separate spatialDistribution operators are used. The first operator delays the temperature to ensure that the heat is not instantaneously transferred through the pipe. The second operator use time as the transported quantity and the result from it is used to calculate τ which is the time the water spends inside the pipe and together with equation 3.5 is used to calculate the temperature of the outgoing water temperature. Calculating τ using the spatial operator in Dymola is done by the following:

(time_a, time_b) = spatialDistribution (time, time, x/length, noEvent (velocity>=0), {0.0, 1.0}, {0.0, 0.0]);
tau = noEvent (if velocity>=0 then time – time_b else time – time_a);

Time is used as a transported quantitate and depending on the direction of the flow τ is calculated from *time_a* or *time_b.* The variable *time* in Dymola is reserved and is the current time of the simulation. If *time* and time_a or time_b is compared it is possible to calculate the time the water spent inside the pipe. The noEvent operator is used to turn off event triggering Events is used to in Modelica to handle discontinuities. By turning off event triggering the models improve performance wise however there are instances where the result is inaccurate. The accuracy loss is also small since it also affects a small timeframe of the simulation.

The pipe is put together using a model with two separate spatialDistribution operators and the flow resistance model. Figure of the pipe can be seen in Figure 3.1.

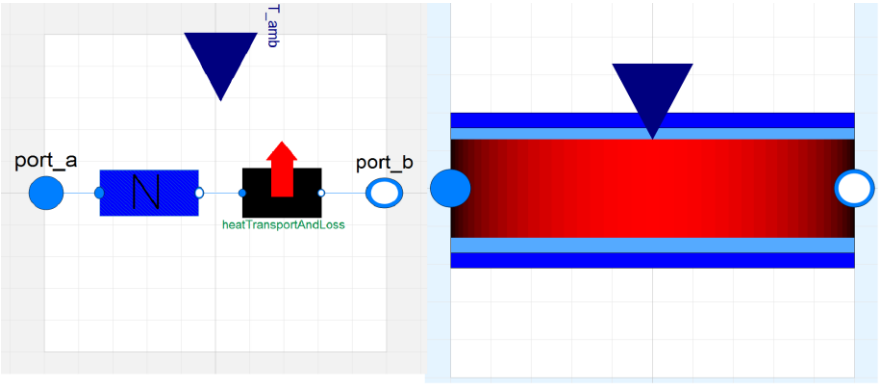

Figure 3.1. Pipe based on the spatialDistribution operator.

The left picture show the components inside the pipe, the blue rectangular model implements mass and momentum balance. The right rectangular is using the spatial distribution operators to calculate the heat delay and heat loss. The top component is used to feed in the ambient temperature. The right picture show the icon for the pipe in Dymola.

Volume models

In Dymola a direct connection between models relating pressure and mass flow, so called flow resistance models, is something that should be avoided. When Dymola tries to solve the system for that connection the pressure between the models is unknown and the solver has to iterate over it for every step. For a large system with many pipes in series and parallel this could lead to large nonlinear systems which is computational expensive. A way to work around the problem is to introduce pressure states that calculates the pressure for that connection point between the models. In this case a volume model is connected between the flow resistance models which introduce a pressure state. The volume model is a simple tank with a fixed size and where the water is ideally mixed. As a bonus the result of the model is more physically representative because there is some mixing occurring of the water inside a pipe and a plug flow approach is not as realistic. These volume models are taken from the MSL or Modelon base library.

If the volume model would be added to the pipe and without altering the spatial operator, the delay of water going through the pipe would become too high. To solve this problem the total volume of the pipe is split between the volume model and the spatial operator that calculates the delay. The split only effect one of the spatial operator inside the pipe to ensure that the heat loss would still be the same but the delay is corrected. The ratio called the split-ratio is set by the user and a split-ratio

of 0.9 means 90 percent of the pipe volume is handled by the spatial operator and the rest is used as a volume model. The second pipe model can be seen in Figure 3.2

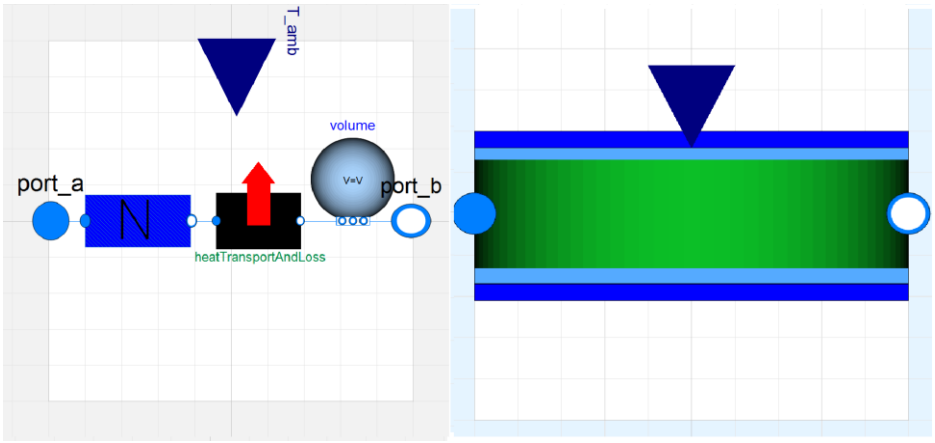

Figure 3.2. Second pipe based on the spatialDistribution operator. The pipe is similar to the first pipe but with an added volume model.

If several pipes would be connected to the right port the flow resistance models of the connected pipes would be connected to each other and a system of nonlinear equations would be created. In order to remove these nonlinear equations a third pipe model has to be implemented, the third pipes creates a direct connection into the volume model for every new connected pipe to avoid that flow resistance models are connected to each other. The pipe with the new connector can be seen in Figure 3.3.

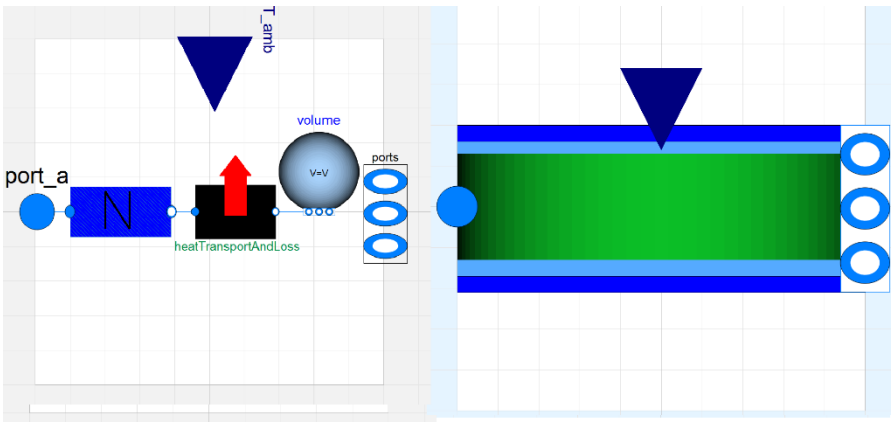

Figure 3.3. Third pipe is the same as the second pipe except the right port is changed making sure it can handle several connections to the same pipe.

Simulations are done to show the effect on the nonlinear system by adding a volume model. The first is done with two pipes in series without a volume model between them. The setup and result from the first simulation can be seen in Figure 3.4. The statistics of the model can be seen to the right. The focus should be on the bottom two lines where it is noticeable for this simulation that a nonlinear system is created and that a numerical Jacobian is derived to solve it.

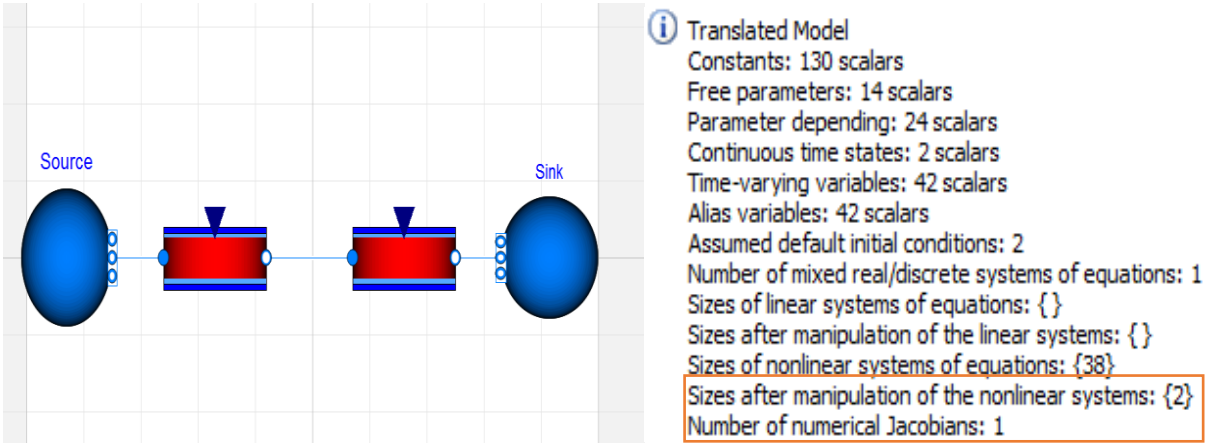

Figure 3.4. A simulation of two pipes with flow resistance models connected to each other and the result for that translated model.

The simulation is re-done adding a volume model between the pipes. The updated model and result is shown in Figure 3.5. For this simulation Dymola is able to remove all nonlinear equations and instead a linear system is created and there is no need for a numerical Jacobian and it's easier for the solver to handle this system than the previous.

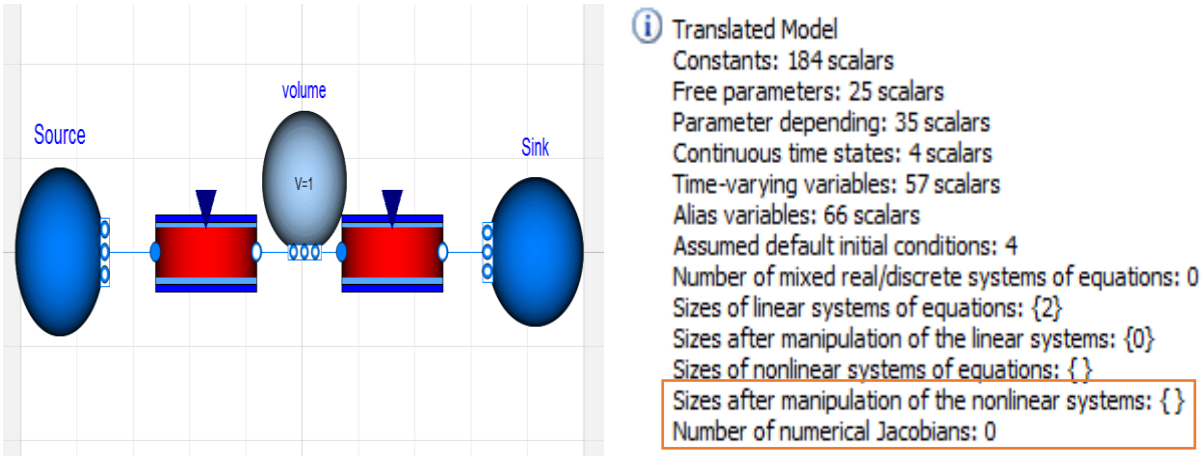

Figure 3.5. The updated simulation layout with a volume model between the pipes and the result from that translated model.

3.1.2 Consumer models

A consumer in district heating networks consist of three major parts: a valve, a heat exchanger and a controller. The controller together with the valve regulate the mass flow into the consumer to ensure that the heat demand is met. The maximal mass flow into the consumer is decided by the diameter of the pipe that it is connected to the network and the differential pressure between the supply and return pipes for the location of the consumer. The heat flow rate into the consumer corresponding to a certain mass flow is also dependent of the temperature difference of the water before and after the heat exchanger. The heat flow rate is given by equation 3.8.

$$Q = \dot{m} \cdot c_p \cdot (T_{supply} - T_{return}) \qquad (3.8)$$

Depending on the desired complexity and the desired way to control the heat flow rate four different basic Dymola consumer models were created. The basics models could be used to create various other consumer models by combining the different models.

Heat exchanger

To solve equation 3.8 both T_{supply} and T_{return} has to either be known or given ways to be calculated. For the first two consumer models T_{return} is calculated by introducing a heat exchanger model where the incoming cold water to the heat exchanger of the consumer is known and gives a correlation between the return temperature and the temperature of the cold water of the consumer. The temperature of the incoming cold water is highly dependent on the load profile of the consumer. Water used for heating is often warmer than water used as tap water. The third consumer model use a heat exchanger with a fixed return temperature independent of both mass flow and type of load. The fourth consumer is introduced to improve the simulation time

The simulation focuses on district heating networks and the heat exchanger model used in the first two consumer models is simplified to reduce the complexity and instead of counter-current flow the heat transfer in the heat exchanger is based around the same principle as the heat loss of the pipe. The delay of the water inside heat exchanger is much lower than the delay in the pipes and therefore the time spent inside is only being used to calculate the heat transfer and it happens instantaneous and then there is no need to use the spatialDistribution operator to keep track on the time delay τ.

The ambient temperature in the equation is the incoming cold water of the consumer. This assumption is the same as having a pipe inside the heat exchanger straighten out and surrounded by a reservoir of cold water that is not effected by the heat flow rate.

The reason the first two models use this type of heat exchanger instead of having a fixed return temperature is have an interaction between the mass flow and the return temperature. A low mass flow inside a heat exchanger would lead to a temperature close the consumers cold water temperature and a high mass would give a higher return temperature. If more complexity is desired it is possible to do a complex simulation of a house and use the result to get the return temperature. Considering the simulations are about district heating networks the complexity is often unnecessary when performing simulations of networks.

Valves

The valve models inside the consumer models is from the Modelica standard library or Modelon base library and are used to control the mass flow into a consumer model and the pressure drop over the consumer model. Instead of having a pipe dimension determine the highest mass flow into the consumer the valve is given nominal values for the mass flow and pressure drop. The nominal values together with the control signal determine the highest mass flow into the consumer model. The user is then able to control the opening of this valve where zero is fully closed and one is a fully opened valve. This can also be done with a controller in the consumer model. If a split between the different load for heating or tap water is desired there is possible to have two consumer models connected next to each other with the different characteristics for the incoming cold water of the heat exchanger or the return temperature.

Valve controlled consumer

The first consumer model consist of a heat exchanger, and a valve however without controller and can be seen in Figure 3.6. The user directly controls the opening of the valve in this consumer model. The control signal into the valve can have a value between zero and one, where zero is a fully closed valve and one is a fully opened. Since the highest mass flow into the consumer substation is set by the valve the user have a rough control of the flow if the differential pressure over the consumer substation is sufficient.

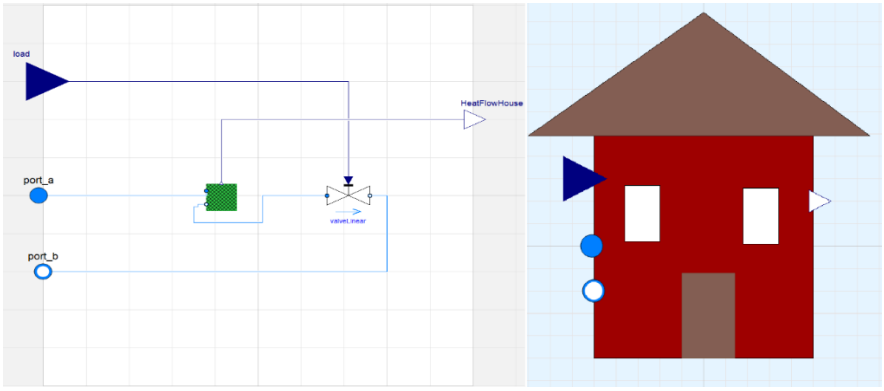

Figure 3.6. The left picture show the valve controlled consumer model, the green block is a heat exchanger that is connected to the valve that controls the mass flow into the consumer. The right picture show the icon used for the consumer model in Dymola.

PI-controlled consumer

The second consumer model is similar to the valve controlled consumer although it has a proportional–integral controller (PI-controller) to control the opening of the valve. This consumer model can be seen in Figure 3.7. The PI-controller is a feedback mechanism where the controller calculates an error based on the desired value and the actual value. The proportional part looks at the present value of the error and if the error is large then the control output is large as well. The integral part integrates the past values of the error and if the current output is not big enough to decrease the error the controller will increase the control signal. Usually this kind of controller also have a derivate part and becomes a PID-controller. The derivative part controls the future values of the error based on the current rate of change. In this case a PI-controller should suffice considering fast changes in load, but these fast changes are not so usual in district heating networks if the network is aggregated. A non-aggregated network might have swift shift in load for example if a water tap is opened and no accumulator tank exists and the controller can then easily be upgraded to a PID-controller.

In this consumer model the PI-controller is calculating the error value based on the heat flow rate calculated by the heat exchanger and the heat demand given by the user. The PI-controller gives a control signal to the valve which controls the mass flow into the consumer trying to minimize the error. The control signal from the PI-controller is restrained to only have a value between zero and one, where one is a fully open valve and zero is a fully closed.

The heat exchanger model is regularized for zero flow meaning the model has a small mass flow even when the mass flow is close to zero. The consumer model still have an outtake from the network even when there is no mass flow going into the model, however this is extremely small. The PI-controller, however, still tries to adjust the opening of the valve but the flow does not change since a fixed mass flow is used. This can lead to unnecessary long simulation time. To handle this problem a minimum function model was connected from the load to the PI-controller to ensure that small load is always used. This should usually be no problem since the load of a consumer is rarely zero and for an aggregated network the risk would decrease even more. The input to this consumer model is the load in kilowatts. These consumer model assure that the heat demand is met but it is slightly more demanding to simulate than the previous model.

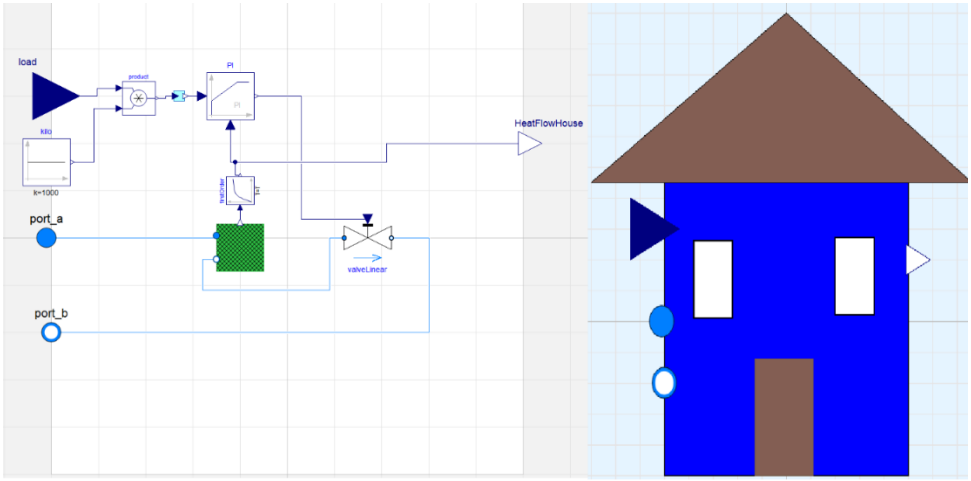

Figure 3.7. Picture of the PI-controlled consumer.

To the left in Figure 3.7 show the different components inside the consumer and the right is the icon used in Dymola. The difference from the first consumer model is that the mass flow is controlled by a PI-controller instead of the user. The product- and kilo-block is used to have the load input as kilowatts instead of watts. The heat exchanger and valve is the same as the valve controlled consumer. The first order block is used to break the algebraic loop.

Between the heat exchanger and the PI-controller there is linear filter block. This is used to break an algebraic loop inside the consumer model. Algebraic loops occur when the input of the block is driven by the output of that same block, through a feedback path or direct feedback. Algebraic loops are difficult to solve mathematically but there are solvers that can handle these kinds of loops. However it's possible to break loops by adding dynamics such as delays or a filter block. This blocks defines the transfer function between the input u and the output y as first order system as following:

$$y = \frac{k}{T \cdot s + 1} \cdot u$$

For this block inside the consumer model the gain *k* is set to 1, *T* is the time constant, and *s* is from Laplace. *T* is controlled by the user. The algebraic loop in the consumer model occurs because the control signal from the PI-controller is depending on the heat flow rate from the heat exchanger, the heat flow rate from the heat exchanger is connected to the mass flow and the mass flow is controlled by the valve which is controlled by the output signal of the controller.

Fixed return temperature consumer

The third consumer model is similar to the PI-controlled consumer except the heat exchanger calculates the heat flow rate using a fixed return temperature which is set by the user. The heat exchanger have a reduced number of equations and can be used if the return temperature is known for the network or the return temperature is independent of the mass flow into the consumer substation. In Figure 3.8 it is possible to see the different components inside the model.

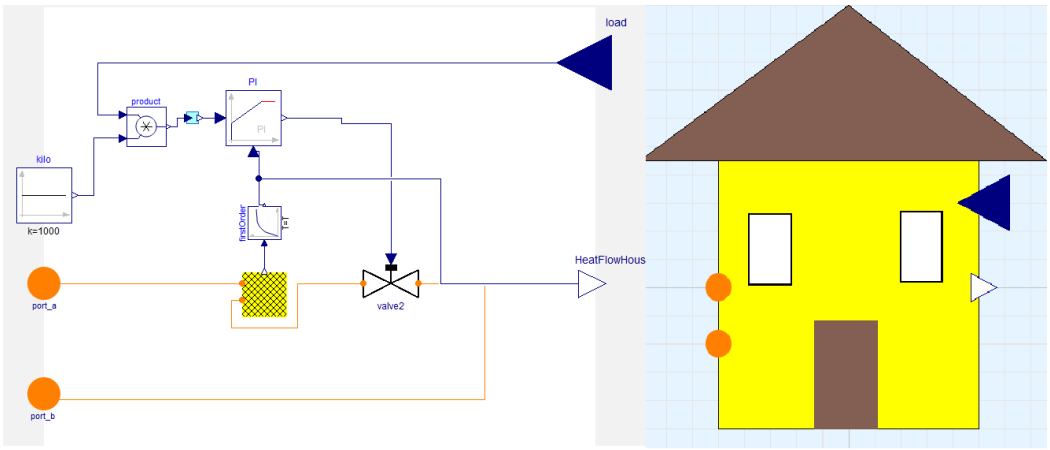

Figure 3.8. Inside the fixed return temperature consumer.

Ideal Consumer

Lastly a fourth consumer model was created, with a fixed heat flow rate but instead of a PI-controller another signal source was used. This is possible to use, since the highest mass flow into a consumer substation is known and the heat exchanger gives a heat flow which makes it's possible to calculate the opening of the valve without the need for a PI-controller. This controller is used mainly to improve the simulation time. The model can be seen in Figure 3.9.

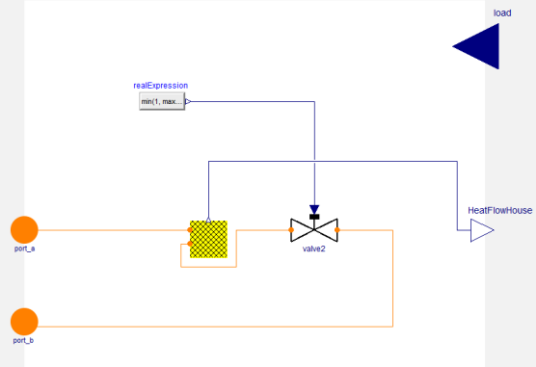

Figure 3.9 Consumer model without a PI-controller

3.1.3 Producer

Producers in district heating networks heat up water from the return pipe and feed it out in the supply pipe with the desired temperature. The model use pressure boundaries to control the pressure of the supply pipe and the pressure at the return pipe. Because the water is flowing out of the top pressure boundary this component also sets the temperature of the outgoing water. The temperature of the high pressured boundary is set to be equal of the incoming cold water temperature, which is given by the temperature of the return water. This is done to mimic a closed system where the temperature into the heater is the same as the temperature of the water in the return pipe. A heater is used to increase the water temperature to the requested temperature. The heater can be controlled by a signal from zero to one, where one is the heater running on full power, and the full power is given by the user. To get the requested supply temperature a PI-controller is used to match the outgoing temperature to the requested temperature. The basic principle of a producer can be seen in Figure 3.10.

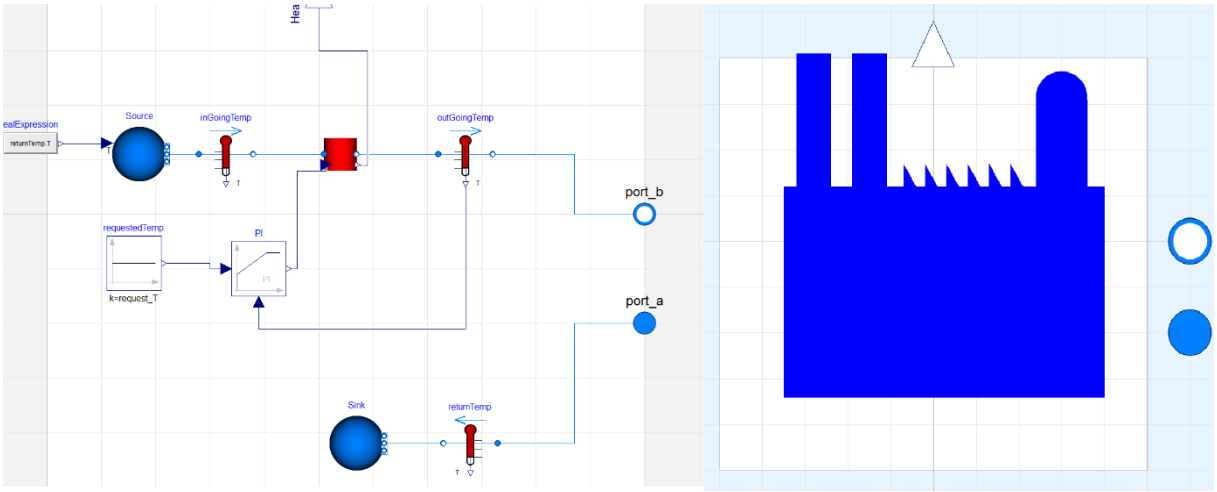

Figure 3.10. Inside the pressure producer component. The blue boundaries are used to set the temperature and pressure of the water. The red block is a simple heater that is controlled by the PI-controller.

A producer model can become very complex depending on the type. This model is created to have a base production unit to use in network representation. The producer model can later be upgraded by adding accumulator tanks, make restrictions how fast the plant can start or shut down, add cost models for fuel to simulate the revenue, integrate it with the electrical grid etc.

3.3 Network representation

A network created in Python using NetworkX has to be translated to Modelica in order to be able to run a simulation. The network can originally be created in Python using either scripts that read information about a network from different files or created directly in Python with NetworkX. The aggregation is also done using Python and to run simulation for an aggregated network it has to be translated into Modelica code. The work was based around previous work of Henning Larsson at Modelon (Larsson, 2015), but had to be updated because the last models aimed at optimization and was not using pressure dependent models.

A simple network was created using NetworkX and Henning's code to show an example how a translation is done. For this network, the pipes, consumers and producers are given parameters that translates into Modelica. NetworkX uses nodes and edges (lines) where consumers and producers are represented by nodes and pipes are represented by edges. Because edges only can be between two nodes splitting of pipes are impossible if there is no consumer node at that location. In order to split pipes an intersection node is created. This node is later removed when running a simulation in Dymola and is only used to connect pipes easier. In the example below node D is an intersection node. The example network can be seen in Figure 3.11.

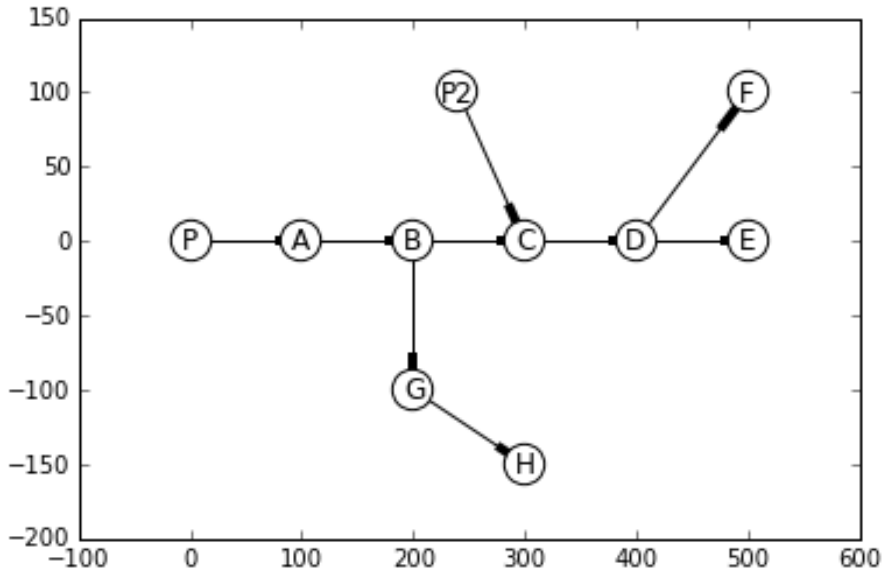

Figure 3.11. A graphical representation of a district heating network using Python and NetworkX

Three major parts have to be done for the translation, node translation, edge translation and connection between the nodes and pipes. The script first has to select which consumer-, pipe- and producer-model should be used and this was done by the following sequence:

```
##### Write code to file #####
f.write("model {}\n\n".format(model_name))
f.write(self.package_definitions)
f.write("  replaceable model Pipe = DHNLibrary.DistrictHeating.ComponentsModelon.Pipes.Dual.ScalingVolume.SpatialScalingVector; \n")
f.write("  replaceable model Consumer =  DHNLibrary.DistrictHeating.ComponentsModelon.Consumer.HouseLoadParameter; \n")
f.write("  replaceable model Producer =  DHNLibrary.DistrictHeating.ComponentsModelon.HeatStation.HeatStation2; \n \n")
f.write("  replaceable package Medium = Modelon.Media.PreDefined.TwoPhase.WaterIF97; \n \n")
for decl in self.decls:
    f.write(decl)
```

Here it's easy to change what model should be used for the different components.

The script translated the nodes and edges into consumer, producer or pipe models. It is important to ensure the parameters that was given in NetworkX are being transferred into Dymola. These parameter can be such things as pipe length, diameter or maximal mass flow into a consumer model.

To get an easy translation, a new dual pipe model was created. The dual pipe model is basically two separate pipes in the same component, one for supply and one for return. The two pipe does not effect each other in any way.

When writing the translation scrip it is important to make sure there are no volume models connected to each other and no direct connection between flow resistance models. This is done by checking the in- and out-degree of a node. The in-degree of a consumer is how many pipes are connected into that node and the out-degree is the amount of edges going out of a consumer. If the out-degree is higher than one for the node which a pipe is connected to, the pipe has to be connected to both the consumer model and the pipe connected to the next consumers. In Figure 3.13 the basic behind the translation is shown.

```
if net.node[u]["type"]=="producer":
        conn_producer_params = {
            "name_pipe": pipe_name,
            "u_name": format(u)
            }
        self.conns.append(conn_templateProducer.format(**conn_producer_params))


if net.node[v]["type"]=="customer":

       conn_cust_params = {
            "name_pipe":pipe_name,
            "v_name": format(v)
            }

       self.conns.append(conn_templateCustomer.format(**conn_cust_params))

       k=1

       for node in net.successors_iter(v):

            conn_pipe_params = {
                "name_pipe":pipe_name,
                "second_pipe": "dualPipe{}_{}".format(format(v),node),
                "portCounter": k+1
                }
            self.conns.append(conn_templatePipes.format(**conn_pipe_params))
            k=k+1
            Nports = k
```

Figure 3.13. The basic behind an edge translation.

The translation shows that an edge is translated. The node that goes into the pipe is called node[u] and the node at the end of the edge is the node[v]. If node[u] is a consumer, nothing should be done because it has been connected by a previous edge, however if it's a producer it has to be connected. If node[v] is a consumer a connection template is being filled in to make the connections in Dymola. The connection template can be seen in Figure 3.14. It's also necessary to connect all outgoing edges of node[v] to ensure there is no connection between flow resistance models. The loop counter k is used to keep track on how many nodes node[V] is connected to, making sure that there are new connections into the vector port of the pipe.

```
        conn_templateCustomer = \
"""\
  connect({name_pipe}.port_b[1], {v_name}.port_a) annotation (Line(points={{{{-37,88}},{{-24,88}},{{-24,8.4}}}}, color={{0,0,127}}));
  connect({name_pipe}.port_b1[1], {v_name}.port_b) annotation (Line(points={{{{-37,88}},{{-24,88}},{{-24,8.4}}}}, color={{0,0,127}}));
"""
        conn_templateProducer = \
"""\
  connect({u_name}.port_a, {name_pipe}.port_a) annotation (Line(points={{{{-37,88}},{{-24,88}},{{-24,8.4}}}}, color={{0,0,127}}));
  connect({u_name}.port_b, {name_pipe}.port_a1) annotation (Line(points={{{{-37,88}},{{-24,88}},{{-24,8.4}}}}, color={{0,0,127}}));
"""
        conn_templatePipes = \
"""\
  connect({name_pipe}.port_b[{portCounter}], {second_pipe}.port_a) annotation (Line(points={{{{-37,88}},{{-24,88}},{{-24,8.4}}}}, color={{0,0,127}}));
  connect({name_pipe}.port_b1[{portCounter}], {second_pipe}.port_a1) annotation (Line(points={{{{-37,88}},{{-24,88}},{{-24,8.4}}}}, color={{0,0,127}})
"""
```

Figure 3.14. Connection template used in Python to get the right connections in Dymola.

The translated result can be seen in Figure 3.15.

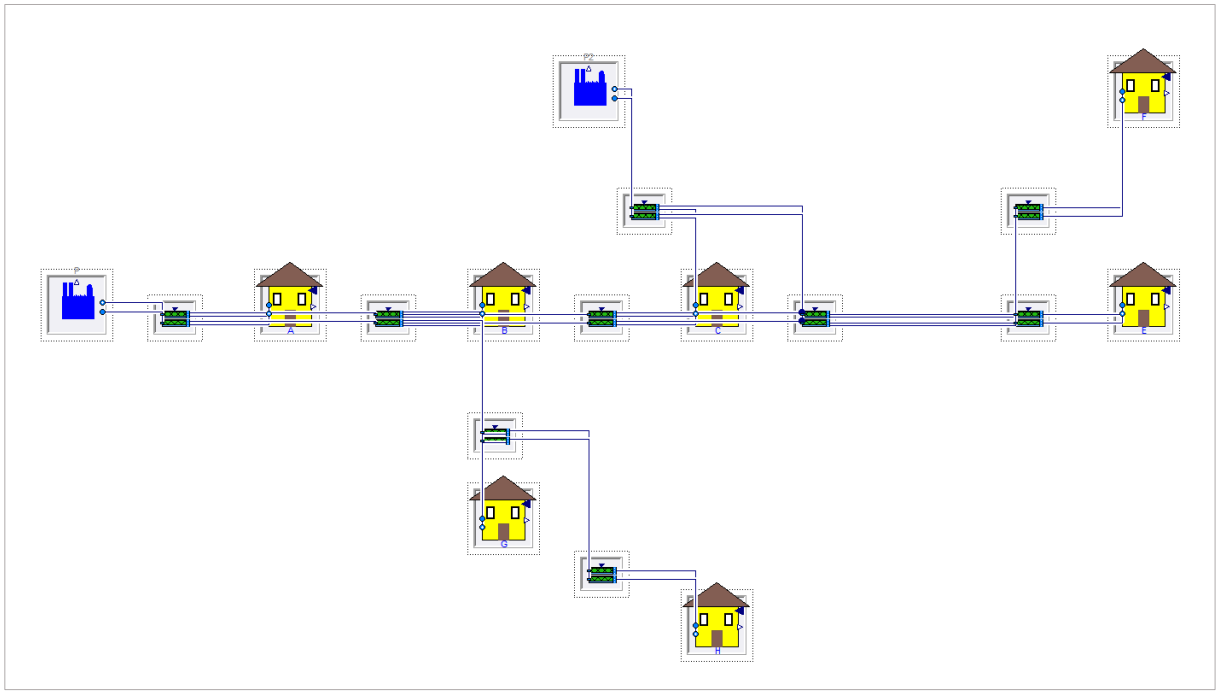

Figure 3.15. The network from Figure 3.11 translated into Dymola.

At the moment there is no ideal way to connect a node if the in-degree is higher than one. Connecting a consumer with an in-degree higher than one would lead to connection two volume models. This creates nonlinear systems when simulating. However simulations are still possible to run and the effect of the simulation time is not too large. There are ways around this, for example, the use of pipes without a volume model when the in-degree of a consumer is higher than one, another implementation would be to use pipes without volume models and then have volumes models between the pipes, however this creates a problem with the delay of heat fronts. For a first implementation this is not taken care of since if a problem arise it's fairly easy to fix manually.

4. Single component simulations

In order to control the accuracy of the models and compare simulation statistics for different setups a series of simulations were performed. The simulations compared the different pipe model implementations simulations statistics and results. Simulations for the consumer models were done to make sure that they were able to meet the heat flow rate for a given heat load.

4.1 Reverse Flow

The water flowing through a pipe usually maintains the same flow direction during a simulation, however there are occasions where the flow is reversed and even changing during a simulation. To check if the pipe works for flow in different directions as well as for zero flow.

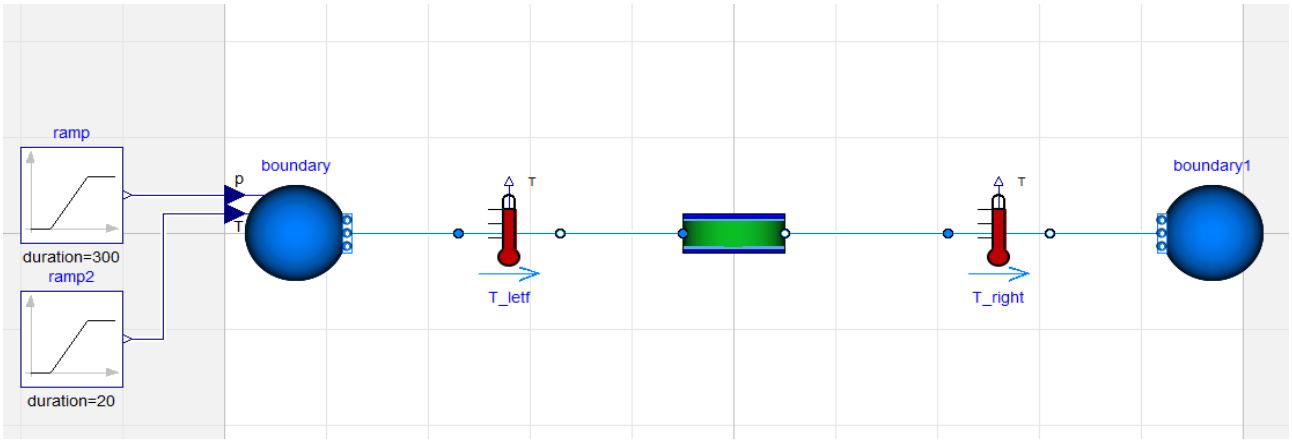

Figure 4.1. The setup of the simulation in Dymola. The boundaries set the pressure and creates a mass flow inside the pipe and the temperature sensor to the left and right of the pipe is used to see the result of the simulation.

The simulation was done for one day, using a three kilometer long pipe with a diameter of one decimeter. The boundaries is used to set a differential pressure of one bar where the left boundary has the higher pressure. At start the temperature in the left boundary was 50°C and the temperature of the water inside the pipe was 20°C. The ambient temperature was set to 0°C. Five hours into the simulation the temperature of the left boundary was increased to 80°C. After twelve hours the pressure started to drop by two bars in the left boundary over a period of 300 seconds, this lead to a change of direction for the flow. The right boundary had the temperature of the water of 50 °C. Four different simulations were done using different pipes to compare the simulation time and results. The first simulation used the MSL pipe which was based on the finite volume and was discretized into 20 segments, the second simulation used the spatial operator pipe without a volume. The third simulation used the same pipe but with a volume and a split ratio of 0.9, the last simulation was done with the same pipe as in the third simulation but the split-ratio was changed to 0.1.

The Modelica standard library pipe does not use the insulation to calculate the heat loss to the surroundings. For this reason the temperature of the water coming out of the different kind of pipes should not be compared but instead the characteristics of the temperatures should be studied.

4.1.1 Results

The simulation that was done with the MSL are shown in Figure 4.2. The simulation took 1.2 seconds to finish. It also had a few nonlinear system; {1, 1}. The number of equation for the simulation for this model was 462.The blue line gives the temperature for the left sensor and the red shows the temperature for the sensor to the right of the pipe.

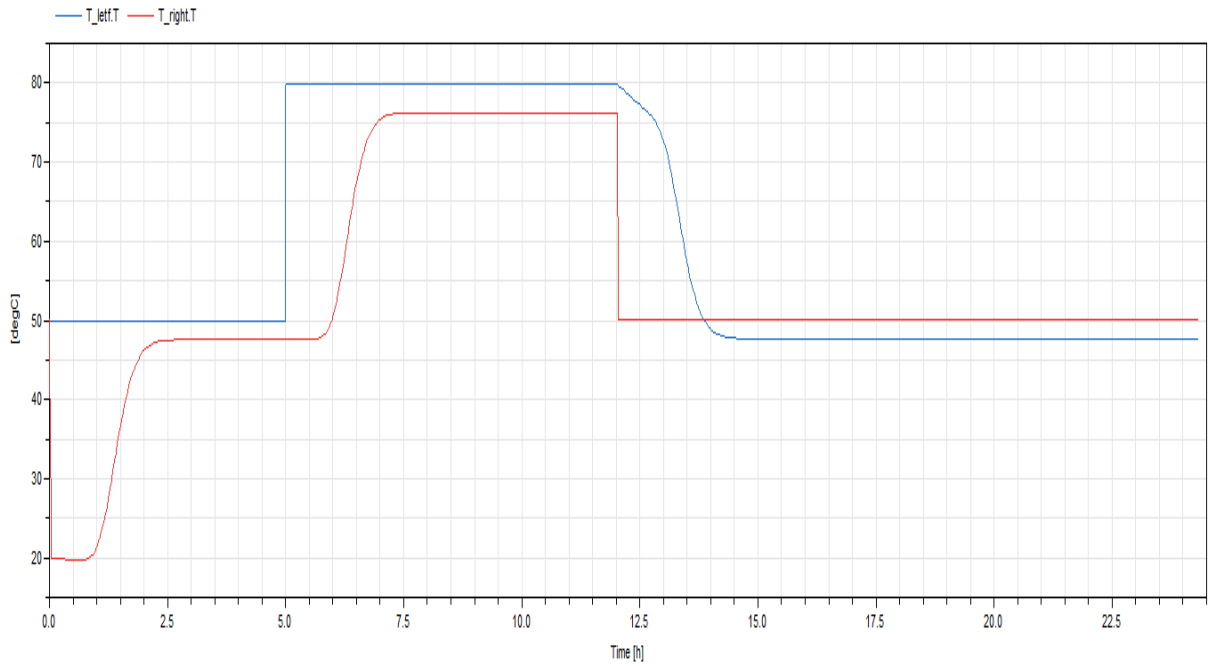

Figure 4.2. Reverse flow simulation with the Modelica standard library pipe. The pipe was discretized into 20 segments. The blue line show the temperature of the left temperature sensor and the red show the temperature given by the sensor to the right.

In Figure 4.3 the same layout was used although the pipe was changed to the one based on the spatialDistribution-operator without a volume model. The simulation run faster compared to the MSL and the time was reduced to 0.12 seconds. After manipulation done by Dymola it have no nonlinear systems. The number of equations were reduced to 119 due to less complexity.

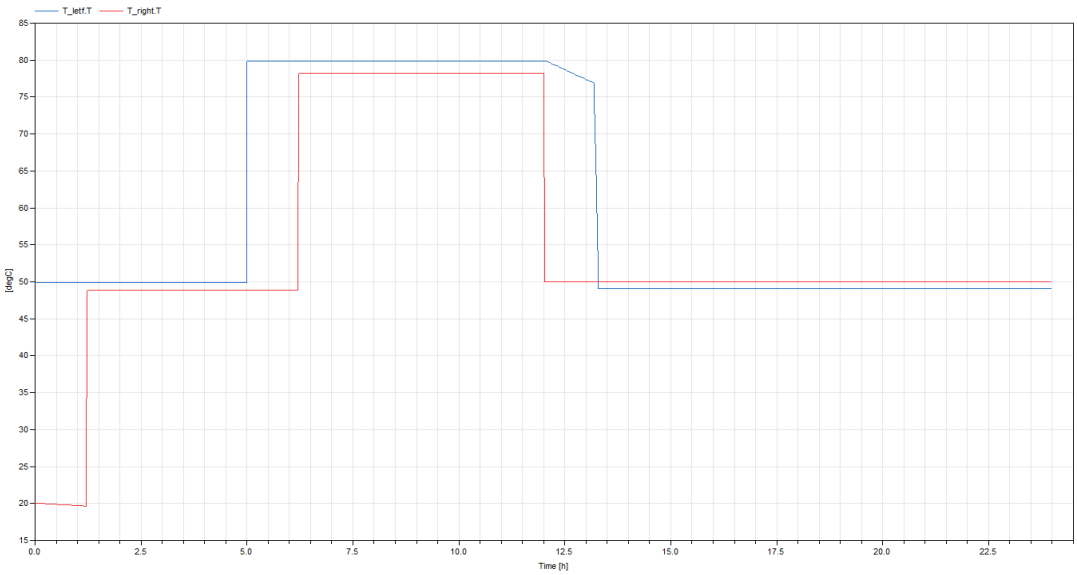

Figure 4.3. Reverser flow simulation with the pipe based on the spatialDistribution-operator.

The result from the third experiment can be seen in Figure 4.4. The pipe is the same as the previous simulation except with a volume model on the right side of the pipe. The simulation times was the same as the previous 0.12 seconds, but since it now had a volume model, there was a volume next to boundary which introduce a nonlinear system into the model, but no Jacobian.

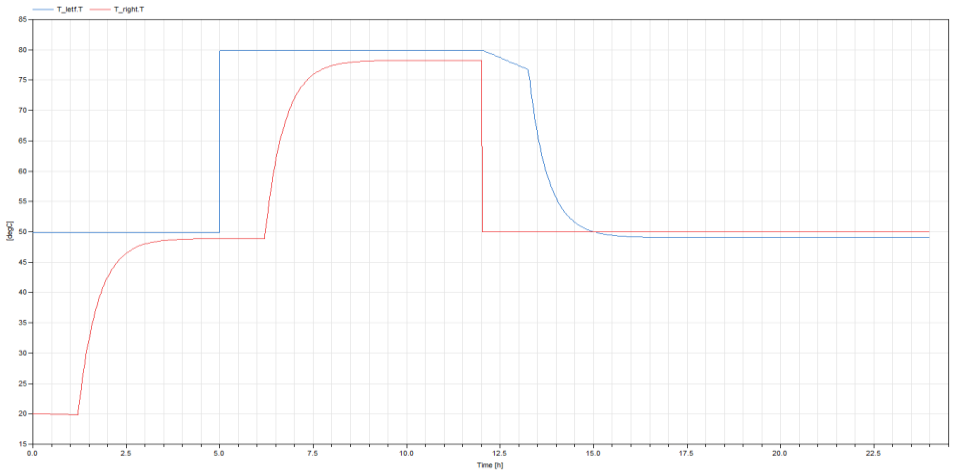

Figure 4.4. The reverse flow simulation for the same pipe but with an added volume model. Simulation time was the same as the pipe with no volume.

The final simulation was done with the same pipe but the split-ratio was changed to 0.1. 90 percentage of the pipe actual volume was given to the volume model and 10 percent was used by

the spatial operator. The result from the simulation can be seen in Figure 4.5. Statistics from the simulation was the same as previous setup since only one parameter has changed.

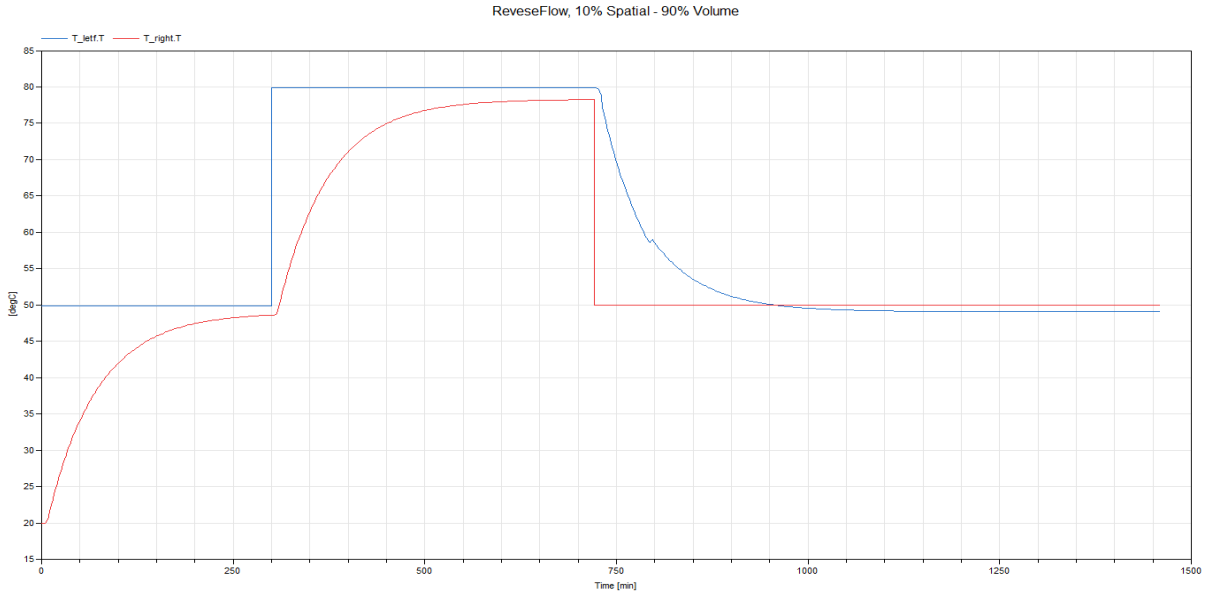

Figure 4.5. Simulation done with a bigger volume model compared to the third simulation.

The water temperature inside the pipe is set to 20°C in the beginning and the right temperature sensor show that that the temperature is slowly decreasing because the water coming out of the pipe has spent more time inside the pipe and the ambient temperature is lower than 20°C. After approximately one hour most of the water that was 50°C at the start of the pipe now reaches the sensor to the right. For the fourth simulation the increase of temperature starts a lot faster due to the pipe acts more as a big tank with ideal mixing. Twelve hours into the simulation the pressure on the left lowers and the mass flow now goes from right to left. The sensor to the right instantaneously change to 50°C because the water is now flowing directly from the boundary. The left sensor show a slow decrease in the temperature, this is because of the 80°C water that was in the pipe now has changed direction. After about one hour all of the 50°C water starts to reach the left sensor, for the pipe with a split ratio of 0.1 the delay is a lot lower. The drop when the 50°C water reach the left sensors is best seen in the pipe without a volume-model since no mixing occurs.

4.1.2 Discussion

Simulation show rather similar result between spatial operator pipe using a split-ratio of 0.9 and the MSL-pipe. When the split-ratio is reduced to 0.1 the pipe model is acting more as a tank than a pipe. The heat loss for that pipe however is the same since only the delay spatial operator is effected by the split-ratio. For a long pipe with a low split-ratio the delay would give incorrect results. Using a pipe without a volume models results in sharper temperature curves in the sensor because there is no mixing occurring in the pipe. In a real world a bit of mixing always occurs because of turbulence and a volume model thus gives more realistic results.

Looking at the statistics of the simulation, the pipe with the spatial operators have a simulation time of 0.12 seconds and the numbers of equations solver are 119, changing very little when introducing a volume model. For the MSL-pipe the simulations takes 1.5 seconds and the number of equations are now 462. A few nonlinear system is also created when translating the system. None of the pipes model had problem handling zero flow or even reverse flow, and the characteristics on the pipe can be controlled by using the split-ratio. A split-ratio closed to one would give results close to the spatial operator pipe without a volume model.

4.2 Heat Loss

To evaluate the heat loss implementations for the different pipe models real world data was used to compare the results. The data was taken from a section of a larger district heating network located in Austria and was provided by the Austrian Institute of Technology. The layout of the used network is shown in Figure 4.6. The pipes for this network have the same diameter, thickness of insulation and thermal conductivity of the insulation and these data was provided as part of the data. The different length of the pipes are given by the same layout.

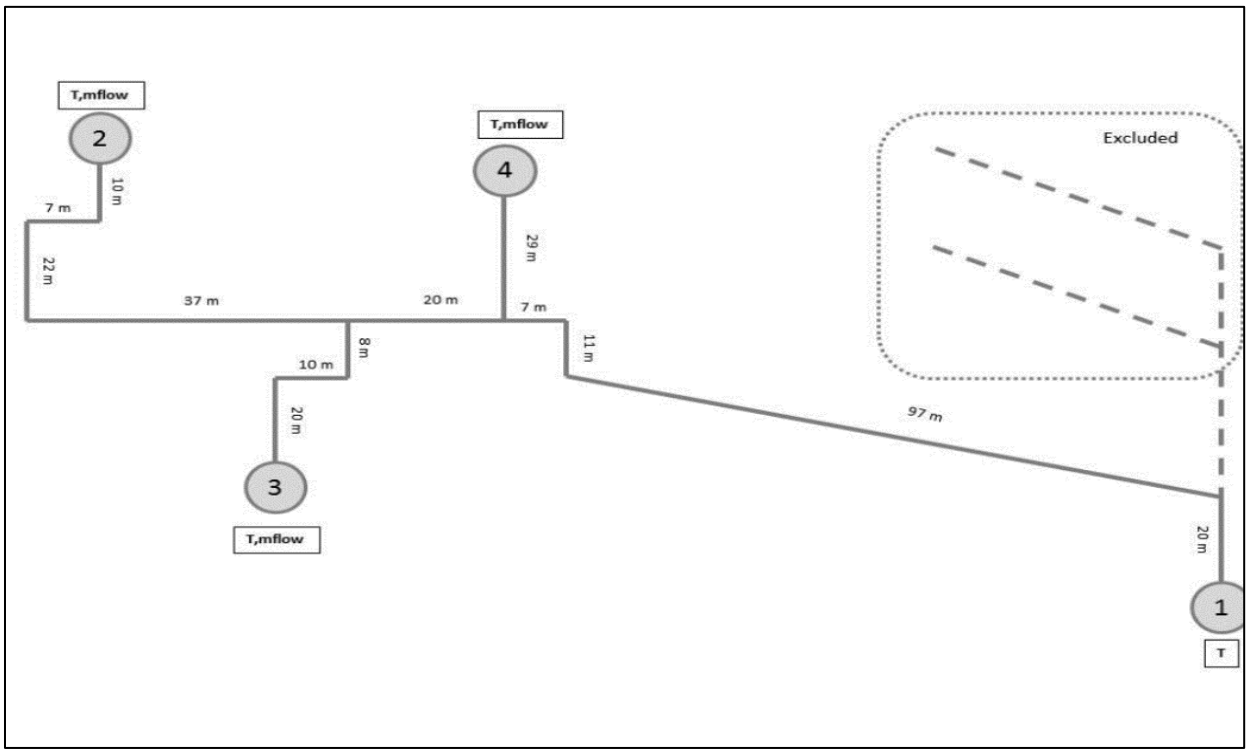

Figure 4.6. Layout of the district heating network used to compare real world data of heat loss with the simulations of the different pipe models.

The mass flow into points 2-4 were given by measured values and were used directly as mass flow sources in Dymola. Because the data contained no information about the pressure this simulation can only be used to compare the heat loss implementations. Because of this the curves of the pipes did not affect the result and the pipes could be treated like longer sections of pipes without curves.

This simulation was done for three different split-ratio in the pipe model and also the MSL pipe. Another simulation was done with a spatial operator pipe with an added volume model without changing the spatial operator used for delay. This was done to study whether there is a need to use the split-ratio. The experiment setup in Dymola is shown in Figure 4.7.

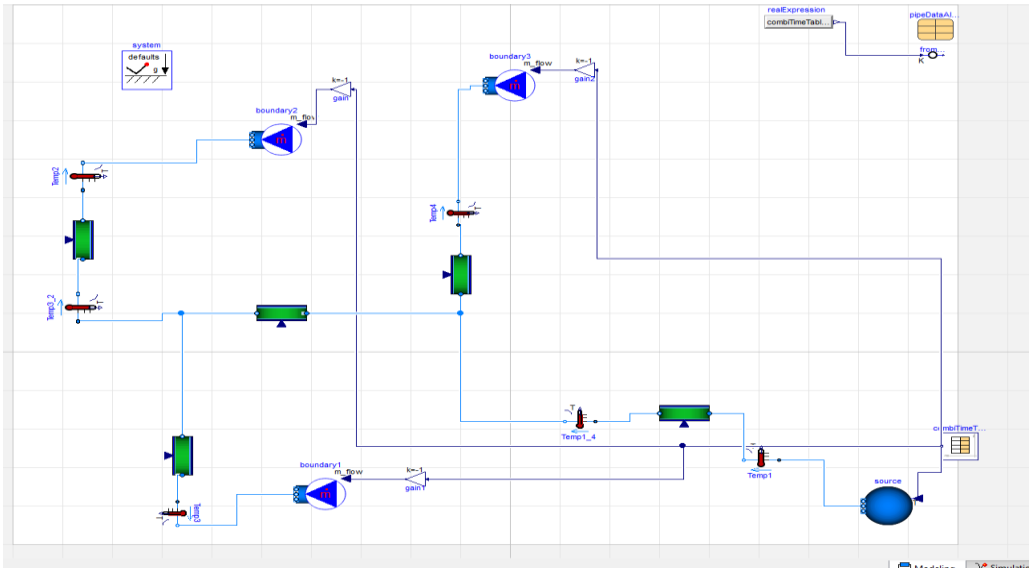

Figure 4.7. The layout used in Dymola to control the implementation of the heat loss calculations for the pipe model. The mass flow sources are connected to the measured value in the network. Temperature sensor are located in the network to compare the measured and simulated temperature.

The outdoor temperature was given for all seven days and can be seen in Figure 4.9. Note that this is the outdoor temperature and not the ambient temperature used to calculate the heat loss. Since information about the ambient temperature was not given from the data the simulation was done with the outdoor temperature. The temperature at point 1 was given by from the data and this is shown in Figure 4.8.

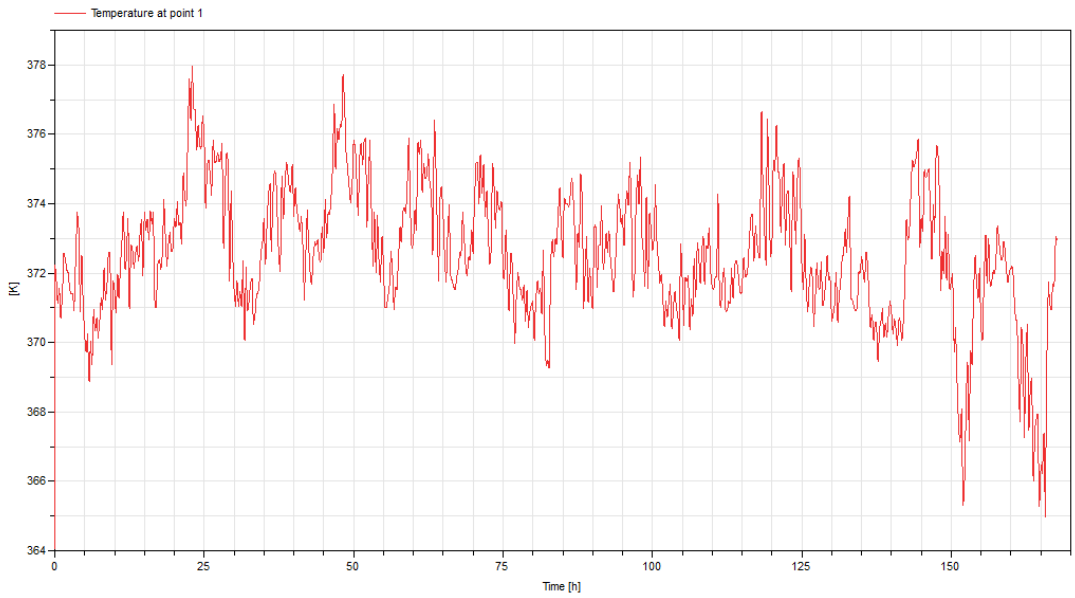

Figure 4.8. Temperature of the water at point 1.

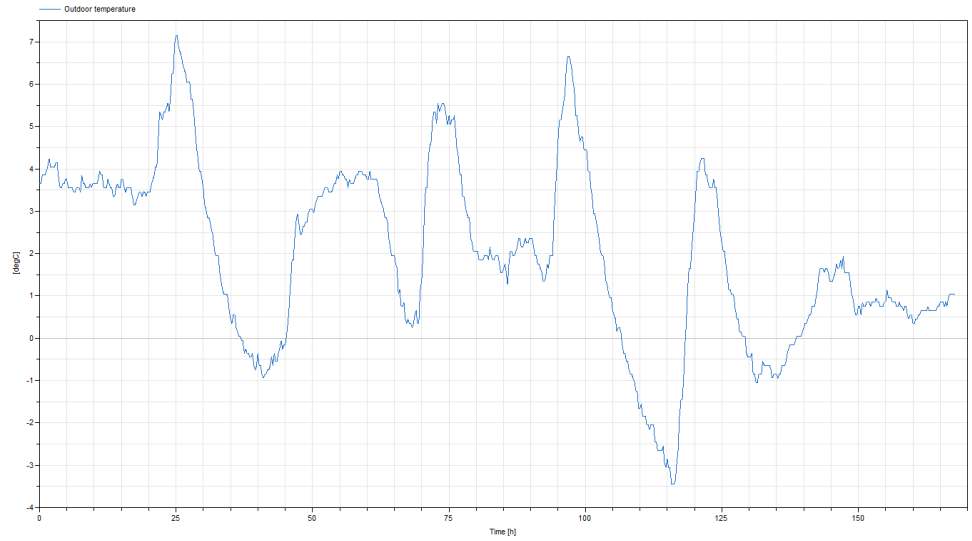

Figure 4.9. The outdoor temperature for the simulation time given by the measured data. The temperature was used as ambient temperature for the pipes because lack of better options.

The mass flows for point one and two are shown in Figure 4.10 This data of the mass flows was used to drive the mass flow sources connected at the different points.

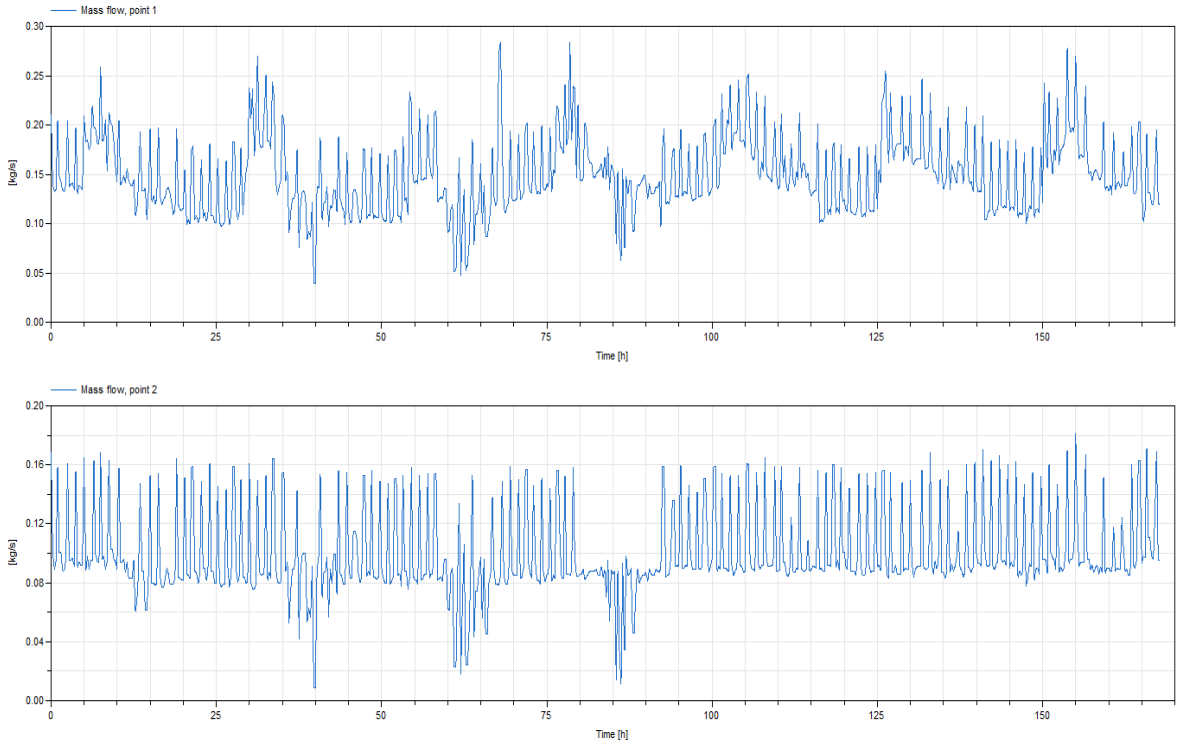

Figure 4.10. The measured mass flow for point one and two for the district heating network.

For the mass flow into point three and four see Figure 4.11.

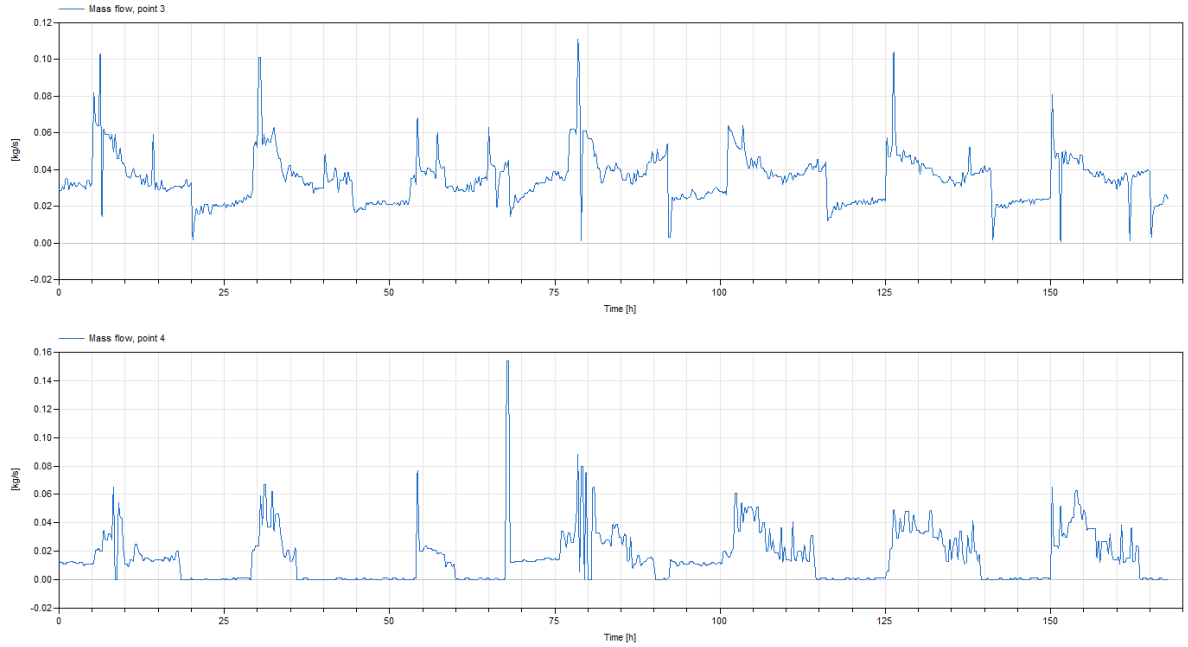

Figure 4.11. The measured mass flow for point three and four for the district heating network.

4.2.1 Results

The first simulation was done with spatial pipe with a volume model with different split-ratios. The first split ratio used for simulation was set to 0.1, for the second simulation the split-ratio was changed to 0.5 and the last simulation was done with a split-ratio of 0.9. In Figures 4.12 to 4.14 the difference between the simulated and measured temperature are compared to each other in point two in the network. The simulation time for the spatial operator pipes was 0.41 seconds.

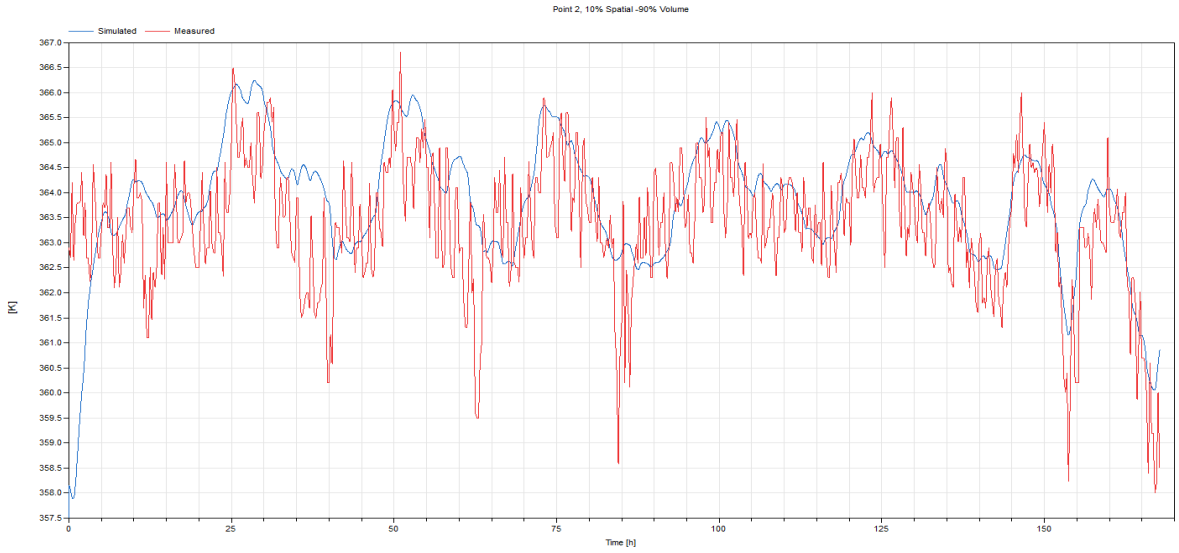

Figure 4.12. A graph of the temperature from the simulation and the measured results. The red line show the measured temperature and the blue show the simulated temperature using a spatial operator pipe with ***split-ratio of 0.1.***

In Figure 4.13 the same setup was used as the previous one although the split ratio was set to 0.5.

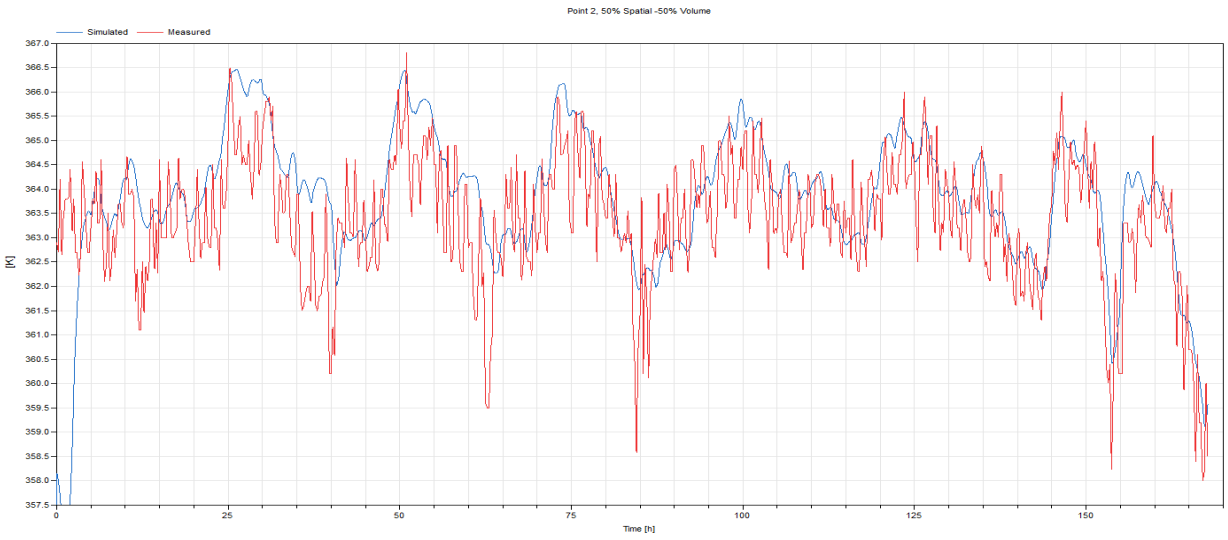

Figure 4.13. Comparison for the pipe with a ***split-ratio of 0.5*** *in* ***point 2.***

The third simulation was done using a pipe with a split ratio of 0.9. The result from the simulated temperature in point two can be seen in Figure 4.14.

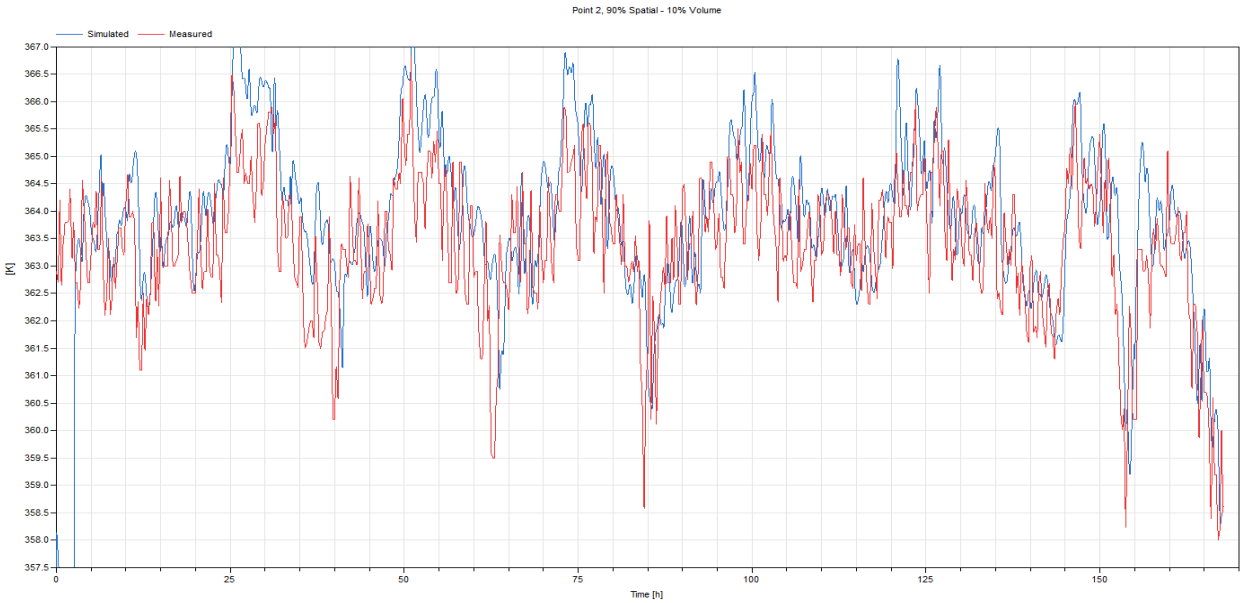

*Figure 4.14. Comparison for the pipe with a **split-ratio of 0.9** in **point 2.***

In Figures 4.15 to 4.17, the difference between the simulated and measured temperature are compared to each other in point three in the network. The simulation result for the pipe with a split ratio of 0.1 is shown in Figure 4.15.

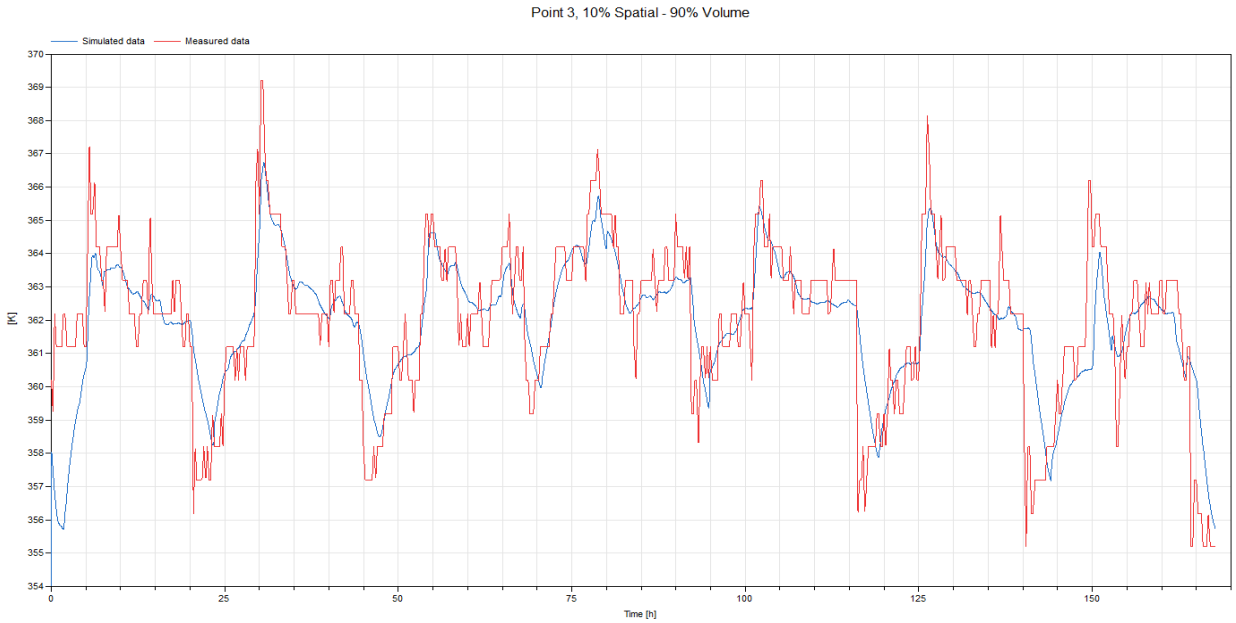

*Figure 4.15. Comparison for the pipe with a **split-ratio of 0.1** in **point 3.***

The result in point three but with an equal split between the volume model and the spatial operator is given in Figure 4.16.

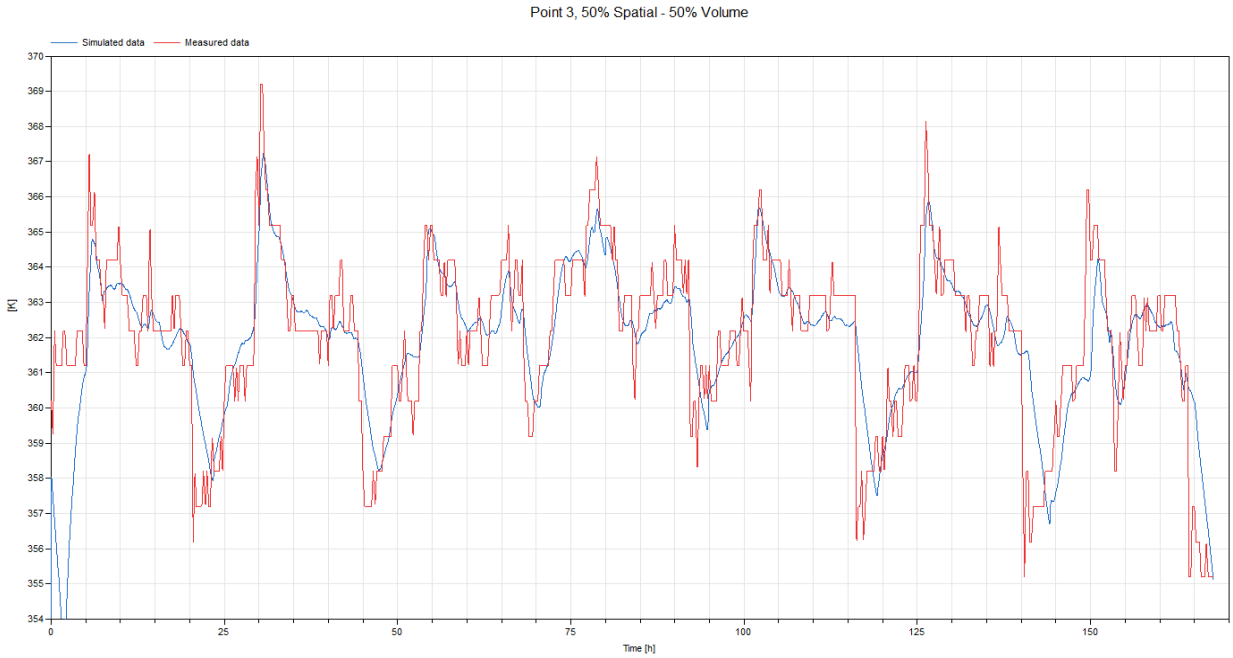

*Figure 4.16. Comparison for the pipe with a **split-ratio of 0.5** in **point 3.***

The result from having the spatial operator taking care of 90 percent of the pipe volume in point three can be seen in Figure 4.17.

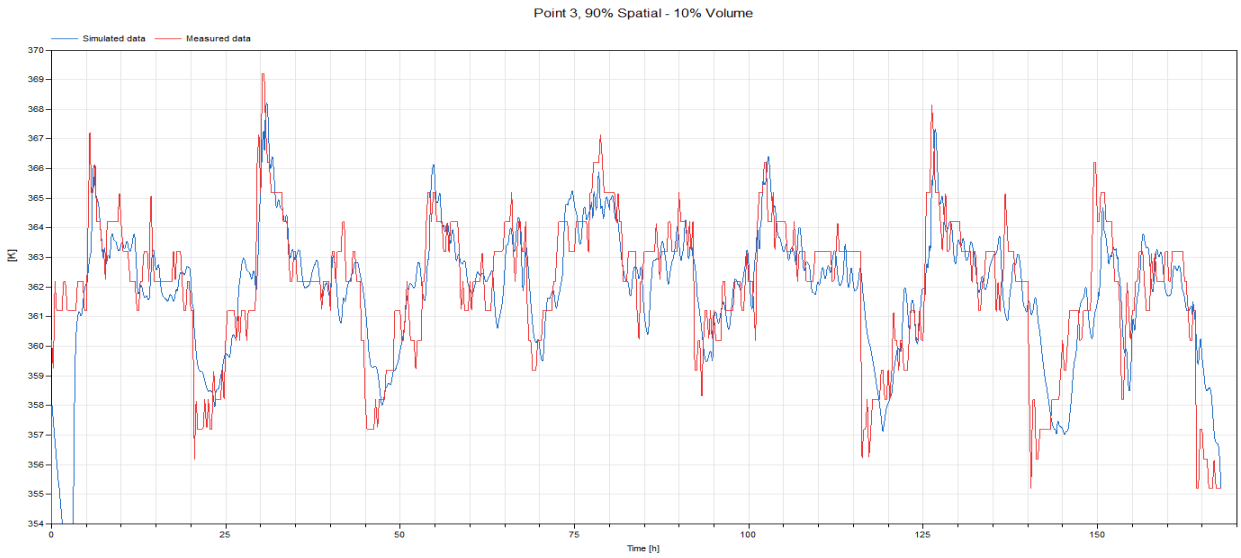

*Figure 4.17. Comparison for the pipe with a **split-ratio of 0.9** in **point 3.***

The simulation result compared to the measured temperature for point four in the network having a pipe where 10 percent of the pipe volume is handled by the spatial and the rest is used in the volume model can be seen in Figure 4.18.

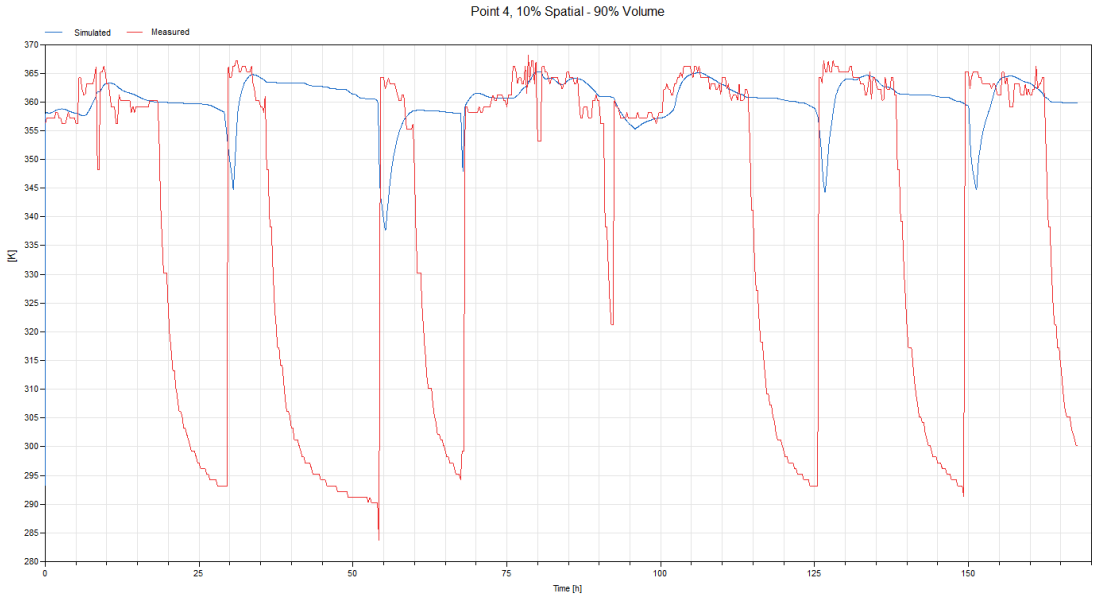

*Figure 4.18. Comparison for the pipe with a **split-ratio of 0.1** in **point 4.***

The result in point four in the network with a pipe splitting the volume equal between the volume model and the spatial operator can be seen in Figure 4.19.

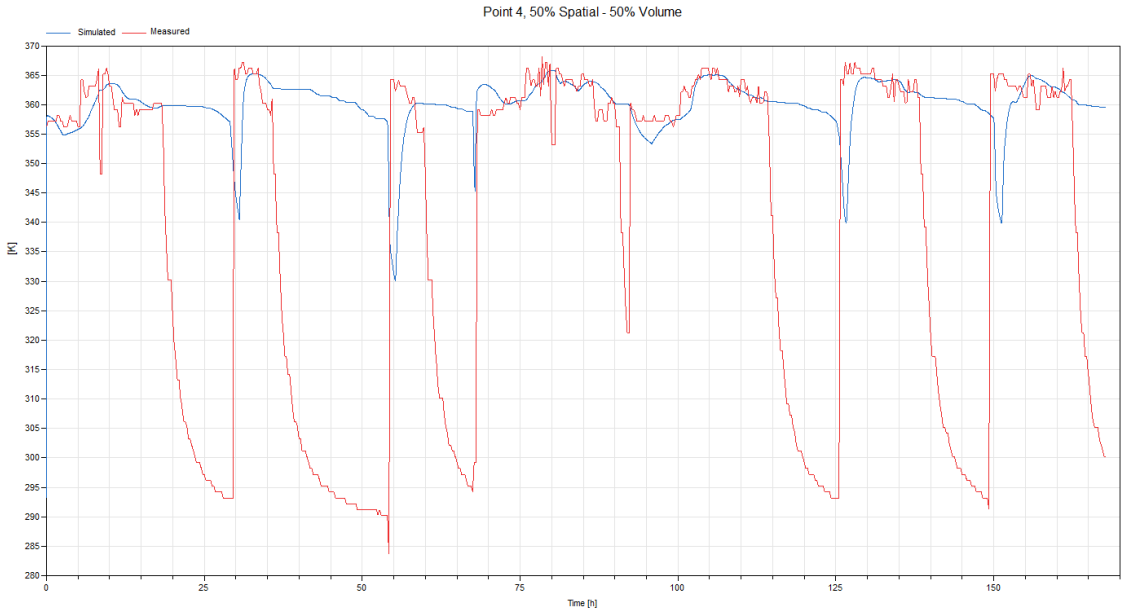

*Figure 4.19. Comparison for the pipe with a **split-ratio of 0.5** in **point 4.***

In Figure 4.20 the difference between the measured temperature and the simulated temperature for when 90 percent of the pipe volume is handled by the spatial operator can be seen.

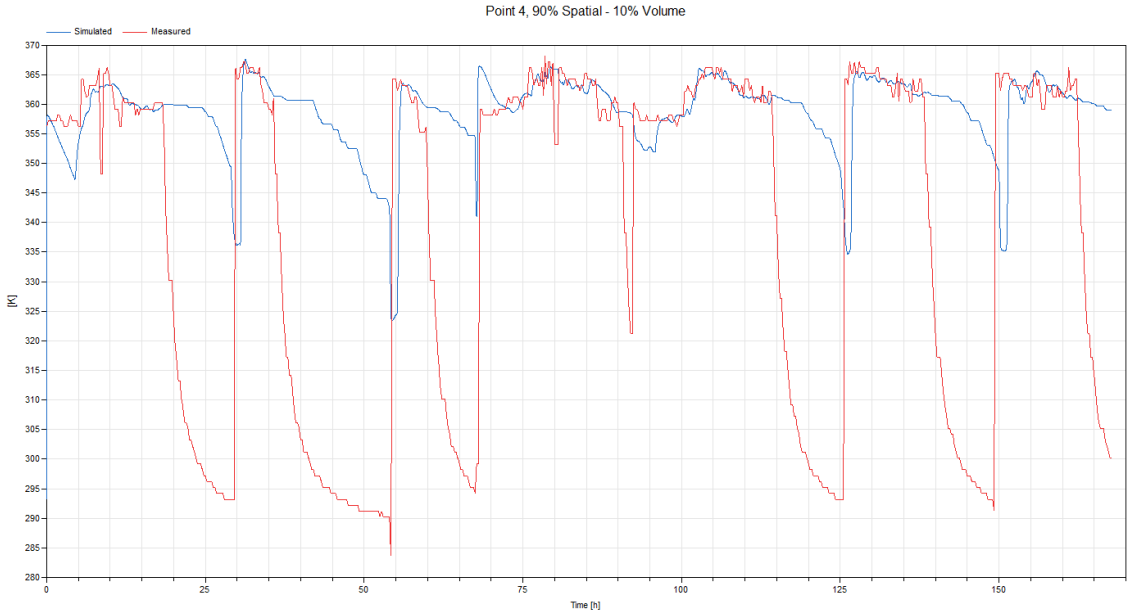

*Figure 4.20. Comparison for the pipe with a **split-ratio of 0.9** in **point 4.***

For the simulation with a spatial operator pipe without using a split-ratio can be seen in Figure 4.21

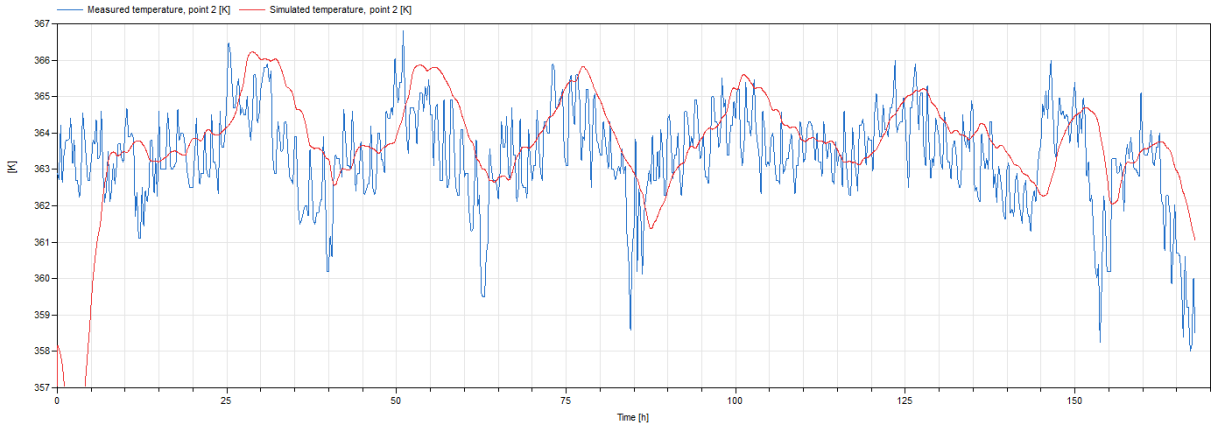

*Figure 4.21. Comparison for the pipe **without a split-ratio** in **point 2.***

To see how the MSL-pipe handles heat loss a simulation was done and the result were studied in the three different points. The result for the different points can be seen in Figure 4.22 – 4.24. The simulation time for the MSL pipe was 7.0 seconds.

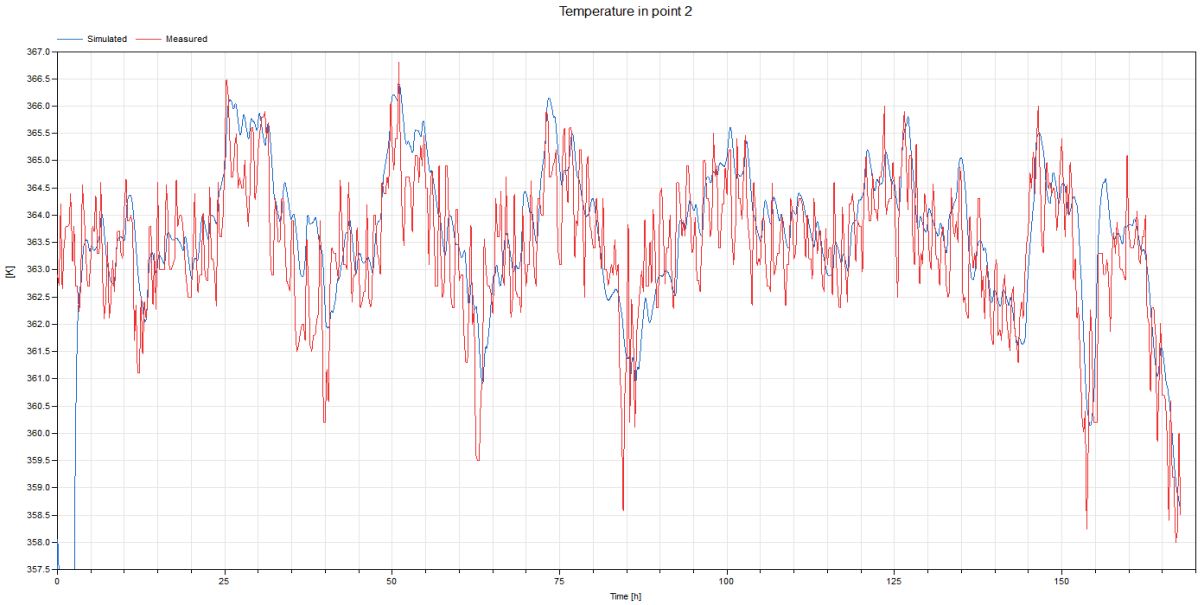

*Figure 4.22. Comparison for **MSL-pipe** with in **point 2.***

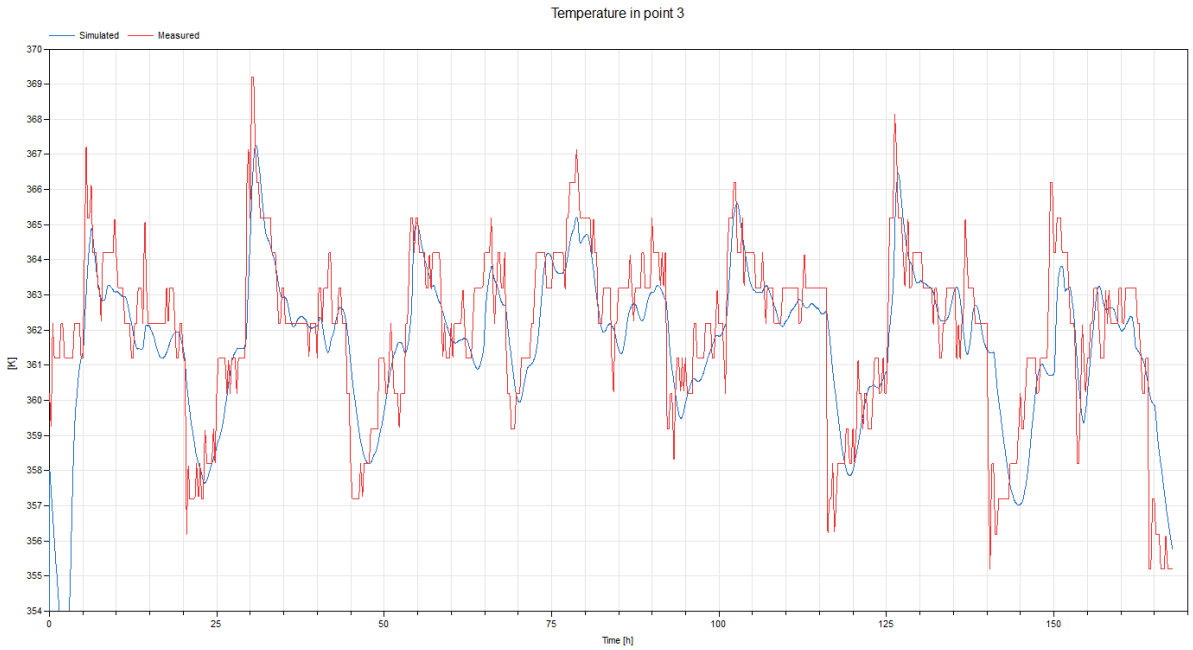

*Figure 4.23. Comparison for **MSL-pipe** with in **point 3.***

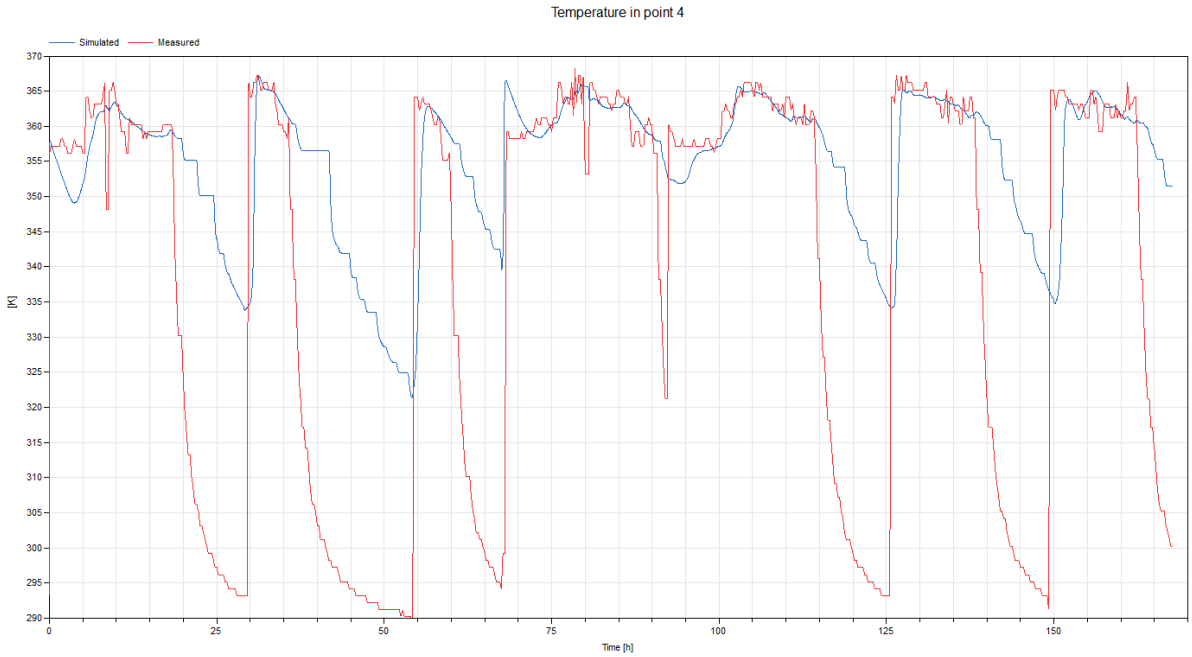

*Figure 4.24. Comparison for **MSL-pipe** with in **point 4.***

4.2.2 Discussion

The result from the simulation in point two that can be seen Figures 4.12 – 4.14 show that the temperature is close to the measured data. The simulated temperature show the same characteristics as the measured data and the simulated are only a few degrees off. The measured temperature is sometimes higher and sometimes is lower than the simulated data implying the total energy loss to the environment is the same for the simulated and measured data. It is possible that the simulated temperature is a bit higher than the measured data in point two for the spatial operator pipe. Studying the effect of the split-ratio, one could see that a large volume model smoothen out the curves but the delay is roughly the same and the same curve characteristics is kept.

In point three the simulated temperature is close to the measured temperature and the measured temperature is both higher and lower than the simulated pipe. Again implying the total heat loss is the same for the measured data and the simulated.

The results in point four show a bigger difference than for the previous points and the difference is higher if a small split-ratio is used. The measured values have a distinct drop in temperature corresponding to when the mass flow into the point is zero. The simulated values are not following this temperature drop and a high temperature difference is seen. When the mass flow is zero, only

the water inside the pipe is cooling down and the differences in energy loss to the environment are only depending of the size of the pipe volume. The simulated temperature also measures water coming out of the pipe and not the temperature of the water inside. Since there is no mass flow, the sensors can't give an accurate result.

The simulated temperature in Figure 4.21 shows why a split ratio is needed. The volume model was added without altering the spatial operator. This means that the spatial operator delays the water as if no mixing occurs but then there is also mixing in the volume model that introduces a delay. Which leads to a delay higher than expected.

Changing the pipe into MSL pipe show that there is only a little difference between the MSL pipe and the spatial pipe. The MSL pipe however use the outdoor temperature as a parameter and not an input, this means that the ambient temperature has to be a fixed value for the simulation. The MSL result show that pipe is not better at reaching the low temperature in point four when the mass flow reaches zero.

The ambient temperature set for the pipes are the outdoor temperature. This is only correct if the pipes are above ground and no wind occur, but mostly the pipes are buried underground. For a more correct result a model should be used to calculate the temperature on the same depth level as the pipe is laying or used measured data for that temperature and not the outdoor temperature. The ground temperature model is not a part of this study and the depth of the pipes are not given.

The results from the simulation show that if the thermal conductivity for the insulation and the diameter of both the pipe and insulation is given, then the simulated temperature is only a few degrees off compared to the measured data. The few degrees difference is both under and over the measured value and for a simulation of a whole district heating system the result should be realistic. Using the outdoor temperature as input for ambient temperature does not seem to have a major effect on the results and a model to calculate the ambient temperature might not be necessary. The spatial operator pipe is also a lot faster for simulation than the MSL pipes. The simulation time for the spatial operator pipe is about 5% of the time it takes to simulate the MSL pipe.

4.3 Consumer models

To ensure that the consumer models are able to control the mass flow and obtain a heat flow rate corresponding to the given heat demand, a simulation was done for a network of four consumers. The simulation setup can be seen in Figure 4.24. The inner diameter of the pipes was set to 0.1 meters and the length to 200 meter, except for the supply- and return pipe closest to the boundaries that were set to 300 meters. The split ratio for the pipes was 0.9. The four different consumers had time varying heat demand profiles. The heat demand was given from the ramps connected to the consumer models and are showed in Figure 4.24. Consumer number three and four were connected in parallel to consumer number two, to ensure that the network both could handle consumer in series and parallels. The starting pressure for the whole network was set to five bars to avoid transients and the starting temperature inside the pipes are set to 20°C. Over a period of 300 seconds at the start the pressure was increased in the top boundary to 15 bars. The highest mass flow into all consumers were set to 60 kg/s by the valves. Separate simulations were done with all consumer models however to the reduce number of graphs, only results from the PI-controlled fixed return temperature is included in the paper.

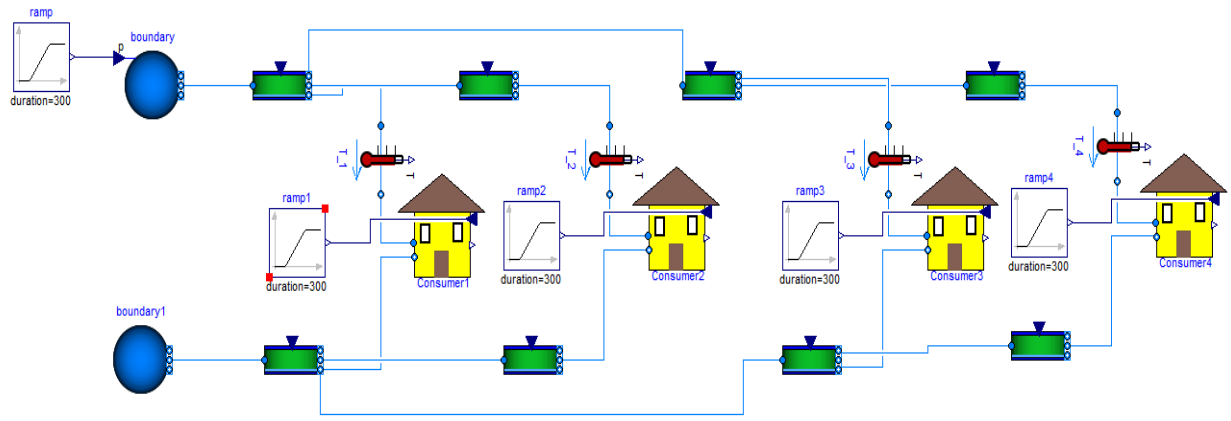

Figure 4.24. Layout of the simulation done with four different consumer models with fixed return temperature. The consumer models were connected to ramp-blocks that gives the load profile for each consumer.

The load profiles that were given from the ramps is shown in Figure 4.25. Here ramp1.y is the demand for Consumer1 in kilowatts and ramp2.y is the demand for Consumer2 in kilowatts etc.

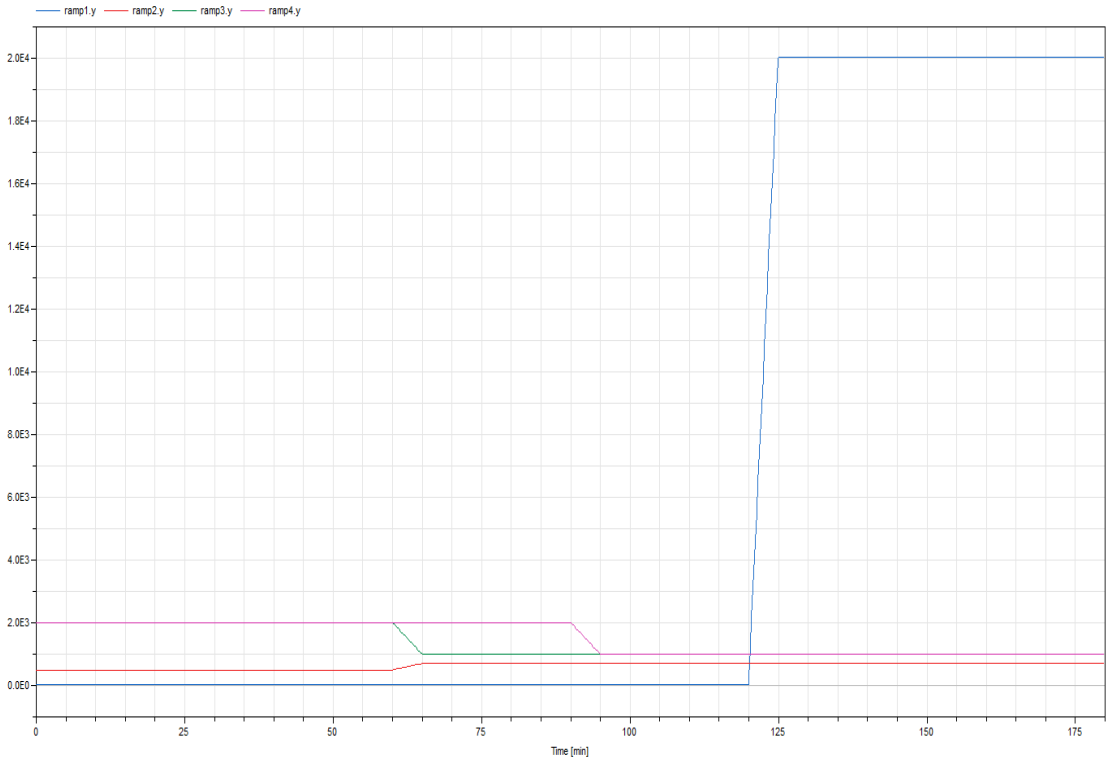

Figure 4.25. The load profiles that are used in the consumer models.

4.3.1 Results

Translation of the models were done and no nonlinear systems were created. The simulation time for the experiment were 3.9 seconds for all consumer models. The simulated heat flow for the four different PI-controlled consumers can be seen in Figure 4.26.

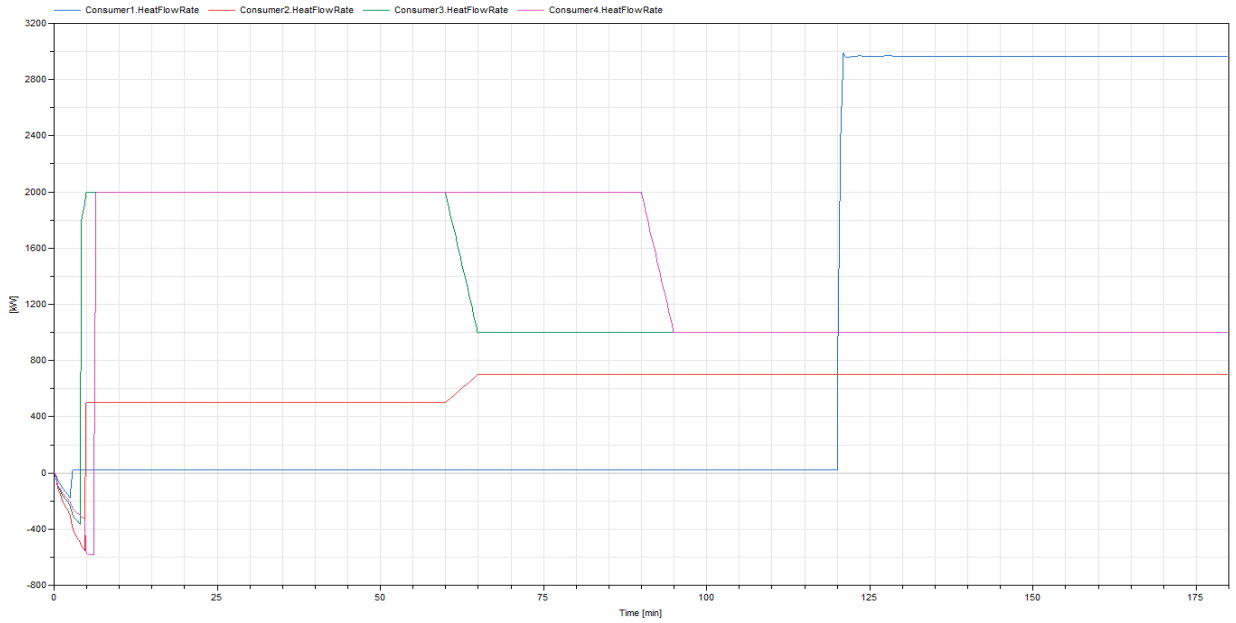

Figure 4.26. Simulation of the heat flow rate for the four different consumer

The output signal from the PI-controller for the different consumers can be seen in Figure 4.27.

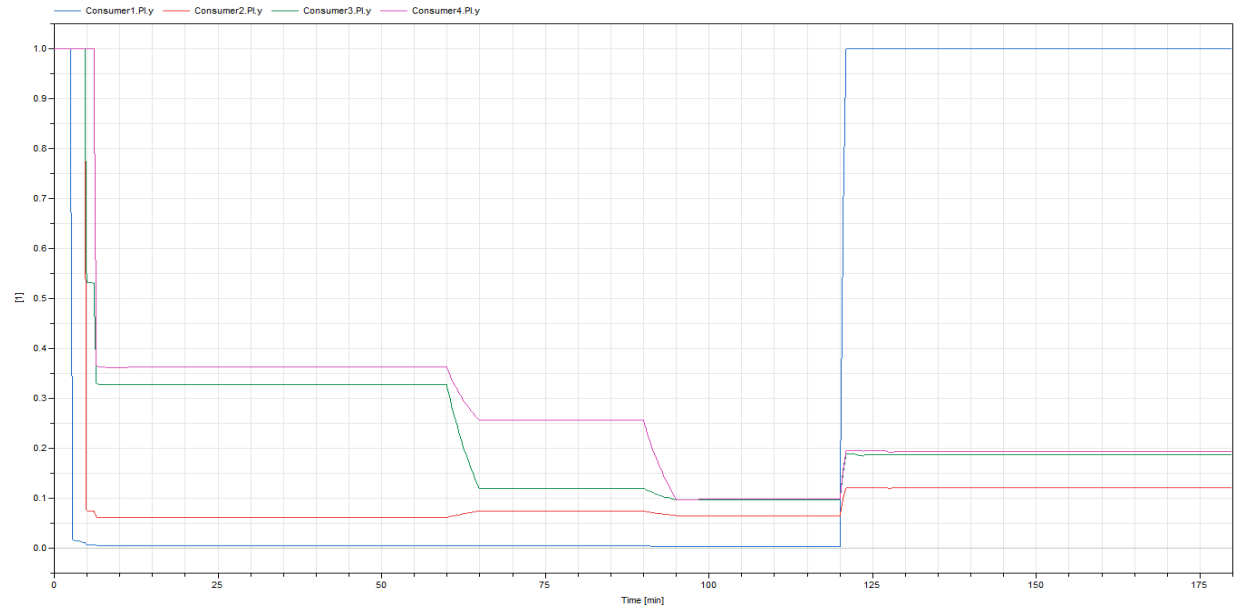

Figure 4.27. The output signal from the PI-controllers inside the consumer models which regulates the opening of the valves.

Figure 4.28 show the simulated result for the fourth consumer. This is included to study what was happening inside the fourth consumer during the simulation.

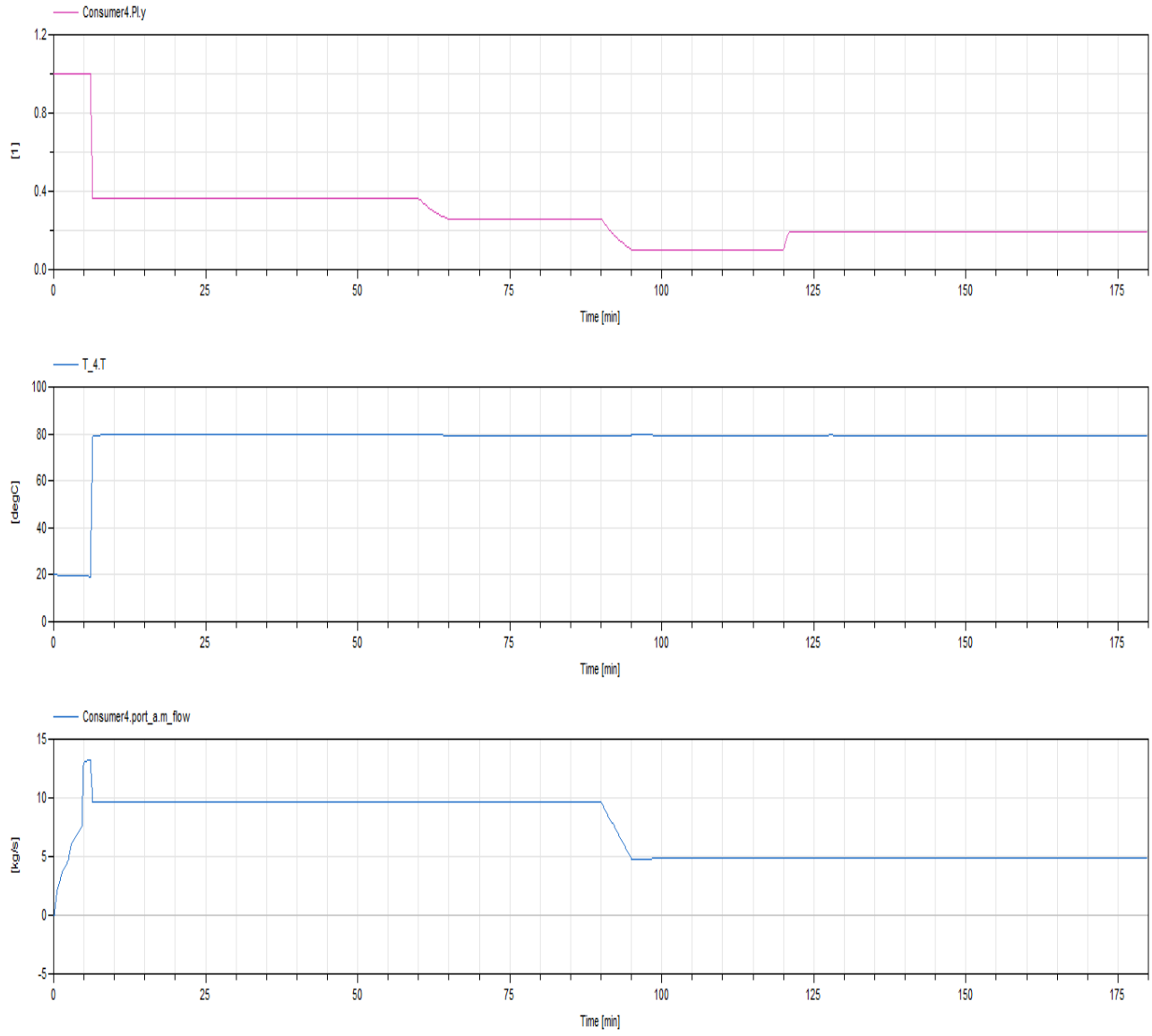

Figure 4.28. A deeper look into Consumer4 to see what is happening inside a consumer model.

The simulations was replicated with the MSL pipe. The results from that simulations differs very little to the result from the spatial operator pipe. The major difference is the simulation time, the time increased to 198 seconds from 3.9 seconds. The translated model also had a few nonlinear system. A comparisons of the statistics of the difference models can be seen Figure 4.29.

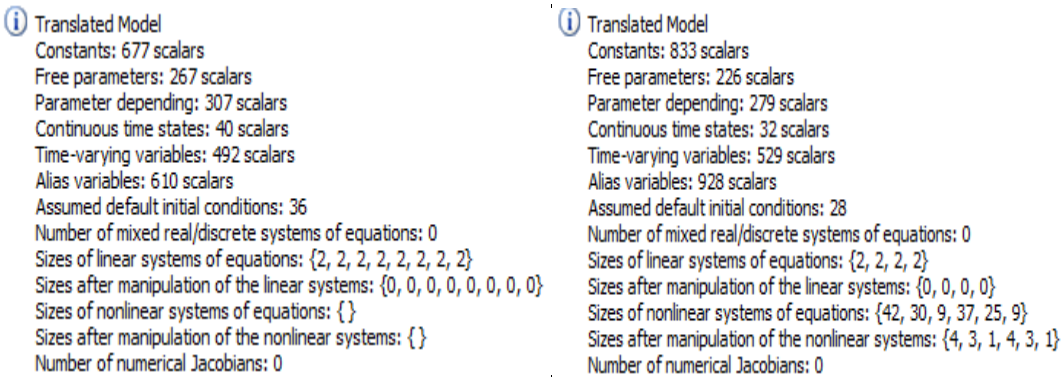

Figure 4.29. Statistics of the different translated models. The left one is the statistic from the spatial operator and the right one is the statistics from the MSL-pipe.

4.3.2 Discussion

The result show that the consumer model have no problem to meet the demanded load if the temperature of the water is high enough and sufficient water can be feed into the consumer model. In Figure 4.26, the heat flow is negative for the first minutes for all consumer. That is because the starting temperature inside the pipes is 20°C and the water of the consumer model is 30°C, so for the first minutes heat is actually taken out from the consumer model. The simulation has this setup to control that the PI-controller does not shut the valve ensuring that there still is a mass flow into the consumer. Zero mass flow into a consumer model would decrease the error but in that case hot supply water would never reach the consumer.

As seen in Figure 4.27, as the hot water reaches the consumers, the heat flow drastically increases and the PI-controllers now have to regulate the opening of the valves to satisfy the consumer's heat demand. After 120 minutes the heat demand for the first consumer is too high and even if the valve is fully opened the heat flow rate from the heat exchanger cannot meet the high heat demand. The valves of the three other consumers has to open up more as well when the first consumer demand is increased, because the mass flow increases in the first consumer model and since there is a max amount of water going through the first pipe depending on the differential pressure set by the boundaries. It is possible to see this in Figure 4.28, where the mass flow into consumer four is constant but the valve opening is increased.

We are able to remove all nonlinear equations using the spatial operator pipe and no numerical Jacobian is created in that case, but when the pipe is changed to MSL-dynamic pipe a few nonlinear system matrixes are created. The number of equations is increased when changing pipe. These results show why there is a need to create a pipe model not based on the finite volume. If the system would be scaled up the simulation time using MSL-pipe would increase even further and this type of pipe would be unable to use for larger networks.

5. Network simulations

District heating networks can have a high amount of consumers and pipes in a network when simulating networks. It is important to determine the highest amount of components in a district heating network that is possible to be simulated, it is important to know if the network has to be aggregated before simulation. The simulations also have to be able to handle loops, branching and meshed networks in the networks. A network occasionally have several producers and it is important that simulations can handle this too. Since the result have shown that the spatial pipe both give better simulation properties and accurate results it's the only pipe used further in network simulations.

A simulation for an imaginary network was created using blocks of five consumers connected in series with five different time varying heat demands. The consumer models used for the simulation were the fixed return temperature without a PI-controller. The return temperature of the consumer was set to 30°C. The connection between the consumers can be seen in Figure 5.1. The heat demand for the different consumers can be seen in Figure 5.2 and the total heat load for one block can be seen in Figure 5.3. The load for the different consumers were made to ensure that the network could handle time-varying load and still provide realist load profiles with peaks in the morning and peaks in the afternoon.

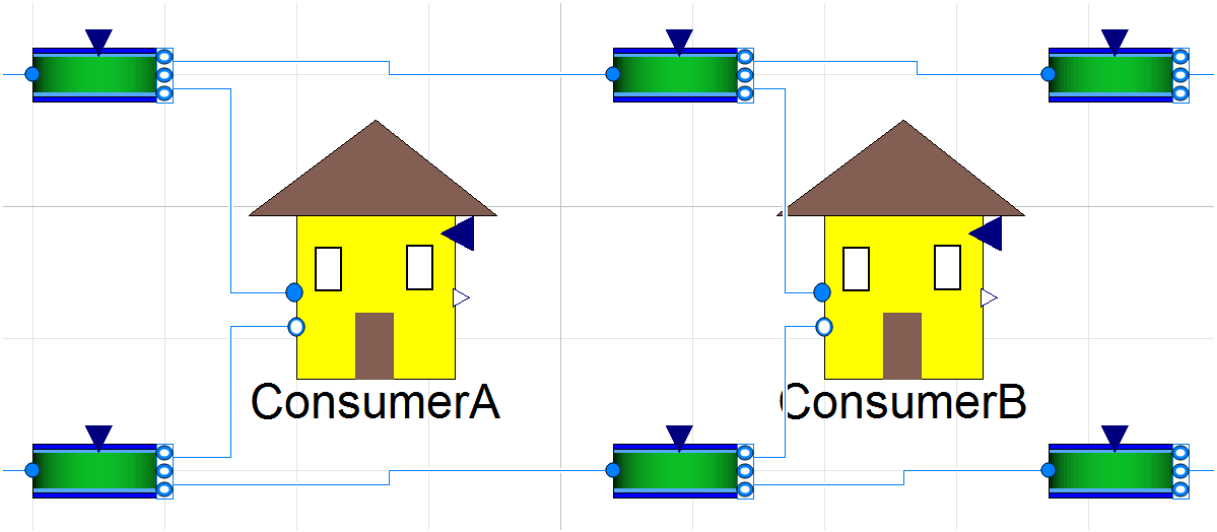

Figure 5.1. A section of the block used in network simulations. The section show how the consumers were connected..

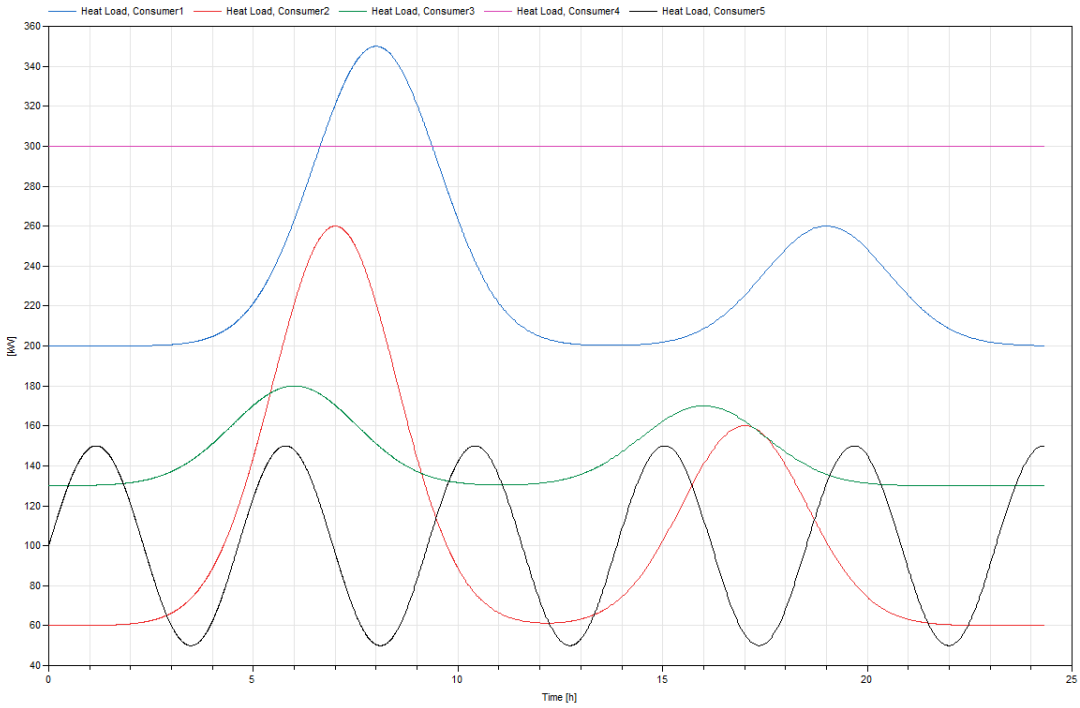

Figure 5.2. The individual heat demand for the five consumer models in the block

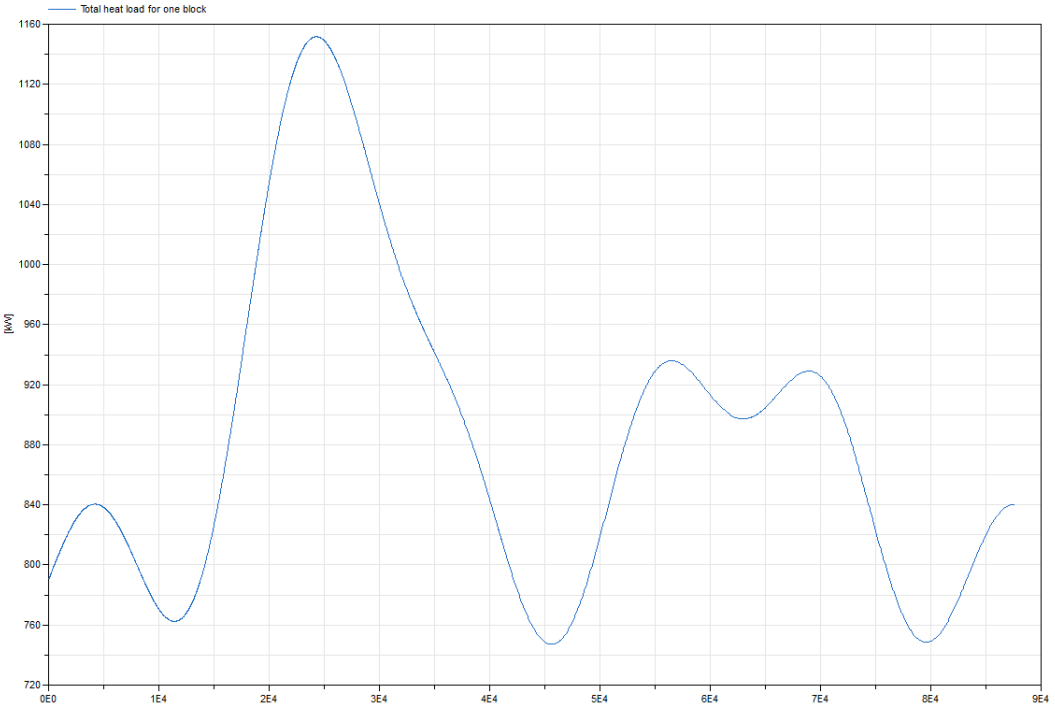

Figure 5.3. The summarized heat demand for one block of five consumers.

5.1 Branched Network

District heating networks rarely consist of consumers connected only in series with each other, often they also contains both branches and loops. A simulation was done with fifty consumers to ensure that the simulation could handle one branch. The layout of the network can be seen in Figure 5.4. The pipe diameter was set to 0.3m through the whole network and the split-ratio to 0.9. The length of all pipes are 500m. The initial pressure is five bars in all nodes of the network and the starting temperature of the water was 60°C. To study if the pressure decreases lower when heat front starts to reach the consumers. In this layout there is a branch after the green pipe. The pressure of the top boundary was controlled by a PI-controller. The control signals was calculated to give the last consumer in the network a differential pressure of two bars to ensure that the consumer could satisfy its heat demand, but without pressurizing the system more than necessary. The lower boundary was set to pressure of five bars and this was not altered during the simulation

The temperature of the top boundary was connected to the temperature of the water in the return pipe. An ideal heater with unlimited power was used to calculate what power was needed to reach the desired temperature which was set to 90°C.

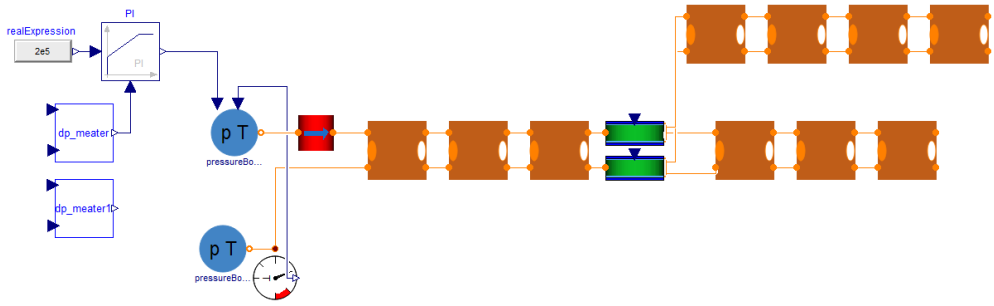

Figure 5.4. Layout of a branched network with fifty consumers. The pressure from the boundary is set to ensure that the last consumer have a differential pressure of two bars.

5.1.1 Results

The simulation time was 30 seconds for this network setup with these properties. The result for the different pressure in the network can be seen in Figure 5.5. There were no nonlinear systems and no Jacobians. The consumer models were able to meet their heat demand, figures of the result for the heat flow rate for the consumer models are excluded to reduce the number of figures. The power of the heater is given by Figure 5.6.

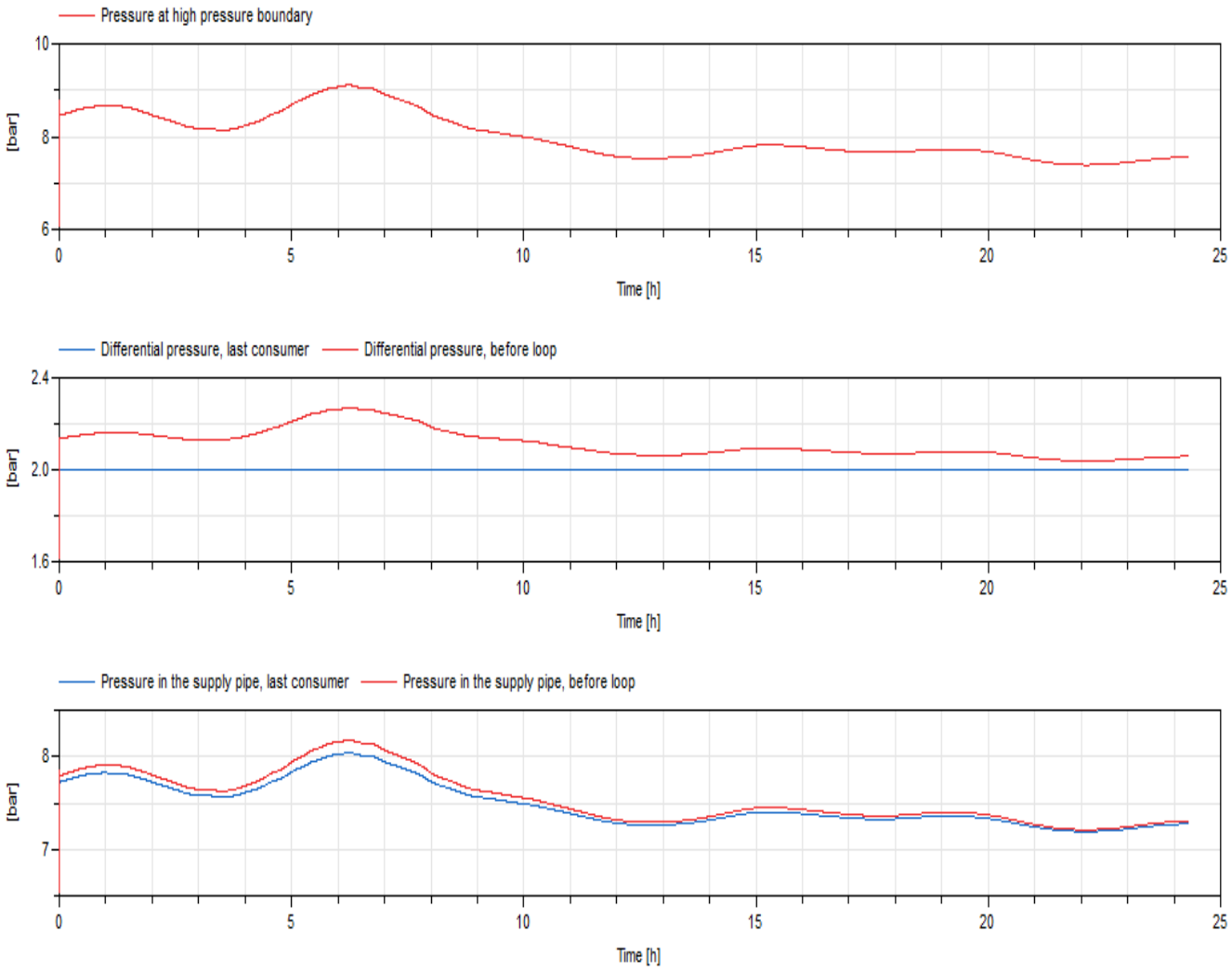

Figure 5.5. Pressure and differential pressure in different places of the network.

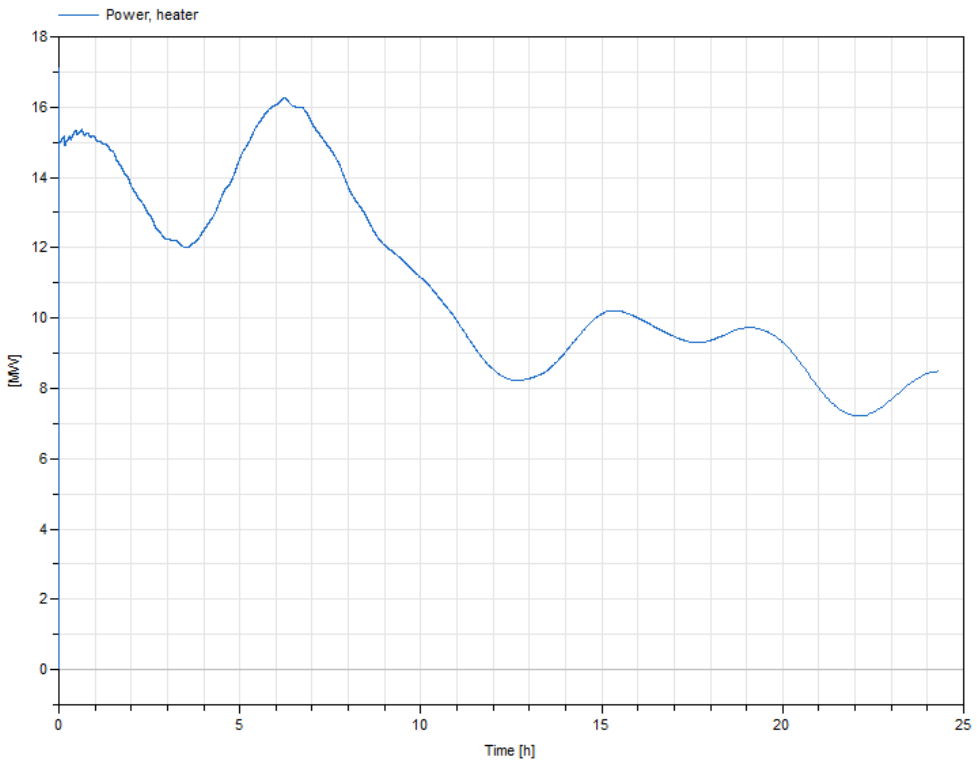

Figure 5.6. The power the ideal heater have to have to reach an outgoing temperature of 90°C.

The results show that the PI-controlled pressure boundary was able to keep a differential pressure of two bars at the last consumer throughout the simulation. During periods of high heat demand, the pressure would have to be increased. This is because there is more water flowing through the consumer models, which would lead to a lower pressure at the last consumer otherwise. When the hot water starts to reach the different consumer models the pressure can decrease due to the fact that a lower mass flow is needed when the temperature of the water is increased.

A higher pressure leads to a higher mass flow and if there is higher mass flow the heater has to work more which can be noted in Figure 5.5. Since it is an ideal heater there is no max output of the power and the power of the heater is only depending on the return temperature and the mass flow since the outgoing temperature is always set to 90°C.

5.2 Looped Network

To ensure the simulation handles loops as well, a test network was created. The layout of this network can be seen in Figure 5.6. The network contains fifty consumers and has the same properties as the branched network, but instead of only a branch this network connects the branch in the end and creates a loop. The network is controlled with the same principle as the branched network with the PI-controller setting the pressure.

The red pipes in Figure 5.6 are used because otherwise volume models would be connected to volume models. Both pressure and differential pressure are measured at the last consumer, consumer before the loop and the last consumer at the bottom inside the loop.

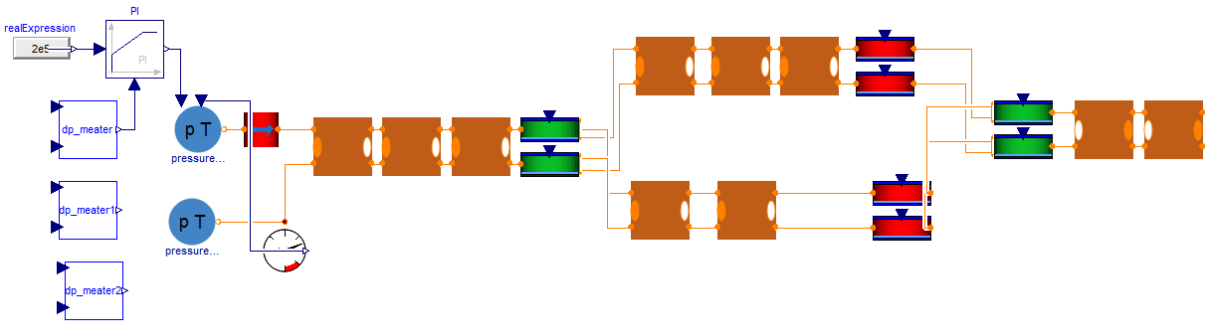

Figure 5.6 Layout of a looped network. It's similar to the branched network but is connected again after the branch to get a loop.

5.2.1 Results

The simulation time for this looped network was 40 seconds. The result for the different pressures can be seen in Figure 5.7. The power of the heater can be seen in Figure 5.8.

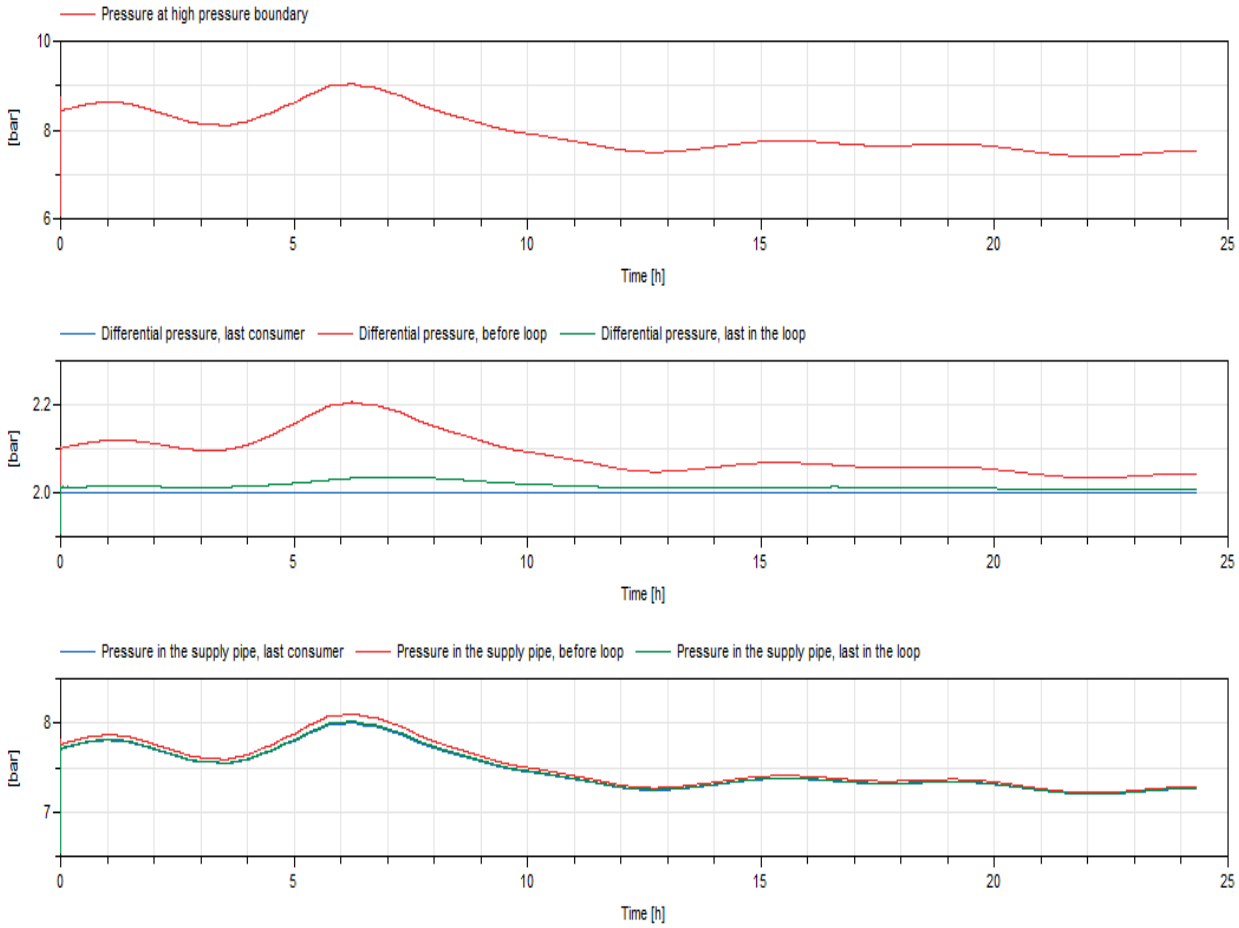

Figure 5.7. Results from the simulations regarding pressure in various points in the network

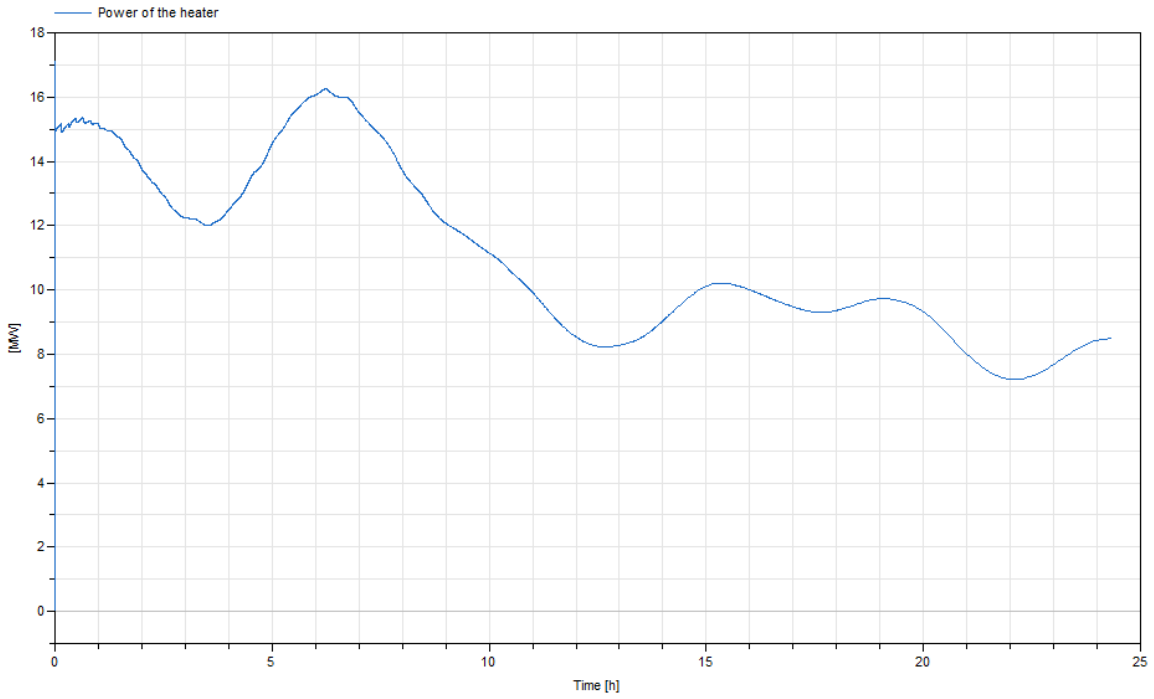

Figure 5.8. The power of the ideal heater.

The result is similar to the branched network, which was expected since the properties and the load profiles were the same. The simulation time was increased a bit compared to the branched network although the looped network is more complex than previous network and the increase in simulation time is not extreme. There seems to be no problem to handle a systems that includes loops using the created models.

5.3 Two producers

In larger district heating networks, there are often several producers in the network. The location of these producers can be at the end of a network or even in the middle. For this simulation a producer model used as network producer was created which is basically ideal heater with a control signal. The control signals was used to decide the mass flow that should be taken from the return pipe to go through the network producer. This simulation was also done to ensure that a whole network handles reverse flow. The layout of the simulation is shown in Figure 5.9.

The setup is that the network producer starts its heater after 30 000 seconds and increase the mass flow up to 20 kg/s into the heater over a 10 000 second period. When the network producer starts its heater, there is likely to be reverse flow. The left producer was controlled as the previous simulations to ensure that the differential pressure over the last consumer was two bars.

The properties were the same as the two previous simulations with the PI-controller and the same pipe diameter, etc. However the number of consumers were only forty in order to easily manipulate the flow through the heater to get reverse flow.

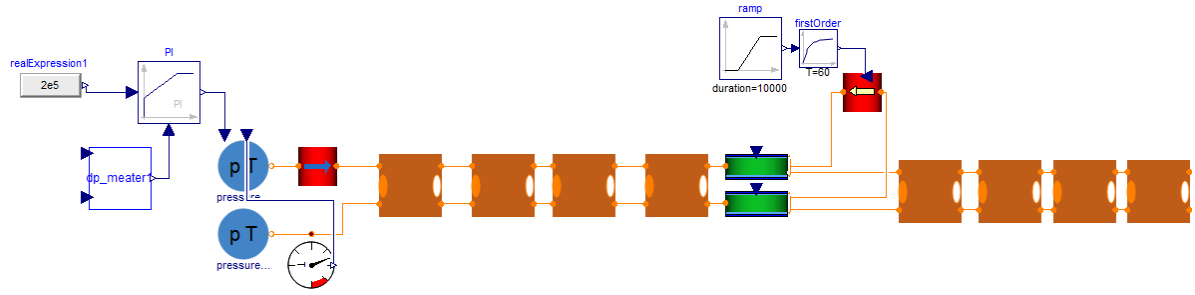

Figure 5.9. Layout of district heating network with two producers.

5.3.1 Results

The result from the simulation can be seen in Figure 5.10. The simulation time was 50 seconds and there were no nonlinear system or numerical Jacobian.

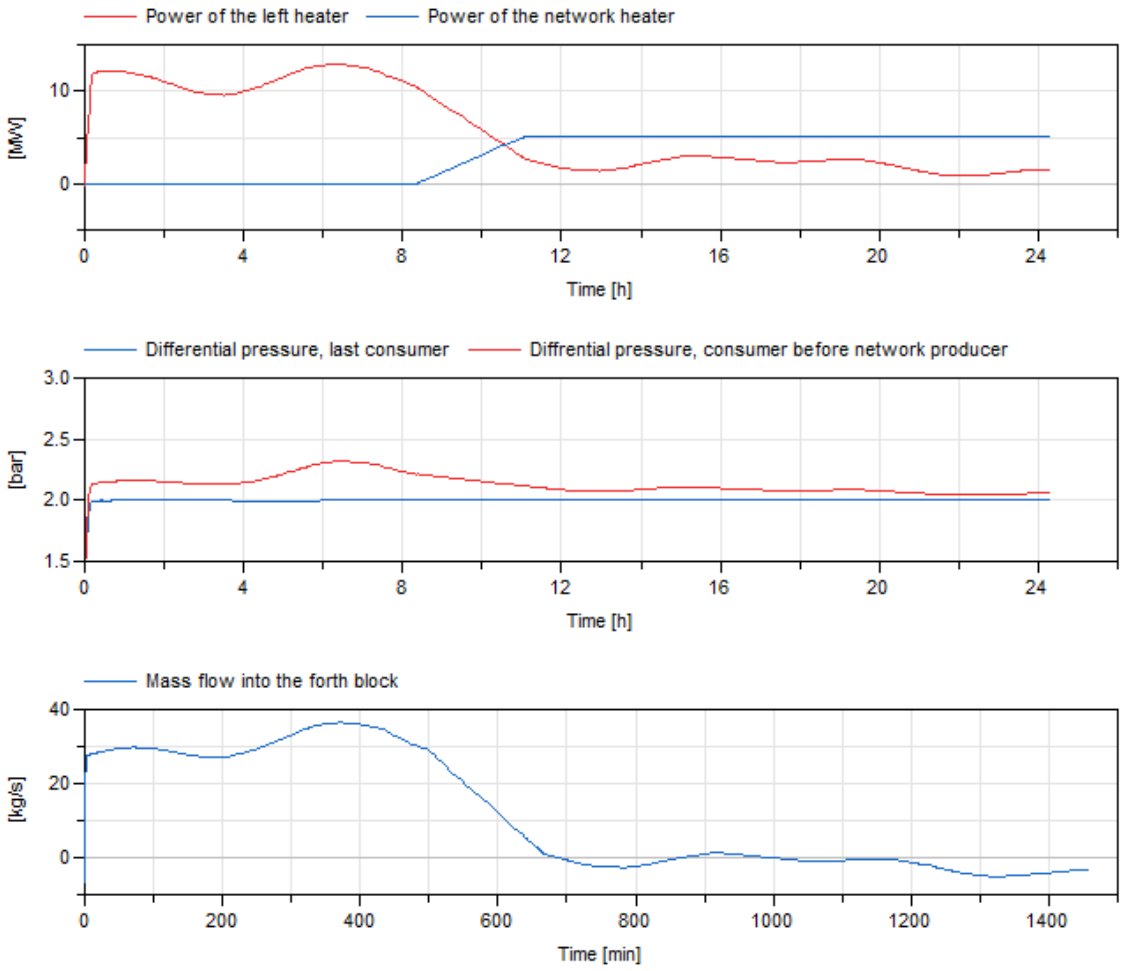

Figure 5.10. The result from a simulation of a district heating network with two producers.

The simulation time for the network did not increase much by adding a second producer. Studying Figure 5.10 it's possible to see that the mass into the left side of the forth block left is negative, that means there is water flowing from the right to the left and we have reverse flow, concluding there is no problem with reverse flow in larger systems as well. The power of the original heater went down as well, as expected, when the network heater started.

5.4 Complex Network

The network layout used for simulation of a complex network can be seen in Figure 5.11. The network included both branching and loops to ensure it is possible to simulate networks containing both. A second producer was added to the network with the same properties as in the previous simulation. The pipe diameter before the first split was 0.3m and after the split all pipes had a diameter of 0.2m, this is done to ensure that the network also can handle networks with different sizes of pipe diameters. The network included 100 consumer models, 2 production unit and 248 pipe models.

The initial pressure for all components in the network was five bar and then the PI-controller increase the control signal until the desired differential pressure was reached for the last consumer. The temperature of the water inside all supply pipe was set to 80°C and the temperature for the water in the return pipe was set to 30°C as initial values. The water going out of the high pressure boundary has the same temperature as the going into the low pressure boundary and is then heated up to 90°C with the ideal heater.

A second simulation was done with the same setup but the control signal of the network producer was always set to zero. This was done to study the effect that the network producer has on the left producer.

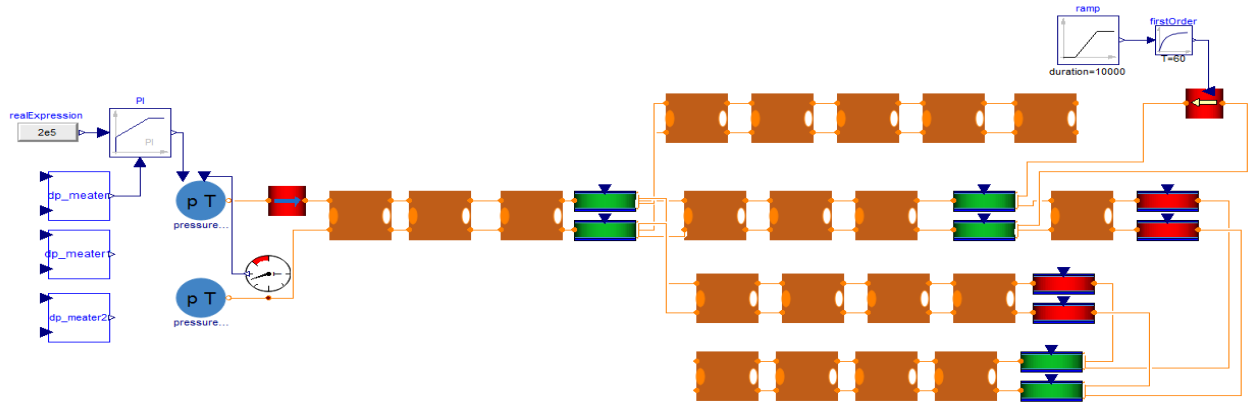

Figure 5.11. Layout of the complex network used for simulation. The network contains both a loop, two branches and two producers.

5.4.1 Results

The result for the pressure in various locations in the network for a simulation over a day's period can be seen in Figure 5.12. The graph show the pressure at the high pressure boundary, the differential pressure at the last consumer and two other consumers, to see that the pressure reduces the further away the producer the consumer is. The simulation time for this network was 100 seconds.

.

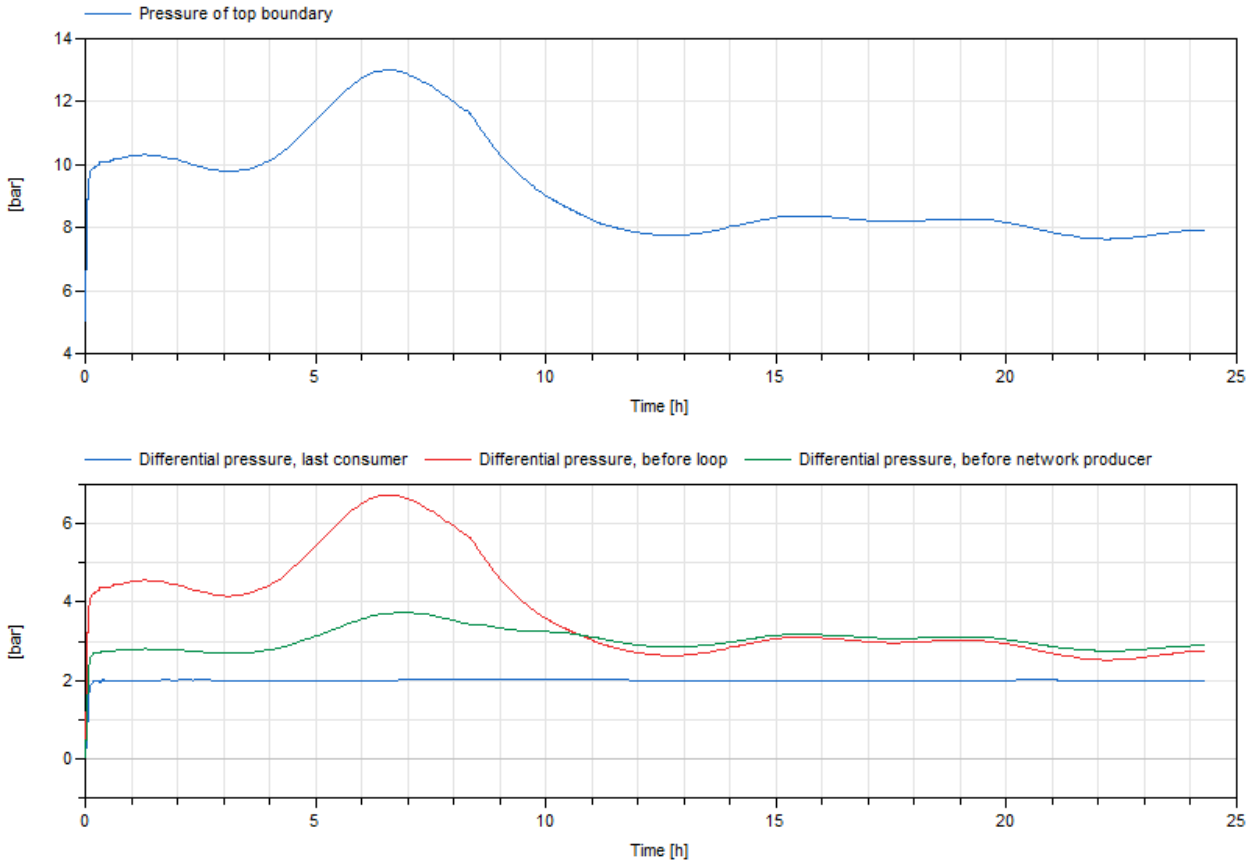

Figure 5.12. The pressure needed to fulfill a differential pressure of two bars at the last consumers and the differential pressure at various point in the network.

Comparisons of the power of the producers in the different simulations can be seen in Figure 5.13. The blue line show the left heater without adding a network producer. Red is the same heater but the network producer now assist to heat up the water

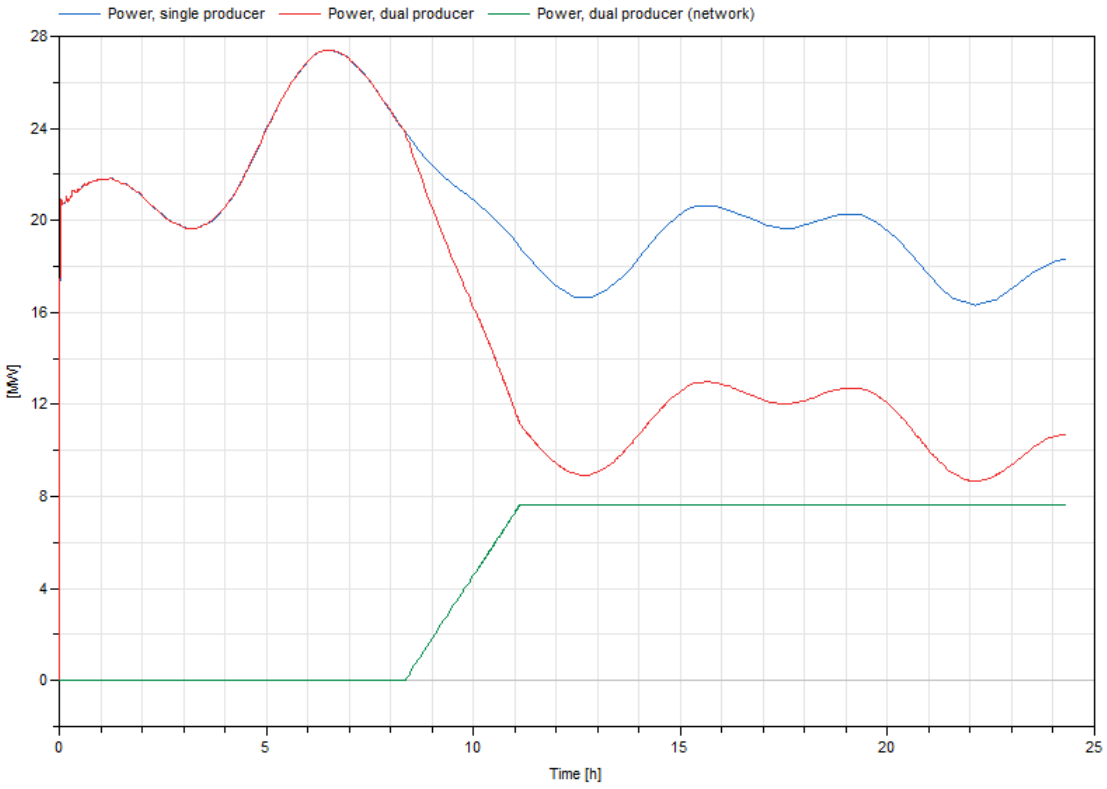

Figure 5.13. Power of heaters from the two different simulations.

The simulation shows that the models are compatible for a complex network with both loops, branches and two producers in the same network. Even with complex network the PI-controller has no problem to reach two bars as differential pressure.

Figure 5.13 shows that the power of left producer is reduced by adding a second producer, as it should.

6. Discussion

The implemented models show that it's possible to simulate larger district heating networks in Dymola. Both the single component simulation and network simulation show the expected results compared to real world pipes and networks. The models have no problem to handle complex networks with loops, branches, several producer or even reverse flow.

Major improvements have been done in simulation statistic by using the spatial distribution operator compared to the finite volume method used in the pipe from Modelica standard library.

The heat loss implementation for the pipes has been compared to the heat loss of an existing network and the simulated temperature shows promising results. However the simulation only test one size of the pipe and one thickness of the insulation since this was the only data available and further investigation should be done on this topic.

6.1. Future work

The base of a pressure driven district heating system have been implemented although some work still remains to be done. The components work as intended and there is room to add complexity to study the results. One of these complexity could be to introduce more realistic heat exchanger in the consumer model. Another is the use of a more realistic producer that include startup time, peak power, cost, etc.

At the moment, the highest amount of consumer models simulated were 100. This should be enough to simulate most networks if parts of the network are lumped together, meaning instead of model with multiple houses with different load profiles a part of a city is modeled as one consumer with an aggregated load profile. But to get more accurate results aggregation should be implemented and studied to see the effect it has on the network. There are probably still possibilities to increase the number of consumers since the simulation time is still fairly fast.

References

Claytex Services Limited. (2011, June 20). *How can I make my models run faster?* Retrieved February 30, 2016, from Claytex: http://www.claytex.com/blog/how-can-i-make-my-models-run-faster/

Dymola. (2015). (Modelon) Retrieved March 30, 2016, from Modelon: http://www.modelon.com/products/dymola/

Frederiksen, S., & Werner, S. (2013). *District Heating and Cooling.* Lund: Svensk fjärrvärme, Studentlitteratur.

Giraud, L., Bavière, R., Vallée, M., & Paulus, C. (2015). *Presentation, Validation and Application of the DistrictHeatingModelica Library.* Univ. Genoble Alpes, Grenoble.

Larsen, H. V., Bøhm, B., & Wigbels, M. (2003). A comparison of aggregated models for simulation. *Energy Conversion and Management*, p1119-1139.

Larsen, H. V., Pálsson, H., Bøhm, B., & Ravn, F. H. (2002). Aggregated dynamic simulation model of district. *Energy Conversion and Management*, p995-1019.

Larsson, H. (2015). *District Heating Network Models for Production Planning.* Lund: Department of Automatic Control.

Modelica. (2012). *Language Specification Version 3.3.* Linköping: Modelica Association. Retrieved December 10, 2015, from https://www.modelica.org/documents/ModelicaSpec33.pdf

Velut, S., & Tummescheit, H. (n.d.). *Implementation of a transmission line model for fast simulation of fluid flow dynamics.* Lund: Modelon AB.