MOBATEC BV | VEESTRAAT 10 | 6134 VJ SITTARD | T 046–4009997 | WWW.MOBATEC.NL
ABN AMRO: 44 59 42 762 | KvK: 14084360 | BTW: NL 8142.56.041.B01

Mobatec Modeller

Concepts and Modelling Methodology

Contents

1 Introduction

Mobatec Modeller

"Rapid development of consistent dynamic process models"

Mobatec Modeller *is designed to effectively assist a model designer in building and maintaining consistent models. Using* ***Mobatec Modeller*** *significantly reduces the modelling time and overall effort.*

Mobatec Modeller *is a software instrument following a systematic modelling method (developed by prof. dr. dipl-ingH.A. Preisig and dr. ir. M.R. Westerweele) for the design of first principles (i.e. physical insight) based dynamic process models for physical, chemical and biological processes.*

Models take a central position in all process engineering tasks as they replace the process for the analysis. They represent an abstraction of the process, though not a complete reproduction. Models make it possible to study the behaviour of a process within the domain of common characteristics of the model and the modelled process without affecting the original process. The mapping of the process into a model does not only depend on the chosen theory, but also on the conditions under which the process is being viewed. The mapped characteristics vary thus not only with the applied theory but also with the conditions.

The construction of process models (including testing, initialisation, tuning, etc) is, in general, seen as a difficult and very time consuming task and is preferably handed over to "modelling experts". This does not have to be the case if a clear, stepwise method is adhered to.

Establishing a model, as it is used in a particular application, involves a number of operations, which can be broken down into a number of principle steps. This documentation will shortly introduce a systematic modelling methodology, which is based on the hierarchical decomposition of processes into thermodynamic systems. After that, the implementation of this methodology in the computer tool **Mobatec Modeller** will be discussed.

For a thorough discussion on the "new" concepts that are introduced in this document (such as Physical Topology, Species Topology and Equation Topology) and other important concepts (such as fundamental time scale assumptions in process models) that play a crucial role, when modelling a physical/chemical/biological process, we refer to the doctoral thesis of dr. ir. M. R. Westerweele. This thesis - "Five Steps for Building Consistent Dynamic Process Models" - can be downloaded from our website http://www.mobatec.nl.

1.1 The Modelling Process

A chemical engineer is often asked to describe the dynamic and/or static behaviour of a physical-chemical-biological (PCB) process (or a set of these processes, which constitute a plant), because information about this behaviour is needed for operations like analysis, control, design, simulation, optimisation or process operation. In order to analyse the behaviour of such a process, the engineer often needs a mathematical representation of the physical and chemical and/or biological phenomena taking place in it.

The representation of a PCB process in form of a mathematical model is the key to many chemical engineering problems. Modelling a chemical process requires the use of all the basic principles of chemical engineering science, such as thermodynamics, kinetics, transport phenomena, etc. It should therefore be approached with care and thoughtfulnes.

A (mathematical) model of a process is usually a system of mathematical equations, whose solutions reflect certain quantitative aspects (dynamic or static behaviour) of the process to be modelled. The development of such a mathematical process model is initiated by mapping a process into a mathematical object. The main objective of a mathematical model is to describe some behavioural aspects of the process under investigation.

The modelling activity should not be considered separately but as an integrated part of a problem solving activity. Heinz A. Preisig analysed and decomposed the overall task of problem solving into the following set of subtasks:

- **(Primary) Modelling**. The first step in the process of obtaining a process model is the mapping of the ”real-world” into a mathematical object, called the primary model. In doing so, one may take different views and accordingly apply different theories, which naturally will result in different models. Within this first step, assumptions are made about the principle nature of the process (such as time scales of hydraulics and reactions, fundamental states, etc.).

- **Model manipulation**. The model can be simplified by applying mathematical manipulations, such as:
    - Model reduction
    - Linearisation
    - Transformation to alternative representations of the model
    - re-arrangement of the mathematical problem equations

- **Problem specification**. Certain variables are instantiated (i.e. defined as known), such that they are available later during solution time and such that the number of equations equals the number of unknown variables.

- **Analysis of the mathematical model**. The analysis of the mathematical problem is done in connection with the specification of the model. On the simple level, a degree of freedom analysis may be done, which for large scale systems is by no means trivial. On the higher level, one may look into things as index of the differential algebraic system.

- **Solution of the mathematical model**. General purpose mathematical packages, such as differential algebraic equation solvers, large scale simulators, linear algebra packages, etc., are used to solve particular problems.

- **Analysis of the results**. The analysis of the results must focus on a verification of the results by comparing them with known pieces of information. This may be experimental data or just experience.

The outcome of the performance of each task must fulfil a set of specifications and requirements, otherwise the design is iterated by looping back to any previous task. This implies that modelling is a recursive and iterative process (and that includes not only the modelling, but everything that is associated with the use of the model). Rarely does one in the first attempt obtain a proper model for the problem under investigation. Usually, an adequate model is constructed progressively through a loop comprising a series of tasks of model development and model validation.

More often than not, the time spent on collecting the information necessary to properly define an adequate model is much greater than the time spent by a simulator program in finding a solution. Most publications and textbooks present the model equations without a description of how the model equations have been developed. Hence, to learn dynamic model development, novice modellers must study examples in textbooks, the work of more skilled modellers, or use trial and error.

During the last decades there tends to be an increasing demand for models of higher complexity, which makes the model construction even more time consuming and error-prone. Moreover, there are many different ways to model a process (mostly depending on the application for which the model is to be used): different time scales, different levels of detail, different assumptions, different interpretations of (different parts of) the process, etc. Thus a vast number of different models can be generated for the same process. All this calls for a systematisation of the modelling process, comprising of an appropriate, well-structured modelling methodology for the efficient development of adequate, sound and consistent process models. Modelling tools building on such a systematic approach support teamwork, re-use of models, provide complete and consistent documentation and, not at least, improve process understanding and provide a foundation for the education of process technology.

This document presents the concepts and design of **Mobatec Modeller**, a computer-aided modelling tool built on a structured modelling methodology, which aims to effectively assist in the development of process models and helps and directs a modeller through the different steps of this methodology. The objective of this tool is to provide a systematic model design method that meets all the mentioned requirements and turns the art of modelling into the science of model design.

The focus of **Mobatec Modeller** is primarily on modelling and not on problem solving (although both activities are supported in the tool). Most of the currently available modelling languages and simulation packages focus on model manipulation, specification, analysis and/or solution and more or less leave out the modelling part. In general it is assumed that the mathematical model of the process under investigation is known or easy to assemble. The development of process models, however, is slow, error prone and consequently a costly operation in terms of time and money.

Modelling is an acquired skill, and the average user finds it a difficult. A modeller may inadvertently incorporate modelling errors during the mathematical formulation of a physical phenomenon. Formulation errors, algebraic manipulation errors, writing and typographical errors are very common when a model is being implemented in a computing environment. Thus any procedure which would allow to do some of the needed modelling operations automatically would eliminate a lot of simple, low-level (and hard to detect) errors.

Mobatec Modeller is a computer-aided modelling tool which is designed to assist a model designer to map a process into a mathematical model, using a systematic modelling methodology. The main task solved by **Mobatec Modeller** is the construction and manipulation of the structure and definition of process models. The output of **Mobatec Modeller** is a first-principles based (i.e. based on physical insight) mathematical model, which is easily transformed to serve as an input to existing modelling languages and/or simulation packages, such as Mobatec LauTrane, Process Studio e-Modeler, gProms, Aspen Custom Modeler, Modelica, Matlab, etc. For certain solvers (LauTrane and e-Modeler) a build in Run Time Simulation Environment is available, such that the build dynamic process models can be tuned and tested.

1.2 Application Dependency of Models

This work is concerned with the mathematical modelling of macroscopic physical and/or chemical processes as they appear in general in chemical or biological plants. A mathematical model of a process usually consists of a set of equations, which describe the dynamic and/or static behaviour of this process.

There are many ways to generate these equations and there are many different ways to describe the same process, which will usually result in different models. The approach a modeller takes when constructing a model for a process depends on:

- *the application for which the model is to be used*. Different models are used for different purposes. For example, a model which is used for the control of a process shall be different from a model which is used for the design or analysis of that same process;

- *the amount of accuracy that has to be employed*. This is of course partially depending on the application of the model and on the time-scale in which the process has to be modelled. In general, a model which needs to describe a process on a small time-scale demands more details and accuracy then the model of the same process which describes the process over a larger time-scale;

- *the view and knowledge of the modeller on the process*. Different people have different backgrounds and different knowledge and will therefore often approach the same problem in different ways, which can eventually lead to different models of the same process.

1.3 Structured Modeling Methodology

The modelling of physcial-chemical-biological processes is one of the most important activities in process engineering, since it constitutes the starting point of most other engineering operations, such as simulation, design or control. Modelling is a creative activity, which requires the use of all the basic principles of chemical engineering science, such as thermodynamics, kinetics, transport phenomena, etc. It should therefore be approached with care and thoughtfulness.

A (mathematical) model of a process is usually a system of mathematical equations, whose solutions reflect certain quantitatieve aspects (dynamic or static behaviour) of the process to be modelled.

A modelling approach (or modelling methodology) can be defined as an algorithmic procedure intended to lead from specific knowledge of physical and topological nature of a process to a mathematical model of that process. The modelling methodology implemented in **Mobatec Modeller** is based on the hierarchical decomposition of processes (in which material and energy exchange are playing a predominant role during normal operation) into networks of elementary systems and physical connections. Elementary systems are regarded as thermodynamic simple systems and represent (lumped) capacities able to store extensive quantities (such as component mass, energy and momentum). The connections have no capacity and represent the transfer of extensive quantities between these systems. The construction of a process model with this methodology consists of the following steps:

1) Break the process down into elementary systems that exchange extensive quantities through physical connections. The resulting network represents the ***physical topology***.

   The process of breaking the plant down to basic systems and connections determines largely the level of detail included in the model. It is consequently also one of the main factors for determining the accuracy of the description the model provides.

2) Describe the distribution of all involved chemical and/or biological species as well as all reactions in the various parts of the process. This represents the ***species topology***.

   This species topology is superimposed on the physical topology and defines which species and what reactions are present in each part of the physical topology.

3) For each elementary system and each fundamental extensive quantity (component mass and energy) that characterises the system write the corresponding balance equation.

   Mobatec Modeller automatically generates all the needed balance equations for component mass and enthalpy of each system, since these balances can trivially formed from the model designers definition of the physical and species topology of the process. The user cannot edit the generated balance equations!

4) Add algebraic equations to the model definition:

- Choose the transfer laws and kinetic laws that express the flow and production rates of the balance equations.
- Express the fundamental extensive variables that characterise each system as a function of intensive variables characterizing the same system.
- Look for dependencies between the intensive and geometric variables that have been introduced and write these dependencies out as equations of state.

The dynamic balance equations (step 3) and the algebraic equations, which are placed on top of the physical topology and species topology, respresent the ***equation topology***.

A detailed discussion on the problems (and their solutions) that can arise when introducing the algebraic equations is given in thesis of Dr. Ir. M.R. Westerweele, which can be downloaded from our website http://www.mobatec.nl.

5) Add the (dynamic) behaviour of the information processing units, such as transmitters, adjusters and controllers.

These steps for building a model do not have to be done strictly in this sequence - at least not for the overall model. It is left to the model designer when the details are being specified in each part of the model.

This five step procedure of building dynamic process models always results in a set of differential algebraic equations (DAE) with an index of one. The model can be used for solving certain problems related to the process or it can be further modified by applying mathematical manipulations, such as linearisation or model reduction.

1.4 Implementation of Mobatec Modeller

Solving process engineering without the help of computer-based tools is for almost any problem an unthinkable proposition. Process simulation, process design, controller design, controller testing, data acquisition and model identification, paramete fitting, valve and pump selection, column sizing are just a few examples taken from a very large catalogue of chemical plant related operations that are almost exclusively done with computer-based tools. Considering the fact that multiple solutions to individual problems are available from different sources such as software houses or supplier companies, the catalogue of tools is rather large. A process engineer, who is more and more involved in the integrated design of processing plants, is thus faced with a multitude of different software tools, which he uses more or less frequently depending on his allocation. Since each of these packages has been developed in isolation, each has a different interface and applies to different problems though the tools may be used to solve problems associated with the same plant.

Often the software is very complex and needs a specialist to define and run problems. Data transfer between tools is not standardised and requires special arrangements or add-ons. All features which do not enhance team work and productivity.

The key to improving effciency is an increased use of common process information and effective exchange of data, models and information between the applications.

Mobatec Modeller is a computer tool that is based on the ideas and concepts which were briefly described in the previous paragraph. It is a computer-aided modelling tool to interactively define and modify process models. It aims to effectively assist in the development of these process models and facilitate hierarchical modelling of process plants through a user friendly interface.

With **Mobatec Modeller** process models are constructed from primitive building blocks, being simple thermodynamic systems and connections. It does not, in distinction to existing flow sheeting packages, build on unit models. **Mobatec Modeller** generates symbolic models in the form of differential-algebraic equations consisting of component mass and energy balances, augmented with transfer laws, physical and geometrical property relations and kinetic laws. In this document the implementation of the modelling method in the computer-aided modelling tool the **Mobatec Modeller** is discussed.

The next chapter starts with describing the construction and manipulation of physical topologies for processes with the aid of **Mobatec Modeller**. A special graphical user interface has been developed to handle physical topologies of arbitrary complexity. With the modelling tool, the physical topology of a process can be built using two main operations, namely refining an existing system (the top-down approach) or grouping systems (the bottom-up approach).

Chapter 3 discusses how one can construct and manipulate the species topology of a physical topology, using **Mobatec Modeller**. The definition of the species topology is initialised by assigning sets of species and reactions to some elementary systems. To aid in this definition, species and reaction databases are used. The user may also specify the directionality (i.e. uni or bi-directional) and permeability (i.e. selective transfer of species) of individual mass transfer connections. The distribution of the species over all systems is automated and uses the facts that assigned species can propagate through permeable mass connections and species may generate “new” species (via reactions), which in turn may propagate and initiate further reactions.

Finally, the implementation and handling of variables and equations (equation topology), which constitute the generated models, is discussed. With the information on the physical and species topology **Mobatec Modeller** can automatically generate balances of fundamental extensive quantities (component mass and enthalpy) of every elementary system. In order to fully describe the behaviour of the process, the flow rates and production rates involved in the balance equations need to be specified. So, in addition to the balance equations other relationships are needed to express thermodynamic equilibria, reaction rates, transport rates for heat, mass, momentum, and so on. A model designer can select a particular relationship from a set of alternatives and connect the selected relationship to a balance equation or to another defined relationship. In this way, each term in a balance equation can be expanded by defining it as a function of some quantities, which in turn may be expanded again and again. The resulting set of equations (the output of the modelling tool) forms a mathematical representation for the behaviour of the process in a specific form, which could serve as an input for problem solving tools, such as Mobatec LauTrane Solver (Mobatec), Aspen Custom Modeler (Aspen), gProms (Process Systems Enterprise), Matlab (Mathworks), Process Studio's e-Modeler (Protomation), Modelica (Dynasim AB) or any other DAE solver.

2 Construction of the Physical Topology

In order to get a mathematical description of a process, a modeller usually has to break the process down into smaller parts for convenience reasons. A process is thus assumed to consist of a set of subprocesses. These subprocesses may in turn be divided into smaller processes, and so on, until the process consists of a number of subprocesses, which each are small enough to be handled individually. In the modelling method that we employ this means that a process is divided into a network of interconnected volume elements. Each volume element of such a network consists of a single phase that is uniformly distributed and hence displays uniform properties over its volume. The network of volume elements describes, so-to-speak, the physical structure of the process and shall be referred to as the *physical topology* of that process.

The physical topology contains, once established, the maximum of information about the dynamic phenomena captured in the model. Any modification on the topology changes the dynamic information contents.

The construction and manipulation of the physical topology in an easy and fast manner, without compromising on the consistency of the process models, is one of the primary objectives of the computer-aided modelling tool **Mobatec Modeller**. The tool therefore allows for any structural change in any order. In this section the basic operations one can perform on a physical topology are discussed.

2.1 Handling Complexity

Elementary systems and connections are the primitive building blocks for constructing a mathematical model for a process. A (correctly defined) connection is always defined between two elementary systems. Thus, a network consisting of elementary systems and connections has a flat structure from a topological point of view. To aid in the handling of large and complex processes, the physical topology is organized in a strictly hierarchical multi-way tree. This means that the systems can be hierarchically grouped in composite systems, such that groups of components can be addressed. So, an additional tree structure is introduced, which is laid over the “flat” physical topology.

The visualization of larger processes can still get rather complicated. The overall model of a large process could easily outgrow the screen and one could easily lose grasp of the whole process, especially if one would only show the flat topology. This implies that we have to find a representation of the tree that only shows one specific part of the tree with detail. The rest of the model should be shown with very little detail.

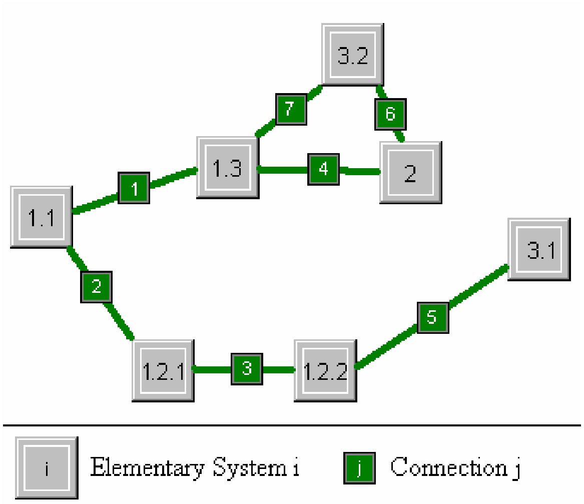

Figure 2.1: Example of a "flat" physical topology

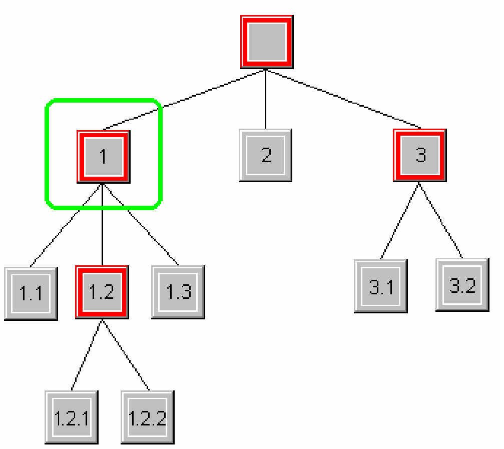

Figure 2.2: Tree structure representation of a process, without showing the Connections

A generic approach to this problem is to associate the view with the nodes of the tree. In this approach we limit the graphical display of a process to two successive hierarchical layers. One (composite or primitive) system is chosen as the "displayed" system, here called the focus system. This splits the overall model into two parts; the **focus system** and its environment. The focus system shows its subsystems (if it has any) in a frame. The environment systems of the focus system are displayed outside this frame. The following example illustrates what this means in terms of graphical representation:

***Example 2.1**: Visualization of a Process*

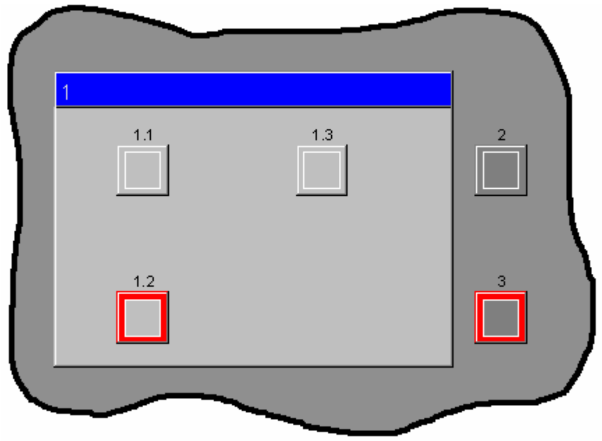

Figure 2.3: *Graphical representation of the tree with system 1 as the focus system. The connections between the systems are not shown.*

Consider a process that can be hierarchically decomposed in a tree of systems as represented in figure 2.2. The complete process, which is modelled as a thermodynamic universe, is represented by the root node. The root node is composed of three subsystems, "1", "2" and "3". These three systems together also represent the complete process. So, if we chose system "1" as the focus system, we can graphically represent the whole thermodynamic universe c.q. as the complete process by showing only the three subsystems. Figure 2.3 shows that graphical representation. The systems "1.1", "1.2" and "1.3" build up the focus system "1" and are therefore located inside the window frame which constitutes the focus system. In order to represent the complete process, the systems "2" and "3" have to be shown as environment systems of system "1". These systems are drawn outside the focus system window, because they are not part of the focus system. In this way the whole process is graphically represented by showing one system (the focus system) in detail, and the rest of the process (the environment systems of the focus system) with little detail.

□

The example shows that in this hierarchical representation of systems, information about the structure of the lower levels is hidden.

2.2 Unique System Identifiers

It is possible to give each system (either composite or primitive) a name, such that a model designer can keep an overview. To help the model designer with the handling of complex processes, a unique label c.q. identifier for each system is introduced. An example of these unique identifiers is already shown in the figures 2.2 and 2.3. The definition of a hierarchical tree of identifiers facilitates a notation for the representation of an arbitrary hierarchical system. The described tree structure does not impose any limitations on the depth of the tree nor on the number of branches from each node. Every node is uniquely labelled by an identifier.

Each identifier consists of a sequence of numbers, which are the branch numbers that must be chosen to arrive in the node being identified, starting at the root node of the tree. So, a node identifier uniquely describes the location of the node in the tree. In **Mobatec Modeller**, the root node of a process is always indicated with "Top Level", because this node represents the "thermodynamic universe". The first subsystem of the root is always identified by "1", the second by "2", etc. The first system of the third subsystem of the root will be labelled "3.1". The first subsystem of the latter will have "3.1.1" as identifier, etc., etc. The node identifiers are thus algebraically constructed by concatenating the consecutively chosen branch numbers. The "deeper down" a node is located in the tree, the "larger" its identifier will be (or the more numbers its identifier will comprise). Whenever the (tree structure of the) physical topology of a process is altered, **Mobatec Modeller** will update all identifiers of the systems that were influenced. In this way the identifiers of the systems (nodes) won't loose their meaning and the systems identifier still describes the location of the system in the tree.

Note: If you are not interested in showing the unique system identifiers, these can be hidden by checking the "Hide System Identifiers" checkbox on the [Preferences][Display] tab.

2.3 Basic Tree Operations

A physical topology is organized as a strictly hierarchical tree. This tree can be built using two main operations, namely by either refining an existing system (the top-down approach) or by grouping systems (the bottom-up approach). The operations that are used for the refining and grouping, and some other operations that are necessary for the manipulation of the physical topology will be discussed with the aid of simple examples. The examples will illustrate the influence of the operations on the tree structure and on the graphical representation of the tree.

o Adding an Elementary System

When constructing a physical topology for a process, one of the most important operations is of course the adding of new systems. A new elementary system is always inserted as an elementary system with a user defined name. The new system will be inserted as a subsystem of the focus system. As a rule a new system is always added to the right of the existing subsystems (if any) of the focus system. If the focus system does not have any subsystems yet, it is still a simple or primitive system. The insertion of a new system will turn the focus system into a composite system with one subsystem.

Several types of systems can be inserted into the physical topology of a process.

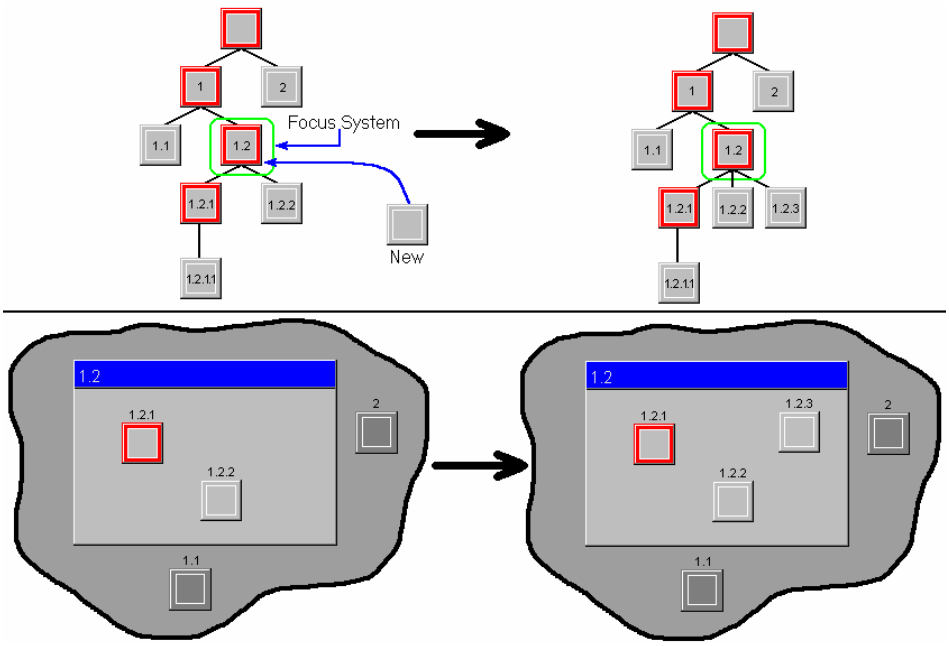

Figure 2.4: *Adding a new system*

- ### Selecting Systems

For a number of operations, such as deleting, moving, grouping, dragging and copying, it is necessary to have the facility of selecting systems. When grouping or copying, only subsystems of the focus system (i.e. internal systems) can be selected for reasons of keeping the (tree) representation consistent.

- ### Removing Selected Systems

Any system, composite or elementary, can be removed from a tree. After the selected systems have been removed, the system identifiers will be automatically updated. Care should of course be taken when one removes one (or more) composite systems, because all the subsystems of that composite system will also be removed and the information of this part of the tree will be lost.

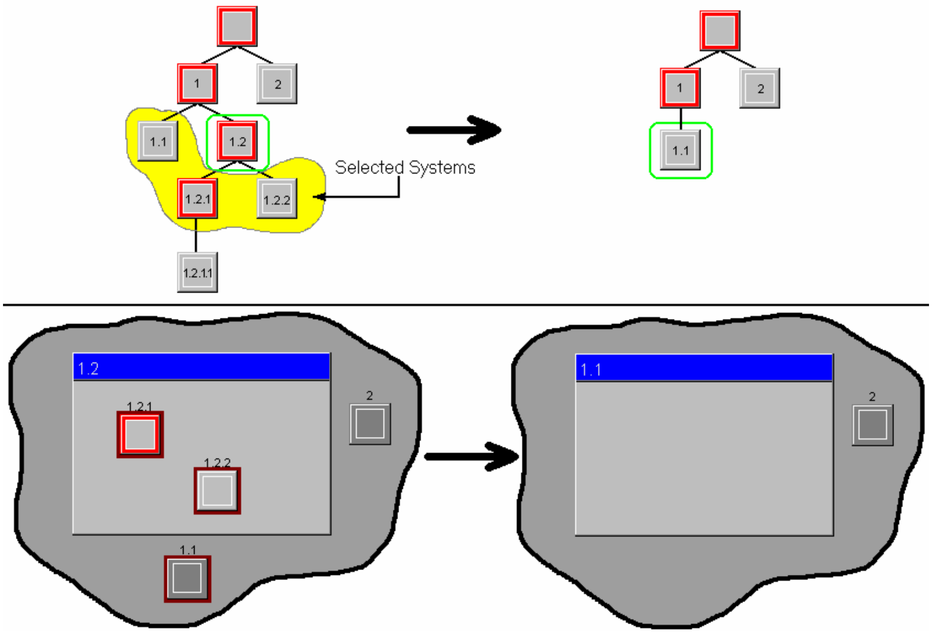

Figure 2.5: *Removing systems*.

o Grouping Selected Systems

As the number of subsystems in a system increases, the management of the subsystems is likely to become easier by grouping a set of those subsystems and introducing an intermediate system to represent them. This operation is called grouping.

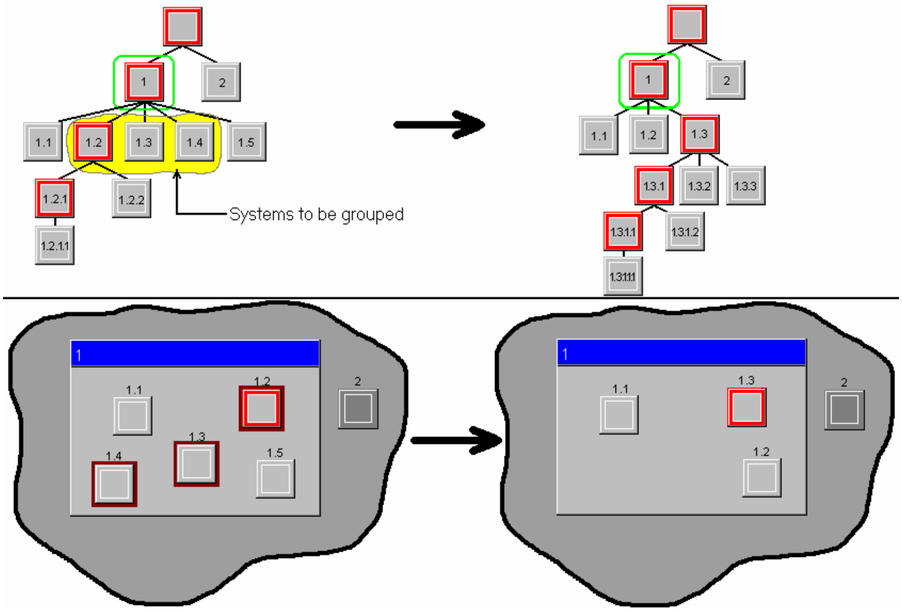

Figure 2.6: *Grouping systems.*

The grouping operation can be defined by a series of previously discussed operations. First we select the subsystems of the focus system that we want to group. Then the selected systems are removed and a new system is inserted to the right of the remaining subsystems (if any). The removed systems are finally reinserted, but now with the new system as their parent system. Grouping systems does not alter the (flat) physical topology of the process. It is merely a matter of convenience and management of the hierarchy.

o Degrouping a System

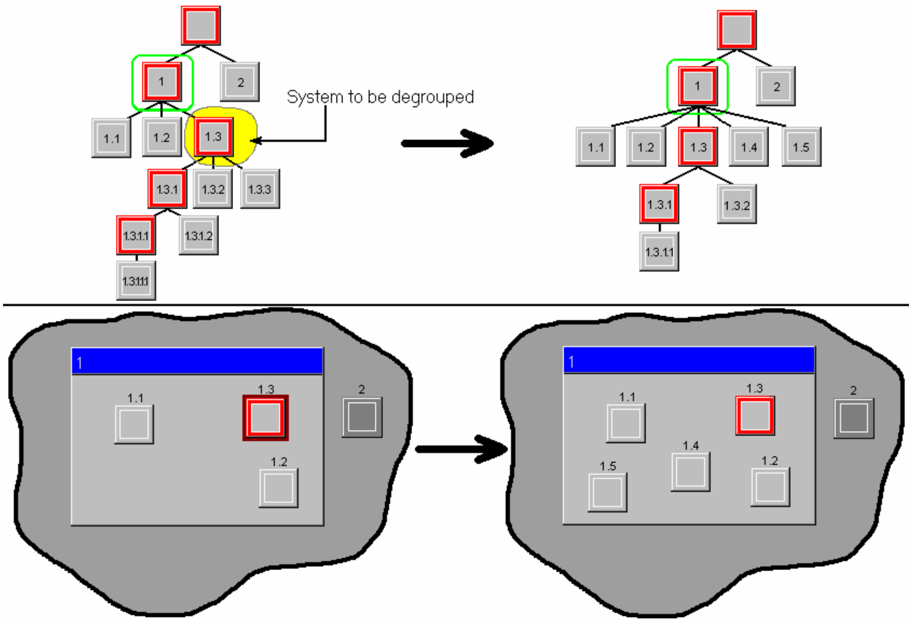

Figure 2.7: *Degrouping a system*.

The degrouping operation is the inverse of the grouping operation. In this case an intermediate system, i.e. the system that is going to be degrouped, is removed after the subsystems of that intermediate system have become subsystems of the intermediate systems parent. Elementary systems can, of course, not be degrouped, because they do not have any subsystems.

o Copying or Saving Selected Systems

The selected systems can be copied. This operation makes an exact copy of the selected elementary and/or composite systems, and stores them in a buffer. For reasons of compatibility and consistency, only a set of systems that are subsystems of the focus system can be copied (or selected for copying). The saving of the selected systems is basically the same operation as the copying, with the difference that the systems are now stored on a disk, so that they can be used any time for the construction of any physical topology.

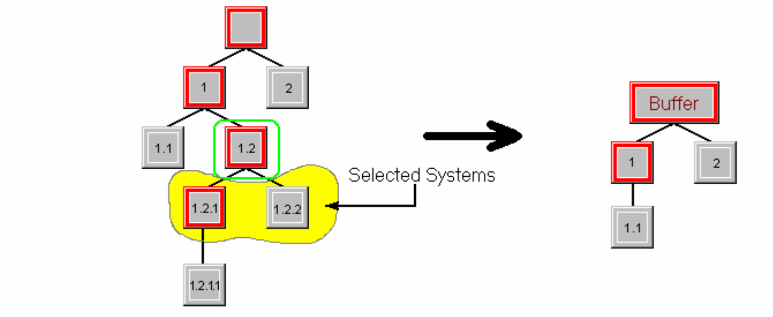

Figure 2.8: *Copying/exporting systems*

o Pasting Copied or Imported Systems

The pasting of copied or (from disk) imported systems is an operation that is very similar to the insertion of a new system. The paste operation adds the systems, which are stored in the buffer or on disk as subsystems of the focus system. Again the rule applies that "new" systems are always added to the right of the existing subsystems.

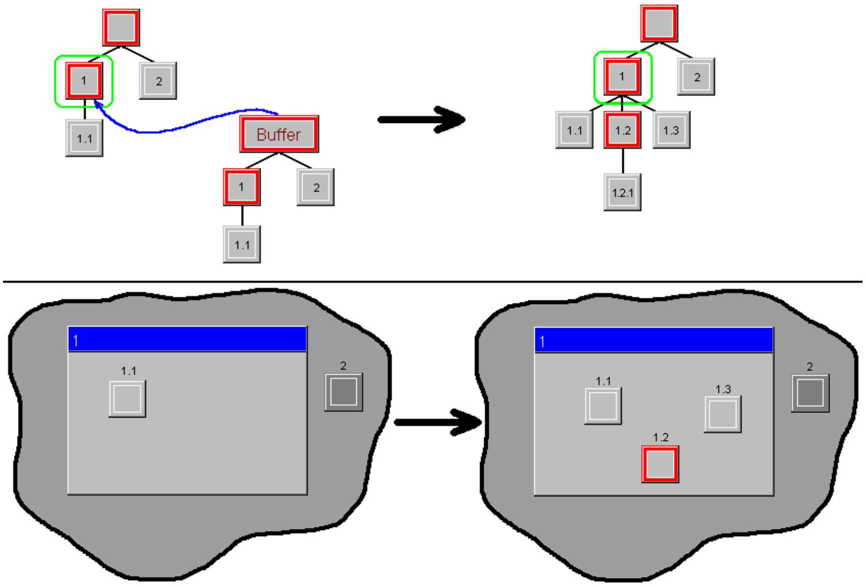

Figure 2.9: *Pasting/Importing Systems*

o Moving a System

Sometimes it is necessary to move a system within the tree structure. The new parent of the system can be any system of the tree that is not a subsystem of the system that is moved. After each operation the identifiers of all the systems that were influenced by the operation, are updated.

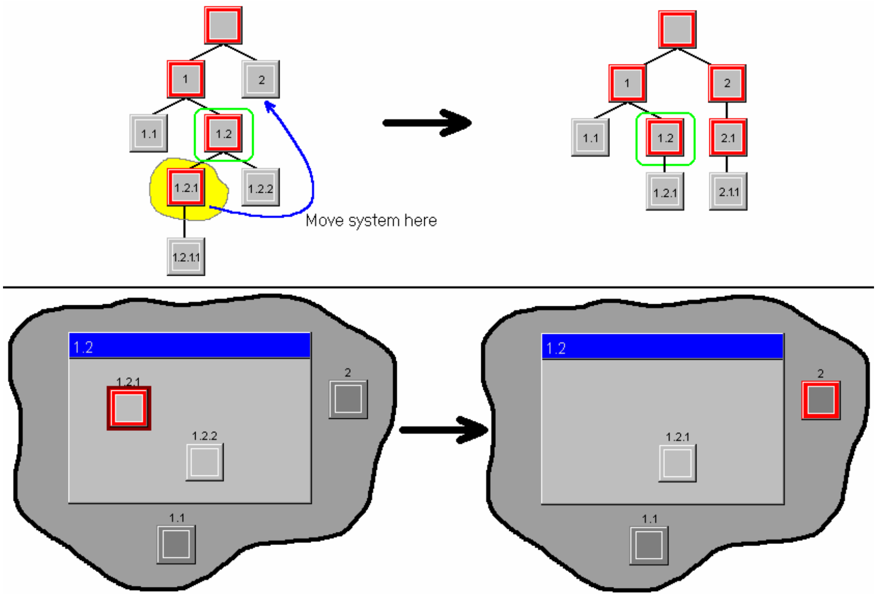

Figure 2.10: *Moving a system within the tree structure.*

2.4 Representation of Connections

The used graphical representation of a hierarchical tree does not show every part of the model in detail. Most parts of the model are shown with very little detail, hiding the structure of their lower levels. Consequently, not every connection will be visible in this representation. Only connections which have an origin and/or target that is a subsystem of the focus system, will be displayed. Connections between two environment systems are not shown for reasons of surveyability. With this approach, we can distinguish between two "classes" of graphical connections, namely *internal* and *external connections*. An *internal connection* is a connection between two systems that are both subsystems of the focus system. An *external connection* has one system as origin or target which is not a subsystem of the focus system. Connections are defined between two elementary thermodynamic systems, but in our representation it is not always possible to show both of these systems at the same time. Therefore, it is necessary to introduce "(graphical) connections" between composite systems. What this means will become clear in the following sections.

2.4.1 Making a New Connection

A correctly defined connection is always defined between two elementary systems. If a model designer wishes to generate a new connection, he must first define the type of connection he wishes to generate. In other words, the modeller must define what kind of extensive quantity will be transferred through the new connection. For now the choices are limited to: a mass, heat or work connection (and information connections for the "transfer" of information). Then he must select a system that will serve as the origin of the connection and, subsequently, he must select a target system.

For ease of establishing a connection, **Mobatec Modeller** automatically invokes a zoom-in operation upon the selection of a composite system as either origin or target system. This means that the selected composite system will be set as focus system and the model designer can select a subsystem of this system. This is done because a connection always has to be defined between elementary systems.

2.4.2 Graphical Representation of Connections

The graphical representation of connections is best explained with the aid of a simple example.

Example 2.2*: Graphical Representation of Connections*

> *Consider a process for which the tree structure representation is defined as in figure 2.2. Let the following connections be given between the elementary systems (see figure 2.11a; se figure 2.1 for a flat topology of this example process):*

Connection number	*Origin System*	*Target System*
1	*1.1*	*1.3*
2	*1.1*	*1.2.1*
3	*1.2.1*	*1.2.2*
4	*2*	*1.3*
5	*1.2.2*	*3.1*
6	*3.2*	*2*
7	*1.3*	*3.2*

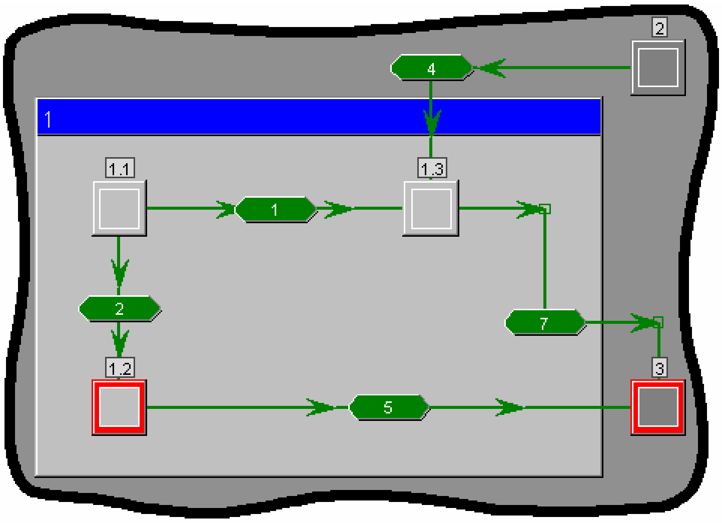

Figure 2.11: *Graphical representation with system 1 as the focus system.*

The following remarks can be made concerning these connections and their representation if system 1 is the focus system:

- *Connections "1" and "2" are both internal connections of system 1 (and are both external connections of system 1.1). Connection "1" is shown as a connection between two elementary systems. For connection "2" this is not possible, because the target of the connection (the elementary system 1.2.1) is not displayed on the screen. The connection is therefore displayed as a connection between the elementary system 1.1 and composite system 1.2, for system 1.2.1 is a subsystem of system 1.2.*
- *Connection "3" is an internal connection of the system 1.2 and is not displayed on the screen because both the origin and the target of this connection are subsystems of system 1.2. If one wants to show this connection on the screen, one must zoom in on system 1.2 (and set this system as the focus system). So, only internal connections of the focus system are shown on the screen. Internal connections of any other system are always hidden.*
- *Connections "4", "5" and "7" are external connections of the focus system 1, because these connections cross the boundary of system 1.*
- *Connection "6" is not shown in this representation, for this connection is established between environment systems of the focus system 1.*

Figure 2.12 shows the graphical representation of the process with the tree root as the focus system.

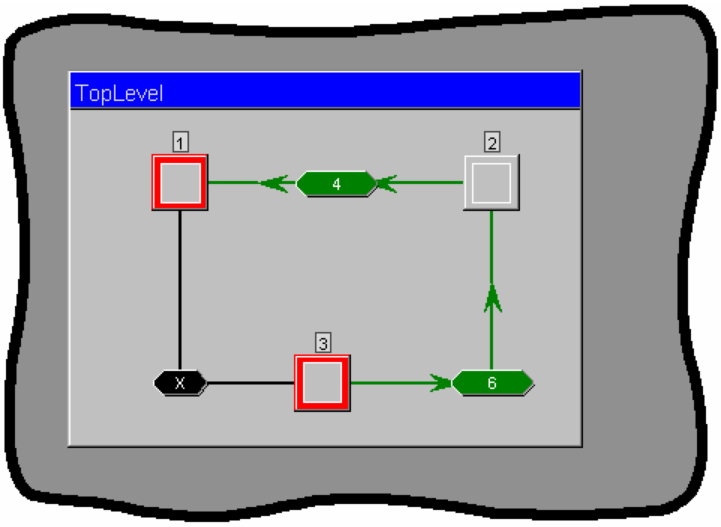

Figure 2.12: *Graphical representation with the tree root as the focus system.*

Remarks:

- As you can see, a problem arises here, for connection "5" and "7" are both defined between the (composite) systems 1 and 3. In the graphical representation it is displayed as only one connection "X", while it in fact is a "composite connection" or "vector of connections", consisting of the connections "5" and "7". In this way an arbitrary number of connections between two (elementary or composite) systems can be defined and represented.

- If every single connection would have to be displayed in the graphical representation, the screen could become very crowded and one would lose a certain amount of surveyability. Imagine, for example, the display of ten connections between a pair of systems.
- If a user does want to see the individual connections, he has to either check the "Always Display as Single Connection" checkbox on the Physical Topology Tab of the connection properties or press the "Expand" button (keyboard shortcut "x") to expand the vector of connections into a set of single connections.
- Information about the connections contained in a "composite" connection can be easily accessed with the modelling tool.
- "Single" connections show the user-defined directionality of the connection graphically and composite connections do not (and can not). The directionality of the connections of a composite connection can be easily accessed with the modelling tool. Remember, however, that the origin and target of a connection only define a reference co-ordinate system for the actual flow (of extensive quantities). The actual direction of the flow depends on a difference in potential between the systems.
- All defined systems are subsystems of the tree root and consequently there are no external systems and no external connections.

2.5 Consequences of Manipulations on the Physical Topology

Manipulation of the physical topology can give rise to two kinds of problems with the already defined connections. The first arises when one wishes to refine an elementary system, which is connected to other systems. The second problem occurs when one deletes, copies or saves a (composite or elementary) system which has one or more external connections. Both of these actions will cause the involved connection to be, at least temporary, not properly defined or "undefined". The two problems give rise to two different kinds of "undefined" connections, namely "*loose connections*" and "*open connections*". The following will discuss how these two types are handled in the modelling tool.

2.5.1 Loose Connections

Loose connections show up when one wants to refine an elementary system that has connections. This means that one "zooms in" on an elementary system, which is already connected to other systems, in order to add subsystems to this system and thus change it into a composite system. As soon as one or more subsystems are inserted to an elementary system with connections, all these connections will loose their meaning/relevance because they are now no longer defined between two elementary systems.

A solution to this problem is to introduce (temporary) loose connections. A *loose connection* is defined as a connection of which one or two ends are connected to a composite system. A loose connection thus has no physical meaning; it is just a temporary state of a connection to help the model designer in fast and easy modelling. The "loose ends" of a loose connection have to be reconnected to a subsystem of the system the connection was originally connected to, in order to form a properly defined connection again.

***Example 2.3**: Loose Connections*

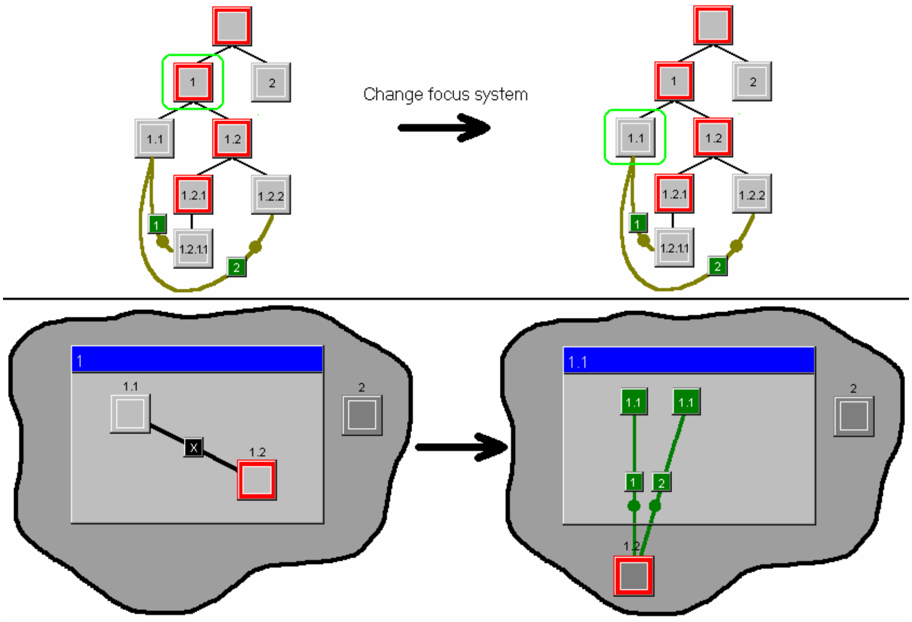

Figure 2.13: *Appearance of loose connections, when zooming in on a connected elementary system*.

Consider a system (system 1 in figure 2.13) which is connected to two other systems (systems 1.2.1.1 and 1.2.2). If we now zoom in on system 1.1 (as shown in figure 2.13), the connections that are connected to the system 1.1.are shown as loose connections, but they are, in fact, still correctly defined connections, because system1.1 is still an elementary system. If we would add a new system to system 1.1, the connections would loose their meaning, because system 1.1 would then turn into a composite system and connections must always be defined between two elementary systems. The "1.1" in the temporary endpoints of the loose connections denote that the loose connections have to be reconnected to a subsystem of system 1.1. The reconnection of the loose connections (if necessary) can be done at any point during the model definition.

□

2.5.2 Open Connections

An *open connection* is defined as a connection that is only connected to one system. The other system of the connection (either the target or the origin) is not yet defined. Connections that are not connected to any system are no connections and therefore do not exist. So, if the result of a manipulation on the physical topology is an open connection with no connected systems, then this connection is deleted.

Just like a loose connection, an open connection has no physical meaning and is just a temporary state to help the model designer. Open connections can appear by one of the following actions:

- Deleting a system that is connected to systems that are not deleted.
- Copying or exporting a set of systems that are connected to systems that are not being copied or exported. The copied systems will contain connections that are open because these connections are only connected at one side.
- Deliberately making a new connection that is only connected to one system (either the origin or the target of the connection). This can be of help, for example, in the construction of not finished, but reusable model parts, such as heat exchangers. In the overall model, a shell and tube heat exchanger must be connected to a source and sink for both the contents of the shell and tube. The heat exchanger as a stand-alone model is not (yet) connected. If a model designer imports the model of the heat exchanger into another model, he will not forget to properly connect the heat exchanger to the existing model, because the open connections of the heat exchanger have to be reconnected.

Open connections may be reconnected at any time to an elementary system of the physical topology. They can, of course, also be deleted.

A special feature of **Mobatec Modeller** that only applies to open connections is that they can be hidden in order to increase the surveyability when the model designer is not focussing on the open connections.

Example 2.4*: Open Connections*

> *Figure 2.14 shows that by deleting system 1.1 three open connections will form. Standard, only those open connections are shown for which the connected endpoint is visible on the screen. In this example this means that only connections 2 and 3 are visible and connection 1 is hidden, because system 1.1.1.1 is not visible on the screen. It is possible, though, to show all the open connections on the screen.*

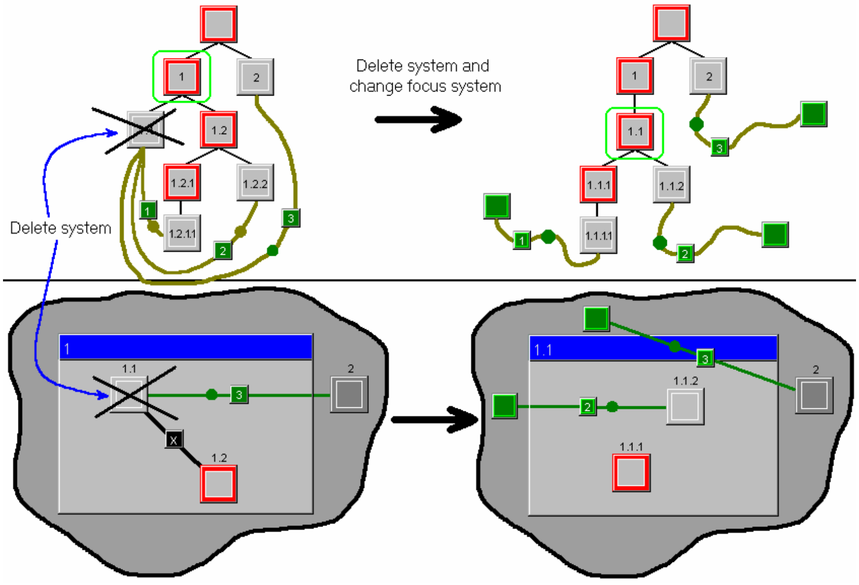

Figure 2.14: *Appearance of open connections, when deleting a connected elementary system.*

In contrast to loose connections, the temporary endpoints of the open connections do not contain a number. This means that an open connection can be reconnected to any elementary system of the physical topology.

□

2.5.3 Possible States of Connections

As mentioned, connections can have different (temporary) states to help the model designer with the construction of process models. The final model can, of course, not contain any "undefined" connections, so these have to be dealt with before the model designer generates the end result. Figure 2.15 summarizes the possible temporary states of connections, during the model construction phase:

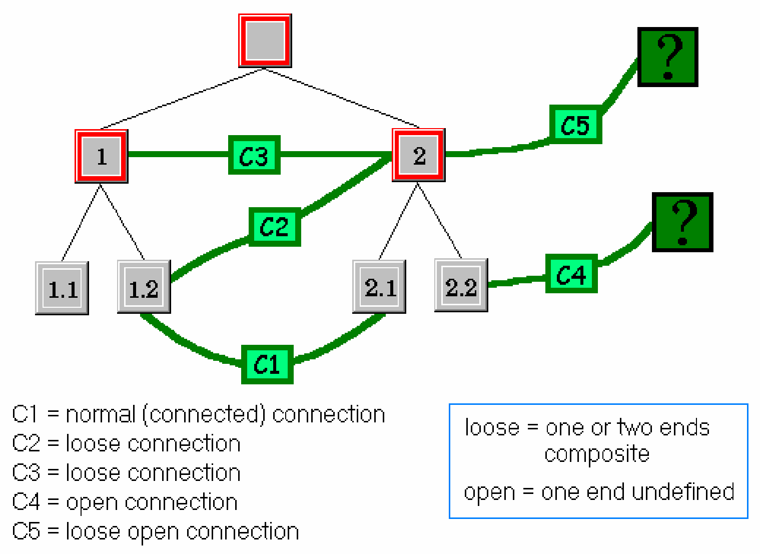

Figure 2.15: *Possible states of connections.*

Remarks:

- The loose end of connection C2 (i.e. system 2) can only be reconnected to (elementary) subsystems of this composite end (i.e. system 2.1 or 2.2). The same accounts for the loose ends of connections C3 and C5
- The open ends of connections C4 and C5 can be reconnected to any elementary system (except the (elementary) system it is already connected to (system 2.2 in case of reconnecting connection C4)).

2.6 Repetitive Structures

An additional advantage of the temporary states of the connections is that these can be used for the definition of so-called repetitive structures. These repetitive structures can be used to approximate the behaviour of distributed systems or to build large, interconnected physical topologies in a very quick manner. The implementation of repetitive structures in **Mobatec Modeller** is limited (for now) to one-dimensional repetitions. Expansion to two and three-dimensional repetitions should, in principle, be trivial.

The following example illustrates the implementation of repetitive structures. In principle, the repeated structure can be a tree of any size and depth with any kind of interconnection, but to keep things simple we use a “flat topology” as repetitive structure.

***Example 2.5**: Repetitive Structures*

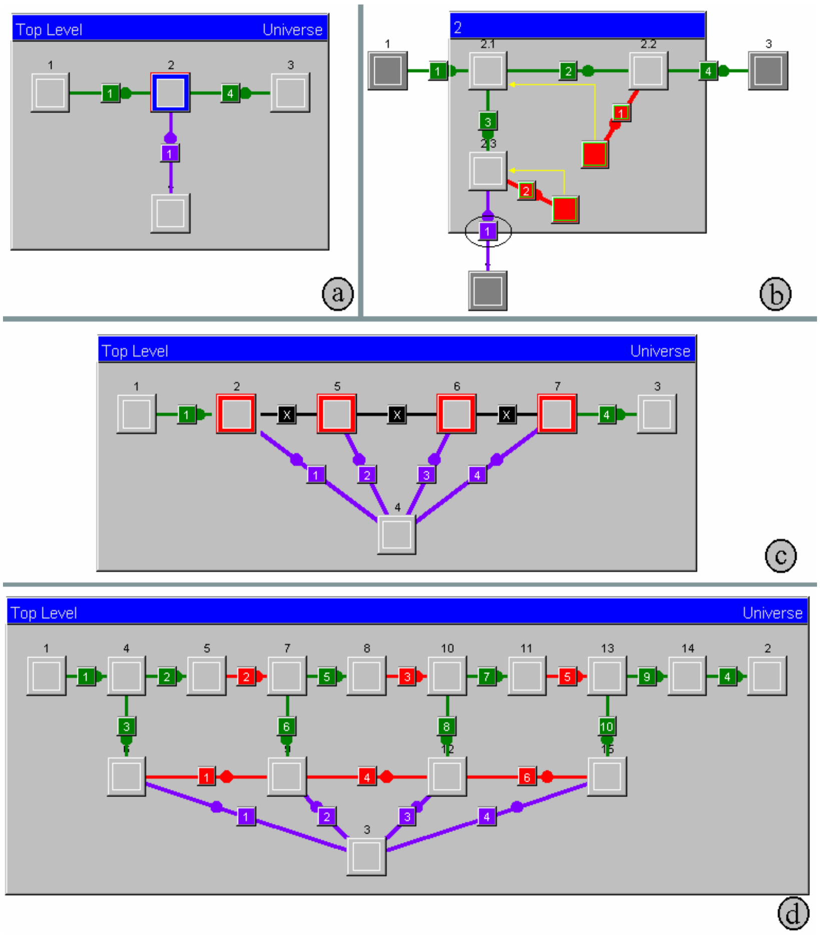

Figure 2.16: Repeating a repetitive structure. a-b) repetitive structure. c-d) repeated repetitive structure.

Consider the physical topology shown in figures 2.16a and 2.16b. The physical topology consists of 6 systems (1, 2.1, 2.2, 2.3, 3 and 4), 4 mass connections, 2 heat connections and 1 work connection. The system 2 is the one that is to be "repeated" (three times, in this example). In this example, the heat connections are used for the interconnection of the repeated structures. The "repetitive system" of heat connection 1 is system 2.1, which means that the connection will be connected to the system 2.1 of the next copy when the structure is repeated. The repetitive system of heat connection 2 is system 2.3. The (encircled) work connection is set to be "connected to all repeated structures". Mass connections 2 and 3 are internal to the structure and will be copied and be internal to each copy of the repeated structure. Mass connection 1 has its target in the repeated structure and will only be connected to the first of the repeated structures. Mass connection 2 has its origin in the structure that is to be repeated and will only be connected to the last copy, when the structure is repeated. Figure 2.16c shows the result when system 2 is repeated three times. Figure 2.16d shows the result of the repetition when the composite systems (2, 5, 6 and 7) shown in figure 2.16c are degrouped.

□

The repetitive structure building function can seriously speed up the construction of larger models, because, when the repetitive structure is properly initialized, the resulting structure will also be properly initialized.

3 Construction of the Species Topology

The definition of the species topology of a process is initialized by assigning sets of species (and/or reactions) to some systems. Species (i.e. chemical and/or biological components), as well as reactions should be selected from corresponding databases. So, before the species topology can be defined, a species and a compatible reactions database must be defined. Such a database contains a list of species and a list of possible reactions between those species. A species and reactions database should, of course, be editable by the user in order to satisfy the specific needs of the user.

In the following, the complete process the model designer wishes to develop is referred to as the plant.

3.1 Defining Plant Species and Reactions

Processing plants involve different types of processes, such as mineral, petrochemical, pharmaceutical, polymeric, or biological with various species and reactions. The definition of the species in various parts of a processing plant requires data about these species. This data can be found in species and reactions databases. In general, a processing plant does not involve all the species and reactions that are listed in a database. Therefore, a model designer has to select a subset from this list, which comprises all the species and reactions that appear in the (model of the) plant and which thus serves as the ”database” for that particular plant. This plant database is then a collection of all species and reactions that may exist in the plant. The selected subset of species will be called the *plant species*. The selected reactions will be called the *plant reactions*. These subsets have to be formed to improve the surveyability and to prevent unnecessary computations involving species and reactions that are not present in the plant.

For the construction of the species topology, the species name and its formula are sufficient, but the database can and must, of course, contain other information, such as specific physical properties. Similarly for reactions, only information about which reactants and which products are involved in a reaction, is sufficient for the construction of the species topology. However, stoichiometric coefficients and other reaction data are most likely to be connected with the database.

With **Mobatec Modeller**, species and reactions can be selected from one or more user specified databases to form the plant species and reaction sets. To aid a model designer in defining a plant species and reaction set for his process, the following actions can be performed:

- Adding species to the plant species set. The user specified species will be copied from the database and added to the plant species set.
- Adding reactions to the plant reaction set. Whenever a new reaction is added to the plant reactions set, the plant species set will automatically be extended with all involved species(i.e. the reactants and products of the reaction), which were not yet listed in this set.
- Removing reactions from the plant reaction set. The user-specified reactions will be removed from the plant reaction set and will consequently also be removed from all the systems in which they were injected.
- Removing species form the plant species set. Whenever a species is removed from the plant species set, all reactions of the plant reactions set, which contain this species (either as a reactant or a product), will be removed from the plant reactions. The species and reactions that were removed from the plant sets will, of course, also be removed from all the systems they were used in.

3.2 Making New Species and Reactions

Databases holding information about species and reactions can be expanded by a user by introducing new species and/or reactions. In order to add a new species to a database, a user must only give a name and specify a formula (which is used in the notation of reactions and for the translation of variables in the generated output code). Optionally, an identifier can be given, which can be used by external physical property databases.

Species specific parameters can be assigned to species by selecting the appropriate (vector) variable from the list of available variables and adding it to the species parameter list. A value for the new parameter can be given.

When introducing a new reaction, the reactants and products with their stoichiometric coefficients need to be defined. The handling of reaction equations is not discussed in this generic document.

3.3 Injecting Species and Reactions

The definition of the species topology is initialized by ”injecting” species and/or reactions into elementary systems. This means that the model designer has to assign a set of species and/or reactions to some elementary systems. These species can only be selected form the defined plant species and reactions, since these sets represent all species and reactions that may exist within the plant.

After the assignment of the injected species and injected reactions to a specific system, the modelling tool will (re)calculate (parts of) the species distribution. This means that the species will propagate into other systems through mass connections. Within the systems, the species may undergo reactions and generate ”new” species, which in turn may propagate further and initiate further reactions. This eventually results in a specific species distribution over the elementary systems, which is referred to as the *species topology* of the processing plant.

The foregoing made clear that the species that are actually associated with a system can have either one of the following sources:

- Initialized species: injected to a system by a model designer
- Transferred species: arrived through one of the mass connections
- Product species: result of a reaction.

The reactions that are associated with a system can only be the result of a manual injection by a model designer.

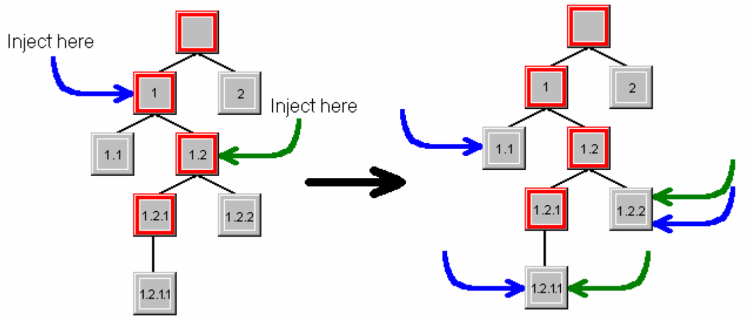

Figure 3.1: *Injecting of species and/or reactions at composite systems.*

Species and reactions may also be injected at composite systems. The injected species and/or reactions will then actually be injected in all the elementary systems that are subsystems of the composite system (see figure 3.1). The same accounts for the removal of injected species or reactions: If a species or reaction is to be removed from a composite system, then this species or reaction will actually be removed from the injected species or injected reactions sets (only if the specific species or reaction is listed, of course) of all the elementary systems that are subsystems of the composite system (see figure 3.2).

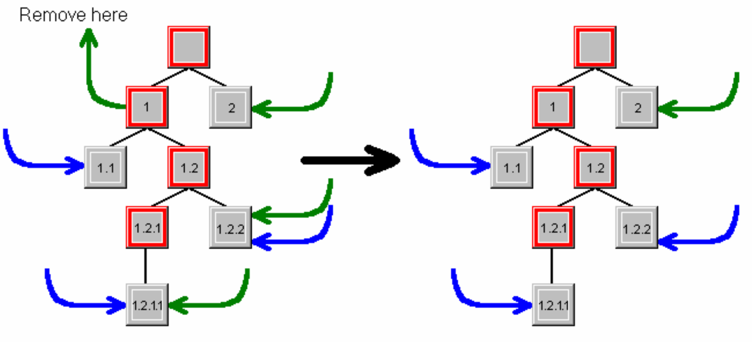

Figure 9.2: *Removing of injected species and/or reactions from a composite system.*

The injection of a reaction into an elementary system does not automatically imply that this reaction can ”happen” in this system and thus that the products of this reaction can be formed. If not all reactants of a reaction are available in a system, then this reaction *cannot* take place in this system. In such a case, the system will have an injected reaction but this reaction will not be active (it is deactivated). So, the reaction will not take place in the system in this case. When the species distribution is changed and the reaction can take place again, it will automatically be ”re-activated”.

It should be noted that the presence of an ”activated” reaction in a system does not imply that this reaction *has* to happen in this system. It implies that this reaction *may* happen in this system, depending on the operating conditions in the system and the driving force for this reaction.

3.4 Permeability and Uni-Directionality of Mass Connections

In the physical topology of a plant, mass connections establish the communication paths between pairs of systems, without specifying which species may or may not be transferred. As mentioned, permeability and uni-directionality are introduced as properties of mass connections. They constrain the mass exchange between systems by making the species transfer respectively selective or uni-directional.

The permeability of a mass connection can be defined in two complementary forms:

- **Permeable Species Set**: This set includes all species that may pass through the mass connection. All other species of the plant species set can not be transferred via this mass connection.
- **Non-Permeable Species Set**: This set is formed by selecting all species that are not transferable through the mass connection.

Both complementary approaches are enabled with **Mobatec Modeller** since not having one or the other approach may force a model designer to (repetitively) define large sets of species.

Often, most connections in a model of a process represent convective flows, which by definition communicate all species. Therefore, the default permeability of a mass connection is defined by an empty non-permeable species set and thus as permeable for all species.

Which species are actually transferred through a connection depends on the species sets of the two connected systems. In reality, it also depends on the actual transfer direction, which depends on the states of the two connected systems. Since the latter information is not available during the construction of the physical and species topology, mass connections are assumed to transfer potentially in both directions (by default). Species may therefore exist on both sides of a connection, that is, if a species appears on one side of a connection and it is not inhibited by a defined permeability, it may also appear on the other side. This policy is very suitable for safety and hazard studies, as the resulting species topology is a maximum solution, showing where the different species *may* occur (and showing which reactions are potentially possible) in different parts of the process' model. This approach, though, will generate more equations than necessary for the description of common processes. That is why the concept of *uni-directional* mass transfer is introduced. This can reduce the number of equations, because species are only allowed to propagate in one direction when the species distribution is (re)calculated and this can reduce the number of species in an involved system. The resulting species topology, based on this policy, could reflect more closely the actual situation in the process.

3.5 Propagation of Species

Whenever an operation is executed which modifies the current species distribution, a mechanism is activated which updates the species distribution over all elementary systems and connections of the affected mass domains. This mechanism first clears all the information about the current species distribution of the affected mass domain. Then all user-defined injected species and reactions are injected in their respective elementary systems. After this, all injected species as well as all products from injected (and activated) reactions are propagated in the mass domains.

3.6 Consequences of Manipulations

The application of some modification to the physical structure of a model, such as deletion of a system or the removal of a (mass) connection, will invalidate (parts of) the constructed species topology. The species distributions of all affected mass domains and the mass domains themselves are therefore reconstructed by **Mobatec Modeller** after any significant change of the physical topology.

Also, modifications to the species topology itself will influence the species distribution. For example, the removal of an injected species, the “removing” of a reaction or the changing of the permeability of a connection will have an effect on (parts of) the species topology. Every action that affects the species distribution of a particular mass domain will invoke an update mechanism which will recalculate the species distribution of that domain.

4 Construction of the Equation Topology

Having completed the first two stages of the modelling process, Mobatec Modeller can automatically generate the dynamic part of the process model, namely the (component) mass and energy balances for all the elementary systems, using the conservation principles. The resulting (differential) equations consist of flow rates and production rates, which are not further specified at this point. In order to fully describe the behaviour of the process, all the necessary remaining information (i.e. the mechanistic details) has to be added to the symbolic model of the process. So, in addition to the balance equations, other relationships (i.e. algebraic equations) are needed to express transport rates for mass, heat and momentum, reaction rates, thermodynamic equilibrium, and so on. The resulting set of differential and algebraic equations (DAEs) is called the equation topology.

From a certain point of view the modelling process can thus be regarded as a succession of equation-picking and equation-manipulation operations. The modeller has, virtually at least, a knowledge base containing parameterized equations that may be chosen at certain stages in the modelling process, appropriately actualized and included in the model. The knowledge base is, in most cases, simply the physical knowledge of the modeller, or might be a reflection of some of his beliefs about the behaviour of the physical process.

The equation topology forms a very important part of the modelling process, for with the information of this topology the complete model of the process is generated. The objective of the equation topology is the generation of a mathematically consistent representation of the process under the view of the model designer (who mainly judges the relative dynamics of the various parts, thus fixes intrinsically the dynamic window to which the model applies). In order to efficiently produce dynamic process models, Mobatec Modeller must, of course, appropriately deal with variables and equations.

In this chapter a classification of the variables and equations that are occurring when modelling physical/chemical systems is made. Next, the balance equations are discussed. After that, the algebraic equations, that are necessary to complete the mathematical model of the process, are classified. The proposed classification makes modelling into an almost trivial activity.

4.1 Variable Classification

The conservation of extensive quantities forms the foundation for describing the dynamic behaviour of systems. Therefore, the set of conserved quantities are called fundamental extensive quantities or the primary state. In the thesis of dr. ir. M.R. Westerweele it is shown that the dynamic part (i.e. the differential equations) of physical-chemical-biological processes can be represented in a concise, abstract canonical form, which can be isolated from the static part (i.e. the algebraic equations). This canonical form, which is the smallest representation possible, incorporates very visibly the structure of the process model as it was defined by the person who modelled the process: The system decomposition (*physical topology*) and the species distribution (*species topology*) are very visible in the model definition. The transport (*z*) and production (*r*) rates always appear linearly in the balance equations, when presented in this form:

$$\frac{d}{dt}x = Az + Br$$

in which

x :: Fundamental state vector

z :: Flow of extensive quantities

r :: Kinetics of extensive quantity conversion

A:: Interconnection matrix

B:: Stoichiometric coefficient matrix

The classification of variables that is presented here is, in the first place, based on the structural elements of the modelling approach (namely systems and connections). The sub classification has its origins in thermodynamics. The variables that appear in the mathematical model of a process are divided into three main groups, namely system variables, connection variables and reaction variables.

4.1.1 System Variables

System variables are variables that are defined within the boundary of a system (or sometimes a group of systems). They can be subdivided into two main groups:

- The basic type of variables are the *fundamental extensive variables* or *fundamental state variables* (*x*) representing the quantities for which conservation laws are valid and for which we write balance equations. In most chemical processes, these are component masses (total mass is the sum of the component masses), energy, and momentum.

- The second group of variables is called *application variables* or *secondary state variables* (y), which is a gathering of two categories, *intensive* and *geometrical variables*, as well as other variables not covered by the previous classification. The secondary state variables are quantities that are derived from the primary state (x). The secondary state must be the result of a mapping of the primary state ($y=f\ (x)$). Thus in any model, this link must exist.

Intensive variables are those that provide a local characterization of the process. We distinguish among them *potentials* that are driving forces for flows such as temperature, pressure, chemical potential, and *physical properties* characterizing the "quality" of the material such as density, or concentration.

Geometrical variables are those that characterize the dimension configuration of the process, independent of the presence of the physical process. They may describe the extent of the process such as volume, area, etc. or give localized characteristics such as porosity, tortuosity, etc.

The choice of the intensive and geometric variables that characterize the process is very much application dependent in the sense that it is determined by the final goal of the model. A common property of the intensive and geometric variables is that, in general, these are the variables for which direct information through measurements is available.

4.1.2 Connection Variables

Connection variables are associated with physical connections and can be divided into two groups:

- *Flow rates* of fundamental extensive quantities represent the rates by which the respective quantities are exchanged between two interconnected elementary systems. An important aspect that we retain for our analysis is that some transfers may induce flows of several fundamental extensive quantities. For example, a mass transfer will induce flows of component masses for each substance present in the transferred material, but also flows of momentum and energy. Similarly, a momentum transfer induces also an energy transfer in the form of mechanical work. In general, each physical connection has assigned a unique fundamental flow and, possibly, several (or no) induced flows.

  We distinguish in our model between *fundamental* or *primary flow variables* (z) and *secondary flow variables*. The primary flow variables are the variables that appear directly in the balance equations. Secondary flow variables are related to the primary flow variables via an algebraic expression and do not appear in the balance equations (when, of course, no substitutions have been made). For example, the mass flow rate (primary flow variable) of a mass stream of incompressible fluid can be easily expressed as a function of the corresponding volumetric flow (secondary flow variable). The flow rates are usually defined as a function of the secondary state variables of the two interconnected systems ($z=h(y_{or}, y_{tar})$).

- *Application variables* are a gathering of *physical properties*, *geometrical variables* and other variables that are related to the connection (i.e. the common boundary of the two interconnected systems).

4.1.3 Reaction Variables

The last group, the reaction variables, will also be split up into two subgroups:

- *Production rates* (r) of fundamental extensive quantities represent the rates by which the respective quantities are produced inside an elementary system. The typical examples are the chemical reaction kinetic rates. These variables are associated with those phenomena in which transformations of fundamental extensive quantities occur inside an elementary system. Such is the case for chemical reactions in which component masses are being transformed into one another. As in the case of flow rates, we can speak of fundamental production rates and induced production rates. In the case of chemical reactions, it is common practice to use the extent rate of a reaction to describe the evolution of the reaction. The consumption rates of reactants and the production rates of the reaction products are expressed in terms of the extent rate of the reaction and thus become induced production rates. The production rates are usually defined as a function the secondary state variables of a system ($r=g(y)$).

- *Application variables* are a gathering of variables that are related to the reaction (for example, the pre-exponential kinetic constant or the activation energy of a reaction).

4.2 Fundamental State Variables and Equations

The behaviour of a system is characterized by the evolution of its state with time. The choice of the elementary system is driven by the choice of the timescale in which the dynamics of the system is to be resolved. This choice is fundamental to the analysis.

In order to characterize the behaviour of a system, information is needed about the natural state of this system (at a given time) and about the change of this state with time. The natural state of a system can be described by the values of a set of fundamental extensive quantities, while the change of state is given by the balance equations of those fundamental variables. The *fundamental extensive* variables represent the ”extent” of the system c.q. the quantities being conserved in the system. In other words: they represent quantities for which the conservation principle and consequently also the superposition principle applies. So, for these variables, the balance equations are valid. In most chemical processes, the fundamental variables are: component mass, total energy and momentum. But other extensive quantities (charge, for example) might be necessary sometimes and will obviously have to be present in a generic modelling tool.

The dynamic behaviour of a system can be modelled by applying the conservation principles to the fundamental extensive quantities of the system. The principle of conservation of any extensive quantity *x* of a system *S* states that what gets transferred into the system must either leave again or is transformed into another extensive quantity or must accumulate in the system (In other words: no extensive quantity is lost). Symbolically we can write a balance equation for the fundamental extensive variable x^f that characterizes the elementary system *S* as:

$$\frac{d}{dt} x_S^f = \sum_c \alpha_c \hat{x}_c + \sum_p \tilde{x}_p$$

Where the first sum is taken over all the physical connections *c* which contribute to the exchange of x_S^f and the second sum is taken over all production/consumption processes *p*. Further, $\hat{x}_c$ represents the corresponding flow rate of extensive quantity *x* through connection *c* and $\tilde{x}_p$ corresponds to the production rate (which is assumed to be of negative sign if it corresponds to a consumption process).

The direction of the flow through a connection is defined relative to the reference co-ordinate system, which, in essence, is introduced for the consistency of the models. This means that a flow is positive if it moves in the reference direction and negative when it moves in the opposite direction. Adding the predicate of direction to every flow allows for dynamic changes of the directionality without affecting the structure of the involved equations. Further, the transfer is defined only once and is then incorporated in the two sets of balance equations describing the behaviour of the two connected systems. The coefficients α_c $\{-1, 1\}$ indicate the conventional direction of the flow rate $\hat{x}_c$.

4.3 Balance Equations

Mobatec Modeller currently supports only component mass and energy (i.e. enthalpy) balanced variables. Extensions to other extensive quantities (such as momentum or charge) will be available in the future. However, a very wide range of physical, chemical and/or biological processes can be modelled with only component mass and enthalpy as balanced quantities.

4.3.1 Mass Balances

Mass is one of the prime extensive quantities when modelling physical-chemical processes. The generic balance equation applies to any of the species present in the system, whereby the number of independent balances is equal to the number of species, that is, if no constraining assumptions are being made. The total mass balance is always the sum of the component balances, thus it is linearly dependent on the component masses.

In most applications it is convenient to use the component mass balances as the basic set of equations. The accumulation of the component masses in a system (*S*) is balanced by the transfer across the system boundary and the internal conversion through reactions:

$$\frac{d}{dt}\underline{n}_S = \sum_{\forall c} \alpha_c \underline{\hat{n}}_c + \underline{\tilde{n}}_S$$

where

$\underline{n}_S$ :: Vector of species, in moles

$\underline{\hat{n}}_i$ :: Vector of molar mass flows of mass connection i

α_i :: Unit direction of reference co-ordinate $\in \{-1, 1\}$

$\underline{\tilde{n}}_S$ :: Vector of production rates for all species

This equation represents the component mass balances of any dynamic system, which is defined in a model of a physical-chemical process.

The mass connections describe the exchange of mass across the boundary separating two adjacent systems. The direction of the flow through a connection is defined relative to the reference co-ordinate system, which, in essence, is introduced for the consistency of the models. This means that a flow is positive if it moves in the reference direction and negative when it moves in the opposite direction. Adding the predicate of direction to every flow allows for dynamic changes of the directionality without affecting the structure of the involved equations. Further, the transfer is defined only once and is then incorporated in the two sets of balance equations describing the behaviour of the two connected systems.

The production term $\tilde{n}$, which appears in the component mass balance for a particular species A_j of an elementary system *S*, includes the information about the rate of consumption or production of this species. This production term is therefore defined as the sum of the rate expressions of each elementary reaction, which involves the species A_j. Kinetic rate equations or "reaction laws" are empirical relations, which describe the rate of conversion of one species (A_j) in an elementary reaction as a function of the concentration of all the species in the system. The empirical relations are very often power laws, with the exponents reflecting the number of molecules of that particular species involved in the reaction.

Chemical reactions are defined as stoichiometric equations that relate a number of reactant molecules with a number of product molecules:

$$|\nu_1|A_1 + |\nu_2|A_2 + \ldots\ldots \leftrightarrow \ldots\ldots + |\nu_{n-1}|A_{n-1} + |\nu_n|A_n$$

With

ν_i :: Stoichiometric coefficient (>0 for products, <0 for reactants)

A_i :: i^{th} species

The coefficients ν_i in this "equation" are called the stoichiometric coefficients. In order to use the reaction concept in mathematical modelling, the "reaction equation" is slightly generalized, such that the chemical reaction can be represented in a form that is almost like an equation. That is, for every reaction a generalized stoichiometric equation is defined:

$$\sum_{i=0}^{n} \nu_i A_i$$

or, in matrix notation:

$$\underline{\nu}^T \underline{A} = 0$$

With,

$\underline{\nu}$ :: Stoichiometric coefficient vector = $(\nu_1\ \nu_2\ \ldots\ldots\ \nu_n)^{\mathrm{T}}$ for *n* species

$\underline{A}$ :: Species vector, containing all 'reactive' species of the system under consideration = $(A_1\ A_2\ \ldots\ldots A_n)^{\mathrm{T}}$ for *n* species

So, for a system involving multiple reactions, one can write the reactions in the following generalized fashion:

$$\underline{S}\,\underline{A} = \underline{0}$$

With,

$\underline{S}$ :: Stoichiometric matrix = $(\nu_1\ \nu_2\ \ldots\ldots\ \nu_m)^{\mathrm{T}}$ for *m* reactions. (Dimension: m x n)

***Example** 4.1: Stoichiometry*

Given a reaction system:

$$\begin{gathered} A + B \Leftrightarrow C \\ 4B + 2D \Leftrightarrow 2E + F \\ C \Leftrightarrow 4D \\ C + 2E \Leftrightarrow A + 2D + G \end{gathered}$$

the stoichiometric matrix is:

$$\underline{A}^T = \begin{pmatrix} A & B & C & D & E & F & G \end{pmatrix}$$

$$\underline{S} = \left(\begin{array}{ccccccc} -1 & -1 & 1 & & & & \\ & -4 & & -2 & 2 & 1 & \\ & & -1 & 4 & & & \\ 1 & & -1 & 2 & -1 & & 1 \end{array} \right)$$

□

For a species A_j participating in a reaction r the following relation always holds:

$$r_{r,A_j} = rf_{r,A_j} - rb_{r,A_j}$$

With

r_{r,A_j} :: Rate of formation of species A_j by reaction r

rf_{r,A_j} :: Rate of production (or consumption) of species A_j by forward reaction of r

rb_{r,A_j} :: Rate of consumption (or production) of species A_j by backward reaction of r

This means that the rate, with which the species A_j is produced (or consumed), is defined as the rate of the forward minus the rate of the backward reaction.

Obviously, the rate of production and consumption of the products and reactants are related through the stoichiometric coefficients. For each reaction, a unique quantity ξ_r can be defined that is a normalized rate of reaction and which can be shown to be the time derivative of the extent of reaction:

$$\xi_r = \frac{r_{r,A_j}}{\nu_{r,A_j}}$$

In homogeneous reaction systems, the production rate for a species is the cumulative production rate for this species over all reactions. If the dimensions of the reaction rates of a system are defined in mass per time unit then the reaction term $\underline{\tilde{n}}$, which occurs in the component mass balances of a system may be written as follows:

$$\underline{\tilde{n}} = \underline{S}^T \underline{\xi}$$

With

$\underline{\xi}$ :: Normalized reaction rates vector = $(\xi_1 \;\; \xi_2 \;\; ....\xi_m)^T$ for m reactions

4.3.2 Energy Balances

Energy is the second most important extensive quantity in chemical process modelling. The energy balance has quite an involved history as it has been the central object of thermodynamics. Many books have been written on the subject of terminology of the thermodynamics and thermostatics and a lot has been talked about the shortcomings of the theories itself. Such a discussion goes beyond the scope of this project and we will simply accept the basic assumptions that form the foundation of the thermodynamic theory.

In the most generic form, total energy is balanced. Total energy is the sum of internal, kinetic and potential energy, all of which are associated with mass. The communication with other systems, though, is not limited to these three forms of energy, but also heat and work affect the energy content of a system.

Particularly heat is an interesting form of energy. Whilst today heat conduction is known to be based on kinetics - energy transfer on the molecular level - early in the last century, heat was thought to be a massless material that could be moved from one body to another. It was named caloric.

Work is an accumulation of various other energy related effects, such as shaft (mechanical) work, volume work, electrical work and others.

A total energy balance for an arbitrary system, assuming reversibility of all processes, may be written in the form:

$$\frac{d}{dt}E = \sum_{\forall m} \alpha_m \hat{E}_m + \sum_{\forall h} \alpha_h \hat{q}_h + \sum_{\forall l} \alpha_l \hat{w}_l$$

with:

E :: Total energy := U + K + P

U :: Internal energy

K :: Kinetic energy

P :: Potential energy

$\hat{E}$ :: Energy stream

$\hat{q}$ :: Heat stream

$\hat{w}$ :: Work stream

The total energy balance is almost never used in this form. Usually a modified version of this basic energy balance is employed for modelling a process component. This modified version is derived from the basic energy balance through some simplifications and assumptions. Caution should be taken, though, when one uses derived energy models, because they are often incorrect or used incorrectly. One could easily introduce faults when one is further simplifying a derived model, because of lack of knowledge about previous assumptions and derivation steps. Knowledge about the common assumptions and the derivation steps is thus essential for the correct use of the different simplified energy models.

In the thesis of dr. ir. M. Westerweele it is shown that the enthalpy balance, which can be derived from the energy balance, can, in most cases, be written as:

$$\frac{dH}{dt} = \sum_{\forall m} \alpha_m \hat{H}_m + \sum_{\forall h} \alpha_h \hat{q}_h + \sum_{\forall k} \alpha_k \hat{w}_k$$

4.3.3 Conclusions

In this section we have derived that the variables that are appearing in the balance equations (i.e. transport and production variables) always appear linearly for physical and chemical systems. The nonlinearities of a process will therefore always emerge in the algebraic relations of the model.

The balance equations can always be abstracted with the following simple form:

$$\frac{d}{dt}\underline{x} = \underline{A}\,\underline{z} + \underline{B}\,\underline{r}$$

In which,

$\underline{A}$ :: Interconnection matrix

$\underline{B}$ :: Stoichiometric matrix,

$\underline{x}$ :: Primary state vector

$\underline{z}$ :: Transport rate vector

$\underline{r}$ :: Reaction rate vector

The matrices $\underline{A}$ and $\underline{B}$ are completely defined by the model designers definition of the physical and species topology of the process under investigation. Therefore these matrices can automatically be constructed by **Mobatec Modeller**. The only things a model designer has to do to complete the model are:

- Provide a link between the transport and reaction rate vectors and the primary state vector. Each element in the transport and reaction rate vectors has to be (directly or indirectly) linked to the primary state vector. This "linking" is done with one or more algebraic equations. If certain elements of the rate vectors are not defined in the algebraic equations, the mathematical system will have too many unknowns and can consequently not be solved.
- Give a mapping which maps the primary state of each system in a secondary state. This mapping is necessary because usually transport and reaction rates are defined as functions of secondary state variables (a heat flow can, for example, be expressed as a function of temperature difference).

4.4 Algebraic Equations

In addition to the balance equations, we need other relationships to express thermodynamic equilibria, reaction rates, transport rates for heat, mass, momentum, and so on. Such additional relationships are needed to complete the mathematical modelling of the process. A model designer should be allowed to choose a particular relationship from a set of alternatives and to connect the selected relationship to a balance equation or to another defined relationship. As with the variables, we divide the algebraic equations into three main classes, namely *system equations*, *connection equations* and *reaction equations*.

4.4.1 System Equations

For each system that is defined within the physical topology of a process, a mapping is needed which maps the primary state variables (*x*) into a set of secondary state variables (*y* = *f*(*x*)). The primary states of a system are fundamental quantities for describing the behaviour of the system. The fundamental state is defined intrinsically through the fundamental behaviour equations. The application of fundamental equations of component mass and energy balances intrinsically defines component mass and energy as the fundamental state variables. Alternative state variables are required for the determination of the transfer rate of extensive quantities and their production/consumption rate.

The equations that define secondary state variables do not have to be written in explicit form, but it has to be possible to solve the equations (either algebraically or numerically) such that the primary state can be mapped into the secondary state. This means that each defined equation has to define a new variable. Equations that link previously defined variables together are not allowed, since the number of equation would then exceed the number of variables and the set of equations of this system would thus be over-determined. Consider the following example:

Example *4.2: Redundant Equations*

It is often seen that people insist on using so-called "normalizing" equations, such as the sum of the fractions equals one, to complete their model definition. Such an equation is actually redundant when you think about it, because the definition of fraction intrinsically implies this:

$$\underline{x} := \frac{\underline{n}}{\underline{e}^T \underline{n}}$$

With

$\underline{x}$ *:: Molar fractions vector*

$\underline{n}$ *:: Molar mass vector*

$\underline{e}$ *:: unity vector =[1 1 1 1.....]^T*

Premultiplying this definition with the transposed unity vector $\underline{e}^T$ *gives:*

$$\underline{e}^T \underline{x} := \frac{\underline{e}^T \underline{n}}{\underline{e}^T \underline{n}} = 1$$

The two presented equations are linearly dependent. So, adding the equation $\underline{e}^T \underline{x} = 1$ *to the model does not add any new information and could actually make computations more* *difficult. With our modelling method, an equation like this is not allowed, since it does not define a new variable.*

□

Due to the nature of the different secondary variables, the system equations are subdivided in some subclasses:

State variable transformations

State variable transformations are relationships that provide links between the internal state of a system and various state variables appearing in kinetic laws, transfer laws, and other definitions such as physical and geometrical properties. For example, the concentration $\underline{c}$ of the components in a system can be characterized by their molar masses $\underline{n}$ divided by the volume of the system V :

$$\underline{c} := \frac{\underline{n}}{V}$$

It should be noted here that, although these equations are called state variable transformations it is not implied that these equations actually have to be used for substitutions and transformations. It is very often seen that modellers insist on doing substitutions, hereby transforming the fundamental balance equations. These often-cumbersome transformations are usually not necessary for solving the problem under investigation and can usually be omitted.

Physical property relations

Processing systems may consist of a variety of materials in the form of pure materials, mixtures, dispersions or any other combination of materials in any state of aggregation. The modelling of these systems potentially requires knowledge about the physical properties of all involved materials. Examples of physical properties are: viscosity, thermal conductivity, diffusivity, partial molar enthalpy and density.

While they are usually thought of as constants, they may change with changing conditions in the system. They can also be a function of geometrical properties and/or indirectly a function of other physical properties. The thermal diffusivity, for example, can be defined as a quotient of the thermal conductivity divided by the density times the heat capacity.

Geometrical property relations

The body volume and the boundary area (surface) of a system are two geometrical properties of a system. These properties can be characterized by alternative sets of geometrical properties. For example, the body volume and boundary surface of a cylindrical system can be characterized by the radius and the length of the cylinder.

Because the system can change its shape as a function of the changing state, the geometrical properties of a system are a function of the systems state, and with it a function of the systems physical properties. For example, increasing the temperature of a system at a constant pressure, can expand the volume and boundary of a system. This can be considered as a result of changing the density (a physical property) of the contents of the system. The volume and mass of a system are always linked through the density.

Equations of state

Equations of state are equations that express algebraic relations between the application variables that characterize an individual system and that are supposed to hold at each moment during the evolution of the process. The term "equations of state" is borrowed from the thermodynamics as it is used, for example, for the well-known relation between the pressure p, the molar volume V_n and the temperature T of an ideal gas:

$$pV_n = RT$$

called the *equation of state of the ideal gas*.

Besides the enumerated types of equations there might be other relations that need to be considered during the modelling process.

4.4.2 Connection Equations

The flow rates, which emerge in the balance equations of a system, represent the transfer of extensive quantities to and from adjacent systems. These flow rates can be specified or linked to transfer laws, which are usually empirical or semi-empirical relationships. These relationships are usually functions of the states, and the physical and geometrical properties of the two connected systems. For example, the rate of conductive heat transfer Q through a surface A_t between two objects with different temperatures can be given by:

$$Q := UA_t(T_{or} - T_{tar})$$

This relationship depends on the temperatures T_{or} and T_{tar} of the origin and target object respectively. Temperature is of course a (secondary) state variable. The rate of heat transfer also depends on the overall heat transfer coefficient U, which is a physical property of the common boundary segment between the two systems, and on the total area of heat transfer At, which is a geometrical property.

A transfer law thus describes the transfer of an extensive quantity between two adjacent systems. The transfer rate usually depends on the state of the two connected systems and the properties of the boundary in between.

4.4.3 Reaction Equations

Depending on the time scale of interest, we can divide reactions into three groups:

- Very slow reactions (slow in the measure of the considered range of time scales). These reactions do not appreciately occur and may be simply ignored.
- Reactions that occur in the time-scale of interest. For these reactions kinetic rate laws can be used.
- Very fast reactions (relative to the considered time scale), for which is assumed that the equilibrium is reached instantaneously.

As the non-reactive parts do not further contribute to the discussion, they are left out in the sequel. For the "normal" reactions the reaction rates of the reactions in the relevant times scale must be defined by kinetic rate equations. The production terms are linked to kinetic laws, which are empirical equations. They are usually written as a function of a set of intensive quantities, such as concentrations, temperature and pressure. For example, the reaction rate *r* of a first-order reaction taking place in a lump is given by:

$$r := V\, k_0\, e^{\frac{-E}{RT}}\, c_A$$

where

r :: Reaction rate of a first-order reaction

V :: Volume of the system

k_0 :: Pre-exponential kinetic constant

E :: Activation energy for the reaction

R :: Ideal gas constant

T :: Temperature of the reacting system

c_A :: Concentration of component *A*

Temperature and concentration(s) of the reactive component(s) are state variables of the reactive system. Reaction constants and their associated parameter such as activation energy and pre-exponential factors are physical properties. In some cases, also geometrical properties of the system are part of the definition of the kinetic law, such as the porosity or other surface characterizing quantities.

The fast (equilibrium) reaction rates are not defined, because the equilibrium reactions are considered to have very fast dynamics relative to the time scale of the process. For these reactions only the reaction outcome has to be given in the form of an equilibrium relation, which should hold at every time instant. This is usually a nonlinear, algebraic relation that relates the masses of the involving species to each other (e.g.: for a reaction $A \Leftrightarrow B + C$ one may write: $K = \frac{c_A}{c_B c_C}$). Consequently, unlike the situation where no equilibrium reactions occur, the initial values of the masses of the involved species cannot be arbitrarily chosen because the quantities of some species in the system are now directly related to the quantities of some other species in the system. This results in some differential equations of the system being directly related to each other. Also, the production terms of the equilibrium reactions occur only in the component mass balances and cannot be determined directly from system equations. As with the undefined connections, the result is a high index model. The problems and solutions regarding high index models are handled in the thesis of dr. ir. Westerweele.

For each defined reaction a modeller must either specify the kinetics or give an equilibrium relation.

4.5 Substitution, yes or no?

It is often seen that model designers insist on eliminating the extensive variables from the model equations. The main reason that is brought up for this preference to write a model that does not involve the extensive variables is that often only the evolution of the application variables is of interest. Also, the transfer laws and kinetic laws are usually given in terms of intensive state variables. Therefore most model designers think they must transform the accumulation terms and perform a so-called state variable transformation. Most textbooks which cover modelling also perform these transformations, often even without mentioning why. But are these, often cumbersome, state variable transformations necessary to solve the considered problems?

In most cases, the transformations are not necessary. There are several reasons to consider the differential algebraic equations (DAEs) directly, rather than to try to rewrite them as a set of ordinary differential equations (ODEs): First, when modelling physical processes, the model takes the form of a DAE, depicting a collection of relationships between variables of interest and some of their derivatives. These relationships may be generated by a modelling program (such as **Mobatec Modeller**). In that case, or in the case of highly nonlinear models, it may be time consuming or even impossible to obtain an explicit model. Computational causality is not a physical phenomenon, so it is rather inconvenient if a model designer has to determine the (numerically) correct causality of the equations. Also, reformulation of the model equations tends to reduce the expressiveness. Furthermore, if the original DAE can be solved directly it becomes easier to interface modelling software directly with design software. Finally, reformulation slows down the development of complex process models, since it must be repeated each time the model is altered, and therefore it is easier to solve the DAE directly.

These advantages enable researchers to focus their attention on the physical problem of interest. There are also numerical reasons for considering DAEs. The change to explicit form, even if possible, can destroy sparsity and prevent the exploitation of system structure.

Small advantages of transforming the model to ODE form can be that for (very) small systems an analytical solution is available and that sometimes less information of physical properties is needed when substitutions are being made (sometimes, some of the parameters can be removed from the system equations when substitutions are made). Another advantage could be that, by doing substitutions, some primary state variables are removed from the model description which could take the code faster, because less variables have to be solved. In general, though, these advantages do not outweigh the disadvantages.

If one does want to perform substitutions, it is recommended that these are done at the very end of the model development and not, as is generally seen, as soon as possible. Postponing the substitutions as long as possible gives a much better insight in the model structure during model development.

4.6 Control

Control is added in the last step of our modelling methodology and can be seen “separately” from the previous steps, since for most models that are meant for control, first the model without the controllers is constructed. The control part is usually added in a later step and can be superimposed on the model without affecting the previous modelling steps. Controllers process measurement and setpoint information. The first taken from the process state, the latter being an input to the process and determining its desired behaviour. Control generates (information) signals as output, which in general affect the flow of extensive quantities inside the process.

Adding a controller to a process has only one goal, namely to modify the dynamic behaviour to the process to be controlled, with the objective of imposing a desired behaviour. The controller may be introduced for different reasons. The controller may force the process to follow a trajectory (the servo or steering problem) or the controller may serve the purpose of compensating for undesired effects of the environment has on the process (disturbance rejection).

Whatever the control objectives are, there is always a need to monitor the performance of the process that needs to be controlled. This is done by measuring the values of certain process variables (such as concentrations, fluid height, temperatures, flow rates, pressures, etc.). The measurements are processed by the controller, which then steers some input variables in order to control the process. Usually a process has a number of available input variables which can be adjusted freely. Which ones are selected to use as input (i.e. manipulatable) variables is a crucial question, as the choice will affect the quality of the control actions we take.

Clearly, there is a flow of information to and from the controller and therefore two new elements need to be introduced, namely the *information system* (i.e. the controller itself) and the *information connection* (through which the information "flows": The target object of the information connection reads variables of the origin object).

The input (i.e. the measurement) to the information system can, in principle, be any time dependent variable of the process. Consequently, the information connection that provides the input to the information system can be connected to any system (either lumped, distributed, source, sink, steady state or composite), any connection or any other information system (cascade control, providing "new" setpoints) of the physical topology of the process. Most of the time, however, the measurements are closely related to the primary states of a process and the information will come from lumped systems.

The controller receives the measurement information and decides what action should be taken. The variables that can be manipulated by the controller are usually some connection characteristics, such as the position of a valve. The control valve is the most frequently encountered final control element, but not the only one. Other typical final control elements for chemical processes are: relay switches (providing on-off control), variable-speed pumps and variable-speed compressors.

From a physical point of view, the variables that can be manipulated will always be connection variables and never system or reaction variables, since a controller cannot change the state of a system directly. It can only manipulate the flow of extensive quantities, which in turn affect the state of the two interconnected systems. Sometimes, however, either for convenience or as a simplification, a controller is made to directly affect some secondary state variables of a source or sink system. A controller that directly influences the temperature of a heat stream, for example, is physically impossible but often seen in modelling (especially if the temperature of the stream can be adjusted relatively quickly compared to the dynamics of the controlled system). The output of an information system can therefore be connected to any connection, any source or sink system or any other information system.

The possible information flows are summarized in figure 4.1.

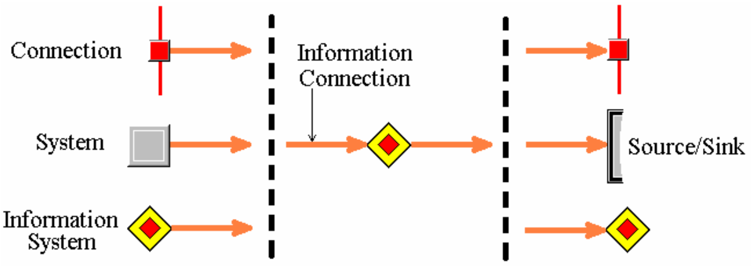

Figure 4.1: *Possible information flows to and from information systems.*

Controllers are dynamic elements. The controller equations can consist of algebraic, differential and integral equations and since the controller normally has a state, the order of the overall model is increased by adding control elements to the physical topology of the process.

Further discussion of control, which obviously has a lot more to it than the few things that are mentioned in this brief section, is not considered to be relevant. It is left to the reader to find out more about the numerous subjects on control, such as feedback control, feedforward control, inferential control, PID, LQR, LQG, H2, H∞, state feedback, cascade control, etc.. All these types of controllers can be implemented in an information system, as long as they are realizable and presented in a DAE form.

4.7 Implementation Details

The construction of the physical topology and species topology of a process are rather straight forward and do, in principle, not need any specific close attention. When introducing the equations into the model, we are faced with some non-trivialities that need some closer look.

This section discusses the most important features of the implementation of the construction of the equation topology.

4.7.1 Equation Classification

The algebraic equations, that are needed to complete the model definition, where divided into three main groups in section 4.4. **Mobatec Modeller** contains a database in which a set of possible algebraic equations is listed. In this database, which can, of course, be further expanded by a model designer, the main groups are further subdivided. Careful consideration should be given, though, were new algebraic equations are placed. An algebraic equation can be placed in one of the following groups (the groups that actually contain the equations are bold faced and underlined):

- **System equations**
- Connection Equations
  - Mass Connections
    - Uni-Directional
      - **Unmodeled / fast**
      - **Rate**
    - Bi-Directional
      - **Unmodeled / fast**
      - **Rate**
  - Phase Transition Connections
    - **Unmodeled / fast**
    - **Rate**
  - Heat Connections
    - **Unmodeled / fast**
    - **Rate**
  - Work Connections

    - **Unmodeled / fast**
    - **Rate**

- Reaction Equations
    - **Unmodeled / fast**
    - **Rate**

- **Species Equations**

- **Control / Information Equations**

Two "new" groups were added to the main groups, namely species and control/information equations.

The species equations are equations that represent physical properties of species (or groups of species). Most of the time, these equations would be assigned to the group of systems equations, but because these equations often hold independent of the system and can be easily coupled to a species database, a separate group is defined.

Control action usually affects some connection characteristics (such as the position of a valve), but sometimes, for convenience or as a simplification, a controller directly affects the secondary state variables of a source or sink system (for example, when a controller directly controls the temperature of a stream, which is, of course, physically impossible, but often seen in modelling). Also, the control equations are usually very different from the other algebraic equations, since they can also hold differential and integral terms. For these reasons the control systems and connections where defined in section 4.6 and their equations are placed in a separate group.

In some cases a system or connection needs "information" of some variables of another system or connection in order to calculate specific quantities. For example, in a tank with a liquid and a gas phase, the gas phase volume is usually defined as the (fixed) volume of the tank minus the liquid volume. So in order to calculate the gas volume, the system (or systems) representing the gas phase needs information (i.e. the liquid phase volume) from the system (or systems) representing the liquid phase. This information can be transferred via information connections.

4.7.2 Modeling Objects

There are 3 "modelling objects" defined within **Mobatec Modeller**, namely *Systems*, *Connections* and *Reactions*. Algebraic equations need to be added to these objects to complete the model that is being made (keeping in mind that the differential equations (i.e. the balance equations) are generated automatically).

Note: *Species* is not listed as a modelling object, since no algebraic equations can be appointed to species. Species can only hold parameters. The algebraic equations that use these species parameters are always calculated in one of the modelling objects. In other cases, no species parameters are even needed, because the physical properties are calculated in external physical property routines.

When a model designer is constructing a process model, the presented equation classification comes in handy, because for each modelling object only a limited range of equations can be chosen and this reduces the chances of making silly mistakes. When, for example, a mass connection is defined as being bidirectional, the model designer *must* choose if he wants to define the flow rate through the connection or if he wishes to leave this flow unmodeled. Either way, he is directed to a set of possibilities: a set of possible rate equations or a set of possible constraints. This way the modeller becomes more aware of what assumptions he makes. Also, the assumptions are automatically documented and very easily traced and thus very easily changed.

4.7.3 Computational Order of Equations

In the fourth and fifth steps of our modelling approach the algebraic equations are added to the model definition. For each modelling object (i.e. system, connection or reaction) these algebraic equations can, in principle, be chosen randomly from the database. In doing so, the problem arises that not every numerical equation solver will be able to solve the equations since the equations are not in the so-called correct computational order and are not always in (correct) explicit form. There are solvers, such as DAE-1 solvers, that can handle implicit algebraic equations, but when the equations are simplified by performing preliminary symbolic manipulations, a much more efficient computational code can be obtained. A very important step to achieve an efficient computational code for DAEs, is to solve the equations for as many algebraic variables as possible, so that it is not necessary to introduce these variables as unknowns in the numerical DAE-solver, since they can be calculated at any call of the residual routine from the information available.

Consider the simple pair of equations:

$$x_2 - 2x_1 = 4 \tag{1}$$

$$x_1 - 7 = 0 \tag{2}$$

In order to solve these equations directly, they must be rearranged into the form:

$$x_1 = 7 \tag{3}$$

$$x_2 = 2x_1 + 4 \qquad (4)$$

The implicit equation (2) cannot readily be solved for x_1 by a numerical program, whilst the explicit form, namely (3), is easily solved for x_1 and only requires the evaluation of the right-hand-side expression. Equation (1) is rearranged to give (4) for x_2, so that when x_1 is known, x_2 can be calculated.

The rearranged form of the set of equations can be solved *directly* because it has the correct *computational causality*. This computational causality is, quite obviously, not a physical phenomenon, but a numerical artefact. Take, for example, the ideal gas law:

$$pV = nRT$$

This is a static relation, which holds for any ideal gas. This equation does not describe a cause-and-effect relation. The law is completely impartial with respect to the question whether at constant temperature and constant molar mass a rise in pressure causes the volume of the gas to decrease or whether a decrease in volume causes the pressure to rise. For a solving program, however, it *does* matter whether the volume or the pressure is calculated from this equation.

It is rather inconvenient that a model designer must determine the correct computational causality of all the algebraic equations that belong to each modelling object, given a particular use of the model (simulation, design, etc.). It is much easier if the equations could just be described in terms of their physical relevance and that a computer program automatically determines the desired causality of each equation and solves each of the equations for the desired variable, for example by means of symbolic manipulation.

Whether the entered equations are in the correct causal form or not, they always have to adhere to some conditions:

- For any set or equations to be solvable, there must be exactly as many unknowns as equations.
- It must be possible to rearrange the equations such that the system of equations can be solved for all unknowns.

The first condition, called the Regularity Assumption, is obviously a necessary condition. It can be checked immediately and all numerical DAE solvers take this preliminary check.

In order to solve a set of equations efficiently, the equations must be rearranged in Block Lower Triangular (BLT) form with minimal blocks, that can be solved in a nearly explicit forward.

When a DAE-1 model is formulated and proper initial conditions have been defined, then the information flow of a simulation can be depicted as in figure 4.2. Starting from the initial conditions $\underline{x}_0$, the secondary state variables $\underline{y}$ of all the systems can be calculated. Subsequently, the flow rates $\underline{z}$ of all the defined connections and the reaction rates $\underline{r}$ of all the defined reactions can be calculated. These rates are the inputs of the balance equations, so now the integrator can compute values for the primary state variables $\underline{x}$ on the next time step. With these variables, the secondary state $\underline{y}$ can be calculated again and the loop continues until the defined end time is reached.

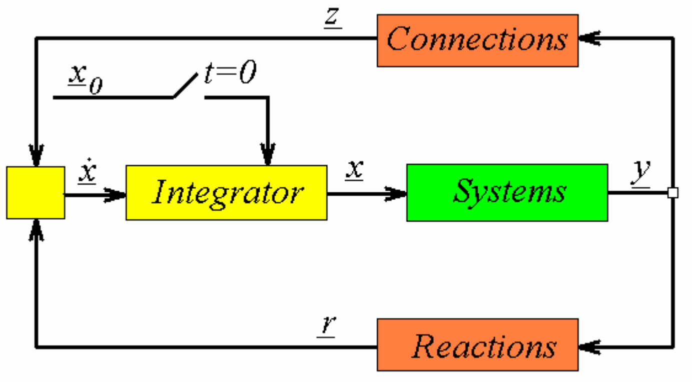

Figure 4.2: *Information flow for a simulation.*

For now, we are only concerned with the algebraic equations (the right hand side of figure 4.2), which means that we can consider the primary state variables $\underline{x}$ of each system as known. Systems are only interacting with each other through connections and therefore the calculation of the secondary variables of each system can be done completely independent of other systems.

The system equations map the primary state into a secondary state for each individual elementary system and each new equation has to define a new (secondary) variable. In some cases two or more equations may introduce two or more new variables, such that these equations have to be solved simultaneously in order to get a value for the variables. This does, however, not occur very frequently.

For connection and reaction equations a similar conclusion can be drawn. For these equations the secondary variables of the systems ($\underline{y}$) can be considered as known (see figure 4.2).

Although it is good to know about computational causality, a model designer does, in principle, not have to worry about BLT forms, because **Mobatec Modeller** handles this automatically.

4.7.4 Adding Equations to a Modeling Object

If equations need to be added to a specific modelling object, select this object and go to the equations tab. The equations tab is subdivided in five tabs, namely*: Defined Equations*, *Sorted Equations*, *Parameters*, *Initial Values* and *Object Class.* On the Defined Equations tab press "Define Equations" and a database of available equations (for the selected object) will be shown. Select the equations that need to be added from this (or any other) database and press "OK". The selected equations will appear in the defined equations dialog box.

The Sorted Equations tab shows all equations that can be solved in the correct solving order (BLT form). In general, when selecting equations, there will be more variables than equations, so not all equations will be shown at first. This means that some of the available variables will have to be set "constant" in order to get a solvable set of equations (in which the number of unknowns equals the number of equations) for the object under consideration. Consider the following simple example:

Example 4.3*: Defining Concentration in a System*

Consider a system with a number of components. If the concentration of those components needs to be known, the following equations could be defined in ***Mobatec Modeller****:*

$$n[\,] = c[\,] * V$$
$$nt = rho * V$$
$$nt = sum(n[\,])$$

In which c[ ] is the vector of component concentrations, n[ ] is the vector of component hold-ups, V is the system volume, nt is the total molar hold-up and rho is the density.

This set of 3 equations contains 5 variables (n[ ], c[ ], V, nt and rho). In the presented modelling methodology all dynamic variables should be either directly or indirectly related to the fundamental extensive quantities (i.e. the primary state variables for which the balance equations are setup). These fundamental extensive quantities are considered to be "known" (i.e. "computed" via the differential equations).

So, for this example the molar hold-up n[ ] is "known", which means that the total hold-up nt can be solved from the last equation. The Defined Equations tab will therefore show only one equation: $nt = sum(n[\,])$ *as a "sorted equation". The "Computed & Knowns" will hold n[ ] and nt. The remaining variables (c[ ], V and rho) will be listed in the "Selectable" list.*

If the density rho of the mixture is considered to be constant, it can be selected from the "Computed & Selectable" list on the Defined Equations tab. By doing this, the variable will be added to the "Constants" list and will be considered a parameter that is known. Consequently, ***Mobatec Modeller*** *will detect that the volume V can now be calculated and* *with this also the concentration c. The “Sorted Equations” will be shown in the correct computational order:*

$$\begin{aligned} nt:&\quad nt = sum(n[]) \\ V:&\quad nt = rho*V \\ c[]:&\quad n[] = c[]*V \end{aligned}$$

The “Computed & Knowns” will list n[ ], c[ ], V and nt. The “Constants” variables will list only the density rho. The number of “defined” equations is now equal to the number of “solvable” equations.

□

After identifying all parameters of a modelling object (and thus making it structurally solvable), the parameters as well as the initial values need to be specified. If this is done for all modelling objects, **Mobatec Modeller** can generate output code that can serve as an input to existing problem solvers (such as e-Modeler (Process Studio), Matlab, ACM, etc.).

4.7.5 Variable Prefixes in Equations

Several “prefixes” of variables in the definition equations are used in **Mobatec Modeller** to aid the user in defining relations that can be used generally. For example, when you want to make an equation that describes the heat flow as a function of the temperature difference of the two interconnected systems, you’ll need to make an equation that looks like this: "*q = U*A*(or.T - tar.T)*".

In this equation the variables *q*, *U* and *A* "belong" to the connection and don't need a prefix. The temperatures *T* refer to the origin and target of the connection and therefore need the prefix "***or***." and "***tar***." respectively.

The prefix "***glob***." is used to refer to a variable (i.e. a parameter) that is used globally and which is the same in the entire model.

Prefix "***spec***." is used when an equation refers to a variable that is specific to a certain species. For example the specific heat *Cp* or specific density *Rho* of a species.

Prefix "***sys***." is used when defining Reaction Equations to refer to variables that are defined in the system the reaction is taking place in. If no prefix is used, when defining a reaction equation, the variable will belong to the reaction.

For example, lets define an equation that defines the first order reaction rate for a reaction *A*➔ *B*. Mathematically one writes: *r = k * c_A * V*.

In this case the variables *r* and *k* belong to the specific reaction and the concentration of component A and the volume are defined in the system the reaction is taking place in. The translation would look something like:

*r = k*sys.c[1]*sys.V*

Where *sys.c[1]* refers to a specific element of the vector variable *c[ ]* (Component A in this case). The specific component should be selected when you install the equation in a specific reaction.

4.7.6 Operators and Functions

Mobatec Modeller allows to construct dynamic process models that are independent of the solver with which the model is going to be solved. In order to achieve this, a syntax has to be defined for representing the most used operators and functions.

The following operators and functions can be used in **Mobatec Modeller:**

Function	**Result**
x + y	Adds x to y
x – y	Subtracts x from y
-x	Minus x
x * y	Multiplies x with y
x / y	Divides x by y
x^y	x to the power of y
x[ ] .* y[ ]	Dotproduct of vectors x[ ] and y[ ] (i.e. x[ ] .* y[ ] = x_1*y_1 + x_2*y_2 + ….)
SUM(x[ ])	Sum of vector x[ ] (i.e. SUM(x[ ]) = x_1 + x_2 + x_3 + ….)
SIN(x)	Calculates sine of x
COS(x)	Calculates cosine of x
TAN(x)	Calculates tangent of x
ddt(x)	Calculates the derivative with respect to time of argument x
SQRT(x)	Calculates the square root of x
ABS(x)	Determines the absolute of x
SIGN(x)	Yields 1, 0 or –1 based on the sign of x
EXP(x)	Calculates the exponent (base e) of x
LN(x)	Calculates the natural (base e) logarithm of x
LOG(x)	Calculates the base 10 logarithm of x
MAX(x, y)	Returns the maximum of x and y
MIN(x,y)	Returns the minimum of x and y
PI	Number Pi = 3.1415

This list of available functions can and will be expanded in future releases.

4.7.7 Boolean Expressions

Very often it is necessary to introduce conditional statements in the form of Boolean expressions into the model definition. For example, when a pipeline has a check valve in it, the gas flow through this line could be pressure driven when the pressure drop is positive. The flowrate should be zero, however, when the pressure drop is negative. When this example is modeled as uni-directional mass connection between two systems (the origin and the target), the used equation could have the following form:

```
$IF\ or.P > tar.P\ THEN$
  $Fv = KV * (or.P - tar.P)$
$ELSE$
  $Fv = 0$
$END$
```

In which *KV* is a measure for the resistance of the line and *Fv* is the volumetric flowrate through the line.

A Boolean expression can, of course, also be nested.

The following relational and logical grammar is used within **Mobatec Modeller:**

String	**Description**
IF	If; compulsory part of conditional statement
THEN	Then; compulsory part of conditional statement
ELSE	Else; compulsory part of conditional statement
END	End; compulsory part of conditional statement
==	Equal to; used within Boolean expression to check equality
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
#	Not equal to
!=	Not equal to
AND	And
OR	Or
NOT	And not