Chemical Reaction Modeling with ThermoFluid/MF and MultiFlash

Tummescheit, Hubertus; Eborn, Jonas

Published in:
Modelica'2002 Proceedings

2002

Tummescheit H., Eborn J.:
Chemical Reaction Modeling with ThermoFluid/MF and MultiFlash
2nd International Modelica Conference, Proceedings, pp. 31-39

Paper presented at the 2nd International Modelica Conference, March 18-19, 2002,
Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR), Oberpfaffenhofen, Germany.

All papers of this workshop can be downloaded from
http://www.Modelica.org/Conference2002/papers.shtml

Chemical Reaction Modeling with ThermoFluid/MF and MultiFlash

Hubertus Tummescheit[†] and Jonas Eborn[‡]

[†]Department of Automatic Control
Lund University, Sweden
hubertus@control.lth.se

[‡]United Technologies Research Center
East Hartford, Connecticut, USA
EbornJP@utrc.utc.com

Abstract

The free Modelica library THERMOFLUID (see [2] and [11]) was developed for simulation of thermo-hydraulic applications, both for single-species applications like the water-steam cycle in a thermal power plant and for multi-species applications with gas mixtures. It has demonstrated its flexibility for modeling thermodynamic and process applications in a variety of industrial and academic projects, see [10], [3] and [7]. This article describes how support for chemical reactions and membrane diffusion has been added to THERMOFLUID, thus expanding the area of possible applications to include reacting flows, chemical batch reactors, catalytic converters, etc. Another crucial part of the modeling work has to be spent on getting physical property data of sufficient accuracy and with acceptable computational complexity for engineering purposes into the model. This has been adressed in the development of a commercial interface to the industry-standard physical property package MultiFlash. The new Modelica library THERMOFLUID/MF provides the modeler with two toolboxes. Firstly, a low-level Modelica function interface to MultiFlash. MultiFlash consists of a core of physical property calculation routines and a basic database of the most comman chemical components and a number of add-on property databases. The interface gives access to multi-component, multi-phase property calculations including gas, several liquid and condensed phases, wax formations and hydrates. Secondly, a high-level Modelica model library which is fully integrated with the THERMOFLUID library and implements robust and efficient dynamical models for the most common process engineering equipment. In addition, reliable crossing functions for detecting phase boundaries in multi-phase, multi-component mixtures have been implemented for the first time in a high-level modeling language. The crossing functions make it possible to simulate processes correctly even at off-design operating points and under start-up conditions. A flash volume may in such cases be filled with only liquid or only gas. Crossing functions for phase transitions ensure high performance simulation even in these cases.

1 Flexible handling of chemical reactions

In standard chemical textbooks, reactions are treated as source terms in component concentration balances:

$$\frac{dc_i}{dt} = c_i^{in} - c_i^{out} + r_i \tag{1}$$

where r_i are the component reaction rates, given by a kinetic expression. In a more general way, we can include the reaction terms in the component mass balance and total energy balance

$$\frac{dM_i}{dt} = \dot{m}_i^{in} - \dot{m}_i^{out} + rZ_i \cdot MW_i \tag{2}$$

$$\frac{dU}{dt} = \dot{q}^{in} - \dot{q}^{out} + \sum_{i=1}^{nc} rZ_i \cdot H_i^f \tag{3}$$

where rZ are reaction rates in moles/s, $\dot{q}$ is convective heat flow, $\dot{m}$ mass flow and H^f is component enthalpy of formation.

1.1 ThermoFluid balance equations

In THERMOFLUID, the general balance equations are implemented in the package `BaseClasses.Balances`. The basic balance equations should not be modified by the average user and thus

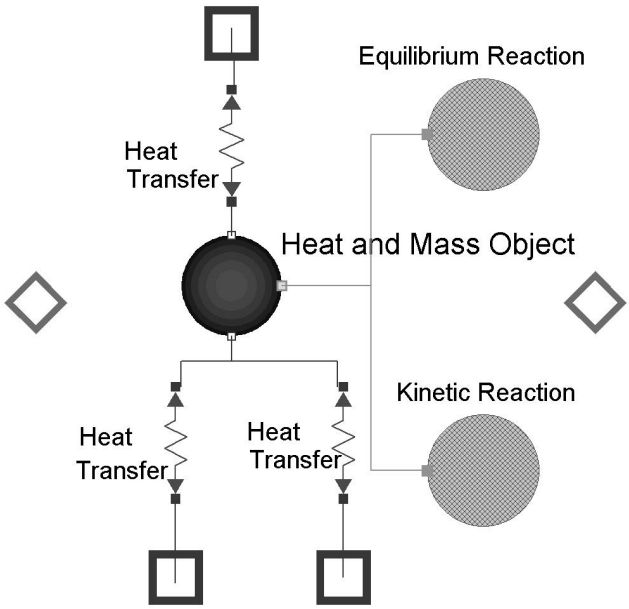

Figure 1: Schematic of the HeatAndMassObject with heat interaction and reaction objects (no diffusion connector present).

need to be general enough to handle all cases from an isolated gas volume to a reactor with added mass- and heat transfer laws. The model structure has to provide the option to add any kind of heat- and mass transfer interaction with the control volume later, as an add-on component. The basic balance equation of a control volume with two connectors is implemented as

```
dM_x = a.mdot_x + b.mdot_x + rM;
dU   = a.q_conv + b.q_conv + Q_s;
```

This means that `rM` and `Q_s`, corresponding to the source terms in (3) should be unspecified in the general base class, and then specified at a later stage when the balance class is reused in the model of a specific component. When there are no reactions or heat interaction with the volume there is no need for any source terms. In this case the model should provide a default value of zero production.

1.2 The HeatAndMassObject, a gateway to the balances

The contradiction of leaving the option open to specify production terms but not having to add a default value of zero can be handled with open flow connectors. In Modelica, all quantities which are flows are marked with the **`flow`**-prefix. Flow variables obey the zero-sum rule (Kirchhoffs' current law) and have in unconnected connectors a zero default value. Since these connectors should be internal to the volume, they need to be attached to an object inside the volume model. This is the `HeatAndMassObject`,see Figure 1, which acts as a gateway between the balance equations and possible heat- and mass transfer objects. External connectors can also be connected to the `HeatAndMassObject`.

Interfaces

The `HeatAndMassObject` interact with other objects through a number of different connectors. The currently implemented connectors include the `HeatFlow` connector for pure heat interaction (conduction/radiation), the `ChemFlow` connector for chemical reactions, both kinetic and equilibrium, and a connector for membrane diffusion.

```
connector HeatFlow
  parameter Integer         n;
  Temperature[n]            T;
  flow Power[n]             q;
end HeatFlow;

connector ChemFlow
  parameter Integer         n, nc;
  parameter String          MediumType;
  Temperature[n]            T;
  Pressure[n]               p;
  Concentration[n,nc]       conc;
  flow MolarFlowRate[n,nc] rZ;
  flow Power[n]             q;
end ChemFlow
```

The diffusion connector is similar to the `ChemFlow` connector but has mass flow rate instead of molar flow rate since this is standard for diffusion.
The flow semantics of Modelica for the molar flow `rZ` and the heat flow `q` make sure that all contributions to the mass- and energy balances are correctly taken into account, no matter whether there are zero, one or many connections to the HeatAndMassObject, see Figure 1. Inside the `HeatAndMassObject` the contributions from the different connectors are summed up and transferred to the balance equations in the volume.

1.3 Objects for encapsulating reactions

To be able to drag and drop reaction models into a volume model (a reactor), they are encapsulated in reaction objects. As shown in the code example below, the `Basic` reaction inherits interfaces and basic parameters from the reaction `BaseObject`.

```
package Reactions

partial model BaseObject
  parameter Integer       n, nc, nr;
  MolarReactionRate[n,nr] reacRate;
  Enthalpy[nc]            compHf;
  parameter StoichiometricNumber[nr,nc]
    stoich=zeros(nr,nc);
equation
  for i in 1:n loop
    r.rZ[i,:] =
      transpose(stoich)*reacRate[i,:];
  end for;
end BaseObject;

model Basic "Simple Arrhenius reaction"
  extends BaseObject;
  parameter Rate[nr]                A0;
  parameter Real[nr]                b;
  parameter MolarInternalEnergy[nr] Ea;
  Concentration[n,nc]               conc;
equation
  for i in 1:n loop
    reacRate[i,:] = ...;
    r.q[i] = -compHf*r.rZ[i,:];
  end for;
end Basic;

end Reactions;
```

The reaction rates are calculated from standard Arrhenius expressions using the concentrations and the parameters. To use the reaction component, like in the kinetic reaction in Figure 1, the user simply needs to specify the parameters. The stoichiometry matrix is constructed as shown in Table 1 and the heat of formation parameters are added from the medium model.

To construct models of other types of reactions the reaction `BaseObject` can be reused. The customized reaction model needs to give expressions for the reaction rates, either by adding equations or by calling a rate function. In this way packages of reactions can be built and reactions can graphically be added to standard reactor models.

	$\{O_2$	H_2	H_2O	O	H	OH	Ar$\}$
$H + O_2 \rightarrow OH + O$	−1	0	0	1	−1	1	0
$OH + O \rightarrow H + O_2$	1	0	0	−1	1	−1	0
$O + H_2 \rightarrow OH + H$	0	−1	0	−1	1	1	0
$H_2O + H \rightarrow H_2 + OH$	0	1	−1	0	−1	1	0
$H_2 + OH \rightarrow H_2O + H$	0	−1	1	0	1	−1	0
$H_2O + O \rightarrow 2\,OH$	0	0	−1	−1	0	2	0
$2\,H + Ar \rightarrow H_2 + Ar$	0	1	0	0	−2	0	0
$2\,O + Ar \rightarrow O_2 + Ar$	1	0	0	−2	0	0	0

Table 1: Reactions included in the H_2 O_2 reaction system and the corresponding stoichiometric matrix.

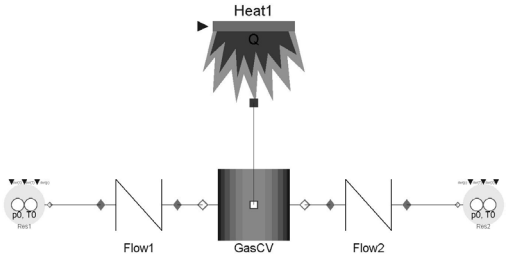

Figure 2: Schematic of example system with H_2–O_2 reaction.

1.4 Example, combustion of hydrogen

As an example, we consider the combustion of hydrogen and oxygen into water. In a simple setting, see Figure 2, the system consists of a reservoir supplying the reactants, a reactor volume and a sink for the product flow. A heat source is added to provide the heat necessary to ignite the mixture.

The complete set of sub-reactions for this process involves a large number (> 40) of very fast reactions, see [12]. Here we only consider the 8 main reactions, involving the components $\{ O_2, H_2, H_2O, O, H, OH, Ar\}$. Argon is included as an inert gas. The included reactions are listed in Table 1. The corresponding stoichiometry matrix and reaction rate parameters have been coded into a `Basic` reaction object inside the `GasCV` reaction vessel.

The result plots show clearly that the reactions are extremely fast once they started. They saturate when all H_2 is burned up and the flow through the volume reaches steady state. The mass flows in Figure 3 show a violent explosion when the mixture ignites. After the

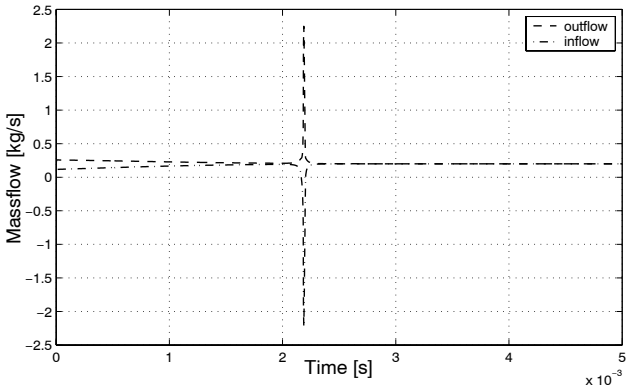

Figure 3: Mass flows into and out of the control volume during the ignition phase

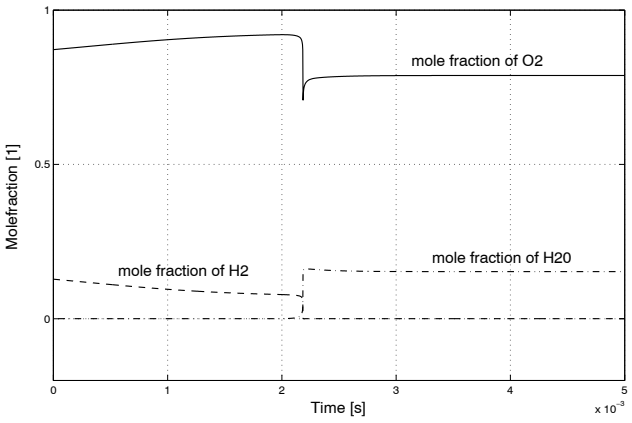

Figure 4: Molar fractions of the principal reactants and products

initial ignition, a steady inflow of premixed gases leads to a steady combustion with plenty of surplus oxygen. The speed of the reactions makes the system very stiff. The whole simulation shown in Figures 4-3 spans only a few milliseconds.

2 The MultiFlash interface

MultiFlash is the generic name for a physical property software from Infochem Ltd. It is a comprehensive system that calculates the thermophysical properties of pure substances and mixtures and carries out phase and chemical equilibrium calculations for fluid and solid phases. MultiFlash consists of several software modules: databases with the raw property data, access software to the databases and different modules for pure component property calculation, mixture models for thermodynamic properties and transport properties, handling of binary interaction parameters and phase and chemical equilibrium calculations. A number of process simulators, e. g., gPROMS from PSE, uses an interface to MultiFlash for the calculation of physical properties. For use in a dynamic simulation program typically only a small fraction of the MultiFlash functions are needed. The current interface is kept as simple as possible, with all necessary interaction with the property database encapsulated into one medium property object. The interface has been tested with both Dymola by Dynasim AB [4] and MathModelica by MathCore AB [6].

2.1 The low-level interface

The low-level interface between Modelica models and the MultiFlash modules, which are accessible via a Win32 Dynamic Link Library (dll) under Windows, consists of the standard Modelica foreign function interface for the C-language. This means that all calls to MultiFlash routines are provided exactly as documented in the MultiFlash programmers guide, including identical variable names. There are two minor, but necessary exceptions. Modelica does not allow to overwrite inputs with outputs in the calling of functions. This is common practice in Fortran numerical routines and in MultiFlash this is exclusively used to provide estimates of the solutions in the input variables. In the Modelica interface, the estimated solutions are provided as *additional* input arguments to the function and the original MultiFlash variables are kept as outputs. The second exception is the handling of error message strings. The handling of errors and warnings is done in the C wrapper functions. Diagnostic messages are written to the simulation log. A flag with the number of errors is returned in the Modelica function call for error trapping purposes.

2.2 Computational efficiency

Simulation time is an important issue and the interface library uses all available methods to make function calls computationally efficient. A simple rule is to get as many physical properties as possible from one call to MultiFlash. All essential medium properties needed for the default dynamic model are available in one property record which is calculated with one single function call to MultiFlash. The dynamic state model and other ThermoFluid models need everything in this standard property record, so it is computationally not efficient to slim down this function call. Boolean flags to the MultiFlash routines are used to ensure that only the medium properties of interest are calculated. High level functions that return only a single property have not been implemented in order to close the door on unnecessarily slow models. However, all low-level MultiFlash functions are available and thus single function calls to obtain properties can be used if it is desired.

Providing good estimates of the solution makes a big difference in the solution time for any nonlinear system of equations, especially for phase equilibrium calculations. It is obvious that for continuous, dynamic simulation the result from the last time step usually provides such an estimate. Internal caching of the last solution in the same control volume is therefore implemented in the THERMOFLUID/MF MultiFlash interface.

2.3 Using MultiFlash with ThermoFluid/MF

Setting up a model that uses MultiFlash properties is currently done through the MultiFlash windows user interface. The user selects the wanted components by querying the available MultiFlash databases. For complex systems, the MultiFlash stream facility is used to define different component streams to be used in different parts of the system. The global stream which has to be defined first contains the union of the components in all streams. If necessary, the thermodynamic models can be changed from the default suggested by MultiFlash through the graphical user interface. All standard equations-of-state (EOS) models, e. g., Redlich-Kwong-Soave or Peng-Robinson, can be used with a selection of mixing rules. Binary interaction parameters can be entered if needed. Transport property models are selected independently of the EOS models. The problem setup is saved in an mfl-file which is then read and parsed during initialization. The file name is the main parameter to the medium models in the THERMOFLUID/MF library. Problem setup files are read from the current directory or from a repository, where all problem setup definitions can be managed in a centralized way.

2.4 Dynamic state equations

The most efficient method of combining the dynamic states and the physical property calculation is to choose the dynamic states of the model such that they are inputs to the physical property calculation routines. That avoids the solution of non-linear equation systems during simulation. Otherwise, inputs to the property functions have to be computed from outputs of that functions through a non-linear equation system. This happens when the outputs are dynamic states or time-invariant parameters, like the volume in a closed vessel. If the property functions are computationally expensive relative to the rest of the model, the saving in computation time by using a model which is explicit in the states is significant. When this can not be achieved, as is the case with the MultiFlash routines, it is still preferable to get non-linear equation systems of the lowest possible dimension. Due to the MF calling structure with pressure p, temperature T and N (mole amounts) as inputs and total volume V as output a special state model has been defined. It is incorporated in the free THERMOFLUID library and can be used interchangeably with the simple ideal gas models in the free THERMOFLUID library and the commercial MultiFlash property models. The state model uses temperature T and mole amounts N as dynamic states, while p can be regarded as an algebraic variable that contains state information. For the standard case of a constant volume control volume, the pressure is solved for iteratively to ensure that the total volume is kept constant, $V_{fixed} = V(p)$. This is the only non-linear equation system in the model for single phase calculations. The dynamic state model is derived from the standard text-book form of an energy and mass balances. In block matrix notation, the inner energy and mole amount balance can be recast into temperature and moles as dynamic states as follows (boldface for vectors and matrices, sizes follow from dimension of N, the number of components in the mixture.):

$$\begin{pmatrix} \mathbf{N}_t \\ U_t \\ V_t \end{pmatrix} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \frac{\mathbf{dU}}{\mathbf{dN}}\big|_{T,V} & \frac{dU}{dT}\big|_{\mathbf{N},V} & \frac{dU}{dV}\big|_{\mathbf{N},T} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{N}_t \\ T_t \\ V_t \end{pmatrix} \tag{4}$$

The subscript t is used for the time derivative, N stands for mole amounts, U for total inner energy. The inverse of the jacobian is used to make this model explicit in the mole vector, the temperature and the volume as dynamic states:

$$\mathbf{J}^{-1} = \frac{1}{\frac{dU}{dT}\big|_{\mathbf{N},V}} \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ -\frac{\mathbf{dU}}{\mathbf{dN}}\big|_{T,V} & -\frac{dU}{dT}\big|_{\mathbf{N},V} & -\frac{dU}{dV}\big|_{\mathbf{N},T} \\ 0 & 0 & 1 \end{pmatrix} \tag{5}$$

The structure of the jacobian inverse reveals that only the equation for the inner energy is transformed into one for the temperature. The mole balance equations remain unchanged from (4).

The partial derivatives occuring in the transformed dynamic and initial equations can be calculated from derivatives that are returned by MultiFlash by setting the appropriate flags. However, the derivatives must be transformed using thermodynamic determinants since MultiFlash returns derivatives at constant pressure and the THERMOFLUID balances are derived at constant volume. As an example we pick the derivative of total enthalpy w. r. t. temperature (all derivatives are at constant composition):

$$\left.\frac{\partial H}{\partial T}\right|_V = \left.\frac{\partial H}{\partial T}\right|_p - \left.\frac{\partial H}{\partial V}\right|_T = \left.\frac{\partial H}{\partial T}\right|_p - \left(\frac{\partial H/\partial p|_T}{\partial V/\partial p|_T}\right) \tag{6}$$

All derivatives on the right hand side of the equation are returned by standard MultiFlash property calls.

2.5 Initialization

In the Modelica 2.0 specification and Dymola version 4.2 the possibility to separate the equations for steady state initialization from the dynamic states was introduced. Due to that separation, a control volume can now easily be initialized at steady state pressure, even when the pressure is not a dynamic state. In complex flow sheets, the calculation of an initial steady state is usually numerically much more challenging than the subsequent dynamic simulation. A helpful way around that problem is to use a suitable "pseudo steady state" instead which avoids harsh initial transients. One possibility to do so is to use steady state initialization only for the states with relatively fast eigenvalues. Setting the pressure gradient to zero, but supplying initial estimates for temperature and composition is one such suitable choice of a "pseudo steady state". The fast modes of the system (pressure and, if a dynamic momentum balance is used, mass flows) are initialized in steady state, while the much slower modes of temperature and composition are set by a non-steady state initial guess. The pressure gradient becomes:

$$\frac{dp}{dt} = \left.\frac{dp}{dT}\right|_{\mathbf{N}} \frac{dT}{dt} + \sum_{i=1}^{n} \left.\frac{dp}{dN_i}\right|_{T} \frac{dN_i}{dt} = 0. \qquad (7)$$

This equation together with given initial composition and temperature is much easier to solve than a full steady state, especially for large networks, but the initial transients due to errors in the initial guesses are orders of magnitude smaller than the ones obtained from non steady state pressures. The new initialization method has been implemented for all state models in the THERMOFLUID and THERMOFLUID/MF libraries and has improved the handling of model initialization considerably. Before implementation of the improved initialization, computation time for small problems was dominated by the time to simulate past the initial transients. With that obstacle removed, typical simulation times for small systems are an order of magnitude faster than before.
This initialization is the default setup when the THERMOFLUID/MF high-level models are used. The initial state is defined by given temperature and mass fractions and an initial pressure estimate. The initialization then solves for the mole amount states.

2.6 Debugging

In order to improve feedback and error messages for debugging, an identifier for each control volume is allocated during the initialization of the model. The identifier is passed to the wrapper functions calling the MultiFlash property routines. Using the unique control volume identifier, it is possible to connect error- and warning messages from the MultiFlash routines to the location in the flow sheet where the error occured, e. g., if a temperature rises above the range of validity of the property function. All error and warning messages from MultiFlash are written to the Dymola simulation log. Information about the version, the configuration, the number and composition of streams etc. is also included in the log.

2.7 ThermoFluid/MF high level models

Modeling of process engineering problems can not be cast into fixed, unchangeable model library components as for example multibody systems. Instead flexibility is needed to have basic building blocks taking care of the standard parts of any dynamic model. These basic models need to be easy to adapt to a specific problem. A large part of the physical property calculations is identical for all modeling problems. The THERMOFLUID/MF library provides such basic models and building blocks for control volume models based on MultiFlash properties. Extensions are simple to add by using elements of the THERMOFLUID or THERMOFLUID/MF libraries. Some examples of lumped and distributed models demonstrate how to build components and larger systems from the building blocks in the library. A mixture which is typical for fuel cell reformer systems is used to demonstrate how the minmimal physical property model is used and also how to add transport properties. Transport properties are not included in the THERMOFLUID library except for water, but MultiFlash includes several models for viscosity, thermal conductivity and surface tension for pure components and mixtures.

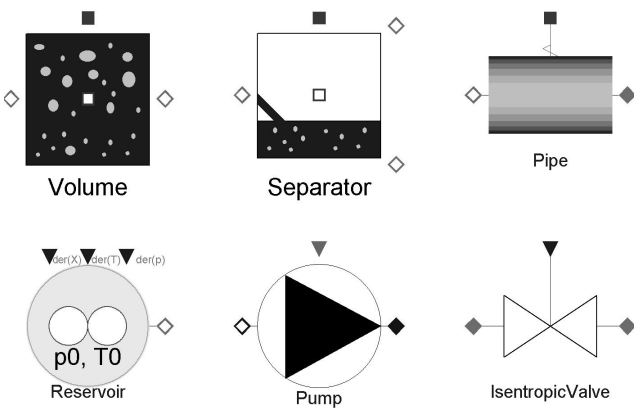

Figure 5: Example models from the library.

The number of high level models in the THERMOFLUID/MF library is fairly small, because most standard models can be used from the THERMOFLUID library. The only base models that are different are control volume models. For flash models new flash control volumes are introduced. They determine which phase is flowing in or out at a connector from the position of the flow connector and the liquid level in the volume. Tray models for destillation columns will be added later.

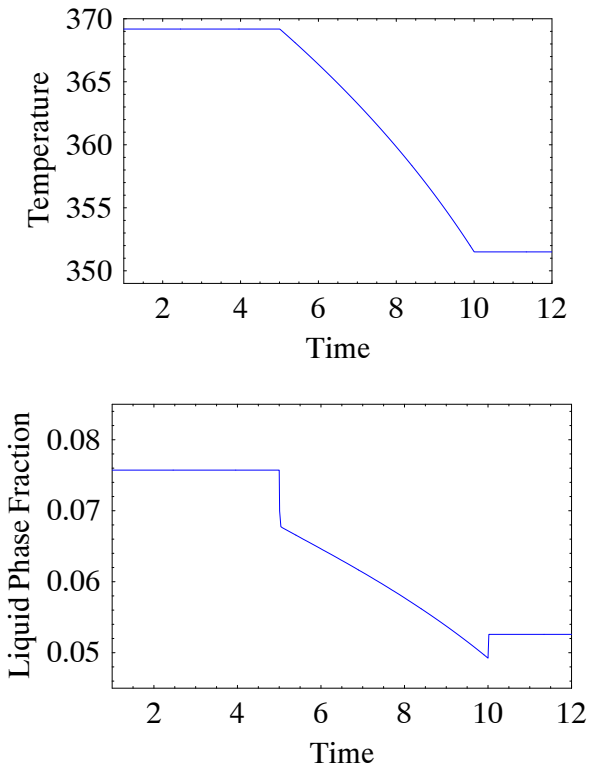

Figure 6: Change of temperature and liquid phase fraction in a water-ehtanol mix during a pressure transient.

The simulation result from the depressurization of a flash volume filled with a water-ethanol mixture in thermodynamic equilibrium of the two phases is shown in Figure 6. A ramp from 1 bar to 0.5 bars is imposed on the volume which has a feed flow of constant composition. The jump in the liquid phase fraction at the start and end of the transient is due to the changing in- and outflow phase fractions.
Using MultiFlash properties with the THERMOFLUID/MF library is very simple and requires only few steps of setup:

- Define the components, phases and models to be used in the MultiFlash user interface and save the result in a model setup file.
- Define a THERMOFLUID-compatible property model, following the examples in the THERMOFLUID/MF library.
- Use that property model in a suitble control volume model from the THERMOFLUID/MF library.

3 Crossing Functions for multi-component multi-phase mixtures

In Modelica, crossing functions are usually automatically generated from all equations that contain statements which indicate that a function $f(x)$ is discontinuous at a certain point x_0. For example, in the following equation:

```
phase = if h < hliq or h > hvap or p > pcrit then 1 else 2;
```

three crossing functions are introduced to monitor the states of the boolean conditions. This is necessary because numerical integration routines assume continuity of their right-hand side functions. This assumption is violated in most if-clauses. This can not be automated for external functions that are discontinuous at a point x_0. Thus crossing functions have to be provided by the user in order to make the simulator detect the discontinuity. These crossing functions have to be consistent with the actual discontinuities, otherwise they will not work. In the context of phase equilibrium calculations for multi-component fluid mixtures this means that a unique function of composition, pressure and temperature (N, p, T) must be returned from the phase equilibrium calculations which has a sign change at the point where a new phase is formed or one phase ceases to exist. At a phase boundary thermodynamic variables have discontinuous first derivatives or are discontinuous by itself, like the heat capacity at constant volume c_v. For a mixture with n components the crossing function is a function $\Re^{(n+1)} \mapsto \Re$. It calculates a measure for the distance to the phase boundary surface which is in $\Re^n$.

3.1 Deviation index

Collaboration with Infochem Ltd. [5] brought forth an implementation of such a function to increase efficiency and reliability of phase equilibrium calculations in dynamic simulations. It is available in the latest release of MultiFlash, version 3.1. This is the first time that a multi-phase property package has been equiped with this feature, which is indispensable for being able to reliably simulate the formation or disappearance of phases in a control volume with high

quality integrators with event detection and error control. Infochem calls the new function the *deviation index*. The calculation of the deviation index is numerically much more efficient than other possibilities to determine the number of phases at a given (N, p, T) during dynamic simulation. Geometrically, the deviation index can be interpreted as a normalized length of the normal vector from the current point in the space spanned by composition, pressure and temperature to the n-dimensional tangent hyperplane to the phase separation surface. The tangent plane is also known as *Gibbs' tangent plane*. It has been used for stability analysis in phase equilibrium calculations before, see [8], [9] and [1]. The new feature is to assign a value to the distance from the hyperplane which allows a solver using interpolation to exactly locate the point in time when the simulation trajectory in the N, p, T- space will pass through the hyperplane. At equilibrium, the Gibbs energy of the system is at a minimum. This condition may be expressed as the equality of fugacities for each component in all phases or equivalently ([1]) as

$$ln(K_{ij}) + ln(F_{ij}) - ln(F_{ir_i}) = 0 \quad i = 1..n_c; \;\; j = 1..n_p \tag{8}$$

where n_c is the number of components, n_p is the number of phases, K_{ij} is the K-value for component i in phase j, F_{ij} is the fugacity coefficent for component i in phase j and r_i is the the reference phase for component i. The K-values are defined as

$$K_{ij} = \frac{y_{ij}}{y_{ir_i}} \tag{9}$$

where y_{ij} is the mole fraction of component i in phase j. In the vicinity of the phase split surface, the left hand side of (8) gives the value of the desired crossing function, the deviation index.

Furthermore, the function needs to reliably calculate the properties used in equation (8) of a phase which is unstable at the current (N, p, T). Considering for simplicity single component mixtures and the calculation of thermodynamic properties for a phase which is unstable inside the 2-phase dome (i. e., superheated liquid or subcooled vapour), it becomes clear that the numerical computation is only possible to the limit of the so called *spinoidal lines*. For simple cubic EOS the spinoidal lines are defined by the connection lines of the maxima and minima of the theorectical isothermes inside the two-phase dome. An implementation thus has to guard against erroneous results far from the phase boundary.

4 Conclusions

The inclusion of reaction calculations and the interface to the physical property database MultiFlash into the THERMOFLUID library opens new possibilities of modeling process systems and combustion processes which up to now have been blocked by the large initial investment in modeling work to set up the physical property calculation.

The general reaction, diffusion and heat transfer object provides a clean and unified way of encapsulating submodels for heat and mass transfer. Base classes never need to be changed no matter how many connections to the control volume exist. Standard reactions can be stored in component libraries and used with any reactor model that has a compatible medium property model. Membrane diffusion uses the same mechanism to couple into the standard dynamical equations.

The THERMOFLUID/MF library provides two sets of models: low level models which are one-to-one wrappers to the MultiFlash physical property routines and high level base models for multi-component liquid-gas two phase models. Care has been taken to make the time consuming VLE-calculations as efficient as possible and at the same time numerically robust.

Crossing functions for multi-phase, multi-component mixtures have been implemented in collaboration with Infochem Ltd. They allow a numerically robust detection of the formation of new phases in a multi phase mixture.

References

[1] J. F. Counsell, R. A. S. Moorwood, and R. Szczepanski. Caclulating Multiphase Equilibria. In *Proceedings of the Conference on Vapour-Liquid Equilibria*, Aston, 1990.

[2] Jonas Eborn. *On Model Libraries for Thermohydraulic Applications*. PhD thesis, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, March 2001.

[3] Rüdiger Franke. Formulation of dynamic optimization problems using modelica. In Martin Otter, Hilding Elmqvist, and Peter Fritzson, editors, *Proceedings of the International Modelica Conference 2002*. Modelica Association and DLR, March 2002.

[4] http://www.dynasim.se.

[5] http://www.infochemuk.com.

[6] http://www.mathcore.com.

[7] Jakob Munch Jensen and Hubertus Tummescheit. Moving Boundary Models for Dynamic Simulations of Two-Phase Flows. In Martin Otter, Hilding Elmqvist, and Peter Fritzson, editors, *Proceedings of the International Modelica Conference 2002*. Modelica Association and DLR, March 2002.

[8] M. L. Michelsen. The isothermal flash problem. I. Stability analysis. *Fluid Phase Equilibria*, 9:1–19, 1982.

[9] M. L. Michelsen. The isothermal flash problem. II. Phase-split calculation. *Fluid Phase Equilibria*, 9:21–40, 1982.

[10] Torge Pfafferot and G. Schmitz. Numerische Simulation von CO2-Kühlprozessen mit Modelica. In *DKV-Tagungsbericht 2001*, volume IV 28. Jahrgang. DKV, Stuttgart, 2001.

[11] Hubertus Tummescheit, Jonas Eborn, and Falko Wagner. Development of a Modelica base library for modeling of thermo-hydraulic systems. In *Modelica 2000 Workshop Proceedings*, pages 41–51, Lund, October 2000. Modelica Association.

[12] Stephen R. Turns. *An Introduction to Combustion*. McGraw Hill International Editions, 1993.