Mathematical and Computer Modelling of Dynamical Systems 1387-3954/02/0000-000$16.00
2002, Vol. 00, No. 0, pp. 000–000 © Swets & Zeitlinger

Object-Oriented Design of Reusable Model Libraries of Hybrid Dynamic Systems – Part Two: A Case Study

A. URQUIA[1] AND S. DORMIDO[2]

ABSTRACT

A novel object-oriented methodology for the design of model libraries is exemplified by means of the design of JARA. JARA is a set of libraries of dynamic hybrid models of some fundamental physical-chemical principles. Its main application field is the modeling of physical-chemical processes in the context of automatic control. Only a reduced set of phenomena has been selected for modeling, focusing mainly on illustrating relevant aspects of the proposed design methodology. The five steps of the design methodology are followed and the *design rules* of JARA are obtained.

Keywords: Object-oriented modeling languages, first principles modeling, hybrid models, chemical engineering models, mathematical models.

1. INTRODUCTION

The design methodology presented in the first part of this paper is based on the principles of the object-oriented modeling of hybrid dynamic systems [1–3], fully exploiting the principles of abstraction and information encapsulation. It distinguishes between the role of the library designers and the role of the library users. A methodological key idea is the following: the identification and solution of the numerical difficulties associated to the library use should be carried out (as far as possible) by the library designers. Leaving this job in the hands of the library users is contrary to the abstraction principle.

Another methodological key concept is the *library design rules*. The design rules cover the numerical aspects of the design and the use of the libraries. They define: (1) the complete set of requirements that each library class has to fulfill; (2) how each library class can be used and what classes are interchangeable; and (3) how the libraries can be enlarged with new classes to be used in conjunction with the existing ones.

[1]Address correspondence to: A. Urquia, Department of Computer Science and Automatic Control, U.N.E.D., Senda del Rey 9, 28040 Madrid, Spain. E-mail: aurquia@dia.uned.es
[2]Department of Computer Science and Automatic Control, U.N.E.D., Madrid, Spain.

The design methodology takes into account all the phases of the library life cycle (i.e., design, programming, maintenance, modifications, extension, etc.) and it defines the role of the library designers and their relationship with the library users. The two pieces of information that the library designers have to provide to the library users are: (1) the hypothesis which each class of the library is based on; and (2) the library design rules.

The proposed design methodology is composed of the five following steps.

Step 1. *Phenomena and hypotheses*. The phenomena to be modeled and the general modeling hypothesis are defined. Frequently, the same phenomena are modeled in different ways using different sets of hypotheses. Once all the pairs (phenomenon to be modeled – modeling hypothesis) have been identified, they can be grouped into libraries.

Step 2. *Phenomena interaction: causal connectors*. Design of the library containing the causal connector classes. A causal connector class is defined by means of the following four pieces of information:

1. The variables composing the connector, classified into *across* and *through* [4].
2. The allowed computational causalities of the connector class.
3. The way of breaking the computational causality loops.
4. A graphic icon.

Modeling languages support: (point 1) the description of the connector variables; and (point 4) the graphical icon. The pieces of information numbered 2 and 3 are part of the causal connector class documentation. In order to facilitate the causal connectors use, the library designer should establish a correspondence between all these causal connector properties and the graphic attributes of its icon.

Step 3. *Connection rules of the causal connectors*. These consist of the definition of the nature and number of the allowed connections among causal connectors. The *design rules* of the libraries are stated during the second and third steps of the design: they are composed of the causal connector classes' definition and their connection rules.

Step 4. *Interfaces definition*. Interfaces are (only) composed of an arbitrary number of causal connectors. The interface classes are gathered in the *interface library.*

Step 5. *Internal description of the phenomena*. The phenomena formulation is strongly conditioned by the rules imposed by the interface causal connectors. The way of facing this conditioning depends on each particular case. However, some general techniques which can be adapted for solving the particular problems posed are discussed in this paper.

Next, the application of the proposed methodology to the JARA design is described [5]. JARA has been programmed using the modeling language Dymola [4] and it is simulated using the modeling environment Dymodraw-Dymola-Dymosim-Dymoview

[4]. The phenomena modeled in JARA allow the composition of many macroscopic and maximum gradient, "academic" models of the process units. For instance, the process units such as: continuous and batch stirred reactors, tubular reactors, heat exchangers, boilers, condensers, and flow conductions with and without delay (i.e., pipes, valves, flow junctions, etc). The interested reader is referred to [5] for the discussion and simulation of several of these JARA application cases.

2. FIRST STEP: PHENOMENA AND HYPOTHESES

The usual way of enunciating the mass, energy and momentum balances is by means of the definition of a control volume (abbreviation: CV), in whose interior the medium is considered a continuum with uniform properties [6, 7]. The CV exchanges mass and energy with its environment through certain control planes (abbreviation: CP). The transport equations describe this interaction. This system decomposition into control volumes and transport phenomena (abbreviation: TP) constitutes a powerful modeling strategy. All the interactions among CVs and all the interactions of a CV with itself can be considered as a TP. For instance, the chemical reactions taking place inside a CV (i.e., all the reaction product and reactive components are in the same phase) can be modeled as a TP: an output flow of the consumed components and an input flow of the generated components. Similarly, the chemical reactions with both liquid and gaseous components and the liquid-vapor phase changes can be modeled as TP between the liquid mixture CV and the gaseous mixture CV. As it will be discussed later, the system boundary conditions can be classified into two groups: those substituting CVs and those substituting TP. This system decomposition into CVs and TP suggests that:

1. The CV models should contain the equations describing the properties of the medium inside the CV as a function of the mass and energy transport through its CPs.
2. The TP models should contain the equations describing the mass, energy and momentum flows through the CPs as a function of the CPs' medium properties.
3. The interface variables of both kinds of models (i.e., CV and TP) should be composed of:
    - 3.1. The physical magnitudes for describing the state of the medium inside the CV, which are needed for calculating the mass, energy and momentum flow through the CPs.
    - 3.2. The physical magnitudes for describing the transport of mass, energy and momentum through the CPs, which are needed for calculating the properties variation of the medium inside the CV.

Next, some of the CVs and TP modeling hypothesis are discussed.

2.1. Modeling Hypotheses for Control Volumes (CVs)

The macroscopic or maximum gradient approximations are very common when modeling chemical process units for automatic control: the CVs and the CPs are considered macroscopic and fixed in the space [8]. The properties of the medium inside the CVs are considered time-dependent, but independent of the spatial coordinates (this is the stirred mixture approximation). There is only one exception to this rule: the pressure inside liquids.

The following three systems have been modeled:

1. *Liquid CV*: the CV contains an ideal, homogeneous liquid mixture composed of an arbitrary number of components.
2. *Gaseous CV*: the CV contains an ideal mixture of an arbitrary number of semi-perfect gases.
3. *Solid CV*: the CV contains a homogeneous solid.

The liquid and gaseous CVs are considered open systems (i.e., they can exchange mass and heat with their environment) and chemical reactions can take place inside them. The solid CV is considered a closed system (i.e., it only exchanges energy, not mass, with its environment). The only modeled characteristic in solids is the heat conduction (for modeling the walls of reactors, pipes, etc.).

One relevant property of a CV is its spatial volume. The liquid and gaseous CVs impose the following constraint on the volume of the fluid stored inside them: *the volume of the gaseous* (*liquid*) *mixture contained into a CV is equal* (*smaller or equal*) *to the CV volume*. The JARA models of the liquid CV and the gaseous CV contain these constraints (this aspect of the liquid CV model is discussed in Section 6.1). In general, the CV volume is considered a time-dependent property.

A very common way of selecting the liquid and gaseous CVs is so that they coincide partially with the interior wall of a tank, pipe, etc (see Fig. 1). The CV surfaces that do not coincide with the wall are the mass-flow CPs. Consider the modeling of the storage of fluids into recipients (tanks, pipe segments, etc.).

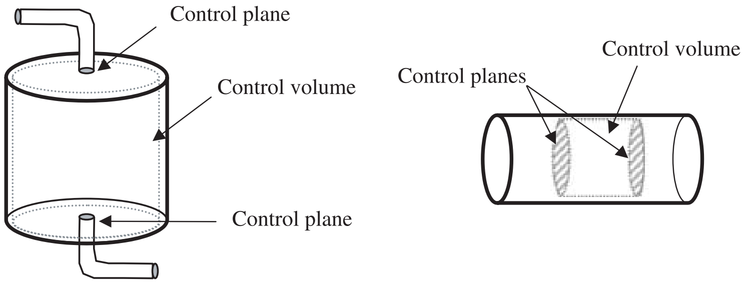


Fig. 1. Control volume and control planes selection.

The following three basic situations are identified and modeled in JARA:

Case 1 The recipient contains a *homogeneous liquid mixture*. The recipient space not occupied by the liquid is full of a gas whose properties are not modeled. Its pressure is the only relevant characteristic, because it is supposed to be equal to the pressure at the liquid surface (it is considered as a boundary condition).

Case 2 The recipient contains an *ideal mixture of semi perfect gases*.

Case 3 The recipient contains a *homogeneous liquid mixture* and an *ideal mixture of semi-perfect gases*. The gaseous mixture is placed over the liquid and the interface between them is clearly defined.

The CV selection is made as follows. The CV coincides with the internal wall of the recipient in Cases 1 and 2. In Case 3, the liquid CV coincides with the internal wall of the recipient and the gaseous CV coincides with the recipient volume not occupied by the liquid mixture (see Fig. 2).

In this context, solid recipients (tanks, pipes, reactors, etc.) have two main sets of interesting properties: (1) those related with their volume; and (2) those related with the transmission and storage of heat. The heat conduction (convection) is modeled as a TP established between the thermal CPs of two adjacent solid CVs (solid CV – liquid CV or solid CV – gaseous CV). The heat storage is modeled by means of the solid CVs energy balances. The volumes of the solid recipients are assumed constant, so the liquid CV volume is constant in Cases 1 and 3. The gaseous CV volume is constant in Case 2, but it is time dependent in Case 3 (i.e., gas volume = recipient volume – liquid volume).

Some comments about the CPs selection. Two kinds of control planes are distinguished in JARA: *mass-flow CPs* and *heat-flow CPs*. An arbitrary number of flows can flow through each CP. The hypothesis made about the properties of the medium inside the CV determine the nature and number of the CPs:

1. A *solid CV* contains only one heat-flow CP: (i) the solid properties are spatially homogeneous, so that all the heat-flow CPs are equivalent; and (ii) the solid CV is a closed system, so it has no mass-flow CPs.

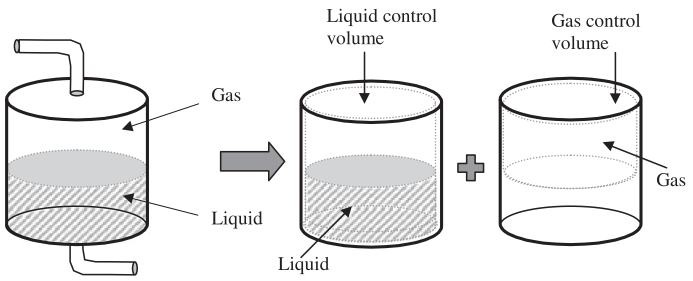


Fig. 2. CVs selection in Case 3.

2. A *gaseous CV* contains one heat-flow CP and one mass-flow CP: all the gaseous mixture properties are spatially homogeneous.
3. A *liquid CV* has two mass-flow CPs and one heat-flow CP: (i) as the liquid properties related with the heat-flow are spatially homogeneous, all the heat-flow CPs are equivalent; (ii) the liquid pressure depends on the position: the simplest and most general CP selection is placing one CP at the CV bottom (point of highest pressure) and the other at the CV top (point of lowest pressure). Any arbitrarily complex configuration can be modeled by decomposition into this kind of CVs [5].

2.2. Modeling of Transport Phenomena (TP)

All the flow models of JARA make the flow direction completely reversible, that is, the flow direction changes during the simulation according to their operation conditions. The JARA TP can be divided into two main groups: (1) *mass transport* due to the pressure or concentration gradient, the gravitational acceleration, chemical reactions, liquid-vapor phase changes, etc.; and (2) *heat transport* due to the temperature gradient. The interface variables are grouped into connectors according to these criteria, so that they allow describing the transport of mass and heat independently.

A hypothesis related to the stirred mixture approximation is to assume that the fluid going out from a CV has the same properties as the fluid contained in it. In JARA, this approximation is applied to the calculus of: (1) the temperature and the composition of the fluid leaving a CV by convection; and (2) the temperature of each mixture component leaving the CV by diffusion. The flow direction changes during the simulation as a function of the operating conditions. In consequence, the properties of the flow established between two CPs have to be calculated from the appropriate CP at any time.

Instead of this approximation, there are others for estimating the properties of the output fluid. For instance, it could be assumed that the property value of the fluid inside a CV, say T, is equal to the mean value between the property value of the input fluid and the property value of the output fluid, that is, $T = \frac{T_{in}+T_{out}}{2}$. This assumption leads to an undesired effect called "*see-saw*" [9]. In order to explain this effect, the previous expression is differentiated. It is obtained: $T = \frac{T_{in}+T_{out}}{2} \longrightarrow \frac{dT_{out}}{dt} = 2\frac{dT}{dt} - \frac{dT_{in}}{dt}$. In consequence, if the increase of the input fluid property value satisfies $\frac{dT_{in}}{dt} > 2\frac{dT}{dt}$, then the output fluid property value decreases: $\frac{dT_{out}}{dt} < 0$. Clearly, this has no sense. The use of CVs with smaller inertia to changes in T (frequently, using CVs of smaller volume) alleviates the problem, but at the cost of increasing the number of CVs of the model.

An important property associated to the TP is the *transport delay*. There are different ways of modeling delays in one-dimensional geometry systems. To illustrate two of them, consider the case of mass transport [9].

1. The first method consists in using a delay memory in which the internal energy value of the input fluid is stored. This value is assigned to the output fluid after the

delay time. It is assumed that the internal energy is transported at the fluid speed, while the pressure is transported almost instantaneously. For this reason, only the internal energy is delayed. The delay value depends on the time: it is calculated as the mass of the fluid stored within the element divided by the fluid-flow (mass per unit of time). The most important disadvantage of this delay representation is that it is assumes that the element does not exchange heat with its environment.

2. The second method (often called the energy balance method [7]) for modeling transport delays consists in dividing the flow path into multiple CVs. The mass balance and the energy balance are established in each CV. The mass transport between adjacent CVs is modeled as a function of the stored fluid properties. Therefore, the flow path is modeled as a succession of CVs connected by the TP describing the heat and mass transport between them. As the number of CVs increases, the solution gets closer to a transport delay (a demonstration can be found [9]). This is the method used in JARA.

Another relevant phenomenon to model is the *junction and bifurcation of several flows*. As previously discussed, one of the JARA modeling hypothesis is that the TP describe the mass, heat and momentum flow through the CPs. Consequently, the CV models must be connected to the TP models and vice versa. The CV models cannot be connected to one another and the TP models cannot be connected to one another either. Apparently, this means an important limitation for modeling the flow bifurcations. The solution adopted in JARA is modeling the flow junctions as CVs [9].

As it is demonstrated in [5], a microscopic (or mathematical) flow junction is a limit case of a CV, when the volume of the CV gets closer to zero. The smaller the volume of the CV, the smaller its memory that is, the faster the transitory of the CV properties is in response to changes in the input flow conditions. The model user determines the CV volume adequate to each experiment, balancing the required precision with the computational cost of introducing fast dynamics in the model. The error produced with this approximation can be estimated comparing how the output flow properties follow the input flow ones. However, it is important to note that the usefulness of this macroscopic junction model is not only to serve as an approximation of the microscopic junction. In addition, it allows modeling the flow properties dependence with respect to the spatial coordinates and the transport delay. To complete this, the flow path is split into several CVs, modeled as macroscopic junctions. TP classes are connected between adjacent CVs in order to describe the flow between the CVs.

2.3. Models Organization into Libraries

At this point of the design, the phenomena to model and their modeling hypotheses are defined. They can be grouped into libraries according to their application context. JARA is composed of seven libraries (see Fig. 3). The causal connectors are defined in

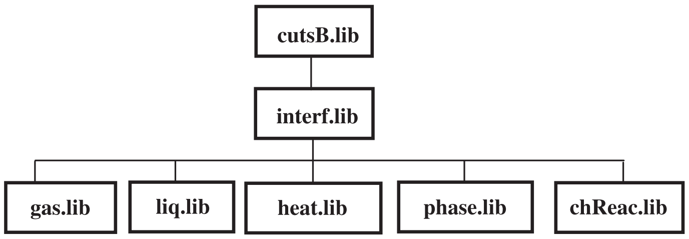


Fig. 3. JARA's libraries.

the *cutsB.lib* library and the interfaces in *interf.lib*. The *gas.lib* library contains some models of gaseous CVs, gaseous mixtures TP (e.g., gas-flow by convection and diffusion, valves, pumps, etc.) and boundary conditions (i.e., gas-flow and pressure sources). Similarly, the library *liq.lib* contains the equivalent models for liquid mixtures. The mixtures of liquids or gases are considered ideal and they can be composed of an arbitrary number of components. The models related with the heat transport are collected in the *heat.lib* library: models of solid CVs, thermal resistances and boundary conditions (heat and temperature sources). The *phase.lib* library contains models of vapor–liquid phase change: boiling and condensation. The classes modeling chemical reactions are in the *chReac.lib* library. All the details can be found in [5].

3. SECOND STEP: PHENOMENA INTERACTION – CAUSAL CONNECTORS

A set of physical magnitudes is grouped into the same causal connector if their connection corresponds to the same model application context. Four different model application contexts have been distinguished in JARA [5]: *mass-flow*, *heat-flow*, *information transmission* and *volume constraint*. The causal connectors defined in JARA have only one allowed computational causality. This is not imposed by the design methodology, but it is a result of the JARA modeling approximations.

Given that the interface causality of JARA models must be declared, a legitimate question is: what is the advantage of programming JARA in the Dymola language (or any other object-oriented modeling language) compared with using a general-purpose simulation language (i.e., ACSL Graphics Modeller, SIMULINK, etc.). Some important advantages are the following: (1) modeling languages support the object-oriented modeling methodology; (2) they allow the description of variable structure models and the modeling environments support an adequate treatment of the discrete-time events; and (3) modeling environments incorporate algorithms for solving non-linear systems of simultaneous equations and reducing the model index.

These capabilities are discussed in the first part of this paper and JARA models take advantage of all of them [5].

3.1. Capacitive and Resistive Causal Connectors

Attending to their computational causality, two types of JARA causal connectors are distinguished: *capacitive* and *resistive*.

Definition 1. A connector is *capacitive* if and only if it satisfies the following two conditions:

1. The across variables are computational outputs and they are only a function of state variables and/or time.
2. The through variables are computational inputs.

Definition 2. A connector is *resistive* if and only if it satisfies the following two conditions:

1. The across variables are computational inputs.
2. The through variables are computational outputs.

The capacitive and resistive connectors are causal connectors. The computational causalities of the capacitive and resistive connectors have been schematically represented in Figure 4. The arrow pointing toward the element represents a computational input and the arrow pointing outwards means computational output. The vertical line perpendicular to the arrow represents the breaking of the causality computational loop by means of integrators (the bond graphs notation is adopted [10, 11]).

3.2. JARA Causal Connectors

Two types of *mass-flow connectors* have been defined in JARA: one for liquid mixtures and the other for gaseous ones. A capacitive and a resistive connector class have been defined for each type. The physical magnitudes composing the JARA mass-flow connectors are:

- Across variables:
    1. The amount of matter for each of the mixture components within the CV. It is measured in moles for gases and in mass units for liquids. In some situations,

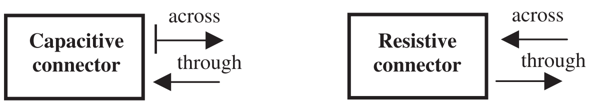


Fig. 4. Computational causality of the capacitive and resistive causal connectors.

this different selection for gases and liquids prevents specifying the molecular mass of the components.
2. Temperature at the CP.
3. Pressure at the CP.

- Through variables:

1. Matter-flow of each of the mixture components through the CP. It is measured in moles per time unit (mass per time unit) for gases (liquids).
2. Enthalpy-flow through the CP associated to the matter-flow.

The JARA's causal connector classes are described in Table 1. The following notation has been used. Suffix C: capacitive connectors; suffix R: resistive connectors; (I): computational inputs; and (O): computational outputs. To illustrate the use of the causal connectors classes, the column "Use examples" contains some JARA models whose interface contains each causal connector class.

Two types of *heat-flow connectors* have been defined in JARA, one for heat conduction, *heat flow – M*, and the other for heat convection, *heat flow – F*. A capacitive and a resistive connector class have been defined of each type. The physical magnitudes composing the JARA heat-flow connectors are (see Table 1):

- Across variables:

1.a. *Heat flow – M* connectors: amount of matter of each mixture component inside a CV. The mixture component masses are included in the *heat flow – M* connector by the following two reasons: (i) if a CV is empty (i.e., zero mass inside it), the heat-flow through its heat CP is zero; and (ii) the heat-flow between two bodies usually depends on their contact surface.
1.b. *Heat flow – F* connectors: matter-flow of each of the mixture components between two CPs. The *heat flow – F* connectors contain the matter-flow because the heat convective transmission coefficient usually depends on the matter-flow [7].
2. Temperature at the CP.

- Through variable:

1. Heat-flow through the CP.

The necessity of describing the information transmission arises when modeling automatic control loops. This is the purpose of the JARA's *information transmission connectors*. Two types of information transmission connectors are defined in JARA: emitter and receiver (see Table 1). Both types of causal connectors are only composed of across variables, whose number and meaning depends on the application. By definition, the emitter (receiver) connector is a capacitive (resistive) causal connector. This implies that the information emitter connector variables have to break the

Table 1. Causal connectors defined in JARA.

Causal connector	Across variables		Through variables		Use examples
Gas flow – R	Moles number (vector) Temperature Pressure	(I)	Molar-flow (vector) Energy-flow	(O)	Gas TP Gas-flow source
Gas flow – C	Moles number (vector) Temperature Pressure	(O)	Molar-flow (vector) Energy-flow	(I)	Gaseous CV Gas pressure source
Liquid flow – R	Mass (vector) Temperature Pressure	(I)	Mass-flow (vector) Energy-flow	(I)	Liquid TP Liquid-flow source
Liquid flow – C	Mass (vector) Temperature Pressure	(O)	Mass-flow (vector) Energy-flow	(I)	Liquid CV Liquid pressure source
Heat flow – M-R	Matter (vector) Temperature	(I)	Heat-flow	(O)	Heat-flow by diffusion Heat-flow source
Heat flow – F-R	Matter flow (vector) Temperature	(I)	Heat-flow	(O)	Heat-flow by convection Heat-flow source
Heat flow – M-C	Matter (vector) Temperature	(O)	Heat-flow	(I)	Liquid CV Gaseous CV Temperature source
Heat flow – F-C	Matter flow (vector) Temperature	(O)	Heat-flow	(I)	Liquid TP Gas TP Temperature source
Info. emitter – C	Signal (vector)	(O)			Automatic controller Signal generator Sensor
Info. receiver – R	Signal (vector)	(I)			Automatic controller Valve All source types
Volume constraint	Three variables		One variable		Liquid CV Gaseous CV Recipient

computational causality loops. This does not imply any practical limitation: it only forces to model the sensor dynamic "appropriately".

Consider, for instance, that the variable x_1 should be part of an information emitter connector and it is not a function of time and/or state variables. In order to model the emitter connector properly, the auxiliary variable x_2 is defined. The following two analogies can be established: (x_1 – physical magnitude to measure) and (x_2 – sensor signal representing the physical magnitude). A first-order sensor dynamic can be arbitrarily considered: $\varepsilon_S \cdot \dot{x}_2 = x_1 - x_2$, where ε_S is a numerical parameter. The class user has to select (in the experiment definition) the value of ε_S small enough to consider that x_2 follows "instantaneously" x_1 for simulation purposes. The desired goal has been achieved: the variable x_2 satisfies the emitter connector condition and it is "equal" to x_1. Of course, the value of ε_S should not be smaller than needed in order to avoid introducing large eigenvalues unnecessarily. An auxiliary variable can be defined to help the user in the selection of the ε_S value, for instance: $\frac{|x_1 - x_2|}{0.5 \cdot (|x_1 + x_2| + \varepsilon)}$. It measures the relative error of considering that x_1 and x_2 are equal. Special note should be taken so that ε is a parameter of small value whose purpose is avoiding the singularity $\frac{0}{0}$.

The fourth type of connector in JARA, the *volume constraint connector* (see Table 1), models the constraints that the recipients impose on the CVs volume:

1. The liquid CV volume is equal to the recipient volume in Cases 1 and 3 (see Section 2.1).
2. The gaseous CV volume is equal to the recipient volume in Case 2 and it is equal to the recipient volume minus the liquid volume in Case 3.

The condition appropriate to each situation is imposed in JARA connecting the volume constraint connector of the CV class to the volume constraint connector of the recipient class [5].

3.3. Graphic Attributes of the JARA Causal Connectors

Causal connectors icons allow identifying easily the type of causal connector (see [5] for further information). The icon color represents the type of flow: liquid, gas or information. Capacitive connectors icons are filled and resistive connectors icons are bordered. The "pointed" shape denotes the existence of through variables (gas, liquid and heat flow connectors). The way of orienting the connector in the interface (inward or outward direction) depicts the sign criteria of the connector through variables.

4. THIRD STEP: CONNECTION RULES OF THE CAUSAL CONNECTORS

The *connection rules* define the nature and number of the allowed connections between causal connectors. Different connection rules are applied depending on whether the causal connectors belong to same-hierarchy classes interfaces or to different-hierarchy classes interfaces. For instance (see Fig. 5), S1 and S2 are different-hierarchy classes (S2 is a molecular class. S1 is a submodel of S2), while S2 and S3 are same-hierarchy classes. The connection of S1 to S2 is internal to S2, while the connection of S2 to S3 is external to both classes.

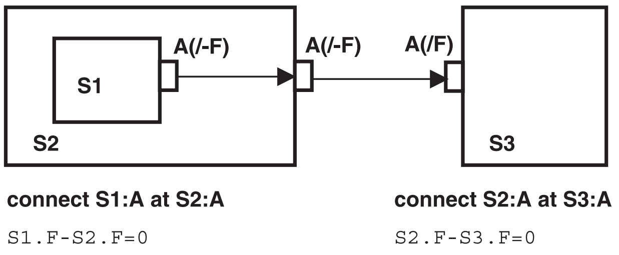


Fig. 5. Same-hierarchy (S2–S3) and different-hierarchy (S1–S2) class connections.

Table 2. Connection rules for same-hierarchy class connectors.

	Liq-R	Liq-C	Gas-R	Gas-C	Heat M-R	Heat M-C	Heat F-R	Heat F-C	Receiver	Emitter
Liquid-R	NO									
Liquid-C	YES	NO								
Gas-R	NO	NO	NO							
Gas-C	NO	NO	YES	NO						
Heat-M-R	NO	NO	NO	NO	NO					
Heat-M-C	NO	NO	NO	NO	YES	NO				
Heat-F-R	NO	NO	NO	NO	NO	NO	NO			
Heat-F-C	NO	NO	NO	NO	NO	NO	YES	NO		
Receiver	NO	NO	NO	NO	NO	NO	NO	NO	NO	
Emitter	NO	NO	NO	NO	NO	NO	NO	NO	YES	NO

Table 3. Connection rules for different-hierarchy class connectors. Vertical: molecular model connector. Horizontal: submodel connector.

	Liq-R	Liq-C	Gas-R	Gas-C	Heat M-R	Heat M-C	Heat F-R	Heat F-C	Receiver	Emitter
Liquid-R	YES	NO	NO	NO	NO	NO	NO	NO	NO	NO
Liquid-C	YES	YES	NO	NO	NO	NO	NO	NO	NO	NO
Gas-R	NO	NO	YES	NO	NO	NO	NO	NO	NO	NO
Gas-C	NO	NO	YES	YES	NO	NO	NO	NO	NO	NO
Heat-M-R	NO	NO	NO	NO	YES	NO	NO	NO	NO	NO
Heat-M-C	NO	NO	NO	NO	YES	YES	NO	NO	NO	NO
Heat-F-R	NO	NO	NO	NO	NO	NO	YES	NO	NO	NO
Heat-F-C	NO	NO	NO	NO	NO	NO	YES	YES	NO	NO
Receiver	NO	NO	NO	NO	NO	NO	NO	NO	YES	NO
Emitter	NO	NO	NO	NO	NO	NO	NO	NO	YES	YES

The information about which causal connectors of the same hierarchy can be connected to each other is summarized in Table 2. In the case where the causal connectors are at different hierarchy classes, the summary is in Table 3. The notation is: YES: connection allowed; NO: connection not allowed. The reader interested in the connection rules of the volume constraint connectors is referred to [5]. The rules

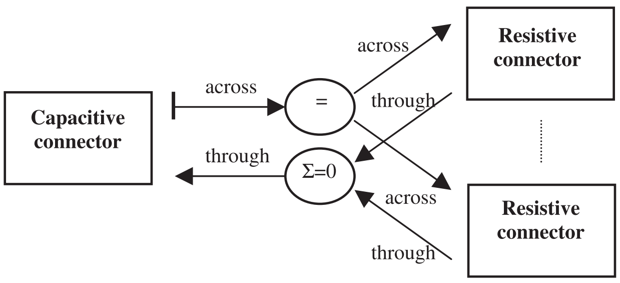


Fig. 6. Connection of same-hierarchy causal connectors.

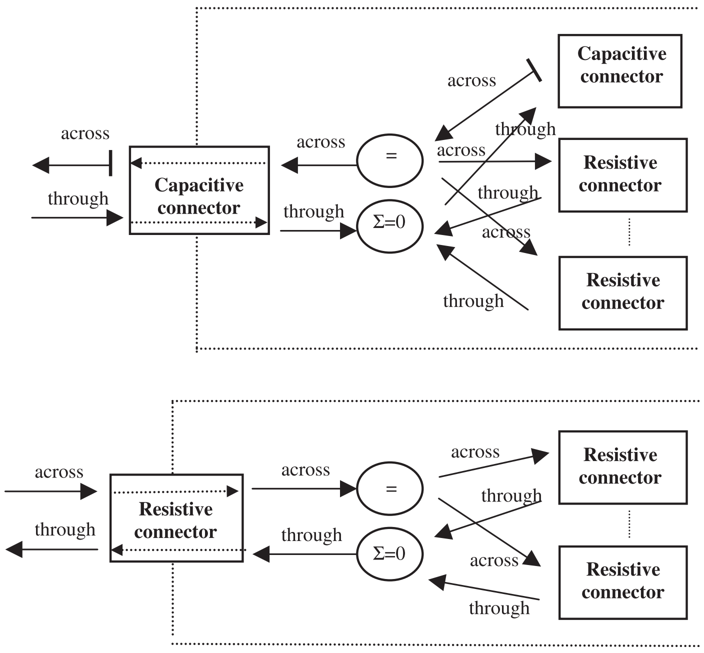


Fig. 7. Connection of different-hierarchies causal connectors.

about the number of connections are:

- *Connection between causal connectors at the same hierarchy* (see Fig. 6):
    1. A capacitive connector can be left unconnected or it can be connected to an arbitrary number of resistive connectors.
    2. A resistive connector must be connected to one, and only to one, capacitive connector.
- *Connection between causal connectors at different hierarchies* (see Fig. 7):
    1. A capacitive connector of the molecular class must be connected to one, and only to one, submodel capacitive connector. It also can be connected to an arbitrary number (zero is allowed) of submodel resistive connectors.
    2. A resistive connector of the molecular class must be connected to at least one submodel resistive connector. It cannot be connected to any submodel capacitive connector.

The previous connection rules can be proposed in terms of the connector graphic attributes (see [5]). For instance, the rules for causal connectors of the same hierarchy:

1. Two icons are connectable if they have the same shape and the same color.
2. A filled icon either can be left unconnected or be connected to an arbitrary number of bordered icons.
3. A bordered icon must be connected to one, and only to one, filled icon.

5. FOURTH STEP: INTERFACES DEFINITION

The proposed design methodology dictates that a valid interface can only be composed of an arbitrary number of causal connectors. This rule establishes the way of extending the libraries with new classes. In addition, the design rules indicate: (1) how to connect, and under which circumstances, any valid interface (by means of the connection rules); and (2) the complete set of conditions that the model internal description must satisfy (by means of the causal connectors definition). The design rules guarantee that all the classes with the same interface are exchangeable in any application context.

All the JARA class interfaces are grouped in the *interf.lib* library. Most of JARA interfaces can be classified into four main groups (all details can be found in [5]):

1. Interfaces used for modeling CVs and across sources (i.e., boundary conditions for the temperature, pressure, etc.), mainly composed of heat-flow and matter-flow capacitive connectors. In some cases, these interfaces also contain volume constraint connectors in order to model the constraint on the fluid volume imposed by the recipient containing it.

2. Interfaces used for modeling TP and through sources (i.e., boundary conditions for the heat-flow, mass-flow, etc.), mainly composed of heat-flow and matter-flow resistive connectors.
3. Interfaces used for modeling pure information emitter or receivers (controllers, signal processors, etc.), mainly composed of information-transmission connectors.
4. Interfaces used for modeling recipients, composed of volume-constraint connectors.

In most of the JARA application examples discussed in [5], new interface classes are defined inheriting the interfaces modeling CVs or TP and extending them with information-transmission connectors.

6. FIFTH STEP: INTERNAL DESCRIPTION OF THE PHENOMENA

The internal description of the phenomena must be formulated so that it satisfies all the conditions imposed by its interface. The proposed design methodology imposes that: (1) the model internal description must be well-posed for all the allowed computational causalities of its interface; and (2) the causality computational loops established when connecting the classes must be broken as dictated by the connected model interfaces.

An additional difficulty arises from the following restriction imposed by the design methodology: a model has to be formulated so that the computational causality of its interface does not change during the simulation. Otherwise, the model connection establishes systems of simultaneous equations [12, 13]. To this respect, the modeling of variable structure systems will be difficult to manage. A variable structure system is defined by: (1) the equations describing the system in each of its states (in general, a different set of equations per state); and (2) the conditions of commutation among their states.

As a result of this design methodology restriction, the interface computational causality of a variable structure system has to be common to all the system states. This limitation strongly conditions the modeling of this system type. However, it does not constitute any practical limitation if appropriate modeling techniques are applied. Next, the modeling of two variable structure systems is discussed: the liquid CV and the boiling process. These fairly simple examples try to illustrate the application of a technique widely used in JARA for formulating the phenomena internal descriptions.

This technique is based on the constraint relaxation by means of the introduction of dummy dynamics. Assuming that the physical phenomena are instantaneous leads to the establishment of simultaneous equations systems [14] and high index problems [15]. Instead, the physical phenomena can be modeled arbitrarily, for instance, assuming first-order dynamics. These dummy dynamics involve additional modeling approximations that the class user has to "tune" for each experiment. The dummy dynamic eigenvalues chosen by the user must be large enough for the dynamics to be considered “instantaneous”, but also small enough to prevent the system from being strongly stiff. When the class designer introduces one of these dummy dynamics, he should also define a variable measuring the error made by the constraint relaxation, that is, the substitution of the instantaneous interaction by the first-order dynamic. These error variables assist the user in the eigenvalues selection.

6.1. Modeling of the Liquid CV

The model interface is defined before the model internal description. The interface of the liquid CV model contains:

1. Two liquid-flow capacitive connectors, *liquid flow – C*, one placed at the CV top and the other at the CV bottom.
2. One heat-flow capacitive connector, *heat flow – M-C*.
3. One volume-constraint connector.

The liquid CV is modeled as a variable structure system with two states: *full* and *not-full*. The state transition conditions are:

1. Old state: *not-full*. State transition condition: the volume of the liquid stored within the CV becomes greater or equal to the CV volume. Completion of this condition gives: New state: *full*.
2. Old state: *full*. State transition condition: the liquid volume becomes less than the CV volume. Completion of this condition gives: New state: *not-full*.

The liquid CV model contains, in both states, the same equation relating the pressure at the top mass-flow CP, the pressure at the bottom mass-flow CP and the liquid weight. However, the way of modeling the pressure at the top mass-flow CP depends on the system state:

1. *Not-full* state. The pressure at the top mass-flow CP is either a constant or a function of time.
2. *Full* state. The pressure at the top mass-flow CP is equal to the pressure exerted on the liquid by the CV, so that the net volumetric-flow of liquid going in and out of the CV is zero. This constant-volume condition could be modeled imposing the equality between the liquid volume and the CV volume: $\sum_i \frac{m_i}{\rho_i} = V_{CV}$. The coefficients ρ_i represent the densities (constant in JARA) of the pure components of the liquid mixture and V_{CV} is the CV volume (constant for liquid CVs).

The analysis to determine whether the previous formulation satisfies the mass-flow causal connectors conditions is the following:

1. The *not-full* state formulation satisfies the conditions. Each component mass of the liquid mixture and its temperature are state variables: they are differentiated in the

mass and energy balances, respectively. The pressure at the bottom mass-flow CP is a function exclusively of state variables and time. The mass-flow and the enthalpy-flow through the mass-flow CPs are computational inputs.

2. The *full* state formulation does not satisfy the conditions. The constraint on the liquid volume reduces the number of freedom degrees (i.e., the system index [15] is higher than one): the mass of one of the mixture components is an algebraic variable and it must be evaluated from the volume constraint equation. The index reduction implies differentiating symbolically [16] the volume constraint equation and others and adding them to the model [17]. This differentiated equation allows the calculation of one of the mass derivatives. In consequence, one of the mass balances of the components is used to evaluate its mass-flow. The pressure at the top CP is not present in the CV model: it must be evaluated from one of the classes modeling the liquid transport.

This model formulation presents one additional difficulty: the model index changes during the simulation run. Most modeling environments do not support the simulation of this model type (a trivial example is shown below). The model needs to be manipulated in order to guarantee that: (1) it has the same index in all the states; and (2) the same set of equations needs to be differentiated in all its states to reduce the index. An algorithm is proposed in [5] to automatically perform these manipulations given certain conditions. However, in this case the *full*-state model index does not only depend on the CV model equations. In addition, it also depends on the internal description of the classes modeling the liquid transport [5]. As a consequence, according to the abstraction principle, the index reduction manipulations should not be carried out during the library design phase.

Example Model with an index which varies during the simulation run. The modeling language is Modelica.

```
model variableIndex
  Real x,y;
equation
  1 = if time > 0.5 then x else y;
  der(x) = y;
end variableIndex;
```

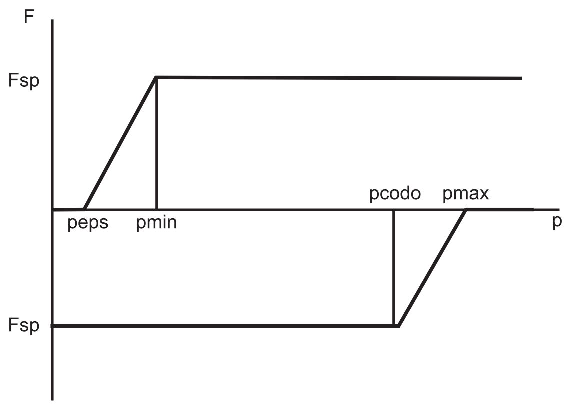
The *constraint relaxation technique* is applied in order to obtain a valid formulation of the *full* state. The volume constraint $\sum_i \frac{m_i}{\rho_i} = V_{CV}$ is equivalent to assuming that the liquid pressure changes are instantaneous. This statement is substituted by an arbitrary model (dummy dynamic) for the pressure at the top mass-flow CP: $p_{topCP} = \frac{1}{\kappa} \cdot \left(\frac{1}{V_{CV}} \cdot \sum_i \frac{m_i}{\rho_i} - 1 \right)$. The value of the coefficient κ has to be chosen by the class user. The coefficient κ has a meaning analogous to the compressibility coefficient only dependent on the pressure (independent of the temperature):

$\kappa = \frac{1}{V} \cdot \frac{dV}{dp} \rightarrow \Delta p = \frac{1}{\kappa \cdot V} \cdot \Delta V$. An indicator of the relative error of this approximation is $\left(\frac{1}{V_{CV}} \cdot \sum_i \frac{m_i}{\rho_i} - 1 \right)$.

This model of the *full*-state CV assumes that the mass-flow through a CP is a function of the CP pressure. This hypothesis conditions the modeling of the liquid TP. Consider, for instance, the following liquid source model. Its interface contains:

1. One *liquid flow – R* connector that represents the liquid-flow CP.
2. One *information receiver* connector. Set-point values for the liquid-flow properties: direction, total flow, composition and temperature.

The functional relationship between the flow set-point and the ‘‘real’’ flow must take into account the pressure at the *liquid flow – R* connector. For this reason, modeling the source behavior should be defined by its characteristic curve (an example is shown in Fig. 8). The characteristic curve relates the pressure and the mass-flow at the *liquid flow – R* connector with the mass-flow set-point at the *information receiver* connector (F^{SP} in Fig. 8). Next, an application example of the liquid CV and the liquid source models is discussed.



Flow going out from the source, $F^{SP} < 0$	Flow going into the source, $F^{SP} > 0$
$F = \begin{cases} F^{SP} & \text{if } p < p_{codo} \\ \frac{F^{SP}}{p_{\max} - p_{codo}}\left(p_{\max} - p\right) & \text{if } p_{codo} \le p < p_{\max} \\ 0 & \text{if } p \ge p_{\max} \end{cases}$	$F = \begin{cases} F^{SP} & \text{if } p > p_{\min} \\ \frac{F^{SP}}{p_{\min} - p_{eps}}\left(p - p_{eps}\right) & \text{if } p_{eps} < p \le p_{\min} \\ 0 & \text{if } p \le p_{eps} \end{cases}$

Fig. 8. Characteristic curve of the liquid source modeled in JARA.

Example Consider a tank dedicated to the storage of a liquid mixture of benzene and kerosene. The mixture goes into the tank through a liquid-flow source connected to the bottom of the tank. Three pipes are connected to the tank at different heights: the first one is placed at one-third of its height, the second one at two-thirds and the third one at the tank top. These three pipes drain the liquid to an atmospheric-pressure sink. The simulation purpose is the study of the filling and emptying dynamics of the tank. The following modeling approximations are made [5]:

1. The benzene – kerosene mixture properties (density, viscosity, etc.) are calculated from the individual component properties assuming that the mixture is ideal. These calculations are made in the liquid CV and the pipe classes. A matrix formulation of the equations has been carried out in JARA, so that the mixture-component number is a parameter of the models.
2. The tank is modeled by splitting it into three CVs. The CVs correspond to the tank portions in between the pipes. In other words, the first CV corresponds to the tank section located between the tank bottom and the first pipe. The second CV corresponds to the section located between the first and the second pipe. The third CV corresponds to the space between the second and the third pipe. The tank section class is composed of the connection of a liquid CV class to a recipient class. This connection is established between their volume constraint connectors.
3. The liquid-flow between two consecutive-numbered CVs has to guarantee that, at any time, the pressure at the top mass-flow CP of the first CV is equal to the pressure at the bottom mass-flow CP of the second CV. The interface of the class, modeling this transport phenomenon, is composed of two *liquid flow – R* connectors and one *heat flow – F-C* connector. In order to fulfill the resistive connector rules, the *constraint relaxation technique* is applied. The pressure equality constraint is substituted by a first-order dynamic: the mass-flow is proportional to the difference between the pressure at the two connectors. The class user has to assign a value to the proportionality constant, K_P. An estimation of the relative error is: the absolute value of the difference of the pressures as the numerator divided by the denominator, which is the pressures average. The composition and temperature of the liquid-flow are calculated from the *stirred mixture approximation* (see Section 2.2).
4. The interface of the pipe model is composed of two *liquid flow – R* connectors and one *heat flow – F-C* connector. The linear momentum balance is formulated as discussed in Table 1 in the first part of this paper. The pipe properties (geometrical dimensions, friction factor parameters, etc.) are known model parameters.
5. The atmospheric-pressure liquid sink is modeled as a pressure source. The model interface contains two causal connectors: a *liquid flow – C* connector and an *information receiver* connector. The information receiver connector has one

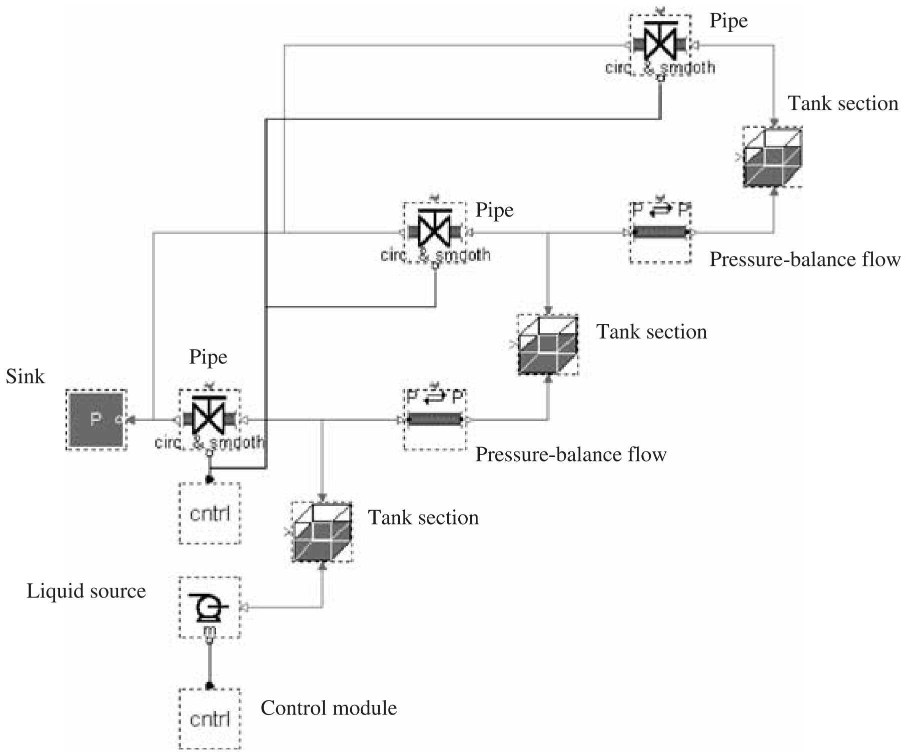


Fig. 9. Graphic representation of the system components. The graphic model editor is Dymodraw.

variable: the pressure set-point. The pressure at the *liquid flow – C* is equal to this same set-point. Its mass and temperature do not matter: the liquid-flow always goes into the sink.

The model's graphic representation is shown in Figure 9. The model graphic editor used is Dymodraw [4]. The following experiment is simulated using Dymola/ Dymosim [4]: the liquid source introduces a 75% kerosene–25% benzene mixture during 1500 s. Then the source changes the flow direction and it extracts liquid from the tank until it is empty. When the liquid level within the tank reaches a pipe, the liquid flows through the pipe. The liquid mixture volume in each of the three tank sections is shown in Figure 10. The linear momentum of the liquid flowing through each of the three pipes is shown in Figure 11. The value of the numeric compressibility coefficient is $\kappa = 10^{-6}$ in this experiment. It has been chosen so that the liquid volume error occurring when the tank is full is as large as the required precision

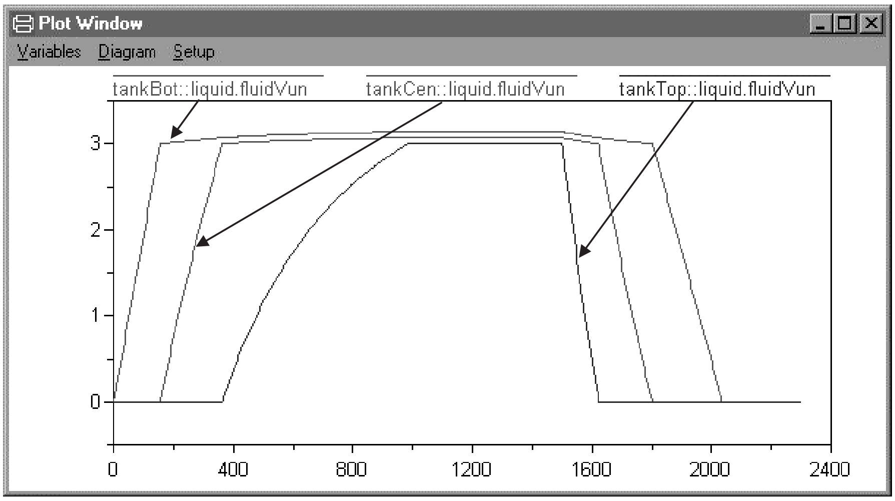


Fig. 10. Liquid mixture volume in each tank section.

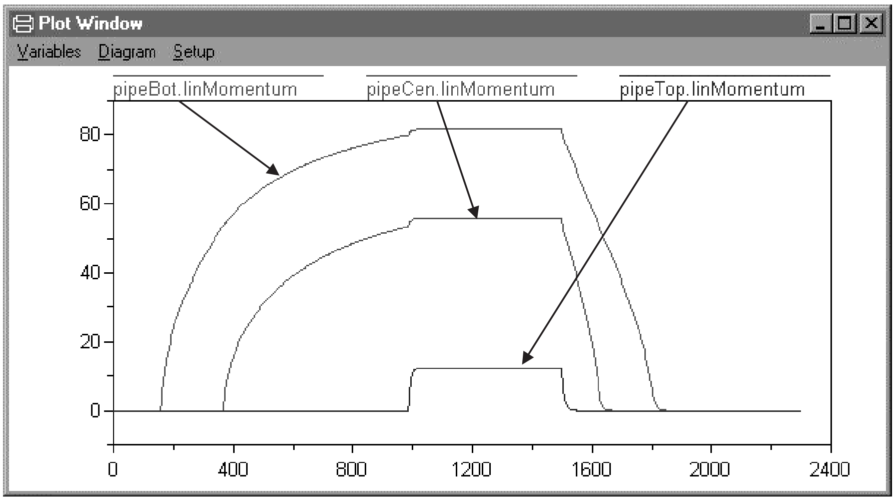


Fig. 11. Linear momentum of the liquid flowing through the pipes.

allows: in this case 2%. The value of the proportionality constant of the flow between CVs is $K_P = 10$. The relative error is shown in Figure 12. The simulation results in Figures 10, 11 and 12 are plotted using Dymoview [4].

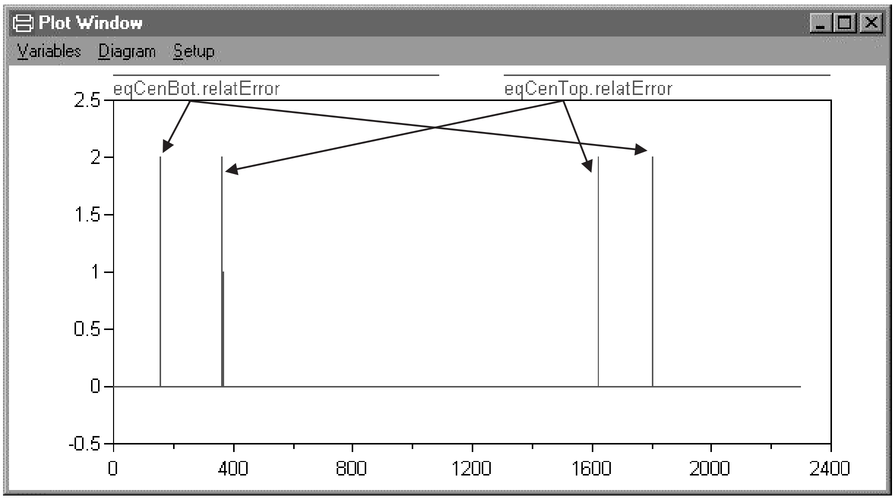


Fig. 12. Relative error occurring when considering that the pressure difference is equal to zero.

6.2. Multicomponent Boiling Modeling

The boiling process is modeled in JARA. Again, the model interface is defined before carrying out the model internal description. The interface of the boiling process model contains two mass-flow resistive connectors: one *gas flow – R* and one *liquid flow – R*. The model connection is carried out connecting:

1. (Boiling model: *gas flow – R*) to (Gaseous CV model: *gas flow – C*)
2. (Boiling model: *liquid flow – R*) to (Liquid CV model: *liquid flow – C*)

The heat transfer to the liquid CV can be modeled connecting the *heat flow – M-C* connector of the liquid CV to the *heat flow – M-R* connector of the heat transport model.

The boiling process is modeled as a variable structure system with two states: *boiling* and *not-boiling*. The transition conditions are the following:

1. Old state: *not-boiling*. State transition condition: the liquid temperature reaches the boiling point (it depends on the vapor pressure and the liquid composition). Completion of this condition gives – New state: *boiling*.
2. Old state: *boiling*. State transition condition: the phase-change mass-flow between the liquid CV and the gaseous CV is not greater than zero. The flow-sign is positive when it goes from the liquid CV to the gaseous CV. Completion of this condition gives – New state: *not-boiling*.

The system model is:

1. *Not-boiling* state. The boiling mass-flow is zero.
2. *Boiling* state. The liquid temperature is equal to the boiling temperature. In consequence, all the additional heat transferred to the liquid is used in carrying out the phase change. This condition allows calculating the phase-change mass transfer. The relationship between the liquid mixture composition and the boiling-produced vapor composition is known. The additional expression needed for calculating the phase-change mass-flow of each liquid mixture component is the energy balance of the liquid CV model.

The analysis to determine whether the so-formulated model satisfies the conditions imposed by its interface is the following:

1. The *not-boiling* state model satisfies the conditions imposed by its interface. It contains expressions to evaluate the mass-flow and the enthalpy-flow: both are equal to zero.
2. The *boiling* state model does not satisfy the interface conditions. It contains a constraint between the across variables of their resistive connectors: the liquid temperature is equal to the boiling temperature and this is a function of the gas pressure and the mass of the liquid components. The index model is greater than one [5]: the liquid temperature is differentiated in the liquid-CV model and it is constrained in the boiling process model. As a consequence, the model index changes during the simulation run.

Again, the model can be conveniently reformulated applying the *constraint relaxation technique*. Setting liquid temperature equal to the boiling temperature is equivalent to establishing that the phase-change mass-flow "instantaneously" reaches the appropriate value. Instead of setting explicitly the constraint on the liquid temperature, the dynamic of the boiling mass-flow can be modeled so that it makes the system evolve "fast enough" to the temperature constraint fulfillment. Arbitrarily, the flow is defined in JARA as proportional to the difference between the liquid temperature and the boiling one, $\sum_i^{Liq.\,Components} F_i = \kappa_b \cdot (T_{liquid} - T_{boiling})$. This condition and the expressions for the phase-change flow composition permit calculating the phase-change mass-flow of all the liquid mixture components. The user has to assign a value to the parameter κ_b adequate to the experiment. The relative error made can be estimated as: $(T_{liquid} - T_{boiling})/T_{boiling}$. Next, an application example of this model is discussed.

Example Consider an industrial boiler in a chemical plant. The vapor is used for generating energy. The liquid input is placed at the bottom of the boiler and the vapor output valve is placed at the top. The water contained in the boiler is continually heated. Usually the boiler operates at a nominal pressure value at the valve output. The simulation purpose is to study the system evolution when this pressure drops abruptly. The following modeling approximations are made:

1. A recipient class models the property "spatial volume" of the boiler.
2. Two CVs are considered:
   2.a. A liquid CV containing the liquid water stored in the boiler. The spatial volume of this CV is equal to the boiler inner volume.
   2.b. A gaseous CV containing the vapor stored in the boiler. The spatial volume of the CV is equal to the difference between the inner volume of the boiler and the liquid water volume. The vapor volume is calculated from the connection among the volume constraint connector of the liquid CV, the gaseous CV and the recipient CV.
3. As discussed previously, the phase-change mass-flow is a transport phenomenon connecting the liquid CV and the gaseous CV. The boiling temperature is a known implicit function of the vapor pressure (i.e., both variables intervene non-linearly in the equation).
4. The heat-flow into the boiler is modeled as a heat-flow source. Its interface contains two connectors: one *heat flow – M-R* connector and an *information receiver* connector.
5. The pump introducing the water into the boiler is modeled as a liquid-flow source. Its interface contains two connectors: one *liquid flow – R* connector and one *information receiver* connector.
6. The set-point signals of the heat-flow source and the liquid-mass source are generated in respective control modules. Their interfaces contain one *information emitter* connector.
7. The vapor valve is modeled. The flow through the valve is a non-linear function of the boiler vapor pressure and the pressure at the valve output.
8. The pressure at the valve output is modeled as a pressure source.

The experimental conditions of the simulation are the following:

1. Initially the valve is closed. It opens after 10 s and it remains opened during the rest of the simulation.
2. The liquid-flow and temperature are constant.
3. The heat-flow into the boiler is constant.
4. The pressure at the valve output is constant during the first 5000 s. Then, it drops abruptly and it remains constant during the rest of the simulation.

The graphic representation of the model is shown in Figure 13. The simulation is executed using Dymola/Dymosim. The boiling temperature (*boil.tempBoiling* variable) and the temperatures of the liquid (*boil.tempL* variable) and vapor

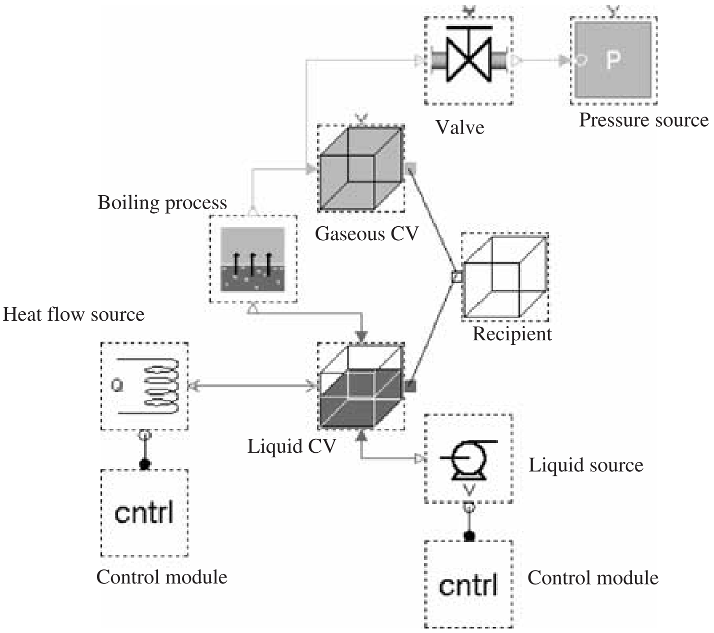


Fig. 13. Graphic representation of the system components. The graphic model editor is Dymodraw.

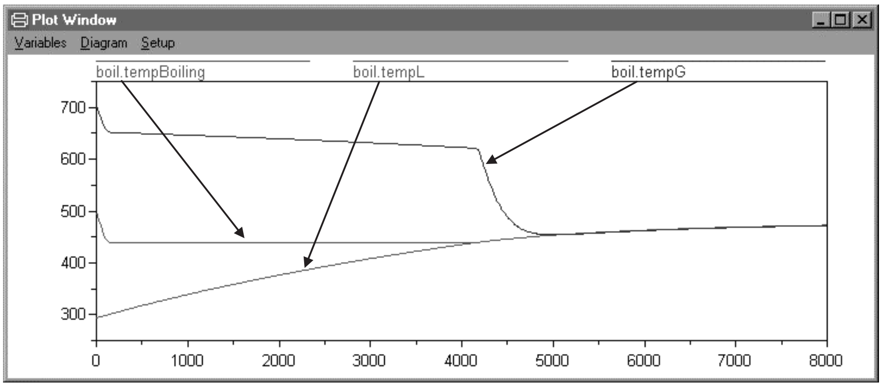


Fig. 14. Boiling temperature and temperatures of the liquid and the vapor contained in the boiler.

(*boil.tempG* variable) contained in the boiler are shown in Figure 14. It demonstrates how the phase-change flow makes the system evolve towards the temperature constraint fulfillment.

7. CONCLUSIONS

There is a trade-off between the problem-solving effort at the libraries design phase and the numerical difficulties that the library user has to face. The fewer the restrictions imposed at the design phase, the more complex and more extensive is the analysis of the complete-model numerical problems. The assumption that the library user has to solve these problems is contrary to the abstraction principle. The proposed design methodology requires the library designer to anticipate and solve the numerical problems arising from the models interaction.

The *library design rules* concept is the cornerstone of the proposed methodology. The design rules completely define the context of the models use and the libraries extension procedure. To this respect, the proposed design methodology takes into consideration the complete life cycle of the libraries. In addition, the role of the library designer is completely defined. The modeling hypothesis and the design rules are the only two pieces of information the library designer has to provide the library users with.

ACKNOWLEDGEMENTS

The authors wish to thank the referees of this paper for their constructive comments.

REFERENCES

1. Andersson, M.: *Object-oriented Modeling and Simulation of Hybrid Systems*. Ph.D. Thesis, ISRN LUTFD2/TFRT-1043-SE. Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, 1994.
2. Barton, P.I.: *The Modelling and Simulation of Combined Discrete/Continuous Processes.* Ph.D. Thesis. Department of Chemical Engineering, Imperial College of Science, Technology and Medicine, London, May 1992.
3. Nilsson, B.: *Object-Oriented Modeling of Chemical Processes*. Thesis. Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, 1993.
4. Elmqvist, H. et al.: *Dymola. Dynamic Modeling Laboratory. User's Manual. Version 4.1b.* Dynasim AB, Lund, Sweden, 2001. http://www.Dynasim.se
5. Urquia, A.: *Modelado orientado a objetos y simulación de sistemas híbridos en el ámbito del control de procesos químicos*. Ph.D. Thesis. Departamento de Informática y Automática, Facultad de Ciencias, UNED, Spain, July 2000.
6. Bird, R.B., Stewart, W.E. and Lightfoot, E.N.: *Transport Phenomena.* John Wiley & Sons, Inc., New York, 1975.
7. Incropera, F.P. and DeWitt, D.P.: *Fundamentals of Heat and Mass Transfer.* John Wiley & Sons, New York, 1996.
8. Himmelblau, D.M. and Bischoff, K.B.: *Process Analysis and Simulation*. John Wiley & Sons, New York, 1992.
9. Electric Power Research Institute (EPRI): *Modular Modeling System. Theory Manual. MMS-02 Release*, Palo Alto, CA, 1984.

10. Karnopp, D.C., Margolis, D.L. and Rosenberg, R.C.: *System Dynamics: A Unified Approach*. Second Edition. John Wiley & Sons. 1990.
11. Thoma, J.U.: *Simulation by Bondgraphs*. Springer-Verlag, Berlin, 1990.
12. Cellier, F., Otter, M. and Elmqvist, H.: *Bond Graph Modeling of Variable Structure Systems*, 1995.
13. Elmqvist, H.: Object-Oriented Modeling and Automatic Formula Manipulation in Dymola. In: *Proc. SIMS'93*, Scandinavian Simulation Society, Kongsberg, Norway, 1993.
14. Elmqvist, H. and Otter, M.: Methods for Tearing Systems of Equations in Object-Oriented Modeling. In: *Proc. ESM'94 European Simulation Multiconference*, Barcelona, Spain, 1994.
15. Brenan, K.E., Campbell, S.L. and Petzold, L.R.: *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. SIAM, Philadelphia, PA, 1996.
16. Pantelides, C.C.: The Consistent Initialization of Differential-Algebraic Systems. *SIAM J. Sci. Stat. Comput*. 9 (2) (1988).
17. Mattsson, S.E. and Söderlind, G.: A New Technique for Solving High-Index Differential Equations Using Dummy Derivatives. In: *Proc. IEEE Symposium on Computer-Aided Control System Design*, California, USA, 1992.