Stream Connectors – An Extension of Modelica for Device-Oriented Modeling of Convective Transport Phenomena

Rüdiger Franke, ABB AG, Germany, Ruediger.Franke@de.abb.com
Francesco Casella, Politecnico di Milano, Italy, Casella@elet.polimi.it
Martin Otter, DLR Institute for Robotics and Mechatronics, Germany, Martin.Otter@dlr.de
Michael Sielemann, DLR Institute for Robotics and Mechatronics, Michael.Sielemann@dlr.de
Hilding Elmqvist, Dassault Systèmes (Dynasim), Sweden, Hilding.Elmqvist@3ds.com
Sven Erik Mattson, Dassault Systèmes (Dynasim), Sweden, SvenErik.Mattsson@3ds.com
Hans Olsson, Dassault Systèmes (Dynasim), Sweden, Hans.Olsson@3ds.com

Abstract

Modelica 3.0 as well as other physical modeling languages have two basic variable types to describe the interaction of physical components: "Potential" (or across) variable and "flow" (or through) variable. It is shown that with these variable types it is not possible to describe in a numerically sound way bi-directional flow of matter. Other alternatives based on signal flow oriented modeling have severe restrictions how components can be connected together.

This fundamental problem is addressed in Modelica 3.1 by introducing a third type of connector variable for physical systems, called stream variable, declared with the prefix stream.

This article motivates and introduces stream variables. Examples are given for their utilization in basic fluid models.

Keywords: thermo-fluid, stream variable, convection, potential/flow variable, across/through variable

1 Introduction

Connectors and connector designs are crucial for the modular modeling of complex physical systems. The understanding of simulation models is generally simplified if the modular model structure corresponds to the structure of actual physical devices. Connectors of such device-oriented models shall represent actual ports, like flanges for fluid flow.

Modelica provides different kinds of connection variables for the definition of connectors. Input and output signals are used for the connection of control blocks. Connections between physical devices are generally treated with pairs of "potential" (or across) and "flow" (or through) variables. Electrical systems are a typical example, where voltage differences are treated as potential variables and currents are treated as flow variables.

Thermo-fluid systems deal with convective transport phenomena. The two basic variable kinds in a physical connector – potential variable and flow variable – are not sufficient to describe in a numerically sound way the bi-directional flow of matter with convective transport of specific quantities, such as specific enthalpy and chemical composition. The values of these specific quantities are determined from the upstream side of the flow, i.e., they depend on the flow direction. When using potential and flow variables, the corresponding models would include nonlinear systems of equations with Boolean unknowns for the flow directions and singularities around zero flow. Such equation systems cannot be solved reliably in general. The model formulations can be simplified when formulating two different balance equations for the two possible flow directions. This is not possible with potential and flow variables though.

The requirements for a good, general-purpose object-oriented modeling framework in this domain are:

- single definition of one fluid connector class;
- intuitive semantics for hierarchical, device-oriented modeling: outer connectors of a functional unit model (e.g., a steam generator) should correspond to physical flanges of the unit itself;
- numerical robustness at zero and reverting flow;
- arbitrary connections between multiple fluid connectors are possible; automatically generated connection equations represent idealized flow junctions.

DOI: 10.3384/ecp09430078

No Modelica fluid library existing so far fulfills all of these requirements. The experience made with different approaches for the modeling of fluid systems during the last years shows that the available kinds of connector variables are not well suited for the description of interfaces and connections for convective transport phenomena, such as thermo-fluid flow.

Without restriction of generality, this article discusses the convective transport of energy. Therefore, the following analysis is formulated in terms of specific enthalpy. Nonetheless, the principle holds for other quantities transported via convection such as substance concentrations as well.

2 Existing Definitions of Fluid connectors

This section discusses two different approaches that are widely used to describe the interaction of thermo-fluid components. Other known approaches are either similar to the discussed ones or are not valid with respect to the connection semantics of Modelica 3.

2.1 Connection Semantics of Balanced Models

In Modelica 3.0 the concept of balanced models was introduced [7]. This approach requires that every model is "locally balanced", which means that the number of unknowns and equations must match on every hierarchical level. It is then sufficient to check every model locally only once, e.g., all models in a Modelica package. Using these models (instantiating and connecting them, redeclaring replaceable models etc.) will then lead to a model where the total number of unknowns and the total number of equations will match - a very important and useful property.

In [7] it is shown that this property can only hold if the number of flow variables in a connector is identical to the number of non-causal, non-flow variables (i.e., variables that do not have a **flow**, **input**, **output**, **parameter**, **constant** prefix). Therefore, this restriction on connectors was introduced in Modelica 3.0 and is utilized below.

2.2 Treatment of Transported Quantities as Signals

Focusing on the requirement for numerical robustness and simplicity for a Modelica tool, transported quantities can be communicated as signals through connectors. This approach has been implemented successfully in the ThermoPower library, see [2]. The basic idea is to use a pair of `input`/`output` variables for each transported intensive quantity (e.g., the specific enthalpy or the composition). Each of them corresponds to a specific direction of flow (into or out of the connector):

```
connector FluidPort_SignalA
  import SI = Modelica.SIunits;
         SI.Pressure         p;
  flow   SI.MassFlowRate     m_flow;
  input  SI.SpecificEnthalpy hBA;
  output SI.SpecificEnthalpy hAB;
end FluidPort_SignalA;
```

It is not possible to connect two connectors of this type together, since then two input variables (`hBA`) would be connected together and this is not allowed due to block diagram semantics. It is also not possible to just remove the input/output prefixes, because the resulting connector would violate the Modelica 3 restrictions, see section 2.1, since the number of non-causal, non-flow variables (`p`, `hBA`, `hAB`) would not be identical to the number of flow variables (`m_flow`).

Therefore, a second connector is needed with the same variables but with exchanged input/output prefixes:

```
connector FluidPort_SignalB
  import SI = Modelica.SIunits;
         SI.Pressure         p;
  flow   SI.MassFlowRate     m_flow;
  output SI.SpecificEnthalpy hBA;
  input  SI.SpecificEnthalpy hAB;
end FluidPort_SignalB;
```

The input variables in both connectors refer to the value the intensive quantity would have assuming the flow is entering the component; the output variables refer to the value that the flow would have if going out of the component. These quantities are always well-defined, and do not have any discontinuity around zero mass flow rate.

The use of signals in physical connectors, however, complicates the device-oriented modeling of fluid systems. Modelica signals are not intended for physical connectors and their use restricts the connection semantics. Each connection set on the same hierarchical level must contain exactly one `FluidPort_SignalA` and one `FluidPort_SignalB` connector. Adaptor models need to be introduced in all other cases.

2.3 Treatment of Transported Quantities as Potential Variables

Focusing on the requirements for intuitive, device-oriented connection semantics, transported quantities can be modeled as potential variables. Previous versions of Modelica_Fluid treat specific enthalpy as potential variable, complemented with enthalpy flow

rate as flow variable, see [4]. The connector definition reads:

```
connector FluidPort_Potential
  import SI = Modelica.SIunits;
       SI.Pressure         p;
  flow SI.MassFlowRate     m_flow;
       SI.SpecificEnthalpy h_mix;
  flow SI.EnthalpyFlowRate H_flow;
end FluidPort_Potential;
```

The pressure `p` and the mass flow rate `m_flow` describe the hydraulic phenomena of fluid flow. The specific enthalpy `h_mix` and the enthalpy flow rate `H_flow` describe the convective transport of energy. Since no input/output prefixes are used in this connector, there are no restrictions how components can be connected together, contrary to the signal flow approach of section 2.2.

A connection set is seen as an infinitesimally small mixing volume and the specific enthalpy in the connector is the mixing enthalpy for arbitrary flow directions. Figure 1 shows an example with three connected components.

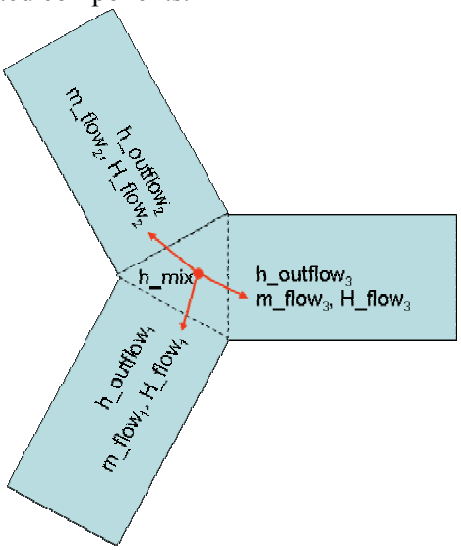

Figure 1: Exemplary connection set with three connected components and a common mixing enthalpy

Consider a connection set with n connectors. The mixing enthalpy is defined by the mass balance

$$0 = \sum_{j=1\ldots n} \dot{m}_j \tag{1}$$

and the energy balance at the infinitesimal small control volume of the connection point

$$0 = \sum_{j=1\ldots n} \dot{H}_j \tag{2}$$

with

$$\dot{H}_j = \dot{m}_j \begin{cases} h_{mix} & \textbf{if } \dot{m}_j > 0 \\ h_{outflow\,j} & \textbf{if } \dot{m}_j \le 0 \end{cases} \tag{3}$$

Herein, mass flow rates $\dot{m}_i$ are positive when entering models (i.e. exiting the connection set). The specific enthalpy $h_{outflow}$ represents the specific enthalpy inside the component, close to the connector, for the case of outflow.

When connecting components together, the Modelica connection semantics for flow variables will result in equations (1) and (2), since `m_flow` and `H_flow` are flow variables. (3) needs to be implemented in a component for every connector of this component.

If the mass flow rates are unknowns, as it is usually the case, then (1)-(3) is a set of non-linear algebraic equations in the mass flow rates $\dot{m}_i$ and the mixing enthalpy h_{mix}. From (3) it can be seen, that the unknown mass flow rates enter this equation set also as Boolean expressions $\dot{m}_j > 0$.

In the most often occurring case of two connected components, it is easy to compute h_{mix} from (1)-(3):

$$h_{mix} = \begin{cases} h_{outflow,2} & \textbf{if } \dot{m}_1 > 0 \\ h_{outflow,1} & \textbf{if } \dot{m}_1 \le 0 \end{cases}$$

Since usually $h_{outflow,1} \neq h_{outflow,2}$, the mixing enthalpy h_{mix} is discontinuous at zero mass flow rate.

If all mass flow rates are zero, $\dot{m}_j = 0$, then (1)-(3) are identically fulfilled for every value of h_{mix}, so these equations do not have a unique solution at this point, which means they are singular. Note, that this singularity is in line with the physics of the problem: The idealized infinitesimally small mixing volume does not have storage; therefore, its thermodynamic state is exclusively defined by the fluid flowing through it. If *no* fluid flows through the connection set, its state is not defined and a singularity arises at zero mass flow.

The above observations can be summarized as follows: Non-linear equation systems might occur, e.g., due to steady-state initialization, steady state models, ideal mixing, or pressure drop components directly connected together. If this is the case and if h_{mix} is an iteration variable of this non-linear equation system, then (1) a singularity is present when all mass flow rates are zero, (2) h_{mix} is usually discontinuous at this singular point, and (3) the directions of the mass flow rates, $\dot{m}_j > 0$, are unknowns, i.e., the equation system consists not only of real but also of Boolean unknowns. No numerical solver is known that is able to compute the solution of such a non-linear algebraic equation system in a reliable way and it is very unlikely that this will ever be the case.

This analysis shows clearly that the mixing enthalpy h_{mix} should not be computed and it should never be used in a connector.

3 Stream Connectors

The goal of stream connectors is to fulfill all requirements on a thermo-fluid connector regarding intuitive device-oriented connection semantics on one hand and numerical robustness on the other.

From the analyses in the previous section, it becomes obvious that, in order to meet the goal of numerical robustness, the plain specific mixing enthalpy should not be computed. Note that the energy balance, together with the piecewise equation given in equation (3), consists of different branches. It can therefore be split. Separate energy balances are the result; each valid for a specific flow direction.

This design decision leads to two primary consequences. First, the corresponding connector variables (called "stream" variables) come in two incarnations, one under the assumption of fluid entering the component, the other under the assumption of fluid leaving the component. The specific enthalpy under the assumption of fluid leaving the component, $h_{outflow}$, is one of the two and has to be established via governing equations of the model. If a model has a "volume/capacity", these equations refer to the corresponding state variables of this volume. If the model does not have a volume/capacity, the specific enthalpy under the assumption of fluid leaving the component at one connector is an algebraic (instantaneous) function of the values of the specific enthalpies under the assumption of fluid entering the component at the other connectors of this component. This is the second consequence: Whenever the mixing enthalpy is *used* in a model it is the mixing enthalpy under the assumption of fluid flowing into said model. We establish this quantity using a dedicated built-in operator

$$inStream\left(h_{outflow,j}\right) = h_{mix}\left(\dot{m}_j \geq 0\right). \qquad (4)$$

This is the definition of the second incarnation of the specific mixing enthalpy on the level of a single flange. This means, the mixing enthalpy is only computed for the case when fluid is entering a component, but not when fluid is leaving. From the perspective of the connection set, this definition leads to n different incarnations of the specific mixing enthalpy (with n as number of connections in a connection set).

After motivating the concept, we will now describe it in detail. First, using stream variables, a fluid connector is defined as follows.

```
connector FluidPort_Stream
  import SI = Modelica.SIunits;
         SI.Pressure         p;
  flow   SI.MassFlowRate     m_flow;
  stream SI.SpecificEnthalpy h_outflow;
end FluidPort_Stream;
```

In this section, only the most important type of connections is treated, where components are connected on the same "level", so called "inside" connections. The special case of "outside" connections, i.e., connections along a hierarchical model, is sketched in appendix A2.

A connector containing stream variables is called a stream connector. A stream connector must have exactly one scalar variable with the **`flow`** prefix. The idea is that all stream variables of this connector are associated with this flow variable. A stream variable

- defines a quantity that is transported by a flow through the stream connector (i.e. large-scale motion via, e.g., `m_flow`);
- represents the value of the transported quantity for the case of outflow through the stream connector, i.e. `m_flow < 0`;
- does not lead to the generation of any connection equations (for "inside" connections).

A model using a stream connector must expose the outflow value, which is known from internal storage or from inflow values of other connectors, independently of the flow direction. The inflow value can be obtained by a model on demand by using the new operator

inStream(h_outflow)

In the Modelica Language Specification 3.1 [6], the definition for this operator is given implicitly. It is based on the balance equation for transported quantities, see (2) and (3):

$$0 = \sum_{j=1\ldots n} \dot{m}_j \cdot \begin{cases} h_{mix} & \textbf{if } \dot{m}_j > 0 \\ h_{outflow,j} & \textbf{else} \end{cases} \qquad (5)$$

However, the common enthalpy h_{mix} for ideal mixing shall not be computed. Instead, for <u>every connector *i*</u> the balance equation (5) is written under the assumption that flow is only entering a component via connector i (so these are "n" equations):

$$0 = \sum_{j=1\ldots n} \dot{m}_j \cdot \begin{cases} h_{mix\_in,i} & \textbf{if } \dot{m}_j > 0 \textbf{ or } j = i \\ h_{outflow,j} & \textbf{else} \end{cases} \qquad (6)$$

$$\textbf{inStream}(h_{outflow,i}) = h_{mix\_in,i} \quad i = 1,2,\ldots,n$$

As a result, the "n" balance equations (6) compute "n" mixing enthalpies $h_{mix\_in,i}$ under the assumption that flow enters the respective connector i. Similarly to the enthalpy for ideal mixing, a singularity is pre-

sent if all mass flow rates vanish: If $\dot{m}_j = 0$ then (6) is identically fulfilled for every value of $h_{mix\_in,i}$. It is therefore necessary to approximate the solution of (6) in an open neighborhood of that point. We neglect this issue for the moment and will come back to it in the subsequent section.

As shown in appendix A1, the enthalpy for ideal mixing for "n" connected components can be explicitly computed from (5) by:

$$h_{mix} = \frac{\sum_{j=1...n} h_{outflow\,j} \max(-\dot{m}_j, 0)}{\sum_{j=1...n} \max(-\dot{m}_j, 0)} \tag{7}$$

In a similar way, the mixing enthalpy under the assumption of a flow into connector i can be explicitly computed from (6) by:

$$\begin{aligned} \mathbf{inStream}(h_{outflow,i}) &= h_{mix}\left(\dot{m}_i \geq 0\right) \\ &= \frac{\sum_{j=1...n,\ j\neq i} h_{outflow,j} \max(-\dot{m}_j, 0)}{\sum_{j=1...n,\ j\neq i} \max(-\dot{m}_j, 0)} \end{aligned} \tag{8}$$

All variables needed for the expansion of the inStream operator are explicitly given in a connection set. The own outflow value $h\_outflow_i$ is not considered as a connector cannot have inflow and outflow at the same time.

Figure 2 visualizes stream variables and the dependencies of the inStream operator for three connected components.

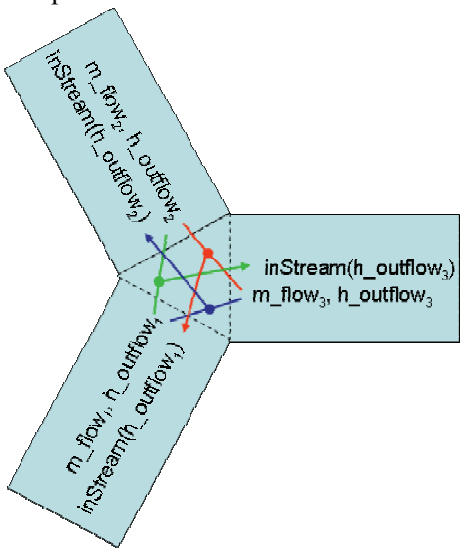

Figure 2: Exemplary connection set with three connected components

The expansion of the inStream operator simplifies for some important special cases.

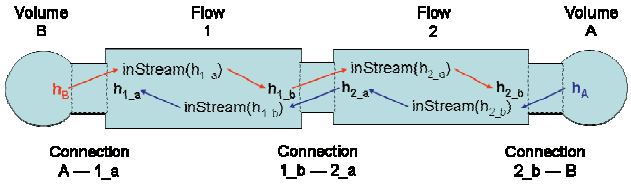

Figure 3: Exemplary series connection of multiple models with stream connectors

Figure 3 shows the series connection of two volumes via two "flow" models that neither store mass nor energy. In this case the inStream operator reduces to a simple propagation of two independent values. This can be easily derived from (8) for the special case of two connected components:

$$\begin{aligned} \mathbf{inStream}(h_{outflow,1}) &= h_{outflow,2} \\ \mathbf{inStream}(h_{outflow,2}) &= h_{outflow,1} \end{aligned} \tag{9}$$

These equations state that the value of the inflowing specific enthalpy is just the value of the specific enthalpy from the upstream side. This result could have been obtained also directly by inspection of a figure of this situation. Therefore, the following relations hold for Figure 3:

$$\begin{aligned} \mathbf{inStream}(h_B) = h_{1\_a} &= \mathbf{inStream}(h_{1\_b}) \\ = h_{2\_a} &= \mathbf{inStream}(h_{2\_b}) = h_A \end{aligned} \tag{10}$$

In volume B the energy balance, with the internal energy U_B of volume B, might have the following form which can be considerably simplified using (10):

$$\begin{aligned} \frac{dU_B}{dt} &= \dot{m}_B \cdot \begin{cases} \mathbf{inStream}(h_B) & \mathbf{if}\ \dot{m}_B > 0 \\ h_B & \mathbf{if}\ \dot{m}_B \leq 0 \end{cases} \\ &= \dot{m}_B \cdot \begin{cases} h_A & \mathbf{if}\ \dot{m}_B > 0 \\ h_B & \mathbf{if}\ \dot{m}_B \leq 0 \end{cases} \end{aligned} \tag{11}$$

Note, that h_A and h_B are either states of the respective volumes or can be directly computed from the states via the media equations, which means that these variables are "known". The equation in (11) still contains a relation of the unknown mass flow rate ($\dot{m}_B > 0$). However, this is uncritical here, because this relation can be directly computed from the states of the volume and from the pressure-drop equations of the two flow models:

$$
\begin{aligned}
\dot{m}_B &= f_1\left(p_{1\_b} - p_B, \rho_{1ba}, \rho_{1ab}\right) \\
&= f_2\left(p_A - p_{1\_b}, \rho_{2ba}, \rho_{1ab}\right) \\
\rho_{1ab} &= \rho\left(p_B, \mathbf{inStream}(h_{1\_a})\right) = \rho\left(p_B, h_B\right) \\
\rho_{1ba} &= \rho\left(p_{1\_b}, \mathbf{inStream}(h_{1\_b})\right) = \rho\left(p_{1\_b}, h_A\right) \\
\rho_{2ab} &= \rho\left(p_{1\_b}, \mathbf{inStream}(h_{2\_a})\right) = \rho\left(p_{1\_b}, h_B\right) \\
\rho_{2ba} &= \rho\left(p_A, \mathbf{inStream}(h_{2\_b})\right) = \rho\left(p_A, h_A\right)
\end{aligned}
\tag{12}
$$

Function $f_1(..)$ is the pressure drop relation of component flow 1 and $f_2(..)$ the respective one of component flow 2. These functions depend basically on the pressure difference between their ports and the upstream density ρ, i.e., ρ_{ba} when the flow is from port "b" to port "a" and ρ_{ab} when the flow is from port "a" to port "b" (other dependencies, e.g., from dynamic viscosity η, can be handled in a similar way). Since p_A, p_B, h_A, h_B are "known" quantities from volumes A and B, respectively, (12) are basically a non-linear implicit equation to compute p_{1_b} and then an explicit equation to compute $\dot{m}_B$. Since $f_1(..)$ and $f_2(..)$ are strictly increasing monotonic functions that are at least continuous, the scalar non-linear equation system is easy to solve. Once $\dot{m}_B$ is computed, the relation appearing in the energy balance of volume B, $\dot{m}_B > 0$, can be calculated as well.

To summarize, the system in Figure 3 is easy to solve numerically and only requires to solve one (uncritical) scalar non-linear equation in one unknown and otherwise only a series of explicit equations needs to be solved. The above analysis holds if the medium equations are a function of pressure p and specific enthalpy h. Other medium dependencies are uncritical as well, as discussed in section 4.

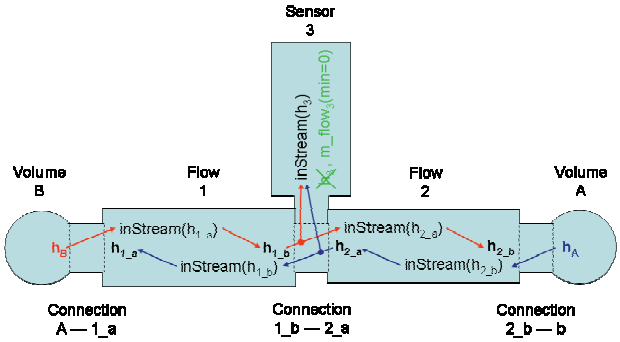

Figure 4: Exemplary series connection of multiple models with stream connectors and a sensor component.

The outflow values of connectors that never provide outflow, like absolute sensors, do not need to be considered for the expansion of inStream operators. Such connectors are defined by using the `min` attribute for the flow variable, see Figure 4.

The inStream value of an unconnected connector is defined to be equal to the outflow value.

The complete specification of stream connectors, for inside and outside connectors is given in [6].

4 Numerical Properties

In this section, the regularization of the inStream() operator is sketched and several other numerical properties are analyzed.

4.1 Regularization of the inStream() Operator

As already analyzed in the preceding section, the return value of the **`inStream()`** operator cannot be uniquely computed if all mass flow rates are zero. When using equation (8), even a division by zero occurs. In [6] this issue is resolved by requiring that the **`inStream()`** operator is appropriately approximated in an open neighborhood of vanishing mass flow rates and that the approximation must fulfill the following requirements:

1. **`inStream`**$(h_{outflow,i})$ must be unique with respect to all values of the flow and stream variables in the connection set, and must have a continuous dependency on them.
2. Every solution of the implicit equation system (6) must fulfill (6) identically (up to the usual numerical accuracy), provided the absolute value of every flow variable in the connection set is greater than a small value ε > 0 ($|\dot{m}_1| > \varepsilon$ **and** $|\dot{m}_2| > \varepsilon$ ... **and** $|\dot{m}_n| > \varepsilon$). This means that the balance equation is approximated only for small mass flow rates and otherwise is exactly fulfilled.

There are several possibilities to fulfill these requirements. In Figure 5, the definition of the inStream() operator is shown for the case of a three-way mixing point before applying any approximations. The region with all mass flow rates but $\dot{m}_i$ zero or positive is left open as the value of the operator $\text{inStream}(h_{outflow,i})$ is arbitrary by definition here.

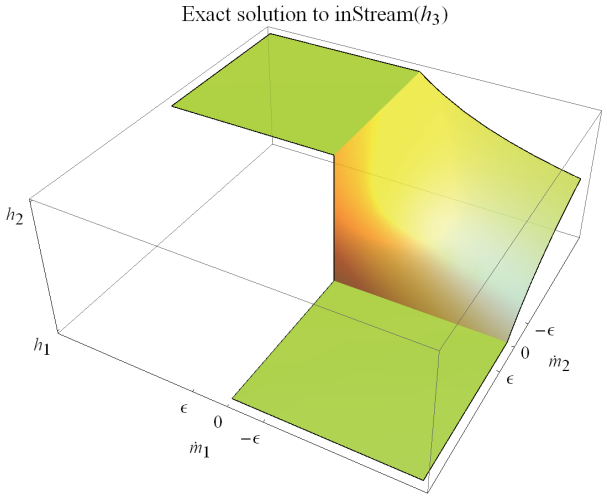

Figure 5: Exact solution for a three-way connection

In the Modelica Language Specification [6], a recommended regularization of the inStream() operator is given. First, let σ_i be the sum of the mass flow rates considered for applying the operator to port *i*.

$$\sigma_i = \sum_{j=1\ldots n,\ j\neq i} \max(-\dot{m}_j, 0)$$

Then, the expressions using the conventional operator max(x, 0) in equation (8) are substituted by a custom operator positiveMax(x, σ_i), which is defined such that it always returns a positive, non-zero value.

$$\text{inStream}(h_{outflow,i}) = \frac{\sum_{j=1\ldots n,\ j\neq i} h_{outflow,j} \cdot \text{positiveMax}(-\dot{m}_j, \sigma_i)}{\sum_{j=1\ldots n,\ j\neq i} \text{positiveMax}(-\dot{m}_j, \sigma_i)} \quad (13)$$

A suitable definition of the operator is a linear combination of max(*x*, 0) and $\varepsilon \in \Re^+$ along a suitably chosen variable α.

$$\text{positiveMax}(x, \sigma_i) = \alpha \cdot \max(x, 0) + (1-\alpha)\cdot\varepsilon .$$

The variable α is a C^1 smooth step function of σ_i.

$$\alpha(\sigma_i) = \begin{cases} 1 & \text{if } \sigma_i > \varepsilon \\ \left(3 - 2\frac{\sigma_i}{\varepsilon}\right)\left(\frac{\sigma_i}{\varepsilon}\right)^2 & \text{if } 0 < \sigma_i \le \varepsilon \\ 0 & \text{if } \sigma_i \le 0 \end{cases}$$

As a result, the value of the inStream() operator is always well-defined and is a continuous function of the variables entering (8). If all mass flow rates are zero, positiveMax(…) = ε, and

$$\text{inStream}(h_{outflow,i}) = \frac{\sum_{j=1\ldots n,\ j\neq i} h_{outflow,j} \cdot \varepsilon}{\sum_{j=1\ldots n,\ j\neq i} \varepsilon} = \frac{\varepsilon \cdot \sum_{j=1\ldots n,\ j\neq i} h_{outflow,j}}{\varepsilon \cdot (n-1)} = \frac{\sum_{j=1\ldots n,\ j\neq i} h_{outflow,j}}{n-1}$$

that is, the operator returns the arithmetic mean of the stream variables (but without $h_{outflow,i}$). Figure 6 shows an illustration of this regularization for the case of a three-way mixing point. Herein, the arithmetic mean value is shown in blue. Note that outside of the regularization domain points remain, which are not continuously differentiable. This is necessary due to the second requirement to the regularization, which states that the approximation must be exact whenever the absolute values of all flow variables are greater than a given small value.

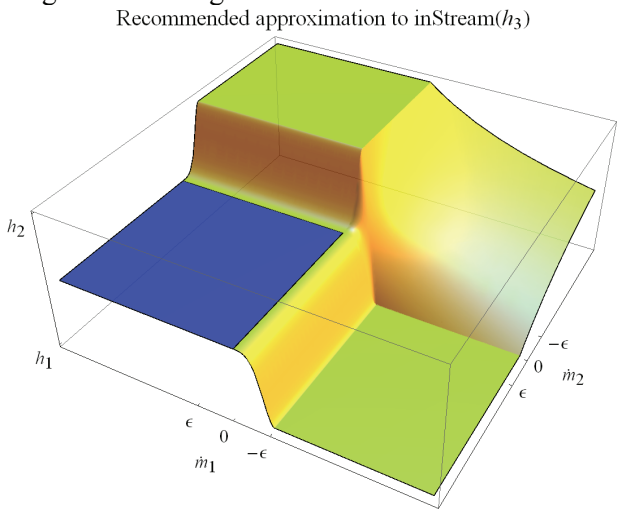

Figure 6: Recommended approximation for a three-way connection

Note, the following trivial implementation of the positiveMax() operator is also allowed according to [6]:

$$\text{positiveMax}(x, \sigma_i) = \max(x, \varepsilon).$$

In this case, the approximation is still exact whenever the absolute values of all flow variables are greater than a given small value. However, in the regularization region the operator is no longer continuously differentiable. See figure 7 for an illustration.

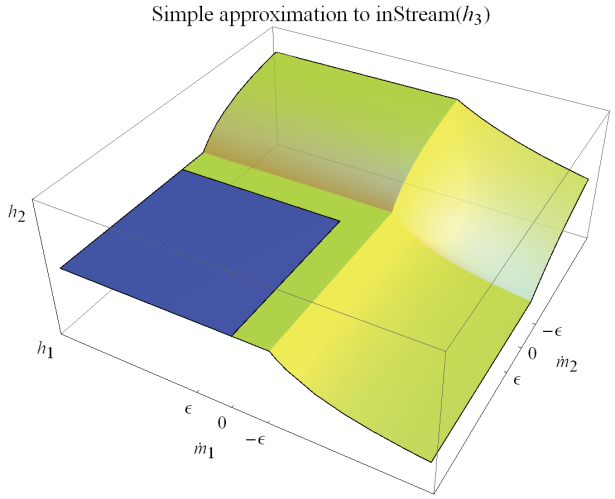

Figure 7: Simplified approximation for a three-way connection

4.2 Iteration Variables for Nonlinear Algebraic Equation Systems

For the robustness of the simulation, the choice and number of iteration variables in non-linear equation systems are crucial. We will discuss this issue at the example of a three-way mixing point with flow models ("detailed wall friction with laminar and turbulent region"), see Figure 8, and in the general case.

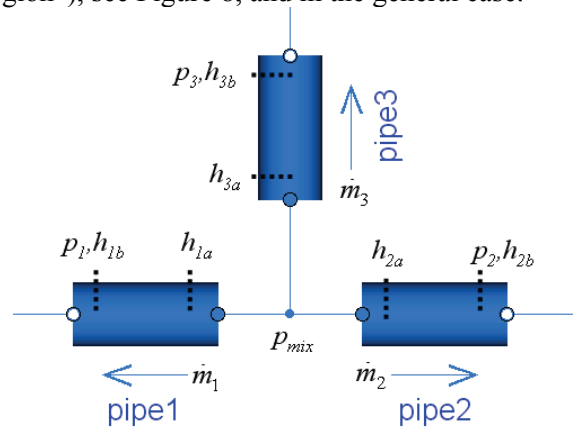

Figure 8: Three way mixing point.

As medium, the fluid "Modelica.Media.IdealGases.MixtureGases.FlueGasSixComponents" is used which is a mixture of six ideal gas substances. The independent medium variables are pressure p, temperature T and 5 independent mass fractions Xi. This system is basically described by the following set of equations:

$$
\begin{aligned}
0 &= \dot{m}_1 + \dot{m}_2 + \dot{m}_3 \\
\dot{m}_1 &= f_1\left(p_{mix} - p_1, \rho_{1ab}, \rho_{1ba}, \eta_{1ab}, \eta_{1ba}\right) \\
\dot{m}_2 &= f_2\left(p_{mix} - p_2, \rho_{2ab}, \rho_{2ba}, \eta_{2ab}, \eta_{2ba}\right) \\
\dot{m}_3 &= f_3\left(p_{mix} - p_3, \rho_{3ab}, \rho_{3ba}, \eta_{2ab}, \eta_{2ba}\right) \\
\rho_{1ab} &= \rho(p_{mix}, T_{1a\_inflow}, \mathbf{inStream}(Xi_{1a})) \\
T_{1a\_inflow} &= T\left(p_{mix}, \mathbf{inStream}(h_{1a}), \mathbf{inStream}(Xi_{1a})\right) \\
\mathbf{inStream}(h_{1a}) &= \frac{h_{2a} \cdot max(-\dot{m}_2, \varepsilon) + h_{3a} \cdot max(-\dot{m}_3, \varepsilon)}{max(-\dot{m}_2, \varepsilon) + max(-\dot{m}_3, \varepsilon)} \\
&\ldots
\end{aligned}
$$

This set of nonlinear algebraic equations can be basically solved with three iteration variables, namely two mass flow rates and the pressure p_{mix} in the mixing point: If two mass flow rates are given, the third mass flow rate can be computed via the mass balance (the first equation in the set of equations above). Furthermore, the inStream operators each depend on h_outflow's of the components attached to the mixing points and can therefore be computed, once the mass flow rates are known (e.g. h_{2a} = inStream(h_{2b}) which can be, e.g., computed from the states of the volume connected to pipe2; similarly h_{3a}, Xi_{2a}, Xi_{3a} can be computed etc., and then **inStream**(h_{1a}), **inStream**(Xi_{1a}) etc. can be computed).

In order to compute all needed intensive quantities for the ideal gas mixture, the (unknown) temperature T must be computed from pressure, specific enthalpy and mass fractions. This requires in general to solve one non-linear algebraic equation in one unknown. In the Modelica.Media package, this is performed with the algorithm of Brent [1] which is completely reliable and very efficient[1]. Once the temperature is known, densities ρ and dynamics viscosities η may be computed for the wall friction correlations. Also, with the pressure in the mixing point, all pressure differences can be computed and finally all mass flow rates via functions $f_1(..)$, $f_2(..)$, $f_3(..)$. The 3 residual equations are then the differences between the mass flow rates given by the solver and the ones computed from the wall friction correlations.

In the general case of a N-way mixing point, a similar analysis shows that N-1 mass flow rates and

[1] An interval is given in the medium definition, in which the root must be present. If possible, a smaller interval is computed by inverse quadratic interpolation (interpolating with a quadratic polynomial through the last 3 points and computing the zero). If this fails, bisection is used, which always reduces the interval by a factor of 2. The inverse quadratic interpolation method has superlinear convergence. This is roughly the same convergence rate as a globally convergent Newton method, but without the need to compute derivatives of the non-linear function.

the mixing point pressure can be used as iteration variables for a reduced nonlinear algebraic equation system of size N. Note, the number of iteration variables is independent of the number of substances in the medium.

Since the iteration variables – pressure and mass flow rates – are always continuous and in many cases also continuously differentiable, the non-linear algebraic equation system is well-posed and fulfills the requirements of a non-linear algebraic equation solver. For example, in Figure 9 , a typical simulation result of the 3-way mixing point of Figure 8 is shown, where it is clearly seen that the two iteration variables $\dot{m}_1(t)$ and $\dot{m}_2(t)$ are continuous and even continuously differentiable as function of time *t*.

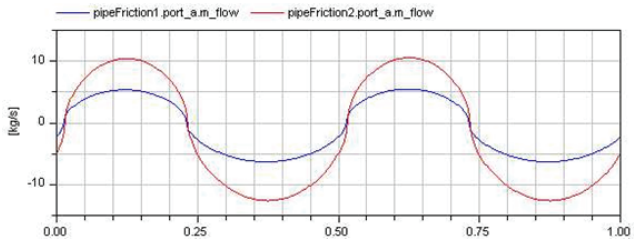

Figure 9: Mass flow rates as iteration variables of the three way mixing point.

The model equations need to be implemented in a specific form, in order to get the desirable properties from above, see section 5.2. The important point is that a function is called to compute the independent variables of the medium (i.e *p*,*T*,*Xi* in the case of the flue gas medium above) from the potential and streams variables of the connector (*p*,*h*,*Xi*) and not vice versa.

In former Beta versions of the Modelica_Fluid library [4], as well as in the ThermoPower library [2], the inverse correlation is used instead: In a "flow" model component a function is present to compute (*p*,*h*,*Xi*) from the medium states (*p*,*T*,*Xi*) via the BaseProperties model which is present in the Modelica.Media models. In such a case, a tool needs to use the temperatures close to the connection points as additional iteration variables. Temperatures are, however, a very bad choice for iteration variables, if bi-directional flow can occur, because they change usually discontinuously when the mass flow changes direction. For example, in Figure 10, the temperatures of Figure 8 are shown that would be used as iteration variables in such a case.

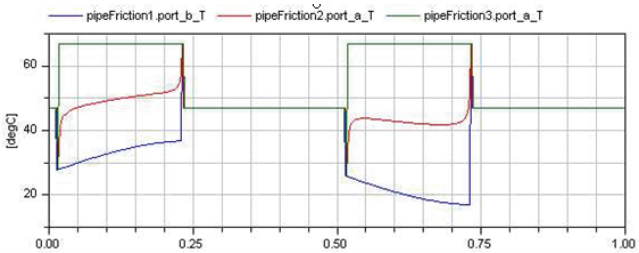

Figure 10: Temperatures as iteration variables of the three way mixing point.

As can be clearly seen, the temperatures are discontinuous and discontinuous iteration variables violate the pre-requisite of every nonlinear algebraic solution method and a reliable solution can therefore not be expected.

The results of the analysis above are summarized in the following table:

Library	*smoothness of iterat. var.*	*number of iterat. var.*
Modelica_Fluid Beta [4]	discontinuous	22
ThermoPower [2]	discontinuous	6
Modelica.Fluid [5]	continuous	3

The above table shows some key results, when simulating the three-way mixing point example from Figure 8 with different Modelica packages with Dymola 7.2 [2]:

- The Beta version of the Modelica_Fluid library [4] uses the connectors from section 2.3 and gives rise to a nonlinear algebraic equation system of 22 iteration variables, where many of the iteration variables are discontinuous when mass flow rate changes direction. So, a reliable solution is not possible.
- The ThermoPower library [2] uses the connectors from section 2.2. If only 1:1 connections are present, the numerical properties of the ThermoPower library and of the streams concept are similar, since a similar technique is used to propagate the specific enthalpy along the connected "flow" models. The main difference is that with the streams concept no longer connection restrictions are present. If an ideally mixing component is added to the ThermoPower library, the three-way mixing example gives rise to a nonlinear algebraic equation system of 6 iteration variables, where 3 of them are discontinuous when mass flow rate changes direction. So, a reliable solution is not possible in this case. The reason is that in the ThermoPower library the connector variables are computed from the medium states and not vice versa.
- The Modelica.Fluid library [5] uses the stream connector concept of section 3. The three-way mixing example gives rise to a nonlinear algebraic

equation system of 3 iterations variables where all of them are continuous. So, a reliable solution is to be expected. This demonstrates that there is a tremendous difference in numerical reliability between the Beta version of Modelica_Fluid and the Modelica.Fluid library released with the Modelica Standard Library 3.1 in August 2009.

The analysis above was performed for an ideal mixing point with a multi-substance medium. One might expect that the described problems are gone if a more detailed model with a volume is used for the mixing point. However, during steady-state initialization and/or for a pure steady-state simulation, the time derivatives of the dynamic states are set to zero and again nonlinear equation systems with similar properties are present. Again, it can be expected that the streams-connector approach leads to equation systems with a small number of continuous iteration variables and can therefore be solved reliably.

Note, if a volume would be used in the connection point of the three-way mixing point, the overall model would have 7 additional states (p, T, 5 mass fractions Xi). If the more precise description is not needed, one can therefore expect that the ideal mixing formulation (leading to a reliably solvable nonlinear algebraic equation system with 3 iteration variables) could result in a more efficient simulation.

4.3 Inverse Function Annotation

The streams connector concept introduced in section 3 is very well suited for media where the potential and stream variables in the connector (e.g., p, h, Xi) are also used as medium states. For media with other media states, there is the disadvantage that for every connection a non-linear algebraic equation has to be solved in order to compute the medium states from the potential and streams variables of the connector. For example, if a medium with independent variables pressure p and temperature T was used for the example in Figure 3, then four scalar non-linear algebraic equations need to be solved to compute (p,T) from (p,h) at points 1a, 1b, 2a, and 2b. This is completely unnecessary for points 1a and 2b, since here the medium states could just be propagated from the volumes.

In order to more efficiently cope with such situations, in Modelica 3.1 [6], section 12.8, a new annotation is introduced to define the inverses of functions. For example, (p,T) media should have the following two basic function definitions:

```
function h_pT
   input  Real p    "pressure";
   input  Real T    "temperature";
   output Real h    "specific enthalpy";
algorithm
   ...
end h_pT;

function T_ph
   input  Real p    "pressure";
   input  Real h    "specific enthalpy";
   output Real T    "temperature";
   annotation(inverse(h = h_pT(p,T));
algorithm
   ...
end T_ph;
```

For a (p,T) media the specific enthalpy h is computed explicitly via function h_pT(..). Additionally, the inverse function T = T_ph(..) is needed which will usually need to call a non-linear algebraic equation solver in order to invert function h_pT(..). In such a case, function T_ph(..) should have the annotation "**inverse**(h = h_pT(p,T)". This annotation states that the inverse of T_ph is function h_pT. The essential requirement is that the inverse function must have the same input/output arguments as the direct function, but the order of the arguments may be permuted. Especially, a variable that was declared as "input", is used as "output" in the inverse function.

The "inverse" annotation signals a tool that it should use this function if possible, because it will be more efficient than using the "direct" function. For example, in Figure 11 two components are connected together and stream variables are used in the connectors. Let us assume that p, T_1 are known variables, since they are states of a volume.

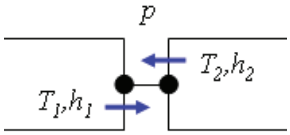

Figure 11: Enhancing efficiency with the inverse annotation when propagating media properties.

This situation is described by the following equations:

$$\begin{aligned} h_1 &= h_{pT}(p, T_1) \\ \mathbf{inStream}(h_2) &= h_1 \\ T_{2,inflow} &= T_{ph}(p, \mathbf{inStream}(h_2)) \\ &= T_{ph}(p, h_1) \end{aligned}$$

Since the inverse function h_{pT} is defined as annotation in function T_{ph}, the tool is advised to try whether the usage of the inverse function will simplify the equations. Indeed this is the case here: If the last equation is replaced by the inverse function:

$$h_1 = h_{pT}(p, T_1)$$
$$\mathbf{inStream}(h_2) = h_1$$
$$h_1 = h_{pT}(p, T_{2,inflow})$$

it can be detected that the same function is called twice. In both cases, the same arguments p,h_1 are used. The second input arguments T_1, $T_{2,inflow}$ must be identical, in order that this can hold. Therefore, the above equations can be simplified to:

$$h_1 = h_{pT}(p, T_1)$$
$$\mathbf{inStream}(h_2) = h_1$$
$$T_{2,inflow} = T_1$$

As a result, function T_{ph} need no longer be called and therefore one scalar nonlinear algebraic equation computation is removed.

The above simplification rule is, e.g., available in Dymola 7.2 [2]. In a few media of Modelica.Media (version 3.1) the inverse annotation is already included. It is planned to include the annotation for all media where this makes sense in the follow-up version of the Modelica Standard Library.

To summarize, with the inverse annotation, appropriate media description and corresponding tool support, the unnecessary scalar nonlinear algebraic equation systems are avoided for 1:1 connections, when using media that do not have (p,h,X) as states.

5 Examples for Basic Fluid Models

Thermo-fluid models are built from components, each representing a control volume. A control volume is defined as a fixed region in space where one studies the masses and energies crossing the boundaries of the region. The boundaries of a control volume usually represent the physical boundaries of the parts through which fluid flow is occurring. Often two basic kinds of component models are distinguished:

- Volume models define a thermodynamic state for a lumped medium.
- Transport or flow models serve to connect volume models. They do not define an own thermodynamic state. Instead they define the transport of fluid for thermodynamic states given at their ports.

Multiple volume and transport models can be assembled to build component models hierarchically. The following examples use the

```
connector FluidPort = FluidPort_Stream;
```

5.1 Volume model

A basic model of an ideally mixed lumped volume is defined below.

```
model Volume "Lumped volume, e.g. vessel"

  replaceable package Medium =
Modelica.Media.Interfaces.PartialMedium;

  parameter Integer nPorts=0
    "Number of ports";
  FluidPort[nPorts] ports;

  parameter Modelica.SIunits.Volume V
      "Volume of device";
  Modelica.SIunits.Mass             m
      "Mass in device";
  Modelica.SIunits.Energy           U
      "Inner energy in device";

  Medium.BaseProperties medium;

equation
  // Definition of port variables
  for i in 1:nPorts loop
    ports[i].p = medium.p;
    ports[i].h_outflow = medium.h;
  end for;

  // Mass and energy balance
  m = V*medium.d;
  U = m*medium.u;
  der(m) = sum(ports.m_flow);
  der(U) = ports.m_flow *
             actualStream(ports.h_outflow);
end Volume;
```

The volume model can be used with an arbitrary medium model out of Modelica.Media. The mass balance and the energy balance sum up the contribution of flows going through nPorts fluid ports.

Alternatively the volume model could be defined with only one fluid port as multiple connections can be made to it. Then, however, ideal mixing would take place in the port outside the volume. Using unary connections to multiple ports ensures that the mixing takes place inside the volume. This is the generally intended behavior for multi-way connections to a volume. Nonlinear systems of mixing equations are avoided.

The actualStream operator used to define the energy balance is a shorthand notation for:

```
actualStream(port.h_outflow) ==
  if port.m_flow > 0 then
    inStream(port.h_outflow) else
    port.h_outflow;
```

Note that with stream variables the Boolean unknown for the flow direction appears in the energy balance, where the flow dependent value of the specific enthalpy is multiplied with the mass flow rate. This is why a reverting flow, i.e. m_flow crossing

zero, does not cause jumping values in the model equations.

5.2 Transport model

The flows are defined with transport models. A basic model for isenthalpic flow is given below:

```
model IsenthalpicFlow
  "Flow model without storage of mass or
   energy, e.g. fitting or valve"

  replaceable package Medium =
Modelica.Media.Interfaces.PartialMedium;

  FluidPort port_a, port_b:

  Medium.ThermodynamicState state_a
    "State at port_a if inflowing";
  Medium.ThermodynamicState state_b
    "State at port_b if inflowing";

equation
  // Medium states for inflowing fluid
  state_a = Medium.setState_phX(
              port_a.p,
              inStream(port_a.h_outflow));
  state_b = Medium.setState_phX(
              port_b.p,
              inStream(port_b.h_outflow));

  // Mass balance
  0 = port_a.m_flow + port_b.m_flow;

  // Isenthalpic energy balance
  port_a.h_outflow =
         Medium.specificEnthalpy(state_b);
  port_b.h_outflow =
         Medium.specificEnthalpy(state_a);

  // (Regularized) Momentum balance
  port_a.m_flow = f(
              port_a.p, port_b.p,
              Medium.density(state_a),
              Medium.density(state_b));
end IsenthalpicFlow;
```

The flow model does not define own thermodynamic states as it has no storage of mass or energy. Instead the mass flow rate is defined for thermodynamic states seen through the ports for the case of inflow. These are generally the (transformed) states defined by connected volume models.

The flow model defines steady-state mass, energy and momentum balances. The function f gives the relationship between pressure drop and mass flow rate.

The use of the ThermodynamicState records state_a and state_b also ensures that appropriate equation systems are generated by a Modelica tool for medium models that do not use pressure and enthalpy as independent variables, such as gas models often using pressure and temperature instead (see section 4 above).

5.3 Sensor model

Ideal sensor models are used to pick up fluid properties, such as temperatures. They do not contribute to the fluid flow. A basic model for a temperature sensor is given below.

```
model TemperatureSensor
  "Ideal temperature sensor"

  replaceable package Medium =
Modelica.Media.Interfaces.PartialMedium;

  FluidPort port(m_flow(min=0))
    "Port with no outflow ever";

  Modelica.Blocks.Interfaces.RealOutput T
    "Upstream temperature";

equation
  T = Medium.temperature(
        Medium.setState_phX(port.p,
          inStream(port.h_outflow));

  port.m_flow = 0;

  port.h_outflow =
    Medium.specificEnthalpy(
      Medium.setState_pTX(
        Medium.reference_p,
        Medium.reference_T))
    "Never used, but seen in plots";
end TemperatureSensor;
```

The mass flow rate in the port is defined to be non-negative. This tells the translation tool that the outflow enthalpy defined by the sensor must not be used in any mixing equations, as outflow does never happen; see also Figure 4 above.

6 Conclusions

So far it has not been possible to agree on a common approach for the formulation of fluid ports. Existing approaches have either been based on control signals, which do not allow the device-oriented modeling of fluid systems, or on pairs of potential/flow variables, which lead to numerically unreliable equation systems.

The new stream variables allow the declarative formulation of fluid ports. Stream variables define the convective transport of specific quantities, such as specific enthalpy or chemical composition. The semantics is simple and can easily be supported by a Modelica tool.

Stream connectors have been standardized in Modelica 3.1. They are used in Modelica.Fluid. Cur-

rently Dymola 7.2 [2] and SimulationX 3.2 [8] support stream connectors. Other tool vendors already announced to support the concept, too.

The streams concept is a big step forward for fluid modeling in Modelica. However, stream connectors are not yet the ultimate solution for fluid modeling because there are still missing features:

- The used medium has currently to be defined for every component. It would be nicer if the medium was defined at one source and the medium definition would then be propagated through the connection structure.
- When components are connected together, the connection semantics ensures that the mass and the energy balance are fulfilled exactly (in the sense of "ideal" mixing). The momentum balance is exact in case of identical dynamic pressure for each of the connected flanges. If this approximation is not sufficiently accurate, an appropriate explicit junction model has to be used. It would be useful if also the momentum balance was automatically fulfilled when 3 or more components are connected together. This requires including information about the geometry of fluid flanges into the connector description.

7 Acknowledgments

Partial financial support of ABB and of DLR for this work within the ITEA2 project EUROSYSLIB is highly appreciated (BMBF Förderkennzeichen: 01IS07022F).

References

[1] R.P. Brent (1973): **Algorithms for Minimization without derivatives**. Prentice Hall, pp. 58-59.

[2] F. Casella, A. Leva (2003): **Modelica open library for power plant simulation: design and experimental validation**. Proceedings of the Modelica 2003 Conference, editor: P. Fritzson, Linköping, Sweden.
www.modelica.org/events/Conference2003/papers/h08_Leva.pdf

[3] Dymola (2009). **Dymola Version 7.2**. Dassault Systèmes, Lund, Sweden (Dynasim).
www.dymola.com

[4] H. Elmqvist, H. Tummescheit, M. Otter (2003): **Object-Oriented Modeling of Thermo-Fluid Systems**. Proceedings of the Modelica 2003 Conference, editor: P. Fritzson, Linköping, Sweden.
www.modelica.org/events/Conference2003/papers/h40_Elmqvist_fluid.pdf

[5] R. Franke, F. Casella, M. Sielemann, K. Proelss, M.Otter, M. Wetter (2009): **Standardization of thermo-fluid modeling in Modelica.Fluid**. Proceedings of the Modelica 2009 Conference, editor: F. Casella, Como, Italy.
www.modelica.org/events/modelica2009

[6] Modelica (2009): **Modelica Language Specification, Version 3.1**.
www.modelica.org/documents/ModelicaSpec31.pdf

[7] H. Olsson, M. Otter, S.E. Mattsson, H. Elmqvist (2008): **Balanced Models in Modelica 3.0 for Increased Model Quality**. Proceedings of the Modelica 2008 Conference, editor: B. Bachmann, Bielefeld, Germany.
www.modelica.org/events/modelica2008/Proceedings/sessions/session1a3.pdf

[8] SimulationX (2009). **SimulationX Version 3.2**. ITI, Dresden, Germany.
www.simulationx.com

Appendix

A1 Determination of the Enthalpy for Ideal Mixing

In previous sections a central formula is an explicit equation to compute the ideal mixing enthalpy of an infinitesimally small connection point. In this section, this formula is derived. For simplicity, it is first derived at hand of 3 model components that are connected together, see Figure 2. The case for N connections follows correspondingly.

The energy and mass balance equations at the infinitesimally small control volume of the connection point of the three connected components are:

$$\begin{aligned} 0 = &\dot{m}_1 \cdot \begin{cases} h_{mix} & \textbf{if } \dot{m}_1 > 0 \\ h_{outflow,1} & \textbf{if } \dot{m}_1 \le 0 \end{cases} \\ &+ \dot{m}_2 \cdot \begin{cases} h_{mix} & \textbf{if } \dot{m}_2 > 0 \\ h_{outflow,2} & \textbf{if } \dot{m}_2 \le 0 \end{cases} \\ &+ \dot{m}_3 \cdot \begin{cases} h_{mix} & \textbf{if } \dot{m}_3 > 0 \\ h_{outflow,3} & \textbf{if } \dot{m}_3 \le 0 \end{cases} \end{aligned} \tag{1a}$$

$$0 = \dot{m}_1 + \dot{m}_2 + \dot{m}_3 \tag{1b}$$

With the max(..) operator:

$$\max(a,b) = \textbf{if } a > b \textbf{ then } a \textbf{ else } b$$

the balance equations above can be rewritten:

$$\begin{aligned} 0 = &\max\left(\dot{m}_1, 0\right) \cdot h_{mix} - \max\left(-\dot{m}_1, 0\right) \cdot h_{outflow,1} \\ &+ \max\left(\dot{m}_2, 0\right) \cdot h_{mix} - \max\left(-\dot{m}_2, 0\right) \cdot h_{outflow,2} \\ &+ \max\left(\dot{m}_3, 0\right) \cdot h_{mix} - \max\left(-\dot{m}_3, 0\right) \cdot h_{outflow,3} \end{aligned} \tag{2a}$$

$$\begin{aligned} 0 = &\max\left(\dot{m}_1, 0\right) - \max\left(-\dot{m}_1, 0\right) \\ &+ \max\left(\dot{m}_2, 0\right) - \max\left(-\dot{m}_2, 0\right) \\ &+ \max\left(\dot{m}_3, 0\right) - \max\left(-\dot{m}_3, 0\right) \end{aligned} \tag{2b}$$

Equation (2a) is solved for h_{mix}

$$h_{mix} = \frac{\begin{pmatrix} \max(-\dot{m}_1,0)\cdot h_{outflow,1} + \\ \max(-\dot{m}_2,0)\cdot h_{outflow,2} + \\ \max(-\dot{m}_3,0)\cdot h_{outflow,3} \end{pmatrix}}{\max(\dot{m}_1,0)+\max(\dot{m}_2,0)+\max(\dot{m}_3,0)}$$

Using (2b), the denominator can be changed to:

$$h_{mix} = \frac{\begin{pmatrix} \max(-\dot{m}_1,0)\cdot h_{outflow,1} + \\ \max(-\dot{m}_2,0)\cdot h_{outflow,2} + \\ \max(-\dot{m}_3,0)\cdot h_{outflow,3} \end{pmatrix}}{\max(-\dot{m}_1,0)+\max(-\dot{m}_2,0)+\max(-\dot{m}_3,0)}$$

This is an explicit formula to compute h_{mix} for three connected components. It is straightforward to generalize this formula to "n" connections:

$$h_{mix} = \frac{\sum_{j=1\ldots n} \max(-\dot{m}_j,0)\cdot h_{outflow\,j}}{\sum_{j=1\ldots n} \max(-\dot{m}_j,0)}$$

A2 Stream Variables in Hierarchical Models

The discussion of stream connectors has been restricted to connections on the same hierarchical level in this paper so far. In this section the handling of stream variables in hierarchical models is considered.

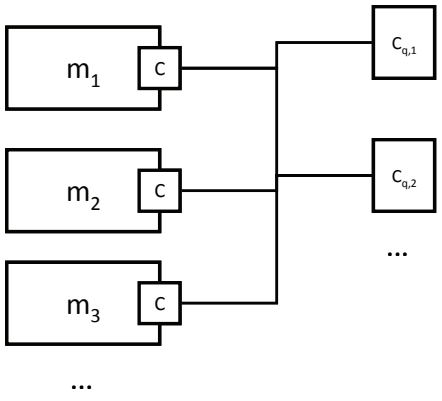

Figure 12: Exemplary fluid system with N=3 inside connectors and M=2 outside connectors

Figure 12 shows an exemplary hierarchical fluid system. If the inStream operator is used in one of the models m_i, i=1…N, then the contributions of the inside and the outside connectors of the connection set need to be taken into account. This results in:

inStream($h_{outflow,i}$)

$$= \frac{\begin{pmatrix} \sum_{j=1\ldots i-1,i+1\ldots N} h_{outflow\,j} \max(-\dot{m}_j,0) \\ + \sum_{k=1\ldots M} h_{outflow\,k} \max(\dot{m}_k,0) \end{pmatrix}}{\sum_{j=1\ldots i-1,i+1\ldots N} \max(-\dot{m}_j,0) + \sum_{k=1\ldots M} \max(\dot{m}_k,0)}.$$

Note the opposite sign for mass flow rates in the outside connectors, following the sign convention for flow variables in hierarchical models.

The models m_i each explicitly define the stream variables `h_outflow` of their port c_i. These ports are inside connectors in the shown connection set. There are no explicit equations in the hierarchical model that define the stream variables h_outflow in the outside connectors $c_{q,k}$, k=1…M. A Modelica tool needs to define the stream variables of outside connectors c_q by establishing one mixing equation for each stream variable $h_{outflow,q}$, q=1…M:

$$h_{outflfow,q} = \frac{\begin{pmatrix} \sum_{j=1\ldots N} h_{outflow\,j} \max(-\dot{m}_j,0) \\ + \sum_{k=1\ldots q-1,q+1\ldots M} h_{outflow\,k} \max(\dot{m}_k,0) \end{pmatrix}}{\sum_{j=1\ldots N} \max(-\dot{m}_j,0) + \sum_{k=1\ldots q-1,q+1\ldots M} \max(\dot{m}_k,0)}$$

This equation considers the contributions of inside and outside connectors in the connection set as well as the sign convention for flow variables in hierarchical models.

The treatment of zero flow for inside and outside connectors is the same as the treatment of zero flow in connections on the same hierarchical level (see Section 4.1).

