K2 Model Database - Tutorial and Reference Manual

Nilsson, Bernt; Eborn, Jonas

1994

Document Version:
Publisher's PDF, also known as Version of record

ISSN 0280–5316
ISRN LUTFD2/TFRT--7528--SE

K2 Model Database - Tutorial and Reference Manual

Bernt Nilsson
Jonas Eborn

Department of Automatic Control
Lund Institute of Technology
December 1994

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
TECHNICAL REPORT

Date of issue
December 1994

Document Number
ISRN LUTFD2/TFRT--7528--SE

Author(s)
Bernt Nilsson and Jonas Eborn

Supervisor

Sponsoring organisation
Sydkraft AB and NUTEK, proj.nr. 9304688-2

Title and subtitle
K2 Model Database - Tutorial and Reference Manual

Abstract

The K2 model database is a set of Omola model libraries for modelling of thermal power plants. The models describe mainly the dynamic behaviour in the water/steam cycle.

The libraries are decomposed into three groups, namely model component libraries, subunit libraries and unit libraries. The model component libraries contain general functions, variables, terminals and super classes. The subunits are models of different media, compartment and flow resistor subunits. Examples of media are subcooled water, water/steam mixtures, superheated steam and flue gas. The compartment models are control volumes containing different media and the flow resistors describe different relationships between flow and pressure. The unit libraries contain models of typical physical objects, like pumps, valves, heat exchangers and more complex units, like boiler and superheater systems.

The K2 model database is used to model a heat recovery steam generation plant, HRSG. The HRSG model is used in a case study to simulate different operating conditions.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280–5316

ISBN

Language
English

Number of pages
82

Recipient's notes

Security classification

1. Introduction

The K2 model database is used to make dynamic models of thermal power plants. The K2 model database consists of a number of Omola model libraries which contain model classes. The primary goal for the use of the model database is to be able to simulate a small power plant. An application study on a *heat recovery steam generation* plant, HRSG, has been done and is presented in a companion report, [Eborn and Nilsson, 1994]. The K2 project is focused on the development of the model libraries but includes also a number of side effects concerning large model databases, version control, static solver and user interface.

The application study in the K2 project was focused on a heat recovery steam generation plant. Flue gas from a gas turbine is used to boil water and the steam is then used to run a steam turbine for power generation. The low pressure steam after the turbine is condensed and recycled to the deaerator. The K2 model database is used to describe the water and steam cycle of the plant. Additional model databases are used for controller descriptions and application dependent configurations. The application study is described in the report "Object-Oriented Modelling and Simulation of a Power Plant", see [Eborn and Nilsson, 1994].

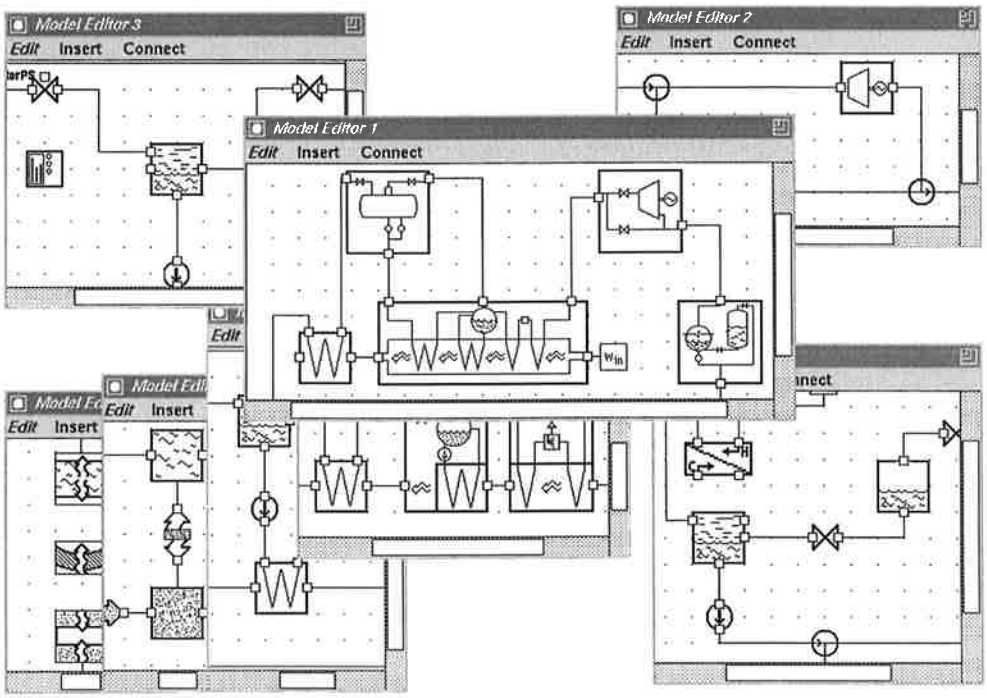

Figure 1. The HRSG application study.

The K2 model database is composed of twelve (12) model libraries. The modelling is supposed to be divided in three different levels. Model components are reused together with the Omola language to describe subunits capturing a certain phenomenon. Subunits can then be used to describe unit models characterizing physical objects. These unit models are used to build up systems. The K2 model database is composed of four model component libraries, four subunit libraries and four unit libraries.

The K2 model database follows the guidelines presented in [Nilsson, 1993] and in [Nilsson, 1994]. The guidelines discuss the decomposition of the structure hierarchy and the class inheritance hierarchy.
Omola is the modelling language used to describe the models in the K2 model database. Omola is an object-oriented modelling language developed at the Department of Automatic Control. An interactive modelling and simulation environment, called OmSim, can handle Omola models. A good overview of Omola is found in [Mattsson and Andersson, 1992] and a tutorial in [Andersson, 1993]. Mats Andersson, the inventor of Omola, presents a detailed description on the language and the idea behind it in [Andersson, 1994].
OmSim uses the file system to develop model databases. An OmSim library is a directory with the library classes in one file each. A set of library directories is grouped under another directory. This directory tree is called an Omola model database. OmSim can load multiple model databases.

Acknowledgements

The authors would like to thank the ones behind OmSim, Sven Erik Mattsson, Mats Andersson and Tomas Schönthal. We also must thank Jan Tuszynski and Ola Bernersson at Sydkraft Konsult AB for many valuable discussions. The K2 project has been financed in three different research programs, namely by Sydkraft, NUTEK power system research program and by NUTEK complex system research program.

References

ANDERSSON, M. (1993): "OmSim and Omola tutorial and user's manual." Report ISRN LUTFD2/TFRT--7504--SE, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

ANDERSSON, M. (1994): *Object-Oriented Modeling and Simulation of Hybrid Systems.* PhD thesis ISRN LUTFD2/TFRT--LUTFD2/TFRT–1043–SE--SE, Lund Institute of Technology.

EBORN, J. and B. NILSSON (1994): "Object-oriented modelling and simulation of a thermal power plant." Technical Report TFRT-7527, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

MATTSSON, S. E. and M. ANDERSSON (1992): "The ideas behind Omola." In *Proceedings of the 1992 IEEE Symposium on Computer-Aided Control System Design, CADCS '92*, Napa, California.

NILSSON, B. (1993): *Object-Oriented Modeling of Chemical Processes.* PhD thesis ISRN LUTFD2/TFRT--1041--SE, Dept. of Automatic Control, Lund Institute of Technology, Lund, Sweden.

NILSSON, B. (1994): "Guidelines for Process Model Libraries using an Object-Oriented Approach." In *European Simulation Multiconference*, pp. 349–353.

PART I: Tutorial

2. How To Use K2

The basic idea is to separate model development into three main modelling levels. These modelling levels are mapped into the class tree and model library organization.

1. System modelling (units to system).
2. Unit modelling (subunits to unit).
3. Phenomena modelling (equations to subunit).

A K2 user of the first level develops system by using the unit models found in the K2 unit libraries. The unit libraries contain flow units, heating units, turbines and sensors.
A K2 user on the second level develops units using the subunit model libraries. The basic idea in the subunit level is the separation into compartment and flow resistors. The compartments are control volumes with dynamic mass and energy balances. The flow resistors are flow descriptions as functions of a driving force from surrounding compartments.
The third level is supported by the Omola language which allows the user to develop his or her own models. There are also K2 model component libraries that supports an expert with predefined classes, like heat resistance functions and terminal class definitions.
The K2 libraries are therefore grouped into three categories, namely unit libraries, subunit libraries and model component libraries. Each individual model class can be found in respective library.

Unit Libraries

The unit models are categorized into three model libraries:

1. Flow units in `K2FlowUnitLib`:
   - Valve
   - Pump
   - Junction
   - Split
2. Heating units in `K2HeatUnitLib`:
   - Heat exchanger
   - Economizer
   - Superheater
   - Superheater system
   - Boiler system
3. Turbine unit in `K2TurbineLib`:
   - Steam turbine
4. Sensor unit in `K2SensorLib`:
   - Temperature sensor

Many of the units are available for different media, namely water, steam and flue gas. Almost all of the unit models have an internal structure of subunits. One example of a unit like this is the heat exchanger.

Heat exchanger example: The heat exchanger model is composed of two compartments describing the two sides of the unit. Each compartment model

describes the dynamics of the medium in it. The heat interaction is described in a heat flow resistor connecting the two compartments. On each compartment outflow there are flow resistors describing the relation between outflow and pressure difference.

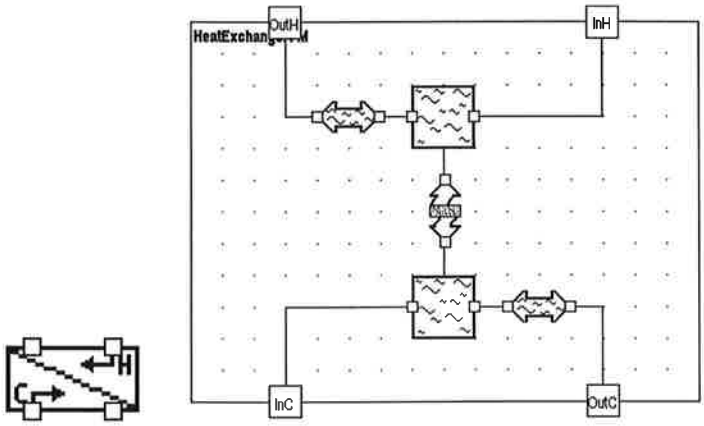

Figure 2. The heat exchanger unit. The icon on the left and internal configuration of compartments and flow resistors on the right.

System modelling: The units are intended to be used as submodels in system descriptions. This system may be a subsystem of a larger system. An example is to build a boiler section out of one heat exchanger, one boiler drum, control valves and one circulation pump. This boiler section is then used in a pan configuration which is used in a water and steam cycle plant. A small

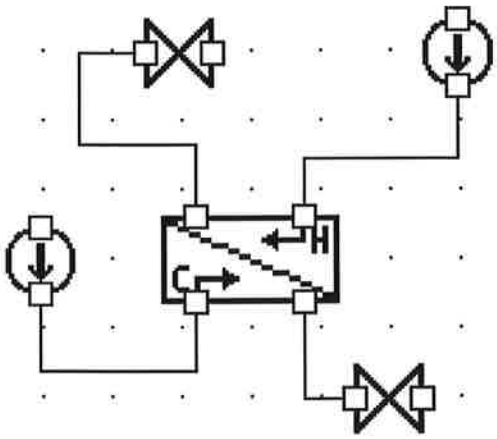

Figure 3. The use of a heat exchanger unit in a system configuration.

system using some units is found in Figure 3.
To make proper connections between units is a problem that may occur. The terminals must be of the same type. The unit models have control terminals and flow terminals. The flow terminals can have different directions and represent different phases. The internal terminal variables are in SI units but there is no guarantee that this means water or steam. Some units allow both and some do not.
A second problem is to assign proper values to parameters and initial values to variables. All parameters and variables are in SI units.

Subunit Libraries

The subunits can be divided into two major categories, namely compartments and flow resistors. The flow resistors are divided into flow and heat flow resistors. There is also a medium model library for medium descriptions, which are used as submodels inside compartment and flow resistor subunits.

1. Compartment subunits in `K2CompartmentLib`.
   - One phase compartments for water, steam and flue gas.
   - Two phase compartment for water and steam.
2. Flow resistor subunits in `K2FlowLib`.
   - One phase flow resistors for water, steam and gas.
   - Critical expansion flow resistor for steam.
   - Loss and friction factor functions for water, steam and gas.
3. Heat flow resistor subunits in `K2HeatFlowLib`.
   - Heat flow resistors.
   - Heat resistance functions.
4. Medium models are found in `K2MediumLib`.
   - Compartment medium models for water, saturated water and steam, steam and gas.
   - Flow resistor medium models for water, saturated water and steam, steam and gas.

Compartment models: The major dynamics are described in the compartment models. Dynamic mass and energy balances are nonlinearly transformed into pressure and enthalpy which are the dynamic states. The states, pressure and enthalpy, are used to calculate other medium specific variables like density, temperature, heat capacity etc. This is done by the use of medium and machine decomposition. All medium specific calculations are described in medium submodels inside the compartment. Inside the medium models the states are used as input arguments to steam table functions which return the desired value. Steam and gas compartments function like this. Water is assumed to be incompressible which means that the water compartment can not have a dynamic mass balance. Flash and drum compartments on the other hand have an additional description of the mixture of water and steam.

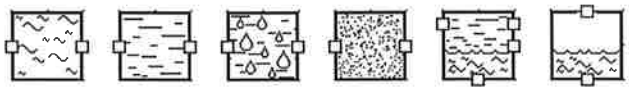

Figure 4. The compartment subunit models for different media.

Flow resistor models: The flow resistors describe the static relation between a gradient and flow. For water, steam and gas flows the gradient is the pressure drop and the medium flow is a nonlinear function of this gradient. Critical expansion of steam describes the flow when the pressure drop is so big that the flow velocity is near to the speed of sound. In the heat flow resistor the heat flow is modelled as a function of the temperature gradient or logarithmic temperature gradient.

Medium models: Medium models are used in both compartment and flow resistor subunits. These models function as ordinary functions with input arguments and output arguments. Compartment medium models need the states,

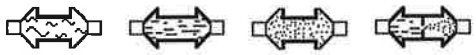

Figure 5. The flow resistor subunit models for different media.

pressure and enthalpy, to calculate physical properties like density, temperature etc. Other physical properties are needed in flow calculations, like viscosity, and therefore there are special medium models for flow resistors. These calculations are done either by calling steam tables implemented in OmSim or using fitted polynomial functions.

Figure 6. The medium subunit models for different media.

The advantage with this decomposition into one machine dependent part and one medium dependent part is that it makes it easy to change the medium description. In this case it is possible to change the steam table function calls because they are encapsulated in the medium model.

Unit modelling: Subunits are intended to be used in order to model a unit. One example of how they are used is seen in Figure 2 which shows the internal structure of an heat exchanger. One basic assumption is that compartments and flow resistors must be connected to each other. Two compartments can not be connected because there will be no flow description for this interaction. On the other hand two flow resistors may be connected, but this will generate an algebraic equation system.

K2 Model Component Libraries

There are four additional libraries containing model components. These model components are used to build subunit and unit model classes but are also necessary during system model building. Examples are terminal classes for flowing media.

1. Functions and variables in `K2BasicLib`:
   - Logarithmic mean function
   - Heat resistance functions for tube geometry and different medium.
2. Terminals in `K2TerminalLib`:
   - Basic terminal classes for physical quantities.
   - Record terminals for flowing media.
3. End terminals in `K2EndTerminalLib`:
   - Record terminals with a given experimental setup.
4. Super classes in `K2ClassTreeLib`:
   - Super classes for the organization of the K2 model database class tree.

The basic library contain function classes of general interest. There are a lot of different heat resistance descriptions.
The terminal library is used at every modelling level, system, unit and subunit modelling. The terminal classes are important because they contain the interaction information.
Units and systems can often not be simulated directly. They lack a number of surrounding conditions. The end terminal classes describe typical conditions and are therefore used connecting to units. Units together with proper end terminals can be simulated directly.
All classes in the K2 model database has super classes that are defined in the class tree library.

PART II: Reference Manual

3. K2 Organization

The K2 model database is composed of twelve model libraries: four unit libraries, four subunit libraries and four model component libraries. In principle the four model component libraries together with the predefined Omola model classes are used to develop the subunit models. The subunit model libraries are used to develop unit models. An application user uses the unit libraries and sometimes the subunit libraries to develop systems in his or her own application model library.

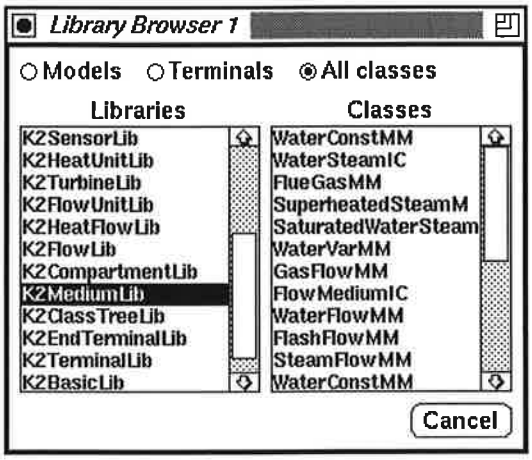

Figure 7. The OmSim browser for the K2 library.

The unit libraries are the first four in Figure 7, `K2SensorLib`, `K2HeatUnitLib`, `K2TurbineLib` and `K2FlowUnit`. The next four libraries contain the subunits, `K2HeatFlowLib`, `K2FlowLib`, `K2CompartmentLib` and `K2MediumLib`. The last four libraries are model components and they are called `K2ClassTreeLib`, `K2EndTerminalLib`, `K2TerminalLib` and `K2BasicLib`.
All K2 libraries are presented in detail in the following sections and they are presented in the reversed order with `K2BasicLib` as the first one. Almost every model class definition are listed in the report, almost 70 classes. Model classes that are not found to be of general interest are not listed, less than 20 classes.

4. K2BasicLib

This K2 library contain variable, parameter and function classes of general interest.

FunctionIC is a common super class for function classes.

LogMean is a function for logarithmic mean temperature calculations.

Th is a function that approximates the flue gas temperature as a function of the enthalpy.

FlowVC* is a flow variable class.

PressureVC* is a pressure variable class.

PressureUpVC* is another pressure variable class.

HeatResistanceIC is the interface class for heat resistance function classes.

GasResistance calculates the convective resistance on the gas side of a tube.

SteamResistance calculates the convective resistance of steam inside a tube.

WaterResistance calculates the convective resistance of water inside a tube.

WallResistance calculates the conductive resistance of the metal wall of a tube.

NoResistance has no heat resistance.

A class with a * is not listed in this report. It is not found to be of general interest.

Function Super Class and LogMean

```
FunctionIC ISA Class WITH
%% This is a super class for other
%%   functions.
variable:
  value TYPE Real;
END;


LogMean ISA FunctionIC WITH
%% This function calculate the
%%   logarithmic mean difference.
%%   For small differences an
%%   approximation is used.
parameters:
  Small ISA Parameter WITH default := 0.05;END;
variables:
  x, y TYPE Real;
equations:
  value = IF (abs(x)<Small) OR (abs(y)<Small)
          THEN 0
          ELSE
            IF abs(x-y) < Small*max(x,y)
            THEN (x+y)/2*(1 - (x-y)*(x-y)/(12*x*y)*(1 - (x-y)*(x-y)/(2*x*y)))
            ELSE (x-y)/ln(x/y);
END;



Th ISA K2BasicLib::FunctionIC WITH
%% This function calculates the fluegas temperature (K)
%%   from enthalpy (J/kg). A polynomial fitted to data
%%   between 350 and 900K is used.
%%   The data was obtained by implicit use of the function
%%   htp with the gas composition [.6911,.078,.1697,0,0,0,.0341]
variables:
  h ISA Variable;
equations:
  value = K2BasicLib::flueTpoly3+h*(K2BasicLib::flueTpoly2+
    h*K2BasicLib::flueTpoly1);
END;
```

Heat Resistance Classes

```
HeatResistanceIC ISA K2ClassTreeLib::HeatResistorCC WITH
%% This is an interface class for heat
%%   resistance functions
terminal:
  Rin ISA K2TerminalLib::HeatMediumTC WITH
    Graphic ISA Layout WITH
      x_pos := 200;
      y_pos := 150;
      invisible := 1;
    END;
  END;
parameters:
  C,n1,n2 ISA Parameter;
variables:
  value TYPE Real;
END;


NoResistance ISA HeatResistanceIC WITH
%% NoResistance is a function class that
%%   calculates no heat resistance.
%%
%% Assumptions: no heat resistance.
%% Model Use:   nothing given
%% Medium:      nothing
%% Model type:  full.
%%
equations:
  value := 0;
END;
```

GasResistance

```
GasResistance ISA HeatResistanceIC WITH
%% GasResistance is a function class that
%%   calculates the heat resistance on
%%   the gas side of a haet exchanger
%%   tube.
%%
%% Assumptions: constant area
%%              turbulent flow
%% Model Use:   given log mean temperature
%%              given gas flow and pressure
%%              given gas mix
%% Medium:      fluegas
%% Model type:  full.
%%
icons:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "GasResistance";
  END;
parameters:
  C := 0.33;
  n1 := 0.6;
  n2 := 0.33;
  d, l ISA Parameter; % diameter and length
  Across ISA Parameter WITH
    value := K2BasicLib::pi*sqr(d)/4;
  END;
  Aheat ISA Parameter WITH
    value := K2BasicLib::pi*d*l;
  END;
variables:
  Cp TYPE Real; % Sp. heat capacity [J/kgK]
  mu TYPE Real; % dynamic viscosity [Pas]
  lambda TYPE Real; % thermal conduct.  [W/Km]
  Re, Pr, Nu TYPE Real; % Dimensionless numbers
  Tm TYPE Real; % Mean temperature  [degC]
equations:
  % ------------------------------------ medium calculations
  Cp = cptp(Tm+K2BasicLib::T0,Rin.p,trans(Rin.Gmix));
  mu = 1e-6*(22+0.035*(Tm-100));
  lambda = 1e-3*(31+0.05775*(Tm-100));
  % ------------------------------------ dimensionless numbers
  Re = abs(Rin.w)*d/Across/mu;
  Pr = Cp*mu/lambda;
  Nu = C*Re^n1*Pr^n2;
  % ------------------------------------ Function evaluation
  value = 1/(Nu*lambda*Aheat/d);
END;
```

SteamResistance

```
SteamResistance ISA HeatResistanceIC WITH
%% SteamResistance is a function that calculates
%%   the heat resistance on the steam side inside
%%   heat exchanger tube.
%%
%% Assumptions: constant area
%%              turbulent flow
%% Model Use:   given log mean temperature
%%              given steam flow
%% Medium:      Superheated steam
%% Model type:  full.
%%
icons:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "SteamResistance";
  END;
parameters:
  C := 0.023;
  n1 := 0.8;
  n2 := 0.4;
  d, l ISA Parameter;                    % diameter and length
  Across ISA Parameter WITH
    value := K2BasicLib::pi*sqr(d)/4;
  END;
  Aheat ISA Parameter WITH
    value := K2BasicLib::pi*d*l;
  END;
variables:
  Cp TYPE Real;                          % Sp. heat capacity [J/kgK]
  mu TYPE Real;                          % dynamic viscosity [Pas]
  lambda TYPE Real;                      % thermal conduct.  [W/Km]
  Re, Pr, Nu TYPE Real; % Dimensionless numbers
  Tm TYPE Real;                          % Mean temperature  [degC]
equations:
  % ------------------------------------ medium calculations
  Cp = 2000; % Avoid chattering
  mu = 1e-6*(12+0.0436*(Tm-100));
  lambda = 1e-3*(24+0.13*(Tm-100));
  % ------------------------------------ dimensionless numbers
  Re = abs(Rin.w)*d/Across/mu;
  Pr = Cp*mu/lambda;
  Nu = C*Re^n1*Pr^n2;
  % ------------------------------------ Function evaluation
  value = 1/(Nu*lambda*Aheat/d);
END;
```

WaterResistance

```
WaterResistance ISA HeatResistanceIC WITH
%% WaterResistance is a function class that
%%   calculates the heat resistance on
%%   the water side inside a of a haet exchanger
%%   tube.
%%
%% Assumptions: constant area
%%              turbulent flow
%% Model Use:   given log mean temperature
%%              given water flow and pressure
%% Medium:      water
%% Model type:  full.
%%
icons:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "WaterResistance";
  END;
parameters:
  C.default := 0.023;
  n1.default := 0.8;
  n2.default := 0.4;
  d, l ISA Parameter;                    % diameter and length
  Across ISA Parameter WITH
    value := K2BasicLib::pi*sqr(d)/4;
  END;
  Aheat ISA Parameter WITH
    value := K2BasicLib::pi*d*l;
  END;
variables:
  Cp TYPE Real;                          % Sp. heat capacity [J/kgK]
  mu TYPE Real;                          % dynamic viscosity [Pas]
  lambda TYPE Real;                      % thermal conduct.  [W/Km]
  Re, Pr, Nu TYPE Real; % Dimensionless numbers
  Tm TYPE Real;                          % Mean temperature  [degC]
equations:
  % ------------------------------------ medium calculations
  Cp = cptp(Tm+::T0,Rin.p);
  mu = 1e-3*(0.25+28*(1/Tm-0.009));
  lambda = 1e-3*(685-0.005*sqr(Tm-140));
  % Appr. expressions for mu, lambda from 'Data och Diagram' p72
  % They are good in the temperature range 50-300 degC
  % ------------------------------------ dimensionless numbers
  Re = abs(Rin.w)*d/Across/mu;
  Pr = Cp*mu/lambda;
  Nu = C*Re^n1*Pr^n2;
  % ------------------------------------ Function evaluation
  value = d/Nu/lambda/Aheat;
END;
```

WallResistance

```
WallResistance ISA HeatResistanceIC WITH
%% WallResistance is a function class that
%%   calculates the heat resistance in the
%%   metalic wall of tube.
%%
%% Assumptions: constant area
%%              homogenouus metal
%%              tube geometry
%% Model Use:   given log mean temperature
%% Medium:      stainless steal
%% Model type:  full.
%%
icon:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "WallResistance";
  END;
submodels:
  Rin ISA Class; % Override medium terminal
parameters:
  d, l ISA Parameter; % diameter and length
  delta ISA Parameter; % wall thickness
variables:
  lambda TYPE Real; % thermal conduct.  [W/Km]
  Tm TYPE Real; % Mean temperature  [degC]
equations:
  % ------------------------------------ medium calculations
  lambda = 13.24+0.0012*Tm;
  % ------------------------------------ Function evaluation
  value = ln(1+2*delta/d)/2/::pi/lambda/l;
END;
```

5. K2TerminalLib

Terminals are important model components that are often defined globally. In the K2 terminal library there are classes for medium flows and heat flows. The terminal class corresponds to a physical variable. Simple terminals are so

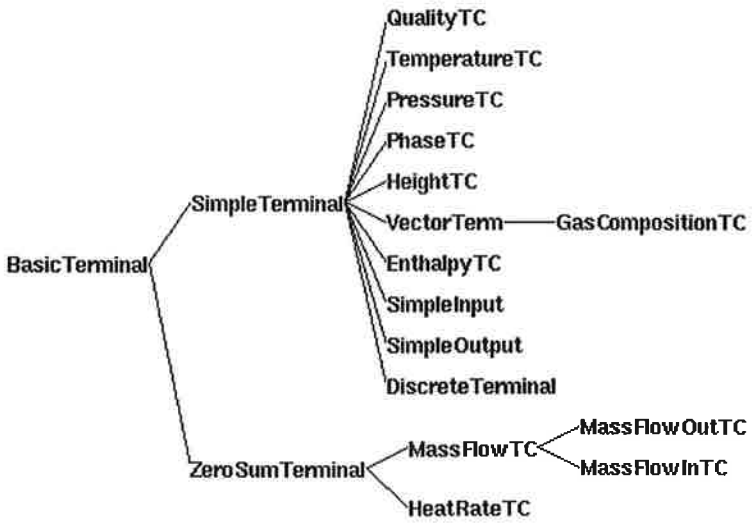

Figure 8. The class hierarchy of basic terminal classes in the K2 terminal library.

called across variables where a connection is interpreted as an equality. The zero sum terminals are so called through variables which are assumed to sum to zero in a connection. They have an attribute describing the direction of a positive value. The basic terminals shown in Figure 8 can be used as subclasses

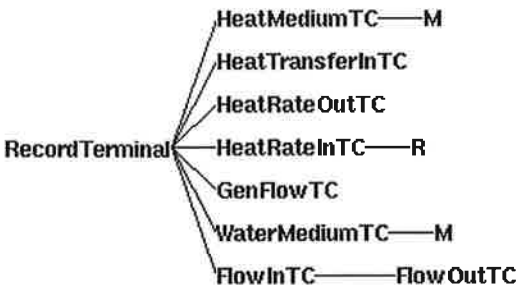

Figure 9. The record terminals in K2 terminal library.

in record terminals like `FlowInTC` below. The flow in terminal class has four attributes describing mass flow, pressure, enthalpy and medium. The flow in and flow out terminals are commonly used to describe medium flows in the K2 models.

```
FlowInTC ISA RecordTerminal WITH
  w ISA MassFlowInTC;
  p ISA PressureTC;
  h ISA EnthalpyTC;
  M ISA WaterMediumTC;
END;
```

Record terminals that describe flows also contain a subterminal containing information about the flowing medium. This information is defined in the medium subterminal which is a record terminal. There are different medium subterminals for different media as seen in Figure 10.

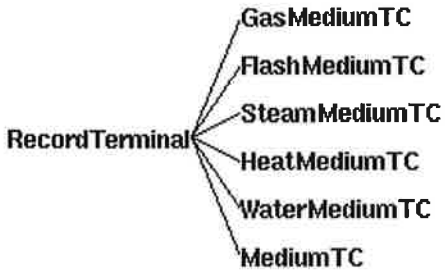

Figure 10. The record terminals describing medium properties in K2 terminal library.

6. K2EndTerminalLib

The terminal classes in `K2TerminalLib` is used inside the model structure hierarchy. To use a unit model from the K2 libraries one has to connect the terminals to a given experimental setup. This setup is described by the end terminals. The listed classes are the most common experimental setups used. Other combinations are possible, but of limited use.

CompWinTC defines a mass flow into a compartment model.

GasCompWinTC defines a gas mass flow into a compartment model.

WaterCompWinTC defines a water mass flow into a compartment model.

SteamCompWinTC defines a steam mass flow into a compartment model.

GasFlowPoutTC defines a gas pressure out from a flow model.

WaterFlowPoutTC defines a steam pressure out from a flow model.

WaterFlowWoutTC defines a water mass flow out from a flow model.

WaterFlowWinTC defines a water mass flow into a flow model.

WaterFlowPinTC defines a water pressure into a flow model.

SteamFlowPinTC defines a steam pressure into a flow model.

7. K2ClassTreeLib

This library contains super classes that branch the class tree into a number conceptual parts. There is almost no important information inherited in this part of the class tree. The super class for all model classes developed in K2

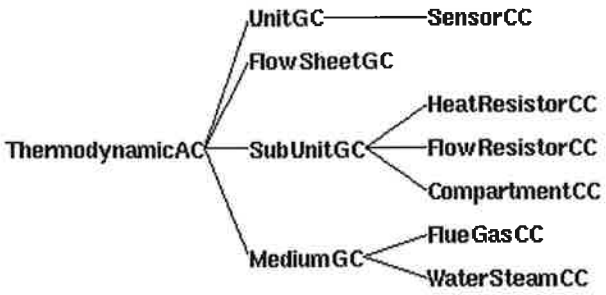

Figure 11. The tree organization levels of the class tree.

model database are `ThermodynamicAC`. AC stands for application class and in an application study there can be a number of different ACs, like ControlAC, ElectricAC etc. The next level are the granularity class level, GC. It branches the tree into medium, subunits, units and systems. The third level is the so called conceptual class level. The classes on this level are super classes for models that are conceptually related. Examples are heating units, vessels, pumps, valves etc.

8. K2MediumLib

The K2 medium library contain model classes that describe medium subunits. These subunits are used in compartment and flow resistor models. The medium subunits are modules that encapsulate all access to medium functions and calculations.

WaterSteamIC is the interface class to medium models used inside compartment subunit models.

WaterConstMM is a subcooled water model with constant density.

WaterVarMM is a subcooled water model with variable density.

SaturatedWaterSteamMM is saturated water/steam mixture.

SuperheatedSteamMM is a model describing superheated steam.

FlueGasMM is a flue gas medium model.

FlowMediumIC is the interface class for medium models that are used inside flow resistor subunits.

SteamFlowMM for steam flow.

WaterFlowMM for water flow.

FlashFlowMM for water/steam mixture flow.

GasFlowMM for flue gas flow.

Figure 12. The compartment medium models for water, flash, superheated steam and for flue gas.

WaterSteamIC

```
WaterSteamIC ISA WaterSteamCC WITH
  %% A medium model interface class for water and steam.
  %%
  %% Assumptions: enthalpy and pressure are medium states.
  %% Model Use:   given interface with compartment models.
  %% Model type:  interface class
  %%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "WaterMedium";
  END;
terminals:
  Min ISA RecordTerminal WITH
    p, hin, hout ISA SimpleTerminal;
  END;
  Mout ISA RecordTerminal WITH
    ap, ah, rho, Tkin, Tkout ISA SimpleTerminal;
  END;
  Mout2 ISA RecordTerminal WITH
    hw, hs, rhow, rhos, alpha ISA SimpleTerminal;
  END;
END;
```

WaterConstMM

```
WaterConstMM ISA WaterSteamIC WITH
%% A medium model describing the thermodynamic
%%   properties of the processed medium.
%%
%% Assumptions : medium states are pressure and enthalpy.
%%               constant density.
%% States      : static
%% Model use   : inside compartment models
%% Model type  : medium model. (Subcooled Water)
%%
parameters:
  T0 ISA Parameter WITH value:=273.16; END;
variabels:
  Tc,Tbelow ISA Variable;
equations:
  %-------------------------- constant density
  Mout.rho = 1000;
  %-------------------------- not used in water models
  Mout.ap := 0;
  Mout.ah := 0;
  %-------------------------- temperatures
  Mout.Tkin  = THP(Min.hin,Min.p);
  Mout.Tkout = THP(Min.hout,Min.p);
  %-------------------------- temp in Celcius
  Tc = Mout.Tkout - T0;
  %-------------------------- Tbelow is temp below boiling
  Tbelow = Mout.Tkout - TP(Min.p);
  %-------------------------- Set Mout2 (Not used in Water)
  Mout2.hw    = 0;
  Mout2.hs    = 0;
  Mout2.rhow  = 0;
  Mout2.rhos  = 0;
  Mout2.alpha = 0;
END;
```

WaterVarMM

```
WaterVarMM ISA WaterSteamIC WITH
%% A medium model describing the thermodynamic
%%   properties of the processed medium.
%%
%% Assumptions : medium states are pressure and enthalpy.
%%               variable density.
%% States      : static
%% Model use   : inside compartment models
%% Model type  : medium model. (Subcooled Water)
%%
parameters:
  eps    ISA Parameter WITH default:=0.001; END;
  s1, s2 ISA Parameter;
variables:
  Tc, Tbelow ISA Variable;
  dvdh, dvdp ISA Variable;
equations:
  %-------------------------- constant density
  Mout.rho = 1/VHP(Min.hout,Min.p);
  % ------------------------------------ thermal properties
  % approximation of derivatives
  dvdp = (VHP(Min.hout,(1+eps)*Min.p) - VHP(Min.hout,(1-eps)*Min.p)) /
    (2*eps*Min.p);
  dvdh = (VHP((1+eps)*Min.hout,Min.p) - VHP((1-eps)*Min.hout,Min.p)) /
    (2*eps*Min.hout);
  %-------------------------- partial derivatives
  Mout.ap := IF s1<0.5 THEN -dvdp * Mout.rho*Mout.rho
             ELSE 0.8115/(Min.p/1e5)^1.77;
  Mout.ah := IF s2<0.5 THEN -dvdh * Mout.rho*Mout.rho
             ELSE -68860/Min.p;
  % Approximate expressions valid for subcooled water in the
  % pressure range 0.2-5bar, perhaps.
  %-------------------------- temperatures
  Mout.Tkin  = THP(Min.hin,Min.p);
  Mout.Tkout = THP(Min.hout,Min.p);
  %-------------------------- temp in Celcius
  Tc = Mout.Tkout - K2BasicLib::T0;
  %-------------------------- Tbelow is temp below boiling
  Tbelow = Mout.Tkout - TP(Min.p);
  %-------------------------- Set Mout2 (Not used in Water)
  Mout2.hw    = 0;
  Mout2.hs    = 0;
  Mout2.rhow  = 0;
  Mout2.rhos  = 0;
  Mout2.alpha = 0;
END;
```

SaturatedWaterSteamMM

```
SaturatedWaterSteamMM ISA WaterSteamIC WITH
%% A medium model describing the thermodynamic
%%   properties of the processed medium.
%%
%% Assumptions : medium states are pressure and enthalpy.
%% States      : static
%% Model use   : inside compartment models
%% Model type  : medium model. (Saturated Water/Steam)
%%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "FlashMedium";
  END;
parameters:
  eps    ISA Parameter WITH default:=0.001; END;
  s1, s2 ISA Parameter;
variabels:
  x, Tc, Ttest ISA Variable;
  dvdh, dvdp, dvsdp, dvwdp, dhsdp, dhwdp ISA Variable;
equations:
  % ------------------------------------ saturation enthalpy
  Mout2.hw = H1P(Min.p);
  Mout2.hs = H2P(Min.p);
  % ------------------------------------ steam ratio (0 -> 1)
  x = MIN(1 , MAX(0 , (Min.hout - Mout2.hw)/(Mout2.hs - Mout2.hw)));
  % ------------------------------------ density
  Mout2.rhow = 1/V1P(Min.p);
  Mout2.rhos = 1/V2P(Min.p);
  Mout.rho = 1/((1-x)/Mout2.rhow + x/(Mout2.rhos));
  % ------------------------------------ amount of water
  Mout2.alpha = (Mout2.rhow-Mout.rho)/(Mout2.rhow-Mout2.rhos);
  % ------------------------------------ thermal properties
  % approximation of derivatives
  dvsdp = (V2P((1+eps)*Min.p) - V2P((1-eps)*Min.p))/(2*eps*Min.p);
  dvwdp = (V1P((1+eps)*Min.p) - V1P((1-eps)*Min.p))/(2*eps*Min.p);
  dhsdp = (H2P((1+eps)*Min.p) - H2P((1-eps)*Min.p))/(2*eps*Min.p);
  dhwdp = (H1P((1+eps)*Min.p) - H1P((1-eps)*Min.p))/(2*eps*Min.p);
  % partial derivative under constant h
  dvdp = x*( (dvsdp - dvwdp) - dvdh*(dhsdp - dhwdp) )
         - dvdh*dhwdp + dvwdp;
  dvdh = (1/Mout2.rhos - 1/Mout2.rhow)/(Mout2.hs - Mout2.hw);
  % alpha-koefficients
  Mout.ap = IF s1<0.5 THEN -dvdp * Mout.rho*Mout.rho
            ELSE Mout.rho/Min.p;
  Mout.ah = IF s2<0.5 THEN -dvdh * Mout.rho*Mout.rho
            ELSE  -Mout.rho/Min.hout;
  % ------------------------------------ temperature
  Mout.Tkin  = THP(Min.hin,Min.p);
  Mout.Tkout = THP(Min.hout,Min.p);
  Tc = Mout.Tkout - K2BasicLib::T0;
  Ttest = Mout.Tkout - TP(Min.p);
END;
```

SuperheatedSteamMM

```
SuperheatedSteamMM ISA WaterSteamIC WITH
%% A medium model describing the thermodynamic
%%   properties of the processed medium.
%%
%% Model type  : medium model (Superheated Steam)
%% Assumptions : ideal gas.
%%               medium states are pressure and enthalpy.
%% States      : static
%% Model use   : inside compartment models
%% Tests       : check of superheated temperature.
%%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "SteamMedium";
  END;
variables:
  Tc, Tbelow ISA Variable;
equations:
  % ------------------------------------ density
  Mout.rho = 1/VHP(Min.hout,Min.p);
  % ------------------------------------ heat coefficients
  Mout.ap = Mout.rho/Min.p;
  Mout.ah = -Mout.rho/Min.hout;
  % ------------------------------------ temperatures
  Mout.Tkin  = THP(Min.hin,Min.p);
  Mout.Tkout = THP(Min.hout,Min.p);
  % ------------------------------------ temp in Celcius
  Tc = Mout.Tkout - K2BasicLib::T0;
  % ------------------------------------ temp below boiling
  Tbelow = Mout.Tkout - TP(Min.p);
  %----------------Set Mout2 (Not used in SuperheatedSteam)
  Mout2.hw    = 0;
  Mout2.hs    = 0;
  Mout2.rhow  = 0;
  Mout2.rhos  = 0;
  Mout2.alpha = 0;
END;
```

FlueGasMM

```
FlueGasMM ISA WaterSteamIC WITH
%% A medium model describing the thermodynamic
%%   properties of the flue gas medium.
%%
%% Model type  : medium model (Flue Gas)
%% Assumptions : ideal gas.
%%               medium states are pressure and enthalpy.
%% States      : static
%% Model use   : inside compartment models
%%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "FlueGas";
  END;
terminal:
  Gin ISA GasCompositionTC;
  Mout ISA super::Mout WITH
    Tkin, Tkout ISA K2BasicLib::Th;
  END;
parameters:
  T0 ISA Parameter WITH value:=273.16; END;
  MW ISA Std::VectorPar WITH
    % Mole Weight vector [ N2, CO2, O2, Ar, SO2, NO, H2O]
    n := 7;
    default TYPE Column[n] := [28.01;44.01;32.00;39.95;64.06;30.01;18.02];
  END;
variables:
  Tc ISA Variable;
equations:
  % ------------------------------------ density
  Mout.rho = Min.p/(Mout.Tkout*(K2BasicLib::R/(trans(MW)*Gin)));
  % ------------------------------------ heat coefficients
  Mout.ap = Mout.rho/Min.p;
  Mout.ah = -Mout.rho/Min.hout;
  % ------------------------------------ temperatures
  % NOTE: fitted equations
  Mout.Tkin.h  = Min.hin;
  Mout.Tkout.h = Min.hout;
  % ------------------------------------ temp in Celcius
  Tc = Mout.Tkout - T0;
  % -------------------------- Set Mout2 (Not used in Flue Gas)
  Mout2.hw    = 0;
  Mout2.hs    = 0;
  Mout2.rhow  = 0;
  Mout2.rhos  = 0;
  Mout2.alpha = 0;
END;
```

FlowMediumIC

```
FlowMediumIC ISA WaterSteamCC WITH
terminals:
  Min ISA RecordTerminal WITH
    p, h ISA SimpleTerminal;
  END;
  Mout ISA RecordTerminal WITH
    rho, my, Cp, lambda, kappa ISA SimpleTerminal;
  END;
END;
```

SteamFlowMM

```
SteamFlowMM ISA FlowMediumIC WITH
%% A medium model describing the thermodynamic
%%   properties of the processed medium.
%%
%% Model type  : medium model. (Superheated steam)
%% Assumptions : medium states are pressure and enthalpy.
%% States      : static
%% Model use   : inside flow and heat modules
%%
variables:
  Tk,Tc,Test ISA Variable;
equations:
  Mout.rho = 1/VHP(Min.h,Min.p); % kg/m3
  Tk = THP(Min.h,Min.p); % K
  Tc = Tk - K2BasicLib::T0; % degC
  Test = Tk - TP(Min.p); % K
  Mout.my = (12+0.0436*(Tc-100))/1000; % Pa/s
  % Approximate expression for my from 'Data och Diagram' p78
  Mout.Cp = CPTP(Tk,Min.p); % J/kgK
  Mout.lambda = 1e-3*(24+0.13*(Tc-100));%
  Mout.kappa = 1/(1 - (K2BasicLib::R/K2BasicLib::M)/Mout.Cp);
END;
```

WaterFlowMM

```
WaterFlowMM ISA FlowMediumIC WITH
%% A medium model describing the thermodynamic
%%   properties of the processed medium.
%%
%% Model type  : medium model. (Subcooled Water)
%% Assumptions : medium states are pressure and enthalpy.
%%               constant density.
%% States      : static
%% Model use   : inside flow modules
%%
parameters:
  T0 ISA Parameter WITH value:=273.15; END;
variabels:
  Tk,Tc,Test ISA Variable;
equations:
  Mout.rho = 1000; % kg/m^3 (water)
  Tk = THP(Min.h,Min.p); % K      (water)
  Tc = Tk - T0; % C
  Test = Tk - TP(Min.p); % K
  Mout.my := 1e-3*(0.25+28*(1/Tc-0.009));% Pa/s
  Mout.lambda := 1e-3*(685-0.005*sqr(Tc-140));%
  % Appr. expressions for mu, lambda from 'Data och Diagram' p72
  % They are good in the temperature range 50-300 degC
  Mout.Cp := CPTP(Tk,Min.p);
  Mout.kappa := 1/(1 - (K2BasicLib::R/K2BasicLib::M)/Mout.Cp);
END;
```

FlashFlowMM

```
FlashFlowMM ISA FlowMediumIC WITH
%% A medium model describing the thermodynamic
%%   properties of the processed medium.
%%
%% Model type  : medium model. (Saturated Water/Steam)
%% Assumptions : medium states are pressure and enthalpy.
%% States      : static
%% Model use   : in flow modules
%%
parameters:
  eps    ISA Parameter WITH default:=0.01; END;
variables:
  x, Tk, Tc, Test ISA Variable;
  hw, hs ISA Variable;
equations:
  % ------------------------------------ saturation enthalpy
  hw = H1P(Min.p);
  hs = H2P(Min.p);
  % ------------------------------------ steam ratio (0 -> 1)
  x = MIN(1 , MAX(0 , (Min.hout - hw)/(hs - hw)));
  % ------------------------------------ density
  Mout.rho = (1-x)/V1P(Min.p) + x/V2P(Min.p);
  % ------------------------------------ temperature
  Tk  = THP(Min.h,Min.p);
  Tc = Tk - K2BasicLib::T0;
  Test = Tk - TP(Min.p);
  % ------------------------------------ medium attributes
  Mout.Cp := x*CPTP(Tk+eps,Min.p) + (1-x)*CPTP(Tk-eps,Min.p);
  Mout.kappa := 1/(1 - K2BasicLib::R/Mout.Cp/K2BasicLib::M);
  Mout.my := 1e-3*(0.25+28*(1/Tc-0.009));% Pa/s
  Mout.lambda := 1e-3*(685-0.005*sqr(Tc-140));
  % Expressions for mu, lambda from 'Data och Diagram' p72
  % Here the expressions for water are used as an approximation.
END;
```

GasFlowMM

```
GasFlowMM ISA FlowMediumIC WITH
%% A medium model describing the thermodynamic
%%   properties of the processed medium.
%%
%% Model type  : medium model. (Heated flue gas)
%% Assumptions : medium states are pressure and enthalpy.
%%               constant density.
%% States      : static
%% Model use   : inside flow modules
%%
variables:
  M TYPE Matrix[1,1];
  Tk,Tc ISA Variable;
  GasMix ISA Std::VectorPar WITH
    % Composition description [ N2, CO2, O2, Ar, SO2, NO, H2O]
    n := 7;
    default TYPE Column[n] := [0.6911;0.078;0.1697;0;0;0;0.0341];
  END;
  MW ISA Std::VectorPar WITH
    % Mole Weight vector [ N2, CO2, O2, Ar, SO2, NO, H2O]
    n := 7;
    default TYPE Column[n] := [28.01;44.01;32.00;39.95;64.06;30.01;18.02];
  END;
equations:
  M = trans(MW)*GasMix; % Moleweight
  Mout.rho = Min.p/Tk/K2BasicLib::R*M; % kg/m3
%  Min.h = HTP(Tk,Min.p,trans(GasMix)); % K, implicit equation
  Tc = Tk - K2BasicLib::T0; % degC
  Mout.my := 1e-6*(22+0.035*(Tc-100)); % Pa/s
  Mout.lambda := 1e-3*(31+0.05775*(Tc-100));
  Mout.Cp := CPTP(Tk,Min.p,trans(GasMix));% J/kgK
  Mout.kappa = 1/(1 - K2BasicLib::R/Mout.Cp/M);
% Assignment doesn't work with 1x1 matrix.
END;
```

9. K2CompartmentLib

This library contains compartment classes. These models describe mass and energy dynamics. The states are transformed into pressure and enthalpy.

CompartmentIC is an interface class with one inflow, one outflow and one heat interaction terminal.

WaterCompartmentFM describes a water medium with a dynamic enthalpy balance and one static pressure balance. Uses a simplified medium description.

WaterVarCompFM* describes a water medium with dynamic enthalpy and pressure balances. Uses a variable water medium description.

SteamCompartmentFM is modelled with dynamic pressure and enthalpy balances.

GasCompartmentFM is similar to SteamCompartmentFM but with additional gas composition description.

FlashCompartmentFM is similar to SteamCompartmentFM but with an additional description of the mixture of water and steam.

OpenCompartmentFM is similar to WaterVarCompFM but with fixed pressure (usually atmospheric) and variable water volume. This means that the pressure dynamics are replaced by volume dynamics.

Compartment2IC* is an interface class to drum compartment.

DrumCompartmentFM describe the dynamic enthalpy and pressure balance for water and steam at saturation. The drum has a developed water level and two outflows: steam and water.

Compartment3IC* is an interface class for water node models.

WaterNode11* is a water compartment with two flows.

WaterNode12* is a water compartment with three flows.

A class with a * is not listed in this report. It is not found to be of general interest. The compartment icons with terminals are seen in Figure 13. The

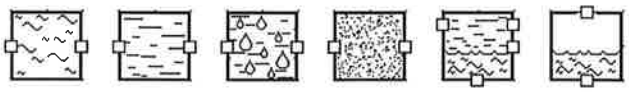

Figure 13. The compartment subunit models for different media.

inflow terminal is usually at the left and the outflow at the right. The heat terminal is hidden at the center of the icon. Note that variations occur.

CompartmentIC

```
CompartmentIC ISA CompartmentCC WITH
  %% Interface class of a compartment model with
  %%   one inflow and one outflow.
  %%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "WaterCompartment";
  END;
terminals:
  Fin ISA FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 1;
      y_pos := 151;
    END;
  END;
  Fout ISA FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400;
      y_pos := 151;
    END;
  END;
  Qin ISA HeatTransferInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201;
      y_pos := 151;
      invisible := 1;
    END;
  END;
END;
```

WaterCompartmentFM

```
WaterCompartmentFM ISA CompartmentIC WITH
  %% A control volume model of a liquid
  %%   medium (water) based on
  %%   dynamic energy and static mass balances.
  %%
  %% Assumptions: constant volume,
  %%              homogenous mixed,
  %%              incompressible (static pressure balance),
  %%              no momentum description,
  %%              no heat interaction,
  %%              no work interaction.
  %% Model Use:   given mass flow direction,
  %%              mass flows are described elsewhere.
  %% States:      enthalpy (h).
  %% Medium:      subcooled water (h < 4.1e5).
  %% Model type:  full.
  %%
parameters:
  V ISA Parameter;
  height ISA Parameter;
variables:
  % states: p = pressure, h = enthalpy
  h ISA Variable;
  p ISA Variable;
  % balance derivitives
  dm ISA Variable;
  de ISA Variable;
equations:
  % ------------------------------------ mass balance (static)
  Fin.w = Fout.w;
  dm := 0;
  % ------------------------------------ energy balance (dynamic)
  de = Fin.w*Fin.h - Fout.w*Fout.h + Qin.R.q;
  % ------------------------------------ state equations (enthalpy)
  Fin.p = p;
  Fout.p = p;
  h' = 1/(M.Mout.rho*V)*de;
  Fout.h = h;
medium:
  M ISA WaterConstMM;
medium_connections:
  %------------------------- medium state to medium model
  M.Min.p    = p;
  M.Min.hout = Fout.h;
  M.Min.hin  = Fin.h;
  %------------------------- medium temp to heat transfer
  Qin.R.Tin  = M.Mout.Tkin;
  Qin.R.Tout = M.Mout.Tkout;
  % ------------------------------------ phase to medium terminal
  Fin.M.q    := 'Water;
  Fout.M.q   := 'Water;
  Fin.M.z    := height;
  Fout.M.z   := height;
heat_medium_connections:
  Qin.M.p    := p;
  Qin.M.w    := Fout.w;
  Qin.M.Gmix := [0;0;0;0;0;0;1.0];
END;
```

SteamCompartmentFM

```
SteamCompartmentFM ISA CompartmentIC WITH
  %% A control volume model of a
  %%   superheated steam medium based on
  %%   dynamic energy and mass balances.
  %%
  %% Assumptions: constant volume,
  %%              homogenous mixed,
  %%              no momentum description,
  %%              no work interaction.
  %% Model Use:   given mass flow directions,
  %%              mass flows are described elsewhere.
  %% States:      pressure (p),
  %%              enthalpy (h).
  %% Medium:      superheated steam.
  %% Model type:  full model.
  %%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "SteamCompartment";
  END;
parameters:
  V ISA Parameter;
submodels:
  Fin ISA super::Fin WITH
    M ISA SteamMediumTC;
  END;
  Fout ISA super::Fout WITH
    M ISA SteamMediumTC;
  END;
variables:
  % states: p = pressure, h = enthalpy
  p ISA Variable;
  h ISA Variable;
  % balance derivitives
  dm, de ISA Variable;
  dV ISA Variable;
  % auxilary variables
  K1, K2 ISA Variable;
  T11, T12, T13 ISA Variable;
  T21, T22, T23 ISA Variable;
equations:
  % ------------------------------------ mass balance
  dm = Fin.w - Fout.w;
  % ------------------------------------ energy balance
  de = Fin.w*Fin.h - Fout.w*Fout.h + Qin.R.q;
  dV := 0;
  % ------------------------------------ auxiliary variables
  K1 = M.Mout.ap*M.Mout.rho + M.Mout.ah;
  K2 = M.Mout.rho*M.Mout.rho/p + M.Mout.ah;
  T11 = (M.Mout.rho + M.Mout.ah*h)/K1/V;
  T12 = -M.Mout.ah/K1/V;
  T13 = -p/K1/V*K2;
  T21 = (T11 - h/V)/M.Mout.rho;
  T22 = (1 - M.Mout.ah/K1)/M.Mout.rho/V;
  T23 = (1 - K2/K1)*p/M.Mout.rho/V;
  % ------------------------------------ transform into pressure and enthalpy
  p' = T11*dm + T12*de + T13*dV;
  Fin.p = p;
  Fout.p = p;
  h' = T21*dm + T22*de + T23*dV;
```

```
  Fout.h = h;
medium:
  M ISA SuperheatedSteamMM;
medium_connections:
  %------------------------- medium state to medium model
  M.Min.p    := p;
  M.Min.hout := Fout.h;
  M.Min.hin  := Fin.h;
  %------------------------- medium temp to heat transfer
  Qin.R.Tin  := M.Mout.Tkin;
  Qin.R.Tout := M.Mout.Tkout;
  % ----------------------------------- phase to medium terminal
  Fin.M.q    := 'Steam;
  Fout.M.q   := 'Steam;
  Fin.M.T    := M.Mout.Tkin;
  Fout.M.T   := M.Mout.Tkout;
heat_medium_connections:
  Qin.M.p    = p;
  Qin.M.w    = Fout.w;
  Qin.M.Gmix := [0;0;0;0;0;0;1.0];
END;
```

GasCompartmentFM

```
GasCompartmentFM ISA CompartmentIC WITH
  %% A control volume model of a gas medium based
  %%   on dynamic energy and mass balances.
  %%
  %% Assumptions: constant volume,
  %%              static gas composition
  %%              homogenous mixed,
  %%              no work interaction.
  %% Model Use:   given mass flow directions,
  %%              mass flows are described elsewhere.
  %% States:      pressure (p),
  %%              enthalpy (h).
  %% Medium:      nonideal gas.
  %% Model type : full model.
  %%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "GasCompartment";
  END;
parameters:
  V, height ISA Parameter;
  GasMix ISA Std::VectorPar WITH
    % Composition description [ N2, CO2, O2, Ar, SO2, NO, H2O]
    n := 7;
    default TYPE Column[n] := [0.6911;0.078;0.1697;0;0;0;0.0341];
  END;
submodels:
  Fin ISA super::Fin WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400;
      y_pos := 151;
    END;
    M ISA GasMediumTC;
  END;
  Fout ISA super::Fout WITH
    Graphic ISA super::Graphic WITH
      x_pos := 1;
      y_pos := 151;
    END;
    M ISA GasMediumTC;
  END;
variables:
  p ISA Variable; % Pressure
  h ISA Variable; % Enthalpy
  dm, de ISA Variable; % Balance equation vars
  dV, K1, K2 ISA Variable;         % aux variables
  T11, T12, T13 ISA Variable;
  T21, T22, T23 ISA Variable;
equations:
  % ------------------------------------ mass balance
  dm = Fin.w - Fout.w;
  % ------------------------------------ energy balance
  de = Fin.w*Fin.h - Fout.w*Fout.h + Qin.R.q;
  dV := 0;                                % pressure-volume work
  % ------------------------------------ auxiliary variables
  K1 = M.Mout.ap*M.Mout.rho + M.Mout.ah;
  K2 = M.Mout.rho*M.Mout.rho/p + M.Mout.ah;
  T11 = (M.Mout.rho + M.Mout.ah*h)/K1/V;
  T12 = -M.Mout.ah/K1/V;
  T13 = -p/K1/V*K2;
```

```
  T21 = (T11 - h/V)/M.Mout.rho;
  T22 = (1 - M.Mout.ah/K1)/M.Mout.rho/V;
  T23 = (1 - K2/K1)*p/M.Mout.rho/V;
  % ------------------------------------ pressure/enthalpy transformation
  p' = T11*dm + T12*de + T13*dV;
  Fin.p = p;
  Fout.p = p;
  h' = T21*dm + T22*de + T23*dV;
  Fout.h = h;
medium:
  M ISA FlueGasMM;
medium_connections:
  %------------------------- medium state to medium model
  M.min.p    = p;
  M.min.hin  = Fin.h;
  M.min.hout = Fout.h;
  M.Gin      := GasMix;
  %------------------------- medium temp to heat transfer
  Qin.R.Tin  = M.Mout.Tkin;
  Qin.R.Tout = M.Mout.Tkout;
  % ------------------------------------ phase to medium terminal
  Fin.M.q    := 'Gas;
  Fout.M.q   := 'Gas;
  Fin.M.T    := M.Mout.Tkin;
  Fout.M.T   := M.Mout.Tkout;
heat_medium_connections:
  Qin.M.p    = p;
  Qin.M.w    = Fout.w;
  Qin.M.Gmix := GasMix;
END;
```

FlashCompartmentFM

```
FlashCompartmentFM ISA CompartmentIC WITH
  %% A control volume model of a mixture of
  %%   saturated water and steam described by
  %%   dynamic energy and mass balances.
  %%
  %% Assumptions: constant volume,
  %%              homogenous mixed,
  %%              no work interaction.
  %% Model Use:   given mass flow directions,
  %%              mass flows are described elsewhere.
  %% States:      pressure (p),
  %%              enthalpy (h).
  %% Medium:      saturated water and steam.
  %% Model type:  full.
icons:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "FlashCompartment";
  END;
parameters:
  V ISA Parameter;
  A ISA Parameter;
  DH ISA Parameter;
  height ISA Parameter;
parameter_propagation:
  Fin.z := height;
  Fout.z := height;
variables:
  p ISA Variable; % state variables
  h ISA Variable;
  dm ISA Variable; % balance equation variables
  de ISA Variable;
  dV ISA Variable; % terms in energy balance
  w ISA Variable;
  K1 ISA Variable; % auxil. variables
  K2 ISA Variable;
  T11 ISA Variable;
  T12 ISA Variable;
  T13 ISA Variable;
  T21 ISA Variable;
  T22 ISA Variable;
  T23 ISA Variable;
  level ISA Variable; % water level
equations:
  % ------------------------------------ mass balance
  dm = Fin.w - Fout.w;
  % ------------------------------------ energy balance
  de = Fin.w*Fin.h - Fout.w*Wout.h + Qin.q - Qout.q + w;
  w := 0;
  dV := 0;
  % ------------------------------------ auxiliary variables
  K1 = M.Mout.ap*M.Mout.rho + M.Mout.ah;
  K2 = M.Mout.rho*M.Mout.rho/p + M.Mout.ah;
  T11 = (M.Mout.rho + M.Mout.ah*h)/K1/V;
  T12 = -M.Mout.ah/K1/V;
  T13 = -p/K1/V*K2;
  T21 = (T11 - h/V)/M.Mout.rho;
  T22 = (1 - M.Mout.ah/K1)/M.Mout.rho/V;
  T23 = (1 - K2/K1)*p/M.Mout.rho/V;
  % ------------------------------------ pressure/enthalpy transformation
  p' = T11*dm + T12*de + T13*dV;
```

```
  Fin.p = p;
  Fout.p = p;
%  Wout.p = p + M.Mout2.rhow*level;
  h' = T21*dm + T22*de + T23*dV;
  Fout.h = if M.Mout2.alpha < (1 - K2BasicLib::eps) then M.Mout2.hs else h;
  % ------------------------------------ water level description
  level = DH*M.Mout2.alpha;
medium:
  M ISA SaturatedWaterSteamMM;
medium_connections:
  % ------------------------------------ medium states
  M.Min.p := p;
  M.Min.hin := Fin.h;
  M.Min.hout := h;
  %------------------------- medium temp to heat transfer
  Qin.Tin    := M.Mout.Tkin;
  Qin.Tout   := M.Mout.Tkout;
  Qout.Tin   := M.Mout.Tkin;
  Qout.Tout  := M.Mout.Tkout;
heat_medium_connections:
  R1.p       = p;
  R1.w       = Fout.w;
  R1.Gmix    := [0;0;0;0;0;0;1.0];
  R2.p       = p;
  R2.w       = Fout.w;
  R2.Gmix    := [0;0;0;0;0;0;1.0];
END;
```

OpenCompartmentFM

```
OpenCompartmentFM ISA CompartmentIC WITH
  %% A control volume model of water
  %%   in an open container, described by
  %%   dynamic energy and mass balances.
  %%
  %% Assumptions: constant pressure,
  %%              homogenous mixed,
  %%              no work interaction.
  %% Model Use:   given mass flow directions,
  %%              mass flows are described elsewhere.
  %% States:      volume (V),
  %%              enthalpy (h).
  %% Medium:      subcooled water.
  %% Model type:  full.
icons:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "OpenCompartment";
  END;
parameters:
  p ISA Parameter WITH default := K2BasicLib::p0;END;
  A ISA Parameter;
  DH ISA Parameter;
  height ISA Parameter;
terminals:
  Fin ISA super::Fin WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201;
      y_pos := 300;
    END;
  END;
  Fout ISA super::Fout WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201;
      y_pos := 1;
    END;
  END;
variables:
  V ISA Variable; % state variables
  h ISA Variable;
  dm ISA Variable; % balance equation variables
  de ISA Variable;
  K1 ISA Variable; % auxil. variables
  T11 ISA Variable;
  T12 ISA Variable;
  T21 ISA Variable;
  T22 ISA Variable;
  level ISA Variable; % water level
equations:
  % ------------------------------------ mass balance
  dm = Fin.w - Fout.w;
  % ------------------------------------ energy balance
  de = Fin.w*Fin.h - Fout.w*Fout.h + Qin.R.q;
  % ------------------------------------ auxiliary variables
  % there is a limiting fcn that takes care of division by zero
  K1 = sqr(M.Mout.rho) + M.Mout.ah*Fin.p;
  T11 = (p-M.Mout.rho*h)/K1/(V+exp(-sqr(V/K2BasicLib::eps)));
  T12 = M.Mout.rho/K1/(V+exp(-sqr(V/K2BasicLib::eps)));
  T21 = (M.Mout.rho + M.Mout.ah*h)/K1;
  T22 = -M.Mout.ah/K1;
  % ------------------------------------ enthalpy/volume transformation
```

```
  h' = T11*dm + T12*de;
  V' = T21*dm + T22*de;
  Fin.p = p;
  Fout.p = p+M.Mout.rho*K2BasicLib::g*level;
  Fout.h = h;
  % ------------------------------------ water level description
  level = V/A;
medium:
  M ISA WaterVarMM;
medium_connections:
  % ------------------------------------ medium states
  M.Min.p    := p;
  M.Min.hin  := Fin.h;
  M.Min.hout := h;
  %-------------------------- medium temp to heat transfer
  Qin.R.Tin  := M.Mout.Tkin;
  Qin.R.Tout := M.Mout.Tkout;
  % ------------------------------------ phase to medium terminal
  Fin.M.q    := 'Water;
  Fout.M.q   := 'Water;
  Fin.M.z    := height;
  Fout.M.z   := height;
heat_medium_connections:
  Qin.M.p    := p;
  Qin.M.w    := Fout.w;
  Qin.M.Gmix := [0;0;0;0;0;0;1.0];
END;
```

DrumCompartmentFM

```
DrumCompartmentFM ISA Compartment2IC WITH
  % not ready
  % no height description
  %% A control volume model of a mixture of
  %%   saturated water and steam described by
  %%   dynamic energy and mass balances.
  %%
  %% Assumptions: constant volume,
  %%              homogenous mixed,
  %%              no heat interaction,
  %%              no work interaction.
  %% Model Use:   given mass flow directions,
  %%              mass flows are described elsewhere.
  %% States:      pressure (p),
  %%              enthalpy (h).
  %% Medium:      nonideal gas, superheated steam (h > 7e6).
  %% Model type:  full.
parameters:
  V ISA Parameter;
  A ISA Parameter;
  DH ISA Parameter;
  height ISA Parameter;
submodels:
  Sout ISA super::Sout WITH
    M ISA SteamMediumTC;
  END;
variables:
  %***  state variabels
  p ISA Variable;
  h ISA Variable;
  %*** balance equation variables
  dm ISA Variable;
  de ISA Variable;
  %*** pressure/volume work
  dV ISA Variable;
  %*** aux variables
  K1 ISA Variable;
  K2 ISA Variable;
  T11 ISA Variable;
  T12 ISA Variable;
  T13 ISA Variable;
  T21 ISA Variable;
  T22 ISA Variable;
  T23 ISA Variable;
  %*** other variable
  level ISA Variable;
equations:
  %-------------------------------- mass balance
  dm = Fin.w + Fin2.w - Wout.w - Sout.w;
  %-------------------------------- energy balance
  de = Fin.w*Fin.h + Fin2.w*Fin2.h - Wout.w*Wout.h - Sout.w*Sout.h;
  dV := 0;
  %-------------------------------- auxiliary variables
  K1 = M.Mout.ap*M.Mout.rho + M.Mout.ah;
  K2 = M.Mout.rho*M.Mout.rho/p + M.Mout.ah;
  T11 = (M.Mout.rho + M.Mout.ah*h)/K1/V;
  T12 = -M.Mout.ah/K1/V;
  T13 = -p/K1/V*K2;
  T21 = (T11 - h/V)/M.Mout.rho;
  T22 = (1 - M.Mout.ah/K1)/M.Mout.rho/V;
```

```
  T23 = (1 - K2/K1)*p/M.Mout.rho/V;
  %-------------------------------- state equations (pressure and enthalpy)
  p' = T11*dm + T12*de + T13*dV;
  Fin.p = p;
  Fin2.p = p;
  Sout.p = p;
  Wout.p = p + M.Mout2.rhow*K2BasicLib::g*level;
  h' = T21*dm + T22*de + T23*dV;
  Sout.h = if M.Mout2.alpha < (1-K2BasicLib::eps) then M.Mout2.hs else h;
  Wout.h = if M.Mout2.alpha > K2BasicLib::eps then M.Mout2.hw else h;
  %-------------------------------- water level description
  level = DH*(1-M.Mout2.alpha);
  %-------------------------------- medium description
medium:
  M ISA SaturatedWaterSteamMM;
medium_connections:
  %------------------------- medium state to medium model
  M.min.p = p;
  M.min.hin = Fin.h;
  M.min.hout = h;
  % ----------------------------------- phase to medium terminal
  Fin.M.q := 'Water;
  Fin2.M.q := 'Water;
  Wout.M.q := 'Water;
  Sout.M.q := 'Steam;
  Sout.M.T := M.Mout.Tkout;
  Fin.M.z := height;
  Fin2.M.z := height;
  Wout.M.z := height;
END;
```

10. K2FlowLib

This library contains flow resistor classes. These models describe medium flow as a function of the pressure on the different sides of the subunit. The physical properties that are needed are calculated in a medium model using the medium states: pressure and enthalpy. There are also loss and friction factor classes which are used as submodels inside the flow resistors.

FlowResistorIC is an interface class with one inflow, one outflow (horizontal layout).

FlowVerticalIC* is similar to `FlowResistorIC` but with another graphical layout (vertical).

WaterFlowResistorFM describes a water medium flow. It has two submodels, the medium model and a loss factor model.

SteamFlowResistorFM is similar to the previous but with another medium model.

SteamFlowResistorFM2* is identical with the previous but with a variable valve loss factor.

GasFlowResistorFM is similar to the two previous ones but with other medium and loss factor models.

CriticalExpansionM is similar to `SteamFlowResistorFM` but with a description of critical expansion flow.

LaminarFlowResistorFM* is similar to `WaterFlowResistorFM` but with a flow description that is solved and simplified for laminar flow losses.

TurbulentFlowResistorFM* is similar to `WaterFlowResistorFM` but with a flow description that is solved and simplified for turbulent flow losses.

GenFlowResistorIC* is an interface class to a flow resistor where the flow direction may change.

GenFlowResistorFM* is a variable direction flow resistor.

ValveLossFactorFunction* is a constant loss factor used in flow resistor models.

ValveLossFactorFunction2* is a variable loss factor.

TubeLossFactor is a super class to the following one. Contains a subclass of `FrictionFactor`.

TubeLossFactorFunction contains a model of laminar flow in a tube.

FrictionFactor is a super class to the two following ones.

LaminarFrictionFactor is a description of the friction losses in laminar flows.

TurbulentFrictionFactor is a description of the friction losses in turbulent flows.

A class with a * is not listed in this report. It is not found to be of general interest. The flow resistor icons with terminals are seen in Figure 14 The inflow

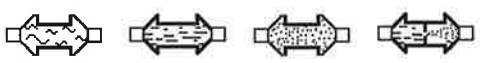

Figure 14. The flow resistor subunit models for different media.

terminal is at the left and the outflow at the right. Note that variations occur.

FlowResistorIC

```
FlowResistorIC ISA FlowResistorCC WITH
%% This is an interface class for
%%   flow resistor subunit classes.
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "WaterFlowResistor";
  END;
terminals:
  Fin ISA FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 0;
      y_pos := 150;
    END;
  END;
  Fout ISA FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400;
      y_pos := 150;
    END;
  END;
END;
```

WaterFlowResistorFM

```
WaterFlowResistorFM ISA FlowResistorIC WITH
%% A resistor model of a liquid based on
%%   static energy and mass balances.
%%
%% Assumptions: constant enthalpy
%%              no heat interaction,
%%              no work interaction.
%% Medium:      water (subcooled).
%% Model type:  full.
%%
parameters:
  length, diameter ISA Parameter;
  Area ISA Parameter WITH
    value := sqr(diameter)*K2BasicLib::pi/4;
  END;
  Zero ISA Parameter; % Always zero
variables:
  w ISA K2BasicLib::FlowVC;
  zloss ISA Variable;
  rho ISA Variable;
  my ISA Variable;
  DeltaP ISA K2BasicLib::PressureVC;
equations:
% ------------------------------------- mass balance
  Fin.w = w;
  Fout.w = w;
% ------------------------------------- enthalpy balance
  Fout.h = Fin.h;
% ------------------------------------- mechanical energy balance
  % 0 = (Fin.p/rho  + vin*ABS(vin)/2) -
  %     (Fout.p/rho + (1 + zloss)*vout*ABS(vout)/2);
  w = Area*sqrt(2*rho*DeltaP/zloss);
% ------------------------------------- auxiliary variables
  DeltaP = Fin.p-Fout.p;
  zloss = zv; % unnecessary
loss_factors:
  zv ISA TubeLossFactor WITH
    Fi ISA TurbulentFrictionFactor;
    diameter := outer::diameter;
    length := outer::length;
  END;
medium:
  M ISA WaterFlowMM;
medium_connections:
  M.min.p := Fin.p;
  M.min.h := Fin.h;
  rho := M.Mout.rho;
  my := M.Mout.my;
END;
```

SteamFlowResistorFM

```
SteamFlowResistorFM ISA WaterFlowResistorFM WITH
%% A resistor model of a liquid based on
%%   static energy and mass balances.
%%
%% Assumptions: constant enthalpy
%%              no heat interaction,
%%              no work interaction.
%%              constant valve loss
%% Medium:      steam (superheated).
%% Model type:  full.
%%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "SteamFlowResistor";
  END;
submodels:
  Fin ISA super::Fin WITH
    M ISA SteamMediumTC;
  END;
  Fout ISA super::Fout WITH
    M ISA SteamMediumTC;
  END;
loss_factors:
  zv ISA ValveLossFactorFunction;
medium:
  M ISA SteamFlowMM;
END;
```

GasFlowResistorFM

```
GasFlowResistorFM ISA WaterFlowResistorFM WITH
%% A resistor model of a liquid based on
%%   static energy and mass balances.
%%
%% Assumptions: constant enthalpy
%%              no heat interaction,
%%              no work interaction.
%%              constant valve loss
%% Medium:      flue gas
%% Model type:  full.
%%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "GasFlowResistor";
  END;
terminals:
  Fin ISA FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400;
      y_pos := 151;
    END;
    M ISA GasMediumTC;
  END;
  Fout ISA FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 1;
      y_pos := 151;
    END;
    M ISA GasMediumTC;
  END;
loss_factors:
  zv ISA ValveLossFactorFunction;
medium:
  M ISA GasFlowMM;
  M.Tk := Fin.M.T;
END;
```

CriticalExpansionM

```
CriticalExpansionM ISA FlowResistorIC WITH
%% A resistor model of flowing steam
%%   under critical expansion.
%%   Velocity becomes independent on
%%   the back pressure.
%%
%% Assumptions: constant enthalpy
%%              no heat interaction
%%              no work interaction
%%              constant valve loss
%% Medium:      superheated steam
%% Model type:  full.
%%
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "CriticalExpansion";
  END;
parameters:
  length, diameter ISA Parameter;
  Area ISA Parameter WITH
    value := sqr(diameter)*K2BasicLib::pi/4;
  END;
  Cd ISA Parameter; % Valve loss coefficcient
submodels:
  Fin ISA super::Fin WITH
    M ISA SteamMediumTC;
  END;
  Fout ISA super::Fout WITH
    M ISA SteamMediumTC;
  END;
variables:
  y ISA Variable; % Valve opening
  k ISA Variable; % Kappa (gamma)
  Fi ISA Variable;
  pratio ISA Variable;
  e, emin ISA Variable;
  krit, dir TYPE DISCRETE Integer;
event:
  Init ISAN Event;
equations:
% ---------------------------------------- Mass balance
  Fin.w = Fout.w;
% ---------------------------------------- Energy balance
  Fin.h = Fout.h;
% ---------------------------------------- Fluid character
  e = Fout.p/Fin.p;
  emin = (2/(k + 1))^(k/(k - 1));
  ONEVENT Init DO
%    new(krit) := 0;
    new(dir) := 1;
  END;
  ONEVENT (e>emin) OR (e<1/emin) DO
    new(krit) := 0;
  END;
  ONEVENT (e<emin) OR (e>1/emin) DO
    new(krit) := 1;
  END;
  ONEVENT e>1 DO
    new(dir) := -1;
  END;
```

```
  ONEVENT e<1 DO
    new(dir) := 1;
  END;
  pratio = ((1 - krit)*(1+dir)*e + krit*(1+dir)*emin +
            (1 - krit)*(1-dir)/e + krit*(1-dir)/emin)/2;
% ------------------------------------ Fluid dynamics
  Fi = sqrt(pratio^(2/k) - pratio^((1 + k)/k));
  Fout.w = dir*Cd*Area*y*sqrt(2*k/(k-1)*Fin.p*M.Mout.rho)*Fi;
% ------------------------------------ Media model
medium:
  M ISA SteamFlowMM;
medium_connections:
  M.min.p = Fin.p;
  M.min.h = Fin.h;
  k = M.Mout.kappa;
END;
```

Loss Factors and Friction Factors

```
TubeLossFactor ISA Variable WITH
%% This is a model component describing
%%   a flow loss factor.
%%   The friction factor is unspecified.
submodels:
  Fi ISA FrictionFactor;
parameters:
  length ISA Base::Parameter;
  diameter ISA Base::Parameter;
assignments:
  Fi.diameter := diameter;
equations:
  value = Fi*length/diameter;
END;


TubeLossFactorFunction ISA TubeLossFactor WITH
%% A Tube loss factor based on laminar flow.
  Fi ISA LaminarFrictionFactor;
END;


FrictionFactor ISA Variable WITH
%% An interface class for friction factor descriptions.
parameters:
  diameter ISA Base::Parameter;
END;


LaminarFrictionFactor ISA FrictionFactor WITH
%% This is a friction factor that can be
%%   used inside loss factor calculations.
%%   Laminar flow description.
variables:
  Re ISA Variable;
  w ISA Variable;
  my ISA Variable;
equations:
  Re = 4*w/diameter/K2BasicLib::pi/my;
  value = 64/Re;
END;


TurbulentFrictionFactor ISA FrictionFactor WITH
%% This is a friction factor that can be
%%   used inside loss factor calculations.
%%   Turbulent flow description.
parameters:
  roughness ISA Base::Parameter WITH default := 0.0003; END;
  % absolute roughness of cast iron
equations:
  value = 1/sqr(3.2 - 2.5*ln(roughness/diameter));
END;
```

11. K2HeatFlowLib

Heat transfer models are grouped together in this library. They are closely related to flow resistors and are therefore called heat flow resistors in the K2 model database. The heat flow resistors have a number of submodels describing different resistance factors, like wall and boundary layer resistances.

HeatTransferIC is an interface class.

HeatTransferFM is a simple model where heat flow is proportional to the temperature gradient.

HeatTransfer2FM is similar to the previous but with logarithmic mean temperature gradient for parallel flows

HeatTransfer3FM* is the same as the previous but for counter current flows.

HeatResistorIC is the interface class to the more complex heat resistor model.

HeatResistorFM is a heat transfer model using logarithmic mean temperature description, wall and boundary layer resistance calculation.

A class with a * is not listed in this report. It is not found to be of general interest.
There are additional descriptions of individual heat resistance in the basic library.

Heat Transfer Classes

```
HeatTransferIC ISA HeatResistorCC WITH
%% This is a interface class for heat
%%   transfer models.
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "HeatResistor";
  END;
terminals:
  Hin  ISA HeatRateTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201;
      y_pos := 1;
    END;
  END;
  Hout ISA HeatRateTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201;
      y_pos := 300;
    END;
    direction := 'out;
  END;
END;



HeatTransferFM ISA HeatTransferIC WITH
%% A simple heat transfer model where
%%   the heat flow is propotional to
%%   the temperature difference.
%%
parameters:
  k, Area ISA Parameter;
equations:
  Hout.q = Hin.q;
  Hin.q = k*Area*(Hin.Tout - Hout.Tout);
END;



HeatTransfer2FM ISA HeatTransferIC WITH
%% A simple heat transfer model for cocurrent
%%   flows where the heat flow is propotional to
%%   the mean logarithmic temperature difference.
parameters:
  k, Area ISA Parameter;
equations:
  Hout.q = Hin.q;
  Hin.q = k*Area*dTm;
submodel:
  dTm ISA LogMean WITH
    x = Hin.Tin  - Hout.Tin;
    y = Hin.Tout - Hout.Tout;
  END;
END;
```

HeatResistorIC and HeatResistorFM

```
HeatResistorIC ISA HeatResistorCC WITH
%% This is an interface class for
%%   heat resistor models.
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "HeatResistor";
  END;
terminals:
  Hin  ISA HeatTransferInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201;
      y_pos := 1;
    END;
  END;
  Hout ISA HeatTransferOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201;
      y_pos := 300;
    END;
  END;
END;
```

```
HeatResistorFM ISA HeatResistorIC WITH
%% A complex heat transfer model with
%%   separate submodels for convective
%%   and conductive heat resistance.
submodels:
  U1 ISA K2BasicLib::GasResistance WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 75.0;
    END;
  END;
  U2 ISA K2BasicLib::WallResistance WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 150.0;
    END;
  END;
  U3 ISA K2BasicLib::SteamResistance WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 225.0;
    END;
  END;
terminals:
  dTm ISA LogMean WITH
    x = Hin.R.Tin - Hout.R.Tout;
    y = Hin.R.Tout - Hout.R.Tin;
  END;
parameters:
  length ISA Parameter;
  diameter ISA Parameter;
  delta ISA Parameter;
parameter_propagation:
  U1.l := length;
  U2.l := length;
  U3.l := length;
```

```
  U1.d := diameter;
  U2.d := diameter;
  U3.d := diameter;
  U2.delta := delta;
variables:
  Tm TYPE Real;
  Tm1 TYPE Real;
  Tm2 TYPE Real;
equations:
  Hout.R.q = Hin.R.q;
  Hin.R.q = dTm/(U1 + U2 + U3);
  Tm1 = (Hin.R.Tin + Hin.R.Tout)/2 - K2BasicLib::T0;
  Tm2 = (Hout.R.Tin + Hout.R.Tout)/2 - K2BasicLib::T0;
  Tm = (Tm1 + Tm2)/2;
connections:
  U1.Tm = Tm1;
  U2.Tm = Tm;
  U3.Tm = Tm2;
  Hin.M AT U1.Rin;
  Hout.M AT U3.Rin;
END;
```

12. K2FlowUnitLib

The subunit models are used to describe typical units. Units are objects that have a physical interpretation, like pumps and valves.

WaterPumpFM is a pump model based on a relationship between pressure and pumping power.

WaterValveFM is a valve model with incompressible water flow and a constant loss factor.

CritValveFM is a valve model for steam with large pressure drops.

TurbulentTubeFM* is a tube model with variable flow direction and turbulent losses.

FlowSplitFM describes a split of one water flow in two flows with equal enthalpy.

FlowSplitSteamFM is the same as the previous but for steam flows.

FlowJunctionFM describes a mixing of two water flows.

SprayTempFM describes a mixing of a water flow into a greater steam flow.

GenPumpFM* can have flow that changes direction.

GenValveFM* can have flow that changes direction.

WaterPumpFM

```
WaterPumpFM ISA FlowVerticalIC WITH
%% A pump model for a liquid based on
%%   static energy and mass balances.
%%
%% Assumptions: constant enthalpy
%%              no heat interaction,
%%              no losses,
%%              work interaction through effective input power
%% Medium:      water (subcooled).
%% Model type:  full.
%%
icons:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "WaterPump";
  END;
variables:
  w ISA K2BasicLib::FlowVC;
  Pe ISA Variable; % Effective input power
  rho ISA Variable;
  DeltaP ISA K2BasicLib::PressureUpVC;
equations:
% ---------------------------------------- mass balance
  Fin.w = w;
  Fout.w = w;
% ---------------------------------------- enthalpy balance
  Fout.h = Fin.h;
% ---------------------------------------- mechanical energy balance
  % 0 = Fin.p/rho - Pe/w + zloss*sqr(w/rho/A);
  w = -Pe*rho/DeltaP;
% ---------------------------------------- auxiliary variables
  DeltaP = Fin.p-Fout.p;
medium:
  M ISA WaterFlowMM;
medium_connections:
  M.min.p := Fin.p;
  M.min.h := Fin.h;
  rho := M.Mout.rho;
END;
```

Valves

```
WaterValveFM ISA WaterFlowResistorFM WITH
%% A valve model for a liquid
%%
%% Assumptions:  constant valve loss
%%
%% Medium:       water (subcooled)
%% Model type:   full.
%%
icons:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "WaterValve";
  END;
variables:
  Area, y ISA Variable; % Valve opening, 0..1
  Area := A0*y;
parameters:
  A0 ISA Parameter WITH
    value := sqr(diameter)*K2BasicLib::pi/4;
  END;
submodels:
  zv ISA ValveLossFactorFunction;
END;


CritValveFM ISA K2FlowLib::CriticalExpansionM WITH
%% A valve model for steam, only a graphical specialization
%%   since the critical flow model also can function as a valve.
%%
%% Assumptions:
%%
%% Medium:       steam (superheated)
%% Model type:   full.
%%
icons:
  Graphic ISA Layout WITH
    bitmap TYPE String := "Valve";
  END;
END;
```

Flow Split

```
FlowSplitFM ISA K2ClassTreeLib::UnitGC WITH
  % A splitting of one flow in two separate flows.
  % Pressure and height is propagated forward.
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "FlowSplit";
  END;
terminals:
  Fin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 0;
      y_pos := 151;
      invisible := 1;
    END;
  END;
  Fout1 ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400;
      y_pos := 151;
      invisible := 1;
    END;
  END;
  Fout2 ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201;
      y_pos := 0;
      invisible := 1;
    END;
  END;
  Fin.w = Fout1.w + Fout2.w;
  Fout1.h = Fin.h;
  Fout2.h = Fin.h;
  Fin.p = Fout1.p;
  Fout2.p = Fin.p;
  Fin.M AT Fout1.M;
  Fin.M AT Fout2.M;
END;


FlowSplitSteamFM ISA FlowSplitFM WITH
  % A splitting of one flow in two separate flows.
  % Pressure and height is propagated forward.
  % Specialized model for steam
terminals:
  Fin ISA super::Fin WITH
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Fout1 ISA super::Fout1 WITH
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Fout2 ISA super::Fout2 WITH
    M ISA K2TerminalLib::SteamMediumTC;
  END;
END;
```

FlowJunctionFM

```
FlowJunctionFM ISA K2ClassTreeLib::UnitGC WITH
  % A junction of two flows that are mixed into one.
  % Pressure and height is propagated backwards.
icon:
  Graphic ISA super::Graphic WITH
    bitmap TYPE String := "FlowJunction";
  END;
terminals:
  Fin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 0;
      y_pos := 151;
      invisible := 1;
    END;
  END;
  Fin2 ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201;
      y_pos := 300;
      invisible := 1;
    END;
  END;
  Fout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400;
      y_pos := 151;
      invisible := 1;
    END;
  END;
  Fout.w = Fin.w + Fin2.w;
  Fout.h = (Fin.h*Fin.w + Fin2.h*Fin2.w)/Fout.w;
  Fin.p = Fout.p;
  Fin2.p = Fout.p;
  Fin.M AT Fout.M;
  Fin2.M AT Fout.M;
END;
```

SprayTempFM

```
SprayTempFM ISA K2ClassTreeLib::UnitGC WITH
  %% A model of a spray attemperator described as
  %%   perfect mixing of steam and spray. No dynamics.
  %%
  %% Model type : full
  %% Assumptions: none
  %% Model Use:   given mass flow direction
  %% States:      none
  %% Medium:      none
  %%
icons:
  Graphic ISA Base::Layout WITH bitmap TYPE String := "SprayTemp"; END;
parameters:
  height ISA Parameter;
terminals:
  Fin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 0.0;
      y_pos := 150.0;
    END;
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Fout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400.0;
      y_pos := 150.0;
    END;
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Win ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 300.0;
    END;
  END;
equations:
  Fout.w = Fin.w + Win.w;
  Fout.h = (Fin.h*Fin.w + Win.h*Win.w)/Fout.w;
  Fin.p = Fout.p;
  Win.p = Fout.p;
  Fin.M AT Fout.M;
  Win.M.q := 'Water;
  Win.M.z := height;
END;
```

13. K2TurbineLib

Contains models describing turbine unit models. Currently one one turbine unit model is available.

SteamTurbineFM is a steam turbine unit model.

SteamTurbineFM

```
SteamTurbineFM ISA K2ClassTreeLib::UnitGC WITH
icon:
  Graphic ISA Layout WITH bitmap TYPE String := "SteamTurbine"; END;
parameters:
  etat, etam ISA Parameter WITH          % Thermal and mechanical
    default := 1;                        % efficiency
  END;
  length ISA Parameter;
  diameter ISA Parameter;
parameter_propagation:
  Flow.diameter := diameter;
  Comp.V := length*sqr(diameter)*K2BasicLib::Pi/4;
terminals:
  sin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 0.0;
      y_pos := 151.0;
    END;
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Sout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400.0;
      y_pos := 151.0;
    END;
  END;
submodels:
  Comp ISA K2CompartmentLib::SteamCompartmentFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 151.0;
      y_pos := 150.0;
    END;
  END;
  Flow ISA K2FlowLib::CriticalExpansionM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 275.0;
      y_pos := 150.0;
    END;
  END;
variables:
  Po ISA Variable; % Output power
equations:
  Po := sin.w*etam*(Comp.h - Sout.h);
connections:
  Sout.h = Comp.h*(Sout.p/Comp.p)^(etat*(1 - 1/Flow.k));
  Sout.p = Flow.Fout.p;
  Sout.w = Flow.Fout.w;
  Flow.Fout.M.T := 0;
  Flow.Fout.M.q := Flow.Fin.M.q;
  sin AT Comp.Fin;
  Comp.Fout AT Flow.Fin;
END;
```

14. K2HeatUnitLib

This library contains units that transfer heat in one way or another. There are typical unit models like the heat exchanger but there are also more complicated units that consist of other units like the boiler.

HeatExchangerFM is a water/water heat exchanger.

EconomizerFM is a water/gas heat exchanger.

SuperHeaterFM is a steam/gas heat exchanger.

SuperHeaterSystemFM contains two super heaters with a steam cooler and a temperature controller.

BoilerFM contains a drum, evaporator with recycled water and a drum level controller.

HeatExchangerFM

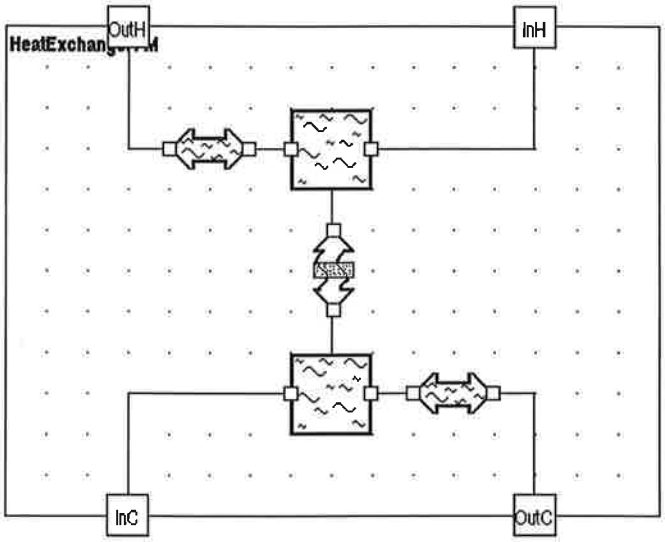

Figure 15. The (water/water) heat exchanger unit model.

```
HeatExchangerFM ISA K2ClassTreeLib::UnitGC WITH
%% A model of a heat exchanger with only one section
%%   water/water interaction.
%%
%% Assumptions: constant volume,
%%              homogenous mixed,
%%              no work interaction.
%% Model Use:   given mass flow directions.
%% States:      enthalpy (h) on primary side.
%%              enthalpy (h) on secondary side.
%% Medium:      Water and water.
%% Model type:  full.
%%
icons:
  Graphic ISA Layout WITH
    bitmap TYPE String := "HeatExchanger";
  END;
parameters:
  length ISA Parameter;
  diameter ISA Parameter;
  delta ISA Parameter;
  V ISA Parameter WITH value := length*sqr(diameter)/4; END;
parameter_propagation:
  HotF.diameter := diameter;
  ColdF.length := length;
  ColdF.diameter := diameter;
  HeatF.length := length;
  HeatF.diameter := diameter;
  HeatF.delta := delta;
  HotC.V := V;
  ColdC.V := V;
terminals:
  InH ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 325.0;
      y_pos := 300.0;
    END;
```

```
  END;
  InC ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 75.0;
      y_pos := 0.0;
    END;
  END;
  OutC ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 325.0;
      y_pos := 0.0;
    END;
  END;
  OutH ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 75.0;
      y_pos := 300.0;
    END;
  END;
submodels:
  HotC ISA K2CompartmentLib::WaterCompartmentFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 225.0;
    END;
  END;
  ColdC ISA K2CompartmentLib::WaterCompartmentFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 75.0;
    END;
  END;
  HeatF ISA K2HeatFlowLib::HeatResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201.0;
      y_pos := 151.0;
    END;
    U1 ISA K2BasicLib::WaterResistance WITH
      Graphic ISA super::Graphic WITH
        x_pos := 101.0;
        y_pos := 101.0;
      END;
    END;
    U3 ISA K2BasicLib::WaterResistance WITH
      Graphic ISA super::Graphic WITH
        x_pos := 301.0;
        y_pos := 101.0;
      END;
    END;
  END;
  ColdF ISA K2FlowLib::WaterFlowResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 275.0;
      y_pos := 75.0;
    END;
    zv ISA K2FlowLib::ValveLossFactorFunction;
  END;
  HotF ISA K2FlowLib::WaterFlowResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 125.0;
      y_pos := 225.0;
    END;
```

```
    zv ISA K2FlowLib::ValveLossFactorFunction;
  END;
connections:
  ColdC.Qin AT HeatF.Hout;
  C2 ISA Base::Connection WITH
    InH AT HotC.Fin;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [325.0, 300.0; 325.0, 224.0; 175.0, 224.0];
  END;
  C3 ISA Base::Connection WITH
    HotC.Fout AT HotF.Fin;
    bpoints TYPE STATIC Matrix[2, 2] :=
      [224.0, 224.0; 100.0, 224.0];
  END;
  C4 ISA Base::Connection WITH
    HotF.Fout AT OutH;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [149.0, 224.0; 75.0, 224.0; 75.0, 300.0];
  END;
  C5 ISA Base::Connection WITH
    InC AT ColdC.Fin;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [75.0, -1.0; 75.0, 74.0; 175.0, 74.0];
  END;
  C6 ISA Base::Connection WITH
    ColdC.Fout AT ColdF.Fin;
    bpoints TYPE STATIC Matrix[2, 2] :=
      [224.0, 74.0; 250.0, 74.0];
  END;
  C7 ISA Base::Connection WITH
    ColdF.Fout AT OutC;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [299.0, 74.0; 325.0, 74.0; 325.0, -1.0];
  END;
  C8 ISA Base::Connection WITH
    HotC.Qin AT HeatF.hin;
    bpoints TYPE STATIC Matrix[2, 2] :=
      [199.0, 224.0; 201.0, 126.0];
  END;
END;
```

EconomizerFM

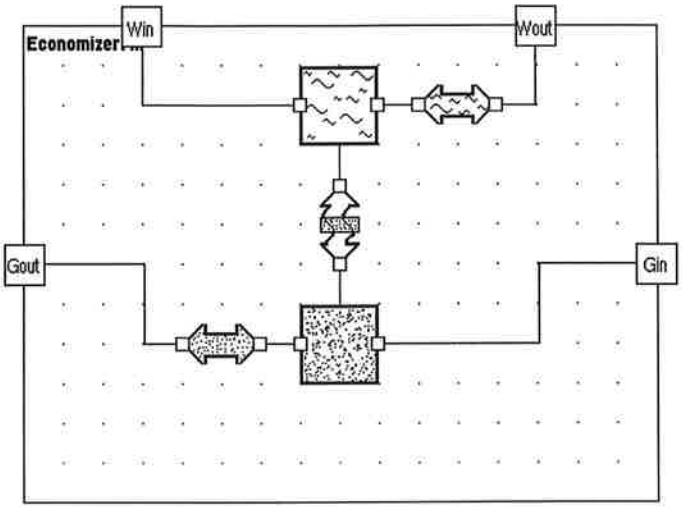

Figure 16. The economizer (water/gas heat exchanger) unit model.

```
EconomizerFM ISA K2ClassTreeLib::UnitGC WITH
%% A unit model of an economizer, basicly a heat
%%   exchanger with one gas and one water compartment.
%%
%% Assumptions: constant volume,
%%              homogenous mixed,
%%              no work interaction.
%% Model Use:   given mass flow directions.
%% States:      pressure (p) and enthalpy (h) on gas side,
%%              enthalpy on water side.
%% Medium:      Subcooled water and flue gas.
%% Model type:  full.
icon:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "Economizer";
  END;
parameters:
  length, diameter, delta ISA Parameter;
  height, zeta ISA Parameter;
  V ISA Parameter WITH value := length*sqr(diameter)/4; END;
parameter_propagation:
  GasFlow.length := length;
  GasFlow.diameter := 2*diameter;
  WaterFlow.length := length;
  WaterFlow.diameter := diameter;
  HeatRes.length := length;
  HeatRes.diameter := diameter;
  HeatRes.delta := delta;
  GasComp.V := 4*V;
  WaterComp.V := V;
  WaterComp.height := height;
submodels:
  GasComp ISA K2CompartmentLib::GasCompartmentFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 100.0;
    END;
  END;
```

```
  WaterComp ISA K2CompartmentLib::WaterCompartmentFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 250.0;
    END;
  END;
  GasFlow ISA K2FlowLib::GasFlowResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 125.0;
      y_pos := 100.0;
    END;
    zv.zeta := outer::zeta/100;
  END;
  WaterFlow ISA K2FlowLib::WaterFlowResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 275.0;
      y_pos := 250.0;
    END;
    zv ISA K2FlowLib::ValveLossFactorFunction;
    zv.zeta := outer::zeta;
  END;
  HeatRes ISA K2HeatFlowLib::HeatResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 175.0;
    END;
    U3 ISA K2BasicLib::WaterResistance;
  END;
terminals:
  Win ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 75.0;
      y_pos := 300.0;
    END;
  END;
  Wout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 325.0;
      y_pos := 300.0;
    END;
  END;
  Gin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400.0;
      y_pos := 150.0;
    END;
    M ISA GasMediumTC;
  END;
  Gout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 0.0;
      y_pos := 150.0;
    END;
    M ISA GasMediumTC;
  END;
connections:
  C6 ISA Base::Connection WITH
    GasFlow.Fin AT GasComp.Fout;
    bpoints TYPE STATIC Matrix[2, 2] := [149.0, 99.0; 175.0, 99.0];
  END;
  C1 ISA Base::Connection WITH
    WaterComp.Fout AT WaterFlow.Fin;
```

```
    bpoints TYPE STATIC Matrix[2, 2] := [199.0, 225.0; 250.0, 249.0];
  END;
  C2 ISA Base::Connection WITH
    WaterFlow.Fout AT Wout;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [299.0, 249.0; 325.0, 249.0; 325.0, 300.0];
  END;
  C3 ISA Base::Connection WITH
    WaterComp.Fin AT Win;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [199.0, 274.0; 75.0, 274.0; 75.0, 300.0];
  END;
  C4 ISA Base::Connection WITH
    Gin AT GasComp.Fin;
    bpoints TYPE STATIC Matrix[4, 2] :=
      [399.0, 150.0; 326.0, 150.0; 326.0, 99.0; 224.0, 99.0];
  END;
  C5 ISA Base::Connection WITH
    WaterComp.Qin AT HeatRes.hout;
    bpoints TYPE STATIC Matrix[2, 2] := [200.0, 250.0; 199.0, 199.0];
  END;
  C7 ISA Base::Connection WITH
    GasFlow.Fout AT Gout;
    bpoints TYPE STATIC Matrix[4, 2] :=
      [100.0, 99.0; 76.0, 99.0; 76.0, 150.0; 0.0, 150.0];
  END;
  C8 ISA Base::Connection WITH
    HeatRes.hin AT GasComp.Qin;
    bpoints TYPE STATIC Matrix[2, 2] := [199.0, 150.0; 199.0, 99.0];
  END;
END;
```

SuperHeaterFM

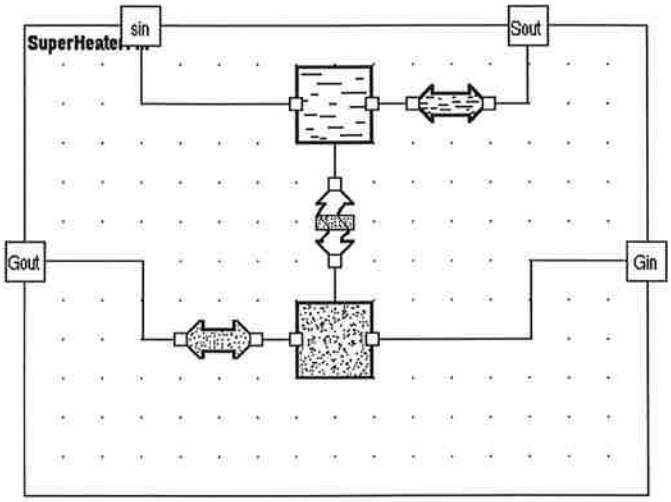

Figure 17. The super heater (steam/gas heat exchanger) unit model.

```
SuperHeaterFM ISA K2ClassTreeLib::UnitGC WITH
  %% A unit model of a superheater, basicly a heat
  %%   exchanger with one gas and one steam compartment.
  %%
  %% Assumptions: constant volume,
  %%              homogenous mixed,
  %%              no work interaction.
  %% Model Use:   given mass flow directions.
  %% States:      pressure (p),
  %%              enthalpy (h) on both sides.
  %% Medium:      superheated steam and flue gas.
  %% Model type:  full.
icon:
  Graphic ISA Base::Layout WITH bitmap TYPE String := "Economizer"; END;
parameters:
  length ISA Parameter;
  diameter ISA Parameter;
  delta ISA Parameter;
  zeta ISA Parameter;
  V ISA Parameter WITH value := length*sqr(diameter)/4; END;
parameter_propagation:
  GasFlow.length := length;
  GasFlow.diameter := 2*diameter;
  SteamFlow.length := length;
  SteamFlow.diameter := diameter;
  HeatRes.length := length;
  HeatRes.diameter := diameter;
  HeatRes.delta := delta;
  GasComp.V := 4*V;
  SteamComp.V := V;
terminals:
  sin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 75.0;
      y_pos := 300.0;
    END;
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Sout ISA K2TerminalLib::FlowOutTC WITH
```

```
    Graphic ISA super::Graphic WITH
      x_pos := 325.0;
      y_pos := 300.0;
    END;
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Gin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400.0;
      y_pos := 150.0;
    END;
    M ISA K2TerminalLib::GasMediumTC;
  END;
  Gout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 0.0;
      y_pos := 150.0;
    END;
    M ISA K2TerminalLib::GasMediumTC;
  END;
submodels:
  GasComp ISA K2CompartmentLib::GasCompartmentFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 100.0;
    END;
  END;
  SteamComp ISA K2CompartmentLib::SteamCompartmentFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 250.0;
    END;
  END;
  GasFlow ISA K2FlowLib::GasFlowResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 125.0;
      y_pos := 100.0;
    END;
    zv.zeta := outer::zeta/100;
  END;
  SteamFlow ISA K2FlowLib::SteamFlowResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 275.0;
      y_pos := 250.0;
    END;
    zv ISA K2FlowLib::ValveLossFactorFunction;
    zv.zeta := outer::zeta/50;
  END;
  HeatRes ISA K2HeatFlowLib::HeatResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 175.0;
    END;
  END;
connections:
  HeatRes.Hout AT SteamComp.Qin;
  C6 ISA Base::Connection WITH
    GasFlow.Fin AT GasComp.Fout;
    bpoints TYPE STATIC Matrix[2, 2] := [149.0, 99.0; 175.0, 99.0];
  END;
  C1 ISA Base::Connection WITH
    SteamComp.Fout AT SteamFlow.Fin;
```

```
    bpoints TYPE STATIC Matrix[2, 2] := [199.0, 225.0; 250.0, 249.0];
  END;
  C2 ISA Base::Connection WITH
    SteamFlow.Fout AT Sout;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [299.0, 249.0; 325.0, 249.0; 325.0, 300.0];
  END;
  C3 ISA Base::Connection WITH
    SteamComp.Fin AT sin;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [199.0, 274.0; 75.0, 274.0; 75.0, 300.0];
  END;
  C9 ISA Base::Connection WITH
    Gin AT GasComp.Fin;
    bpoints TYPE STATIC Matrix[4, 2] :=
      [399.0, 150.0; 326.0, 150.0; 326.0, 99.0; 224.0, 99.0];
  END;
  C10 ISA Base::Connection WITH
    GasFlow.Fout AT Gout;
    bpoints TYPE STATIC Matrix[4, 2] :=
      [100.0, 99.0; 76.0, 99.0; 76.0, 150.0; 0.0, 150.0];
  END;
  C11 ISA Base::Connection WITH
    HeatRes.hin AT GasComp.Qin;
    bpoints TYPE STATIC Matrix[2, 2] := [199.0, 150.0; 199.0, 99.0];
  END;
END;
```

SuperHeaterSystemFM

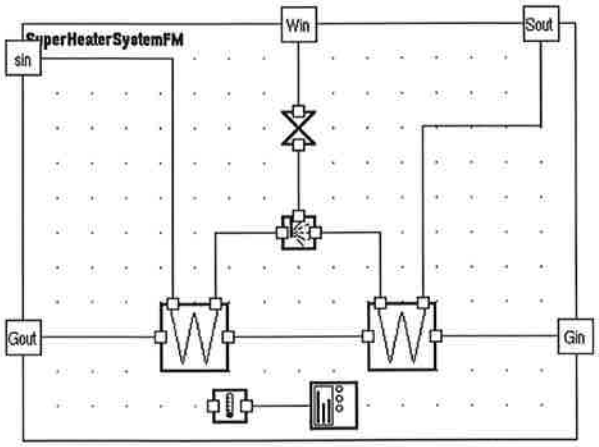

Figure 18. The super heater system with two super heaters and spray.

```
SuperHeaterSystemFM ISA K2ClassTreeLib::UnitGC WITH
  %% A unit model of a complex superheater with spray
  %%   temperature control.
  %%
  %% Assumptions: constant volume,
  %%              homogenous mixed,
  %%              no work interaction.
  %% Model Use:   given mass flow directions.
  %% States:      pressure (p),
  %%              enthalpy (h).
  %% Medium:      superheated steam and flue gas.
  %% Model type:  full.
icon:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "SuperHeater";
  END;
parameters:
  diameter, zeta, delta ISA Parameter;
  T, Ref, uMan, Manual ISA Parameter;
parameter_propagation:
  Heat1.diameter := diameter;
  Heat1.zeta := zeta;
  Heat1.delta := delta;
  Heat2.diameter := diameter;
  Heat2.zeta := zeta;
  Heat2.delta := delta;
  Valve.diameter := diameter;
terminals:
  sin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 1.0;
      y_pos := 276.0;
    END;
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Sout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 376.0;
      y_pos := 301.0;
    END;
    M ISA K2TerminalLib::SteamMediumTC;
```

```
  END;
  Win ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 201.0;
      y_pos := 301.0;
    END;
  END;
  Gin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 401.0;
      y_pos := 75.0;
    END;
    M ISA K2TerminalLib::GasMediumTC;
  END;
  Gout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 1.0;
      y_pos := 75.0;
    END;
    M ISA K2TerminalLib::GasMediumTC;
  END;
submodels:
  Heat1 ISA K2HeatUnitLib::SuperHeaterFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 125.0;
      y_pos := 75.0;
    END;
  END;
  Heat2 ISA K2HeatUnitLib::SuperHeaterFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 275.0;
      y_pos := 75.0;
    END;
  END;
  Spray ISA K2FlowUnitLib::SprayTempFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 150.0;
    END;
  END;
  Temp ISA K2SensorLib::TempSensorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 150.0;
      y_pos := 25.0;
    END;
  END;
  Valve ISA K2FlowUnitLib::WaterValveFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 225.0;
      bitmap TYPE String := "ValveV";
    END;
    Fin ISA super::Fin WITH
      Graphic ISA super::Graphic WITH
        x_pos := 201;
        y_pos := 300;
      END;
    END;
    Fout ISA super::Fout WITH
      Graphic ISA super::Graphic WITH
        x_pos := 201;
        y_pos := 1;
```

```
      END;
    END;
  END;
  PID ISA ControlSystemLib::PIDControllerFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 225.0;
      y_pos := 25.0;
    END;
  END;
equations:
  T*PID.SetPoint' + PID.SetPoint = Ref;
connections:
  PID.AutoMan.Uman := Uman;
  PID.AutoMan.Manual := Manual;
  Temp.in = Heat2.SteamFlow.M.Tc;
  Valve.y = PID.Control;
  C1 ISA Base::Connection WITH
    Heat2.Gout AT Heat1.Gin;
    bpoints TYPE STATIC Matrix[2, 2] := [250.0, 74.0; 149.0, 74.0];
  END;
  C5 ISA Base::Connection WITH
    Temp.out AT PID.Measure;
    bpoints TYPE STATIC Matrix[2, 2] := [161.0, 24.0; 207.0, 24.0];
  END;
  C6 ISA Base::Connection WITH
    Spray.Win AT Valve.Fout;
    bpoints TYPE STATIC Matrix[2, 2] := [211.0, 149.0; 211.0, 224.0];
  END;
  C2 ISA Base::Connection WITH
    sin AT Heat1.sin;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [0.0, 275.0; 109.0, 275.0; 109.0, 99.0];
  END;
  C3 ISA Base::Connection WITH
    Heat2.Sout AT Sout;
    bpoints TYPE STATIC Matrix[4, 2] :=
      [289.0, 99.0; 289.0, 226.0; 375.0, 226.0; 375.0, 300.0];
  END;
  C4 ISA Base::Connection WITH
    Gin AT Heat2.Gin;
    bpoints TYPE STATIC Matrix[2, 2] := [400.0, 75.0; 299.0, 74.0];
  END;
  C7 ISA Base::Connection WITH
    Heat1.Gout AT Gout;
    bpoints TYPE STATIC Matrix[2, 2] := [100.0, 74.0; 0.0, 75.0];
  END;
  C8 ISA Base::Connection WITH
    Heat1.Sout AT Spray.Fin;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [139.0, 99.0; 139.0, 161.0; 199.0, 161.0];
  END;
  C9 ISA Base::Connection WITH
    Heat2.sin AT Spray.Fout;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [259.0, 99.0; 259.0, 137.0; 199.0, 137.0];
  END;
  C10 ISA Base::Connection WITH
    Win AT Valve.Fin;
    bpoints TYPE STATIC Matrix[2, 2] := [200.0, 300.0; 187.0, 224.0];
  END;
END;
```

BoilerFM

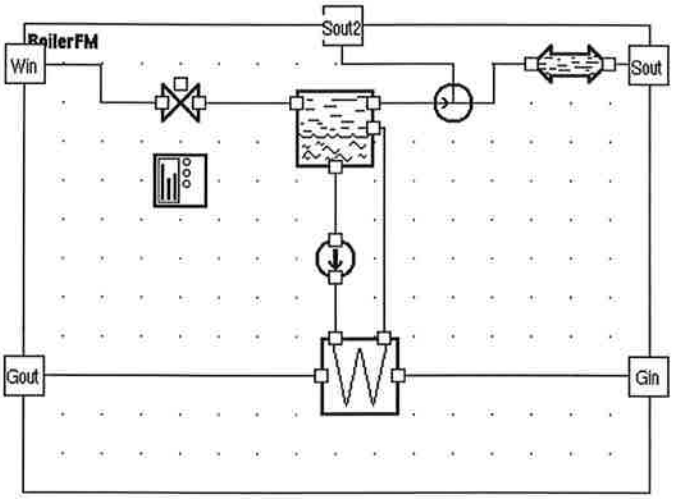

Figure 19. The boiler system with drum and evaporator.

```
BoilerFM ISA K2ClassTreeLib::UnitGC WITH
%% A unit model of a boiler consisting of a
%%   drum containing water and steam and a heat
%%   exchanger.
%%
%% Assumptions: constant volume,
%%              homogenous mixed,
%%              no work interaction.
%% Model Use:   given mass flow directions.
%% States:      pressure (p),
%%              enthalpy (h).
%% Medium:      saturated water and steam.
%%              flue gas.
%% Model type:  full.
icon:
  Graphic ISA Base::Layout WITH
    bitmap TYPE String := "Boiler";
  END;
parameters:
  length, diameter, delta ISA Parameter;
  height, zeta ISA Parameter;
  TubeV ISA Parameter WITH value := length*sqr(diameter)/4; END;
  V, DH ISA Parameter;
  T, Ref, Manual, uMan ISA Parameter;
parameter_propagation:
  Hex.length := length;
  Hex.diameter := diameter;
  Hex.delta := delta;
  Hex.zeta := zeta;
  SteamFlow.length := length;
  SteamFlow.diameter := diameter;
  Valve.diameter := diameter;
  DrumComp.V := V;
  DrumComp.DH := DH;
  DrumComp.height := height;
  Control.AutoMan.uMan := uMan;
  Control.AutoMan.Manual := Manual;
terminals:
  Win ISA K2TerminalLib::FlowInTC WITH
```

```
    Graphic ISA super::Graphic WITH
      x_pos := 0.0;
      y_pos := 275.0;
    END;
  END;
  Gin ISA K2TerminalLib::FlowInTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400.0;
      y_pos := 75.0;
    END;
    M ISA K2TerminalLib::GasMediumTC;
  END;
  Sout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400.0;
      y_pos := 275.0;
    END;
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Sout2 ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 205.0;
      y_pos := 300.0;
    END;
    M ISA K2TerminalLib::SteamMediumTC;
  END;
  Gout ISA K2TerminalLib::FlowOutTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 0.0;
      y_pos := 75.0;
    END;
    M ISA K2TerminalLib::GasMediumTC;
  END;
submodules:
  DrumComp ISA K2CompartmentLib::DrumCompartmentFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 234.0;
    END;
    level ISA SimpleOutput WITH
      Graphic ISA super::Graphic WITH
        x_pos := 201;
        y_pos := 151;
        invisible := 1;
      END;
    END;
  END;
  Valve ISA K2FlowUnitLib::WaterValveFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 100.0;
      y_pos := 250.0;
    END;
    zv.zeta := outer::zeta;
    y ISA SimpleInput WITH
      Graphic ISA super::Graphic WITH
        x_pos := 201;
        y_pos := 300;
      END;
    END;
  END;
  Split ISA K2FlowUnitLib::FlowSplitSteamFM WITH
    Graphic ISA super::Graphic WITH
```

```
      x_pos := 275.0;
      y_pos := 250.0;
      bitmap TYPE String := "FlowSplitUp";
    END;
    Fout2 ISA super::Fout2 WITH
      Graphic ISA super::Graphic WITH
        x_pos := 201;
        y_pos := 300;
      END;
    END;
  END;
  SteamFlow ISA K2FlowLib::SteamFlowResistorFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 350.0;
      y_pos := 275.0;
    END;
    zv ISA K2FlowLib::ValveLossFactorFunction;
    zv.zeta := outer::zeta/50;
  END;
  Pump ISA K2FlowUnitLib::WaterPumpFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 200.0;
      y_pos := 150.0;
    END;
  END;
  Control ISA ControlSystemLib::PIDControllerFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 100.0;
      y_pos := 200.0;
    END;
  END;
  Hex ISA K2HeatUnitLib::EconomizerFM WITH
    Graphic ISA super::Graphic WITH
      x_pos := 216.0;
      y_pos := 75.0;
    END;
  END;
equations:
  T*Control.SetPoint' + Control.SetPoint = Ref;
connections:
  Control.Measure = DrumComp.level;
  Valve.y = Control.Control;
  DrumComp.Wout AT Pump.Fin;
  Pump.Fout AT Hex.Win;
  Valve.Fout AT DrumComp.Fin;
  DrumComp.Sout AT Split.Fin;
  C24 ISA Base::Connection WITH
    SteamFlow.Fout AT Sout;
    bpoints TYPE STATIC Matrix[2, 2] := [374.0, 274.0; 400.0, 275.0];
  END;
  C4 ISA Base::Connection WITH
    Gin AT Hex.Gin;
    bpoints TYPE STATIC Matrix[2, 2] := [399.0, 75.0; 249.0, 74.0];
  END;
  C5 ISA Base::Connection WITH
    Gout AT Hex.Gout;
    bpoints TYPE STATIC Matrix[2, 2] := [0.0, 75.0; 200.0, 74.0];
  END;
  C1 ISA Base::Connection WITH
    Win AT Valve.Fin;
    bpoints TYPE STATIC Matrix[4, 2] :=
      [0.0, 274.0; 49.0, 274.0; 49.0, 249.0; 87.0, 249.0];
```

```
  END;
  C3 ISA Base::Connection WITH
    Split.Fout1 AT SteamFlow.Fin;
    bpoints TYPE STATIC Matrix[4, 2] :=
      [286.0, 249.0; 301.0, 249.0; 301.0, 275.0; 325.0, 275.0];
  END;
  C8 ISA Base::Connection WITH
    Split.Fout2 AT Sout2;
    bpoints TYPE STATIC Matrix[4, 2] :=
      [274.0, 261.0; 274.0, 275.0; 204.0, 275.0; 204.0, 299.0];
  END;
  C10 ISA Base::Connection WITH
    Hex.Wout AT DrumComp.Fin2;
    bpoints TYPE STATIC Matrix[3, 2] :=
      [230.0, 99.0; 230.0, 233.0; 224.0, 233.0];
  END;
END;
```

15. K2SensorLib

Sensor library for measurement classes. Currently only a temperature sensor model.

TempSensorFM contains a first order model of the temperature.

TempSensorFM

```
TempSensorFM ISA K2ClassTreeLib::SensorCC WITH
%% A model of a temperature sensor with first
%%   order dynamics.
%%
%% Model type : full
%% Assumptions: none
%% Model Use  : connecting to controllers
%% States     : none
%% Medium     : none
%%
icons:
  Graphic ISA Base::Layout WITH bitmap TYPE String := "TempSensor"; END;
parameters:
  T ISA Parameter WITH default := 1;END;
terminals:
  In ISA K2TerminalLib::TemperatureTC WITH
    Graphic ISA super::Graphic WITH
      x_pos := 0.0;
      y_pos := 151.0;
    END;
  END;
  Out ISA Base::SimpleTerminal WITH
    Graphic ISA super::Graphic WITH
      x_pos := 400.0;
      y_pos := 151.0;
    END;
  END;
connections:
  T*Out'+Out = In;
END;
```