# INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

**The quality of this reproduction is dependent upon the quality of the copy submitted.** Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

# NOTE TO USERS

**This reproduction is the best copy available.**

UMI®

# A Hierarchical Approach to Dynamic Modeling and Simulation of Complex Processes

by

Hong Tu

A Dissertation Submitted to the Graduate Faculty in Engineering in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy, The City University of New York

2004

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

7/29/03

Date

Prof. Irven H. Rinard

Chair of Examining Committee

7/29/03

Date

Mumtaz K. Kassir

Prof. Mumtaz K. Kassir

Executive Officer

Prof. Morton Denn

Prof. Alexander Couzis

Prof. Jae W. Lee

Dr. Bruce Benjamin

Supervisory Committee

The City University of New York

# Abstract

**A Hierarchical Approach to**

**Dynamic Modeling and Simulation of Complex Processes**

**by**

**Hong Tu**

**Advisor: Professor Irven H. Rinard**

A hierarchical and object-oriented dynamic modeling and simulation system prototype called *ForeSee* has been developed, which demonstrates some of the key aspects of four sets of modeling components to represent the dynamic modeling and simulation of chemical processes. It is based on an object-oriented design of the software and programmed in Java.

There are three key ideas involved in this simulator. The first is that while each of chemical processes is functionally different from the other, if we look at the fundamental components of each, these are quite similar in their physical behavior. This means that simulation models of complex processes can be built up hierarchically starting with relatively simple models of these fundamental components. The second is that the number of fundamental components that are required is relatively small. This confers a number of advantages in the software design of the simulator as well as the technology transfer to the user. The third is that each of these fundamental components can be represented by an icon and that equipment models can be assembled from these fundamental components via a graphical drag-and-drop interface.

In a similar manner higher level models of complex processes can be assembled from the equipment models. While many simulators have the latter capability, none that we know of permit the direct assembly of equipment models. The graphical creation of equipment models also confers a number of advantages with respect of model visualization and verifiability.

These fundamental modeling components can be categorized into four types (*containments, core models, connectors,* and *coordinators*), each of which describes a different aspect of system behavior. It is in the representation of equipment models that ForeSee differs most strongly from all the other simulation systems. In these systems an equipment model consists of a set equations and the associated codes for their execution. In ForeSee an equipment model is specified as an interconnected set of modeling components. The basic premise in this approach to modeling is that a relatively small set of modeling components can be used to generate a variety of equipment models, and in turn a lot of higher level models of complex processes can be assembled from the equipment models.

# Acknowledgements

I would like to express my deepest appreciation and gratitude to Professor Irven H. Rinard for his continuous guidance, inspiration and contribution at all times during the course of this research work.

I would like to thank Professor Morton Denn, Professor Alexander Couzis, Professor Jae W. Lee, Dr. Bruce Benjamin, Professor David S. Rumschitzki, and Professor Izidor Gertner for their help and encouragement.

I am grateful to Andrew Eng, Zhen Rong Xu, and all other members of the faculty and staff for their assistance during my study at the department.

I am deeply indebted to my parents, my wife Xiaodong, and my son Jingwei. They have made many sacrifices to give me tremendous supports. I will always be grateful to them for their encouragement and supports. This work is dedicated to them, for their love, patience and encouragement.

*To my wife Xiaodong and my son Jingwei*

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1 Introduction

Dynamic modeling and simulation is of growing importance in process systems engineering and process operations analysis. This is driven by increasing concerns for maintaining product quality within stringent limits, ensuring safe and smooth operation, and not violating constraints on environmental emissions.

To be effective in achieving these goals, dynamic modeling and simulation must be applied concurrently in the design phase of the project. Unfortunately, this has not been the case heretofore. Dynamic modeling and simulation has been a very time-consuming and labor intensive activity, one requiring highly skilled systems engineers and computer applications specialists. Although there are now commercially available dynamic simulators, few, if any, possess a library of equipment models extensive enough to satisfy the needs of all but the most routine projects. If one has to stop to develop a new equipment model, this will take time and skilled personnel and subject the project to the vagaries of software development.

In addition, simulators for which each equipment model is custom developed are large in terms of the amount of code involved, can be difficult to maintain and modify, and are time-consuming to learn and apply. Clearly, a more efficient approach to this problem is needed. The demands for timeliness and reliability will only increase as the focus of industry shifts from commodity chemicals to fine chemicals and pharmaceuticals.

The purpose of this research has been to develop the appropriate methodology and to create such a simulator. It is based on a hierarchical approach to dynamic modeling and simulation of complex processes. Using this approach a hierarchical and object-oriented dynamic modeling and

simulation environment called *ForeSee* has been developed.

ForeSee is a dynamic modeling and simulation environment intended for the modeling and simulation of a wide variety of complex chemical engineering processing systems. It is based on an object-oriented design of the software and programmed in Java to insure multi-platform portability. It is a hierarchical system in which a large variety of processing systems can be modeled and simulated using a large reusable set of interconnected equipment models that are in turn modeled using a compact and readily comprehensible set of four types of modeling components. The four component level models are *containments, core models, connectors* and *coordinators*. These are represented graphically to the end user rather than as a set of equations. The equipment model can be created by the selection, parameterization and aggregation of the component level models using a graphical drag-and-drop interface. The basic premise in this approach to modeling and simulation is that a relatively small set of component models can be used to generate a wide variety of equipment models, and in turn a lot of higher level models of complex processes can be assembled from the equipment models. We believe that a number of advantages will result from this approach:

- All the basic model equations used by the simulator are precoded in the component level models. The user needs only to understand what these component models do and how to structure them to create a wide variety of equipment models. This eliminates the need for the user to derive model equations and code them, which are both time consuming and error prone.
- If this set of component level models is as compact as we believe, then the simulator itself will be compact. This will minimize the amount of coding that must be done to create the simulator as well as the effort

required to modify and maintain it. What is more important to the user is that the effort to learn the simulator will be greatly reduced and simplified, particularly because of the graphical nature by which models are created and represented. ForeSee demonstrates some of the key aspects of the use of hierarchical modeling to represent the dynamics of chemical process equipment. It supports the graphical drag-and-drop interface for creating both equipment models and assembling them into a flowsheet.

In recent years, the industrial interest in techniques and software packages for dynamic process modeling, simulation and analysis has been growing steadily. Dynamic simulators have been used routinely to design control systems, evaluate process operability, and train engineers and operators. But all too often these applications have become available after the plant has been designed and built, thereby limiting the benefits that might be obtained. In cases where a dynamic simulator has been available before the plant design is frozen, operational difficulties have been identified and corrected, thereby avoiding expensive retrofits after the plant is built.

Various factors have contributed to the growing demand for better dynamic modeling and simulation capabilities. One of them is the need for improvement of chemical productions in a very global sense, which includes safety and risk analysis, the achievement of low and guaranteed limits on concentrations of chemicals emissions, and the need for high and reproducible product qualities. Another factor is the pressure to reduce the time and expense involved in process and product development in face of global competition.

Environmental and safety regulations, growing demands on product quality as well as increasingly competitive markets necessitate the continuous improvement of chemical processes in minimal time at minimal

cost. One approach towards improved processes is a detailed evaluation of a larger number of flowsheet alternatives during the process development and conceptual design phase. Another strategy builds on an increase in the experimental efforts to support the selection of unit operations, their scale-up or the development of process operation policies and control system. In general, both approaches contradict the objective of minimizing the development time to respond to the needs of the markets with minimum delay. In many cases, development time and reasonable specific product cost decide whether decent profits are possible. Hence, theoretical or experimental evaluations of process alternatives are only possible within tight time and cost constraints [Marquardt 1991].

The bottom line is that model families with varying degrees of detail are required for every unit operation in order to meet the needs of the diverse process engineering activities. The development of such model families is a major undertaking that needs to be supported to the extent possible by information technology. Their proper application is demanding and therefore needs to be effectively supported by software tools. Such modeling tools and model libraries should form self-contained modules to be integrated into open computer-aided process engineering environments. Model representation must allow various kinds of interpretation such as steady-state and dynamic simulation as well as optimization to effectively support model-based analysis and synthesis of processes and their control system.

Although mathematical models of processes are now widely used at all levels from process synthesis and process design through operations planning to process control and monitoring and while there are some simulation packages available to assist the process engineers in these tasks, much remains to be done, particularly in the area of dynamic modeling. There are

several reasons for the immature state of dynamic modeling. There is wide agreement among chemical engineers as to what the appropriate steady-state models are for most of the widely used items of equipment. This is not true for dynamic models, partly for historical under-emphasis and partly for model complexity and fidelity considerations. The net result is that while there may be one steady-state model for a particular item of equipment, there may be several dynamic models, depending upon the degree of model fidelity required. Several measures can used to characterize model complexity, for instance, number of equations, DAEs index, stiffness ratio, real time factor, nonlinearity, discontinuity, and so on. Therefore, with the increasing sophistication of applications and the resulting models, it has become clear that the formulation of appropriate consistent models is a major, time-consuming and error-prone task. If the simulator has an adequate library of dynamic equipment models, which can be configured to represent a processing system flowsheet, carrying out the simulation might be mostly a matter of configuration by means of visual manipulation in the simulator. However, if we have to develop a model, either because none exists for the equipment type of interest or because the existing models are inadequate to the task at hand, the simulation task becomes much more formidable.

Because quick turn around is essential in an engineering environment and most chemical engineers do not have expertise in dynamic simulation, engineers must have a dynamic flowsheet simulation package which is easy to learn and use and allows simulations to be quickly built and changed. We believe that the simulator must be interactive, require no programming, and not be overly complex. The ideal dynamic simulator should be configurable, user-friendly, flexible, portable, readily accessible, fast, and inexpensive. Such an ideal dynamic simulation tool would allow process engineers and

control engineers to routinely develop rigorous, nonlinear, dynamic simulations for most processes. It would allow us to study the impact of process design and controller structure on the operation of not just different unit operations, but entire processing units and even plants. Indeed, we argue that convenience and ease of use are the key issues in the design of such modeling tools, and it is from this angle that current and future progress in this area should be judged [Vogel 1991; Tyreus 1992].

We argue that an advanced simulator must provide capabilities for dynamic and steady-state analysis in one unifying framework. There is a need for an integrated tool supporting the solution of all kinds of process systems engineering problems that include modeling, steady-state and transient analysis, optimization and control for process design and operation.

The simulator must have sufficient capability so that the user does not have to do any programming to develop a simulation. The needed modeling tools must be readily available to solve typical problems. Physical properties for common chemicals, unit operations, and control algorithms should all be available from databases or libraries. Deriving equations and programming requires a higher level of expertise that most users do not have. A simulator that only provides an environment for integrating differential equations the user enters is discouraged.

Although there may be several concerns about dynamic simulation (for instance, model fidelity and computational efficiency), the main concern of this research is with the modeling aspects of dynamic simulation. While each of chemical processes is functionally quite different from the others, if we look at the fundamental components of each, these are quite similar in their physical behavior. This means that simulation models of complex processes can be built up hierarchically starting with relatively simple models of these

fundamental components. If the number of fundamental components that are required is relatively small and if a wide variety of complex chemical equipment models can be generated from these relatively small sets of fundamental, then this approach is viable.

# Chapter 2 Literature Survey

## 2.1 Introduction

The state of the art as well as future trends in process modeling and simulation have been reviewed in a number of recent contributions from different perspectives [Marquardt 1991, 1996; Pantelides & Barton 1993; Stephanopoulos 1987; Shacham et al. 1982]. As more and more methodologies and numerical algorithms become available, modeling is expected to become the major bottleneck in the widespread use of model-based techniques in industrial practice. Therefore, we will focus largely on computer-aided mathematical modeling regardless of the type of application with emphasis on concepts for the structuring of process models and of the modeling process.

A natural extension of the modularization on the flowsheet level is a partition of the unit operations in submodels. These submodels can then be used in different composite models, which simplify modeling considerably. The idea of modularization at the unit operation level has gained increasing interests in the last years.

This experience has triggered considerable effort in recent years. All these attempts aim at facilitating model development and maintenance by enhancing the capabilities for model formulation, reuse and adaptation as well as for maintenance and documentation.

## 2.2 Modeling Elements

While each of chemical processes is functionally quite different from

the others, there are quite a lot of similarities in their structural behavior.

### 2.2.1 Networks

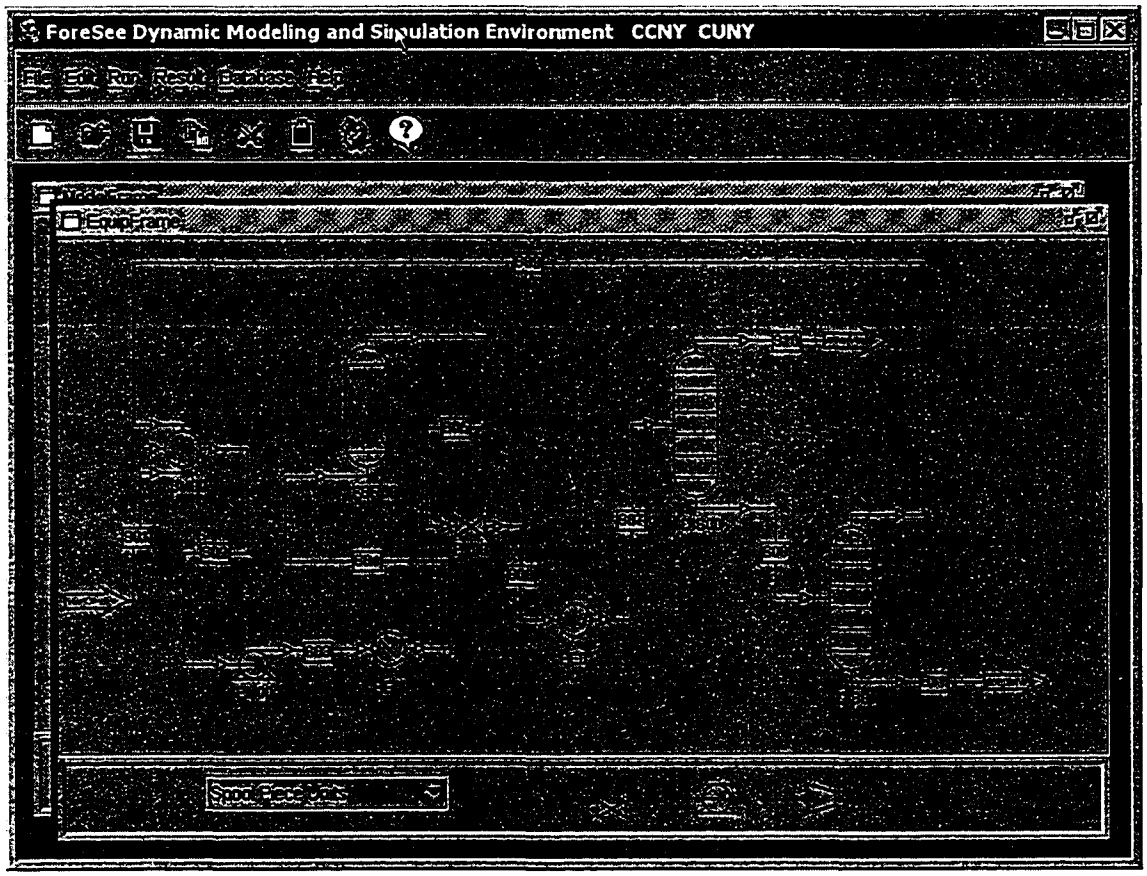



Figure 2-1 The Flowsheet Example for Chemical Process

Process design draws extensively from flowsheeting with the flowsheet being the graphical representation of the plant system. With the two building blocks of devices (nodes) and connections (links), the traditional flowsheet representation was put into the background and was replaced by a network of devices and connections. This network, which now carries the label *physical topology*, is the basic structure on which all the other parts of the model build. A diagram serves the purpose of a graphical representation of these

basic networks, with the vertices being the devices and the arcs representing the connections [Preisig 1995].

This structurization of unit operation is motivated by the idea that mass, energy and momentum are transferred between devices because of potential differences (e.g., of concentration, temperature and pressure). In the connections the fluxes of mass, energy and momentum are then easily calculated from these potential differences.

The typical flowsheet example for chemical process is shown in Figure 2-1.

### 2.2.2 Devices

Device is the physical topology of a plant unit and represents a delimitable part of a process such as a heat exchanger or a distillation column [Bogusch & Marquardt 1997]. The major conceptual distinction between devices and connections is their roles in a real process. The role of a device is the determination of some vector of characterizing state variables ($\mathbf{x}$) like pressure, temperature and composition, etc. from known fluxes ($\phi$) like mass, energy or momentum flows from the surroundings of device. Hence, the device responds to flux information by providing state information. In contrast, the role of a connection is the transformation of a driving force (e.g., a difference in some potential) determined by the known states of two adjacent devices into a flux. Complementary to a device, a connection responds to state information with flux information. Consequently, in dynamic modeling only devices but not connections may display a holdup for extensible quantities. Unfortunately, there is no sharp distinction between devices and connections. Instead it depends on the

knowledge of the modeler and the modeling context, whether a submodel is represented as a device and a connection. To illustrate the concept let us look at a process pipe, which might be abstracted either as a connection or as a device depending on the relevance of the pipe dynamics and the modeling context.

### 2.2.3 Connections

Connections represent communication paths between parts of the overall system. They always connect two devices (i.e. source and sink). A connection being attached to only one device is not allowed. The consequence is that the global system is always closed.

Connections are defined as separate objects. They are not part of either of the two connected devices. Connections also do not represent the dynamics of process but describe the transfer of *extensive* quantities through a boundary assuming pseudo-steady state for the physical system associated with the actual transfer. The transport law, which is the main component of the mathematical representation of a connection, is generally a function of the states of the two devices.

### 2.2.4 Phases

Phases are contained in devices. Clearly, the material must fill the device, so the sum of the phases present must equal the volume of the device. A basic premise is that all material undergoing processing is instantaneously in a stable thermodynamic state, which implies that this state is describable in terms of a finite set of state variables and uniform temperature, pressure,

and composition throughout. It further implies that all material is present in well-defined thermodynamic phases, and hence that any process can ultimately be defined as a collection of interacting phases.

### 2.2.5 Streams

A stream is also a special connection. A stream can be thought of as a collection of one or more substreams flowing together through the same pipe or vessel. For instance, if a stream consists of both vapor and liquid, it is known as a two-substream stream. In a fluidized bed a small amount of the solids are entrained by the gas stream leaving the bed. This stream contains two substreams, one the gas by itself, the other the entrained solids. A liquid-liquid decanter is used to separate a stream containing two immiscible liquids, each liquid being a substream of the feed stream.

### 2.2.6 Ports

A port is a portion of the boundary for a device through which material, energy or signals can flow. A port establishes connectivity between process devices. Flow can occur from one piece of device to another only if there are ports that couple the devices. The notion of a port is consistent with the fundamental chemical engineering concept of a boundary envelope [Hangos & Cameron 1997, 2001].

- Process device can handle multiple ports; each port is distinguished by a name.
- There are three port types: material ports, energy ports and signal ports. A material port transfers chemical substances. An energy

port transfer energy in the absence of any material flow. A signal port conveys information. A signal may be associated with a material flow or energy flow; however, the principle purpose of a signal port is to convey information.

- A port has a normal flow direction that indicates the intended direction of flow. A port with an inlet flow direction normally fills its associated equipment; a port with an outlet flow empties its equipment. A port with a static flow does not ordinarily have flow except for special circumstances (such as a relief valve).
- A port connection is a connection between exactly two ports of the same port type. One port has a normal flow direction of inlet and the other has a normal flow direction of outlet. The flow status through a port connection may be normal (occurring in the intended direction), reverse (occurring opposite of the intended direction) or static (not occurring). A collection of equipment is connected if there exists some path (material, energy and/or signals) through port connections that reach all the equipment. This flow path constitutes the process topology.
- Port connections can represent mass transfer, heat transfer, mechanical or electrical connections and control information.

## 2.3 Classification of Simulators

Defining the scope of computer-aided modeling is not a trivial matter. The modeling tools in current simulators may roughly be classified into two groups: *block-oriented* and *equation-oriented* approaches [Boston et al. 1993; Marquardt 1991]. We briefly mention some of what we perceive to be their

advantages and disadvantages. This list is not comprehensive but is representative of current technology.

### 2.3.1 Block-Oriented Modeling Tools

Block-oriented approaches mainly address modeling on the flowsheet level. Every process is abstracted by a block diagram consisting of standardized blocks that model the behavior of a process unit or a part of it. All the blocks are linked by signal-like connections representing the flow of information, material and energy employing standardized interface and stream formats.

Models of process units are precoded by a modeling expert and incorporated in a model library for later use. The models are built upon an underlying representation of equations and variables that are hidden from the user. Modeling on the flowsheet level is supported either by a modeling language or by a graphical editor. In both cases, the end user selects the models from the library, provides the model parameters and connects them to the plant model. The incorporated chemical engineering knowledge as well as the model structure is usually fixed and not accessible. A common exception is physical property models that can be selected independently of the process unit model.

More recent researches on model construction have been aiming at establishing decompositions of process models in terms of entities that are simpler than the conventional unit operations, thus allowing the description of complex models in a structured fashion in terms of a relatively small number of concepts. The modeling tools being able to set up model blocks for new equipment are referred to as the block-oriented and knowledge-based

modeling systems.

The typical examples of block-oriented systems are as follows:

- **DYNSIM**

DYNSIM [Gani et al. 1992] is a block-oriented modeling system that requires no programming or compiling by the engineers developing the model. The user defines specific characteristics of modular unit/equipment operation and logic operations that are interconnected by streams. The generic building blocks in DYNSIM are shown in Figure 2-2.

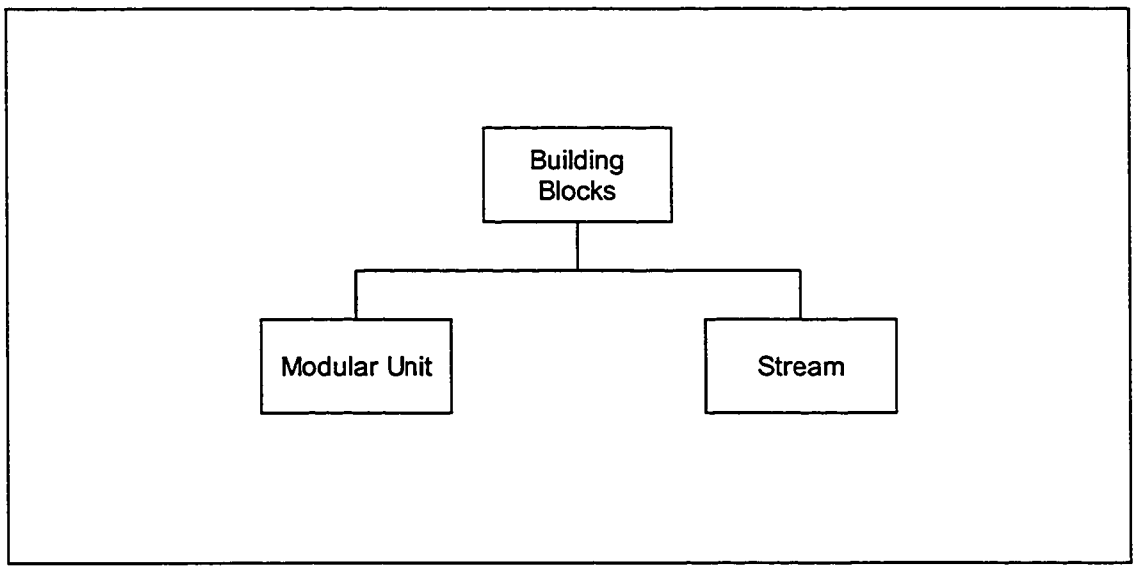



Figure 2-2 Classification of Building Blocks in DYNSIM

- **Modeller**

Modeller [Weiss & Preisig 1997; Preisig 1995] is a block-oriented modeling and simulation environment. It constructs plant models using two principal building blocks, namely *systems* and *connections*. The generic building blocks in Modeller are shown in Figure 2-3.

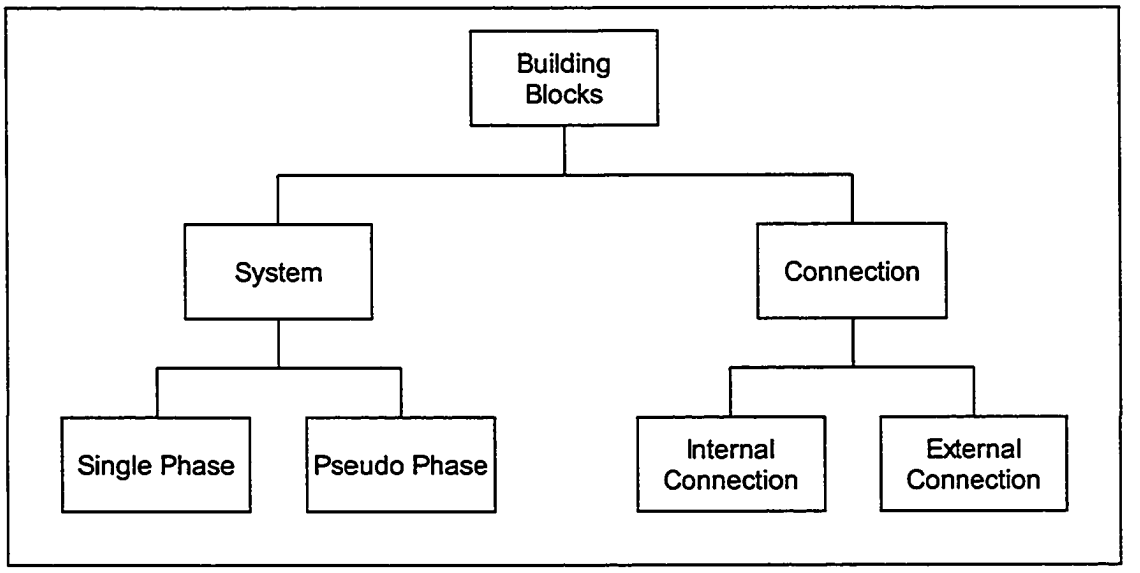



Figure 2-3 Classification of Building Blocks in Modeller

- **ProMot/DIVA**

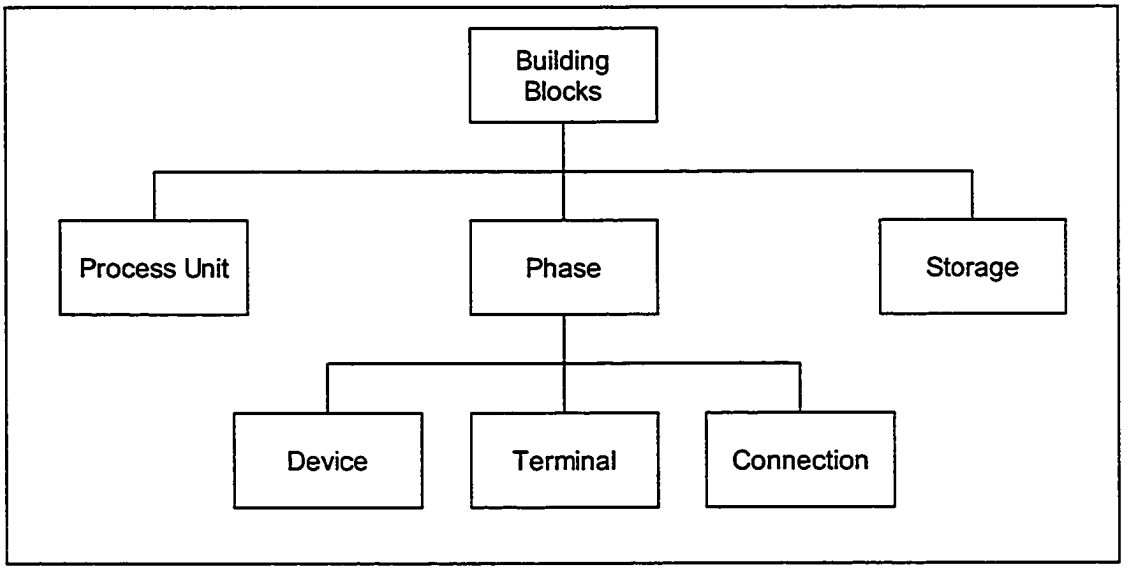



Figure 2-4 Classification of Building Blocks in ProMot/DIVA

ProMot/DIVA [Tränkle et al. 2000; Bär & Zeitz 1990; Kröner et al. 1990; Holl et al. 1988] is a block-oriented modeling system. The generic

building blocks in ProMot/DIVA are shown in Figure 2-4.

### 2.3.2 Equation-Oriented Modeling Tools

Equation-oriented approaches mainly address modeling on the equations level, which represents physico-chemical-biological relations. In this architecture, general-purpose modeling languages are defined in order to capture process modeling knowledge. Unlike block-oriented approach, this approach promotes the development and the use of high-level abstractions while allowing the user to work with the underlying structure when necessary.

More recent researches on model construction have been aiming at supporting hierarchical decomposition of complex models in order to facilitate reuse of modification of existing models. All of them use the concepts from semantic data modeling and object-oriented programming with structured representations of encapsulated submodels organized in inheritance and aggregation hierarchies. These languages are not restricted to chemical engineering applications, since the language definition is confined to a relatively small number of generic elements. The modeling tools being able to set up model equations for new equipment are referred to as the equation-oriented and knowledge-based modeling systems.

The typical examples of equation-oriented systems are as follows:

- **SpeedUp**

SpeedUp [Perkins et al. 1996; Pantelides 1988; Perkins 1984; Perkins & Sargent 1982] is a generic modeling language. The generic building blocks in SpeedUp are shown in Figure 2-5.

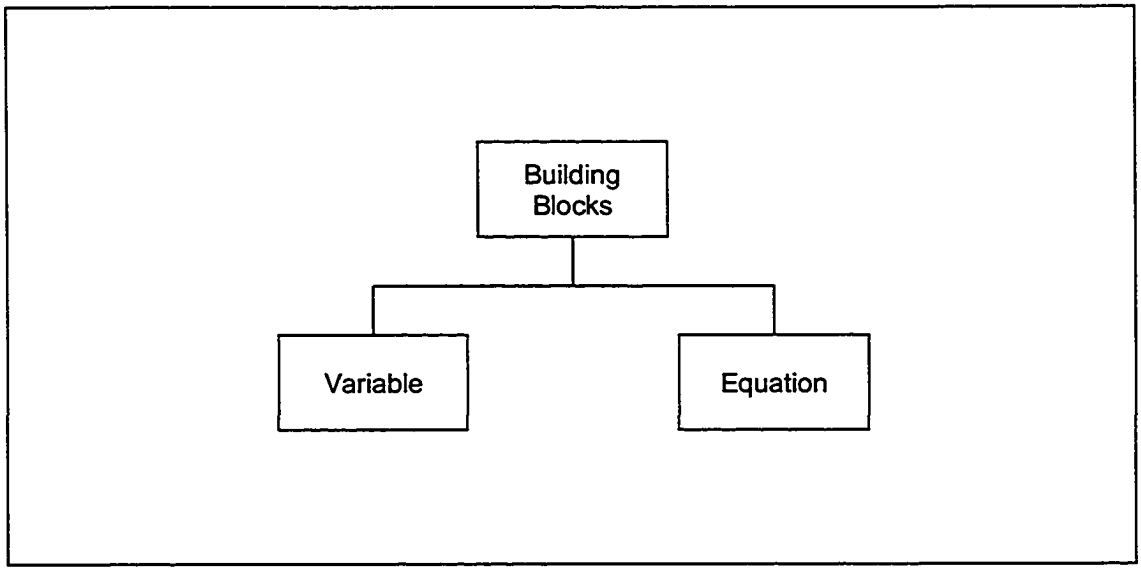



Figure 2-5 Classification of Building Blocks in SpeedUp

- **ASCEND**

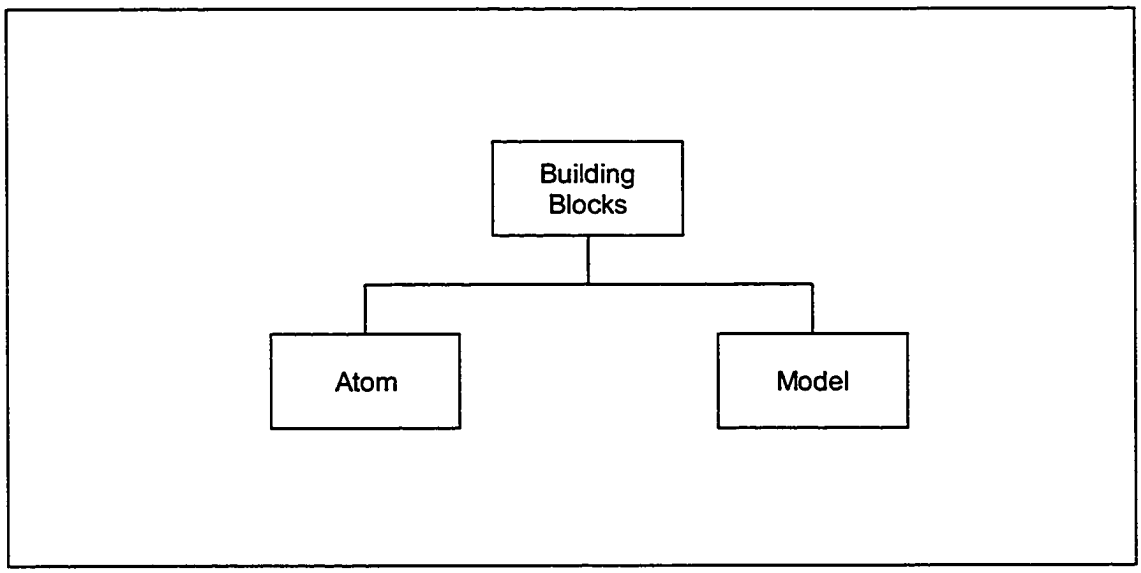



Figure 2-6 Classification of Building Blocks in ASCEND

ASCEND (Advanced System for Calculations in ENgineering Design) [Westerberg et al. 1985; Pièla et al. 1991] is a generic modeling language. The

simulator engine in ASCEND consists of numerical routines that have been defined using the ASCEND modeling language. The generic building blocks in ASCEND are shown in Figure 2-6.

- **gPROMS**

gPROMS [Barton & Pantelides 1994; Oh & Pantelides 1996] is a generic modeling language that contains its own simulator. The generic building blocks in gPROMS are shown in Figure 2-7.

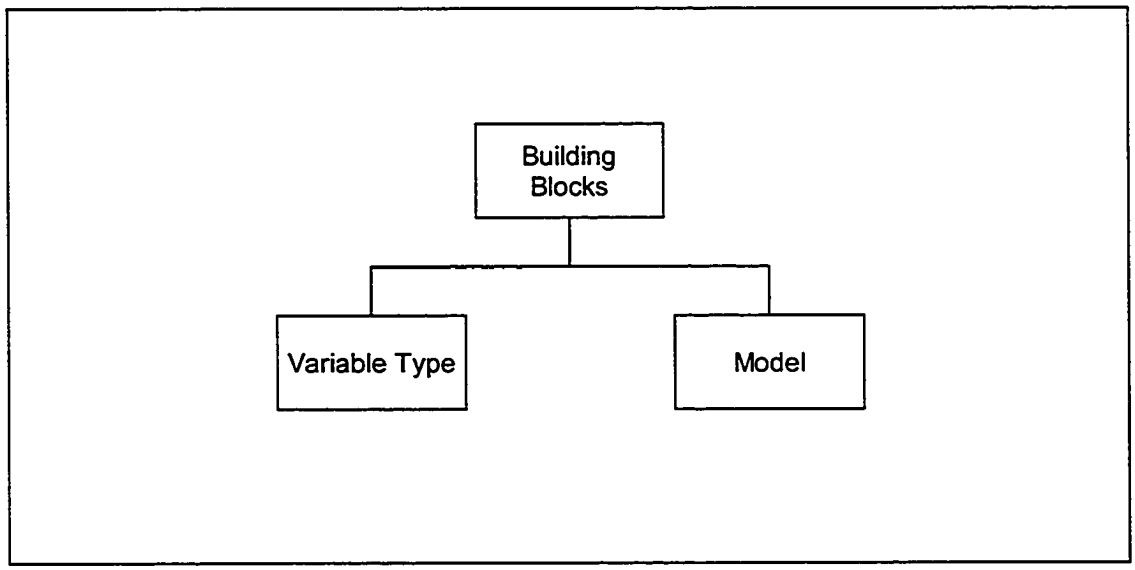



Figure 2-7 Classification of Building Blocks in gPROMS

- **DYMON/DYLAN**

DYMON (DYnamic process MOdeling eNvironment) [Lund 1992] is an integrated modeling and simulation system. The modeling language is called DYLAN. DYLAN is integrated with SpeedUp. The generic building blocks in DYMON are shown in Figure 2-8.

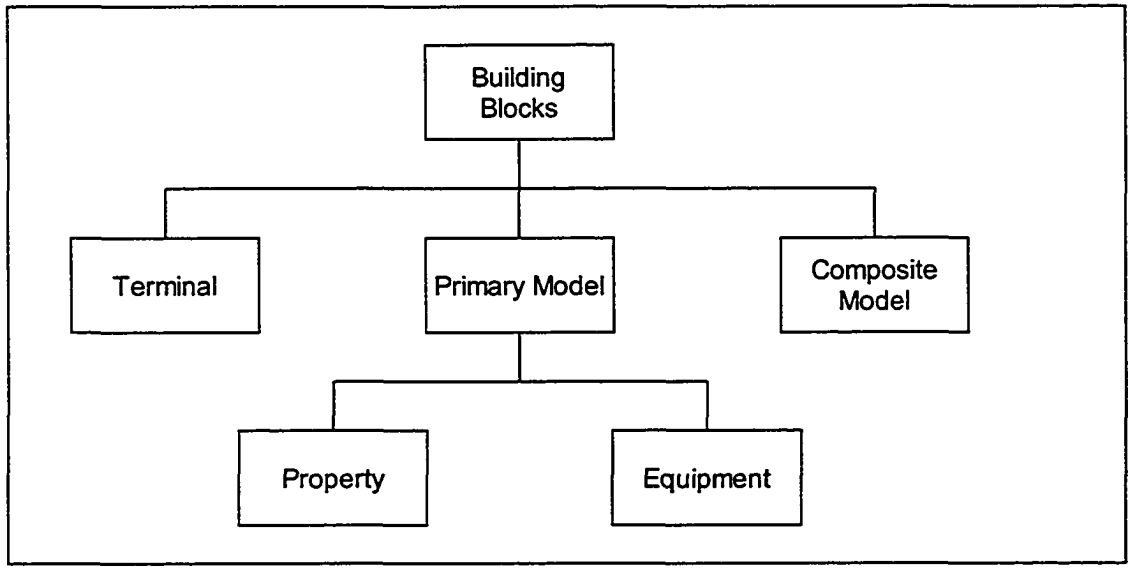



Figure 2-8 Classification of Building Blocks in DYMON/DYLAN

- **OMOLA/OmSim**

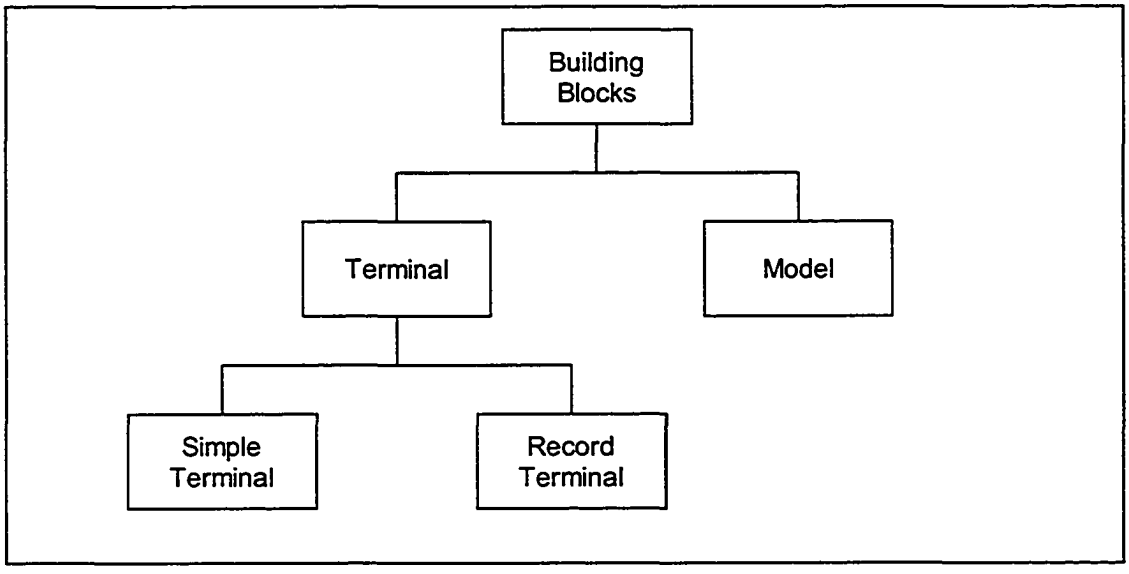



Figure 2-9 Classification of Building Blocks in OMOLA/OmSim

OMOLA (Object-oriented MOdeling LAnguage) [Nilsson 1996] is a generic modeling language that is integrated with a DAEs solver OmSim.

The generic building blocks in OMOLA are shown in Figure 2-9.

- **DesignKit/MODEL.LA**

DesignKit [Stephanopoulos et al. 1990, 1987] is an equation-oriented and knowledge-based system that is integrated with the simulator gPROMS. MODEL.LA is the modeling language of DesignKit. The generic building blocks in DesignKit are shown in Figure 2-10.

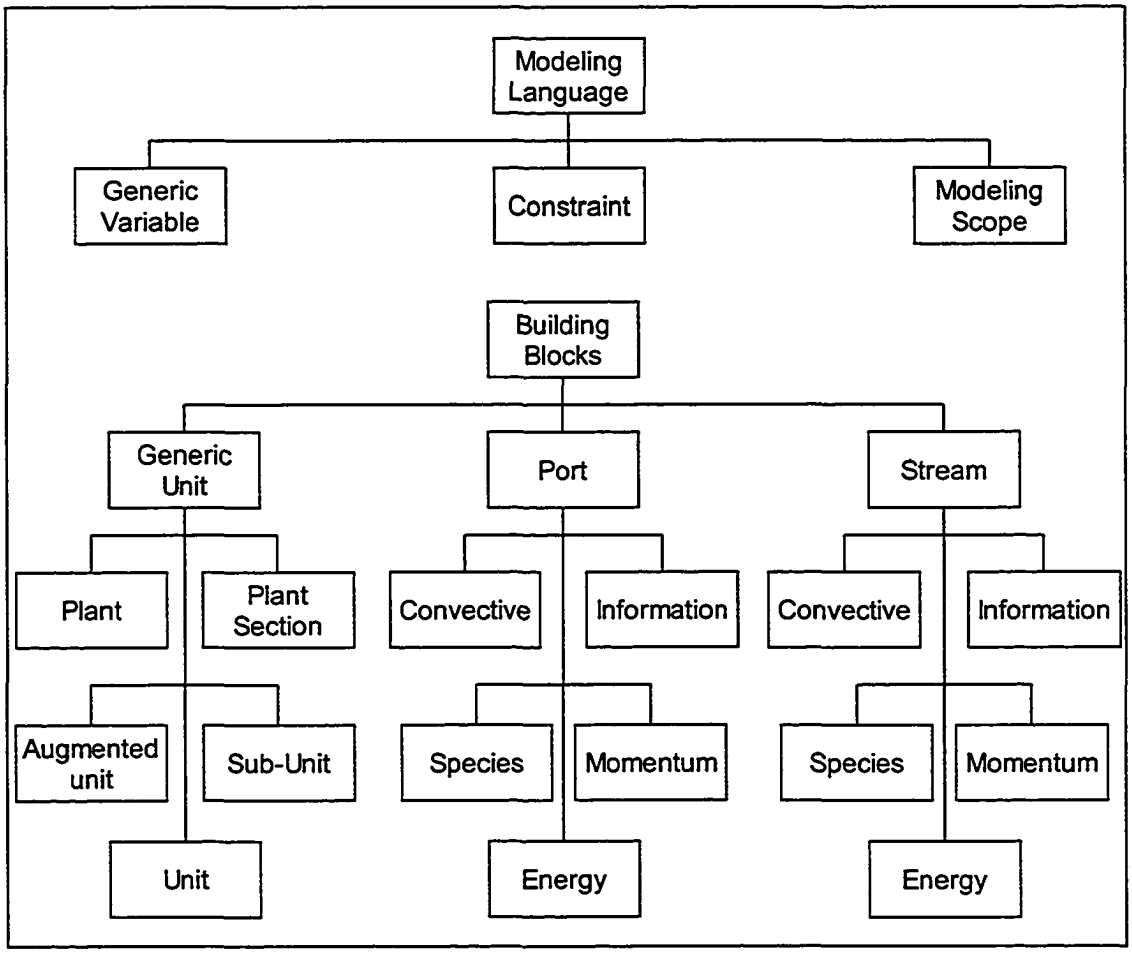



Figure 2-10 Classification of Building Blocks in DesignKit/MODEL.LA

- **ModKit/VEDA**

ModKit [Marquardt 1996, 1992; Bogusch et al. 2001] is an equation-oriented and knowledge-based system that is integrated with simulator gPROMS and SpeedUp. VEDA [Marquardt 1994] is the modeling language of ModKit. The generic building blocks in ModKit are shown in Figure 2-11.

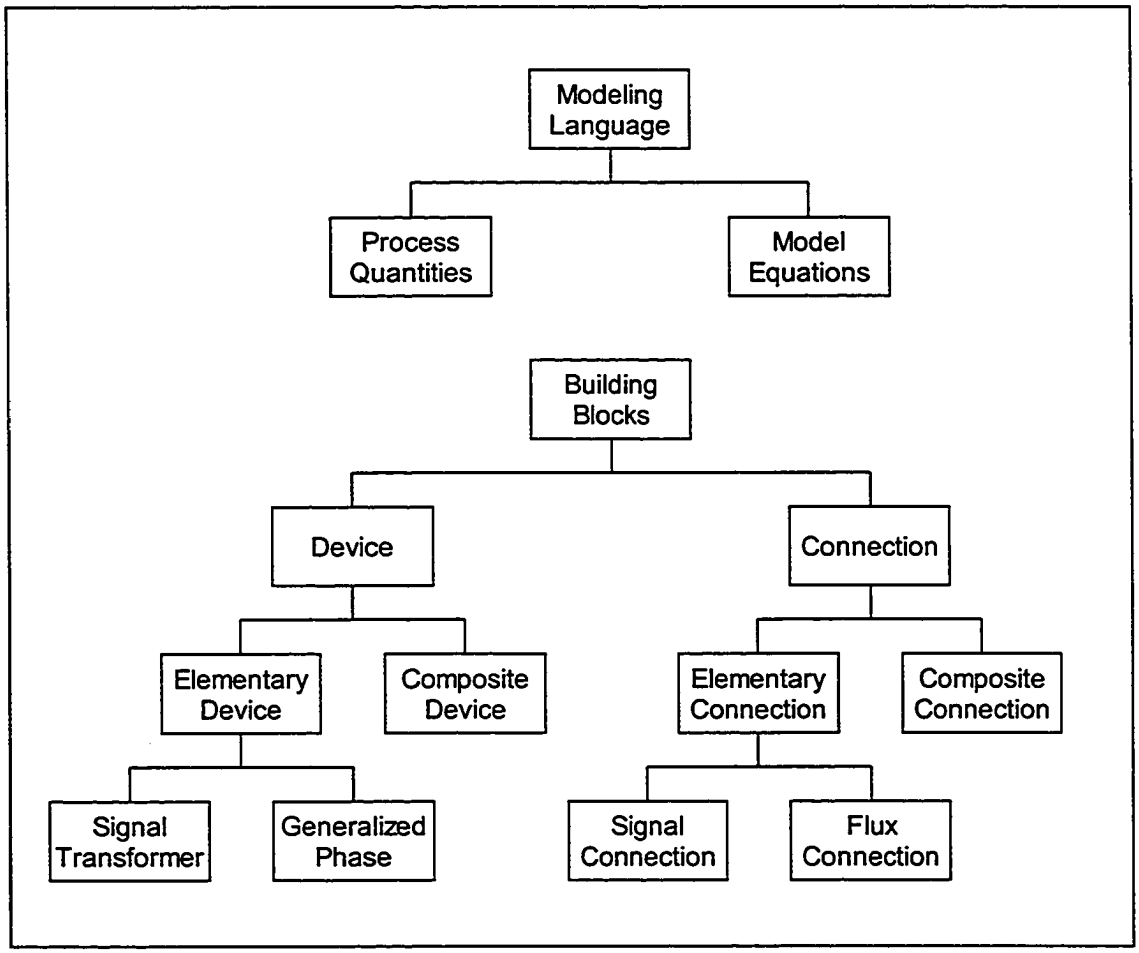



Figure 2-11 Classification of Building Blocks in ModKit/VEDA

- **ModAss**

ModAss is an equation-oriented and knowledge-based integration

environment for processing engineering [Sørlie 1990]. The generic building blocks in ModAss are shown in Figure 2-12.

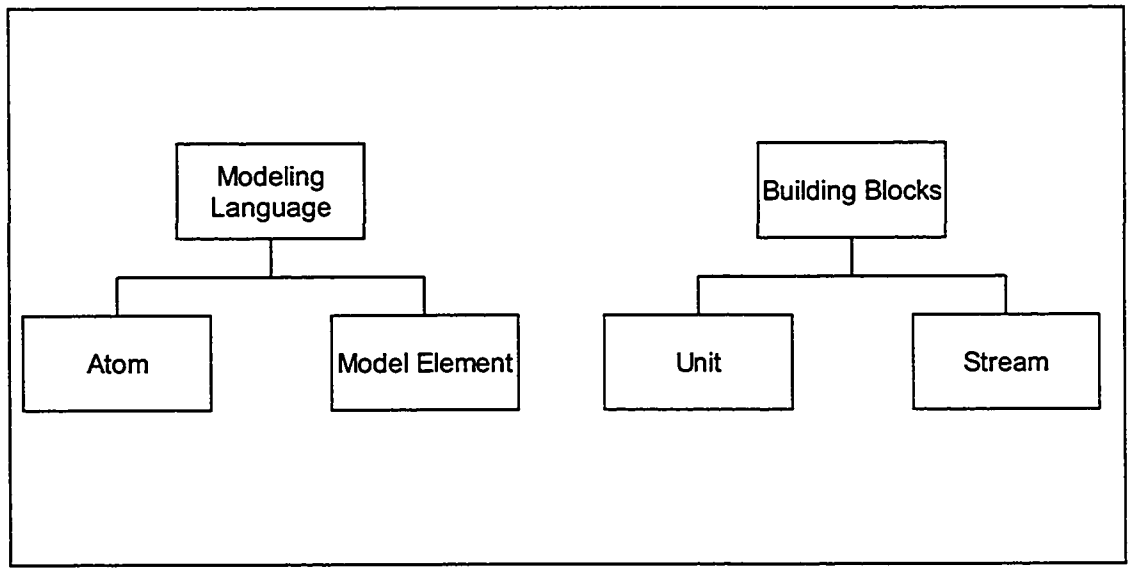



Figure 2-12 Classification of Building Blocks in ModAss

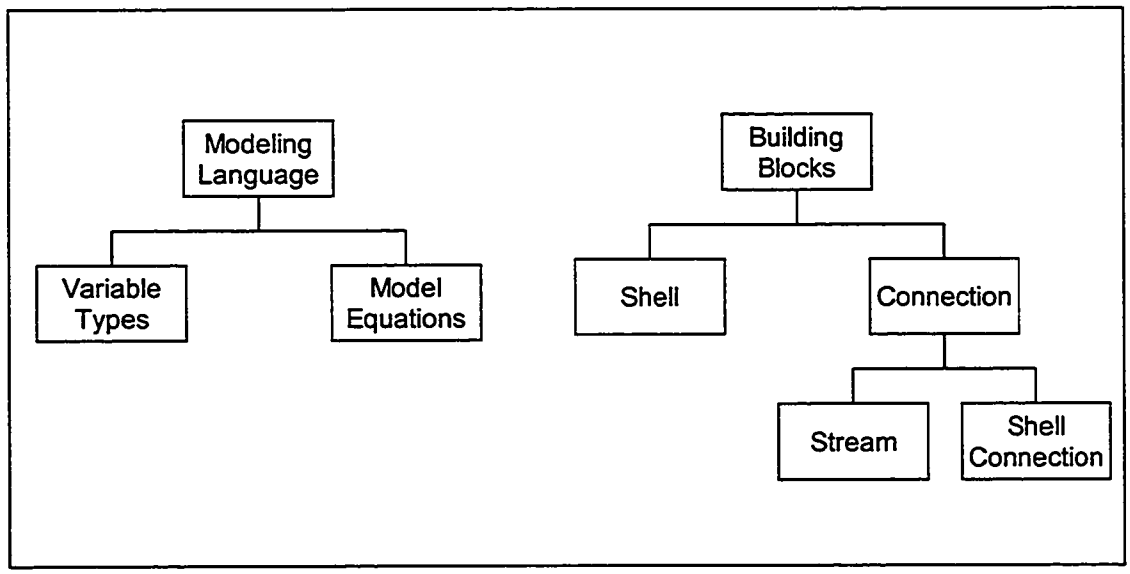



Figure 2-13 Classification of Building Blocks in ModDev

- **ModDev**

ModDev [Jensen & Gani 1998, 1999] is an equation-oriented and knowledge-based system that is integrated with gPROMS. The generic building blocks in ModDev are shown in Figure 2-13.

- **BatchKit**

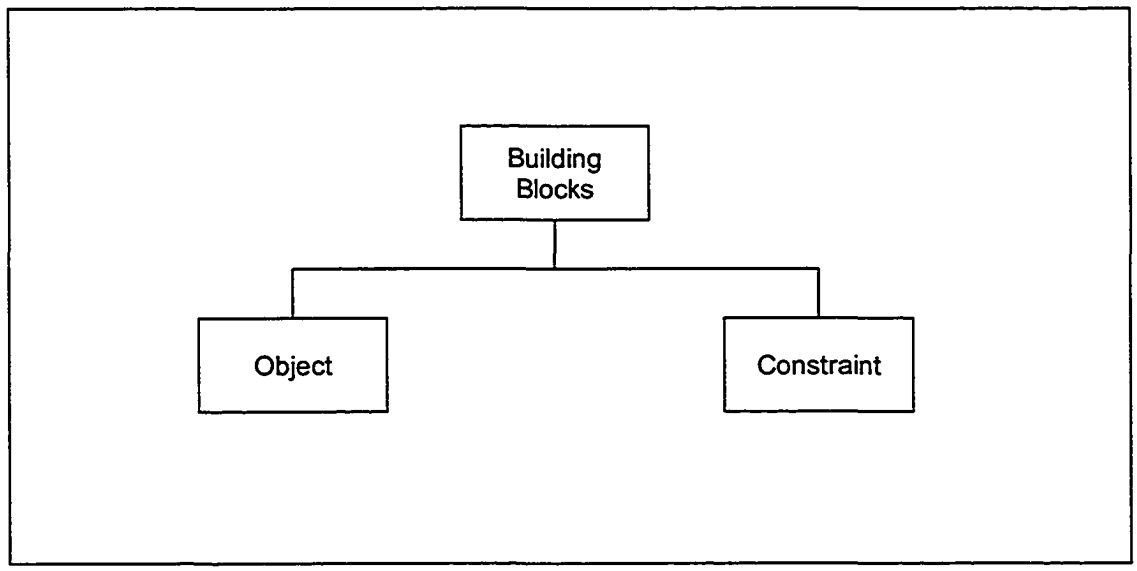



Figure 2-14 Classification of Building Blocks in Batchkit

BatchKit is an equation-oriented and knowledge-based integration environment for processing engineering [Hofmeister 1998]. GAMS and/or SpeedUp [Pantelides 1988] is its modeling language. The generic building blocks in BatchKit are shown in Figure 2-14.

## 2.4 Discussion

Common to both modeling tools is *modular*, *hierarchical* and *object-*

*oriented.* Some of the modeling tools are integrated in computer-aided engineering environments to support modeling, analysis and eventually synthesis of the process and/or its associated control system.

Equation-oriented modeling tools support the implementation of unit models and their incorporation in a model library by means of declarative modeling languages. There are no different tools for the modeling expert or for the end user. Hence, modeling on the unit level requires profound knowledge in such diverse areas as chemical engineering, modeling and simulation, numerical mathematics, and computer science. The development of novel unit models is therefore often restricted to a small group of experts.

The ability to create new unit operation models is particularly important because the construction of a complete library of unit operation models is probably impossible due to the complexity and diversity of dynamic process models.

The block-oriented approach to modeling and simulation, though powerful and easy accessible to many engineers for the solution of standard flowsheet problems, does not adequately support the solution of more involved problems. This is largely due to the lack of precoded models for many unit operations of adequate level of detail. Examples include multi-phase reactors, membrane processes, polymer reactors, and most units including particulates. Should the user want to use a new model, it must be completely written. Therefore, costly and time-consuming model development for a particular unit is often required during project work.

While equation-oriented languages support the implementation of the models largely, they do not assist the user in developing models from using engineering concepts, nor is there support for the documentation of the modeling process during the lifecycle of a process or for proper design and

documentation of the model library. No mechanism is provided for reusing information employed in constructing a model of one unit operation in building the model of another. For instance, models of different types of heat exchangers have to be developed and maintained independently despite the fact that they may share many common characteristics. Reuse of validated unit models by a group of simulator users is therefore almost impossible and redundant modeling is unavoidable. The consistency and soundness of an initially even well-designed model library is inevitably getting lost over time.

Standard models of standard unit operations are included in both approaches mentioned above in the form of model libraries. Thus, standard problems are readily solved with these approaches. If non-standard problems are to be tackled, the model libraries have to be extended. Therefore, time-consuming and error-prone programming has to be performed by specifically trained experts and much engineering effort is spent on building a model of process instead of using it [Dieterich & Eigenberger 1997].

We hopefully combine the advantages of both, the flexibility of nowadays equation-oriented modeling tools and the ease-of-use of current block-oriented modeling tools; abandon the disadvantages of both, difficult accessibility to many engineers of equation-oriented modeling tools and difficult extension to the solution of non-standard equipment of block-oriented modeling tools. We believe that a complex model of (ideally) any chemical process can be configured just by selection, parameterization and aggregation of fundamental modeling components taken from a component model library. It is the nature of these fundamental modeling components and hierarchical and object-oriented dynamic modeling of complex processes that will be widely explored in this research.

# Chapter 3 Hierarchical Structure of the Simulator

## 3.1 Introduction

Chemical plants are often decomposed into plant sections. These plant sections are composed of a number of unit operations (e.g., distillation units, reactors, buffer tanks). Unit operations are often composed of a set of processing objects (e.g., pumps, valves, tubes, vessels, sensors).

In general, modeling of a process involves the following five principle steps: (1) decomposition of the process model into the equipment models; (2) decomposition of the equipment model into the component models; (3) creation of new component models if they do not exist in a model library; (4) aggregation of the component models into an equipment model; (5) aggregation of the equipment models into a complete process model.

In the first step, which represents the process abstraction, a process model is hierarchically decomposed and finally abstracted into a set of component models. When the needed component models are identified, the component models that do not already exist in a model library must be created. Finally, these component models are hierarchically aggregated and finally assembled into a complete process model.

A typical example for hierarchical decomposition of process model is shown in Figure 3-1. The process model is decomposed into two equipment models (e.g., reactor and absorber); the absorber is further decomposed into component models (e.g., absorber tray).

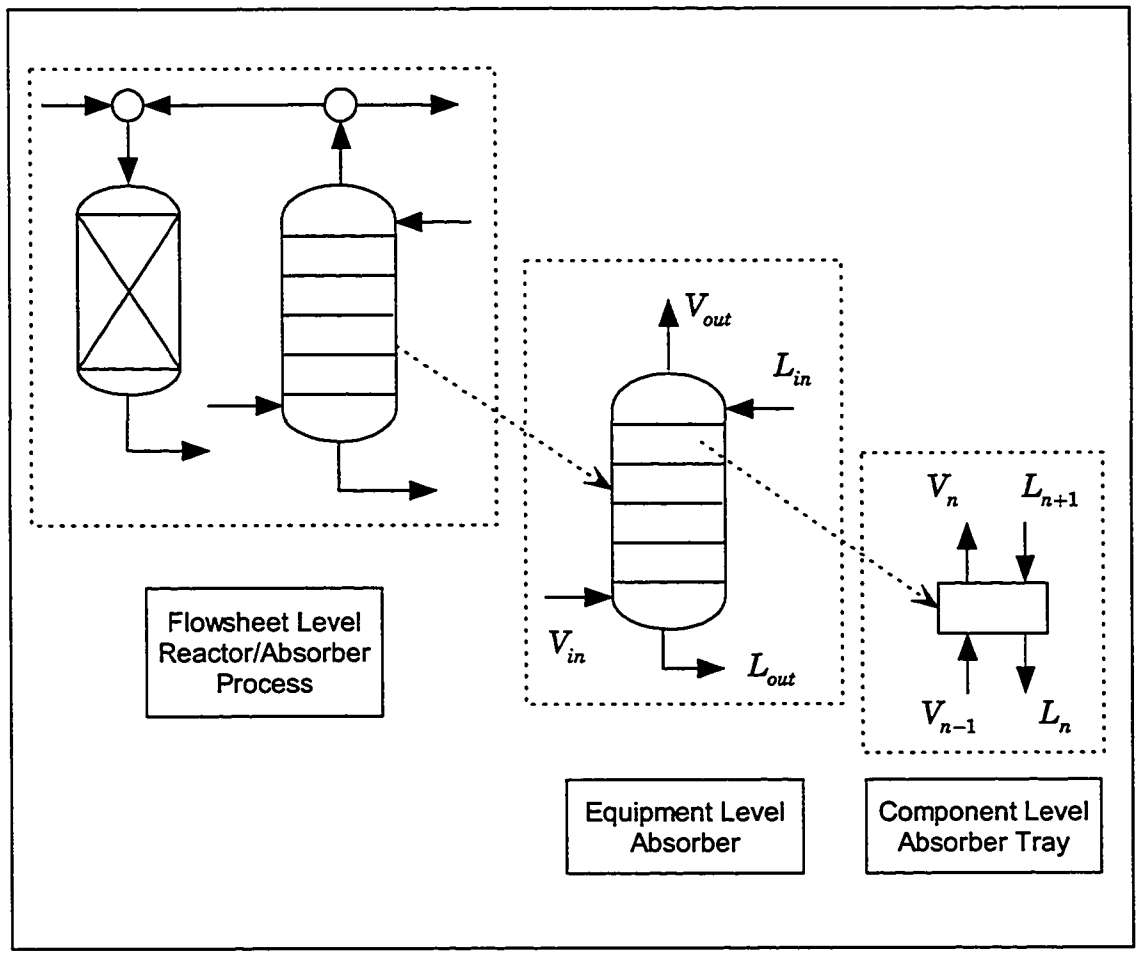



Figure 3-1 Hierarchical Decomposition of Process Model

## 3.2 Hierarchical Structure

There are two programs in ForeSee dynamic simulator structure shown in Figure 3-2. The one shown on the left is the *Simulator* itself. The *Simulator* is operated via the Run Director. The program on the right is the *Initializer*. This program allows the user, using the drag-and-drop GUI techniques, to assemble a flowsheet model using a library of equipment models. Another feature of the *Initializer* is the Modeler that allows the user to develop new equipment models from a set of basic component models.

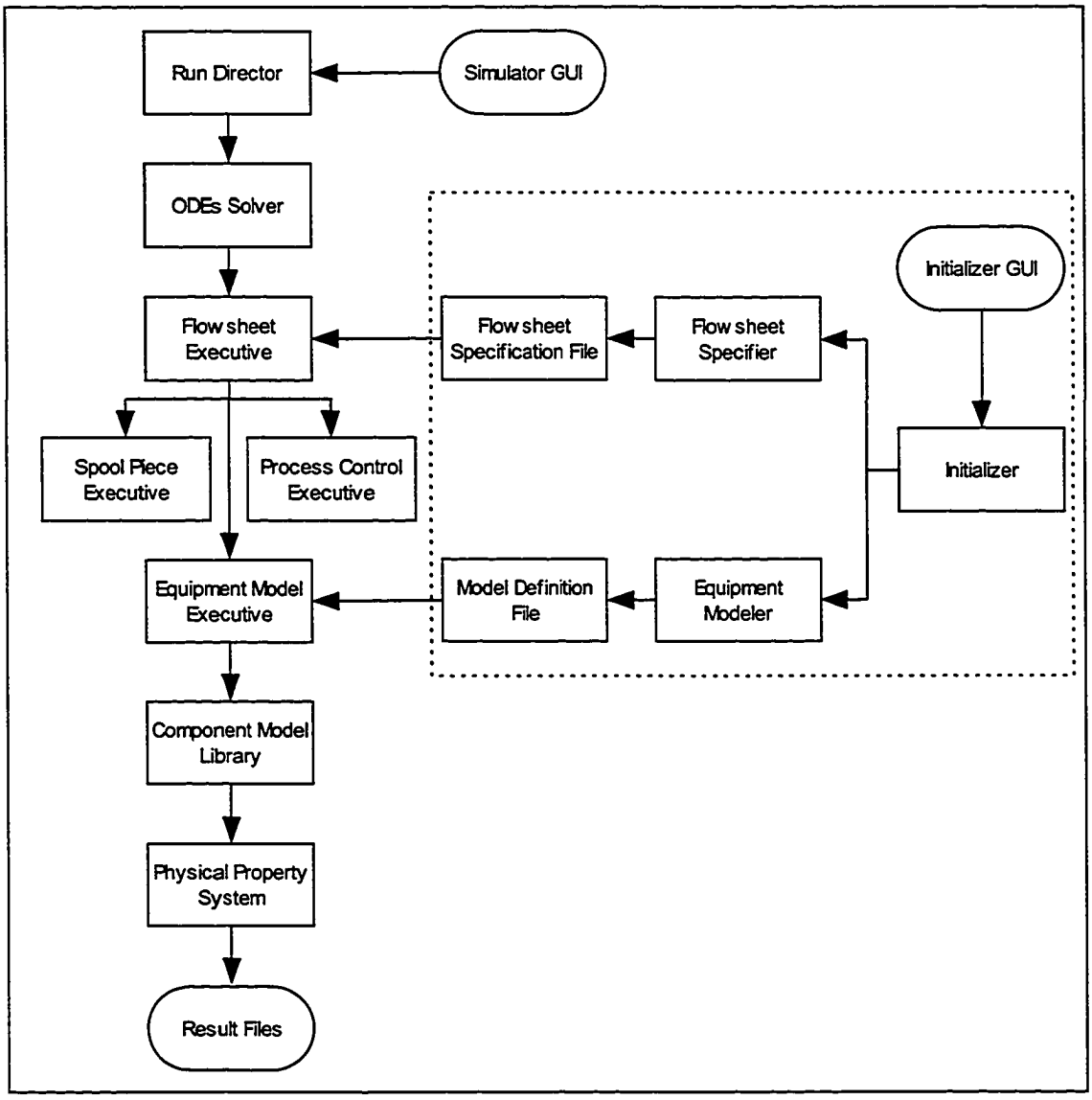



Figure 3-2 ForeSee Dynamic Simulator Structure

The user must also provide the data that describes the equipment model in terms of equipment size, chemical components involved, and other operational parameters. Initial conditions for all state variables including phase composition, pressure, temperature, and initial inventories must also be provided. These all are done via pull-down dialog panels which request the requisite data. All data and parameters can be accessed by clicking **Next** buttons on the dialog panels.

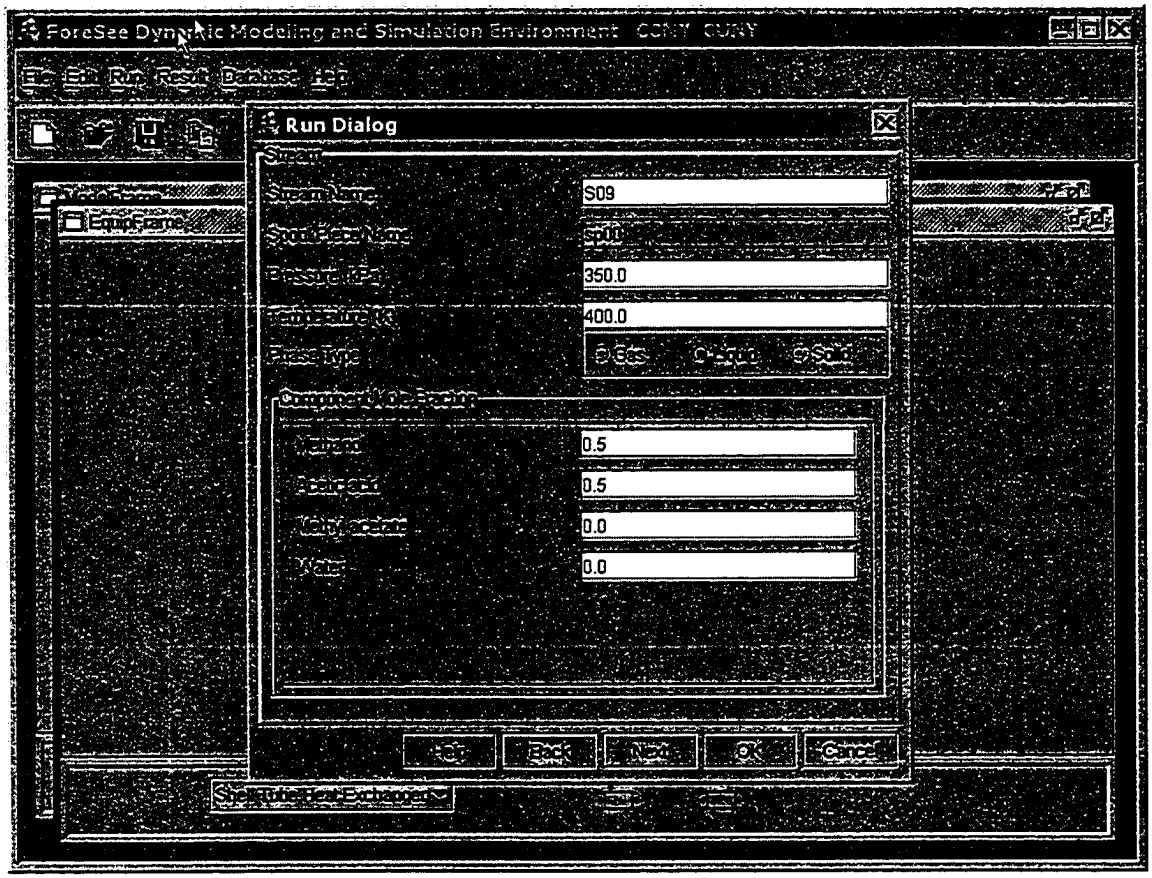



Figure 3-3 Data and Parameters Dialog Panels in ForeSee

### 3.2.1 Run Director Level

Every computer program starts with a main program. In this case, it is called the *Run Director*. It allows the user to tell the program what to do. For most simulators the Run Director is interactive, generally by means of a graphical user interface (GUI). In this simulator, the Run Director calls the *ODEs Solver* which is a routine for solving a set of first-order ordinary differential equations (ODEs). Run Director is a run-time environment to run, disturb, and plot process performance.

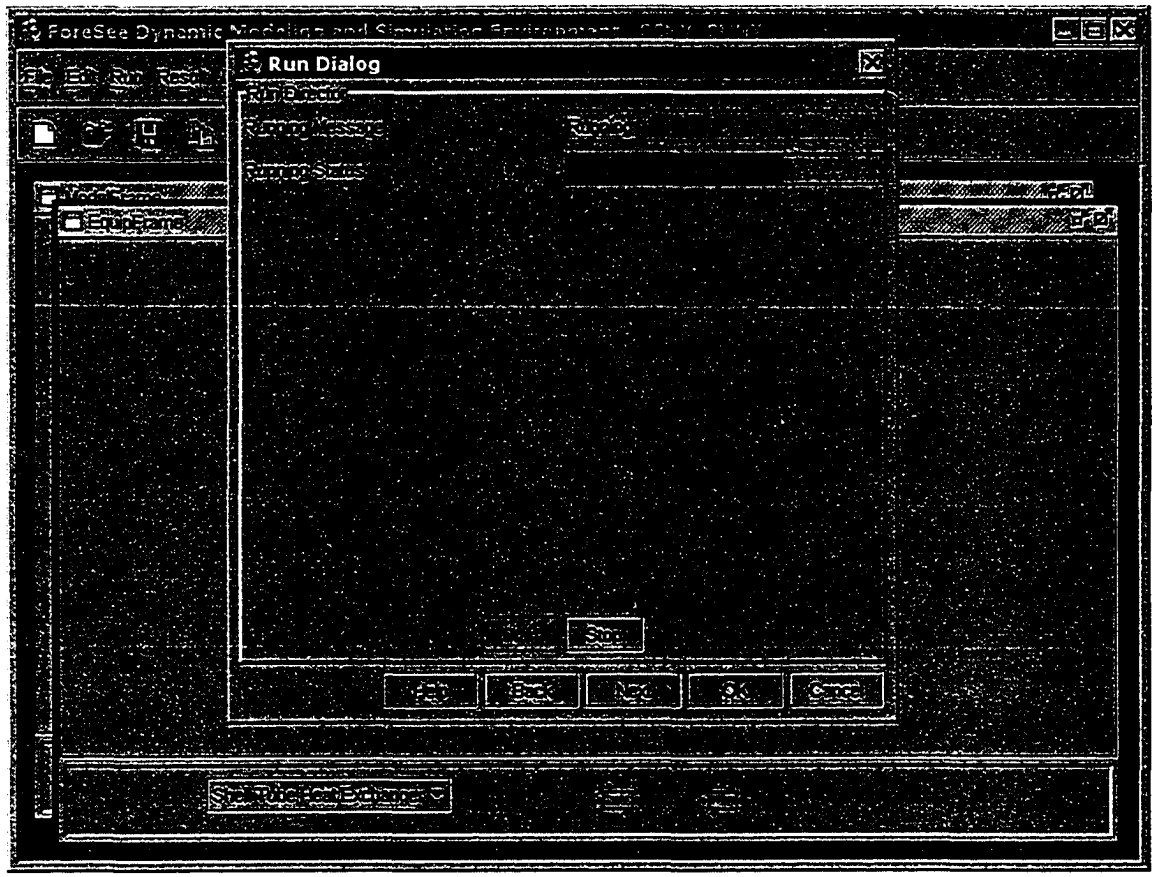



Figure 3-4 Run Director Level in ForeSee

### 3.2.2 ODEs Solver Level

There are many ODEs Solver routines available. Typical are LSODE from the Lawerence Livermore ODEPACK, several ODEs routines from *Numerical Recipes* [Press et al. 1992], and some from proprietary mathematical libraries such as IMSL. All ODEs Solver routines require a function evaluation routine to evaluate the current values of the derivatives of all of the ordinary differential equations (ODEs) being integrated. For dynamic process simulation, we will refer to this as the *Flowsheet* routine. Its function is to invoke the equipment models and spool pieces that represent

the flowsheet.

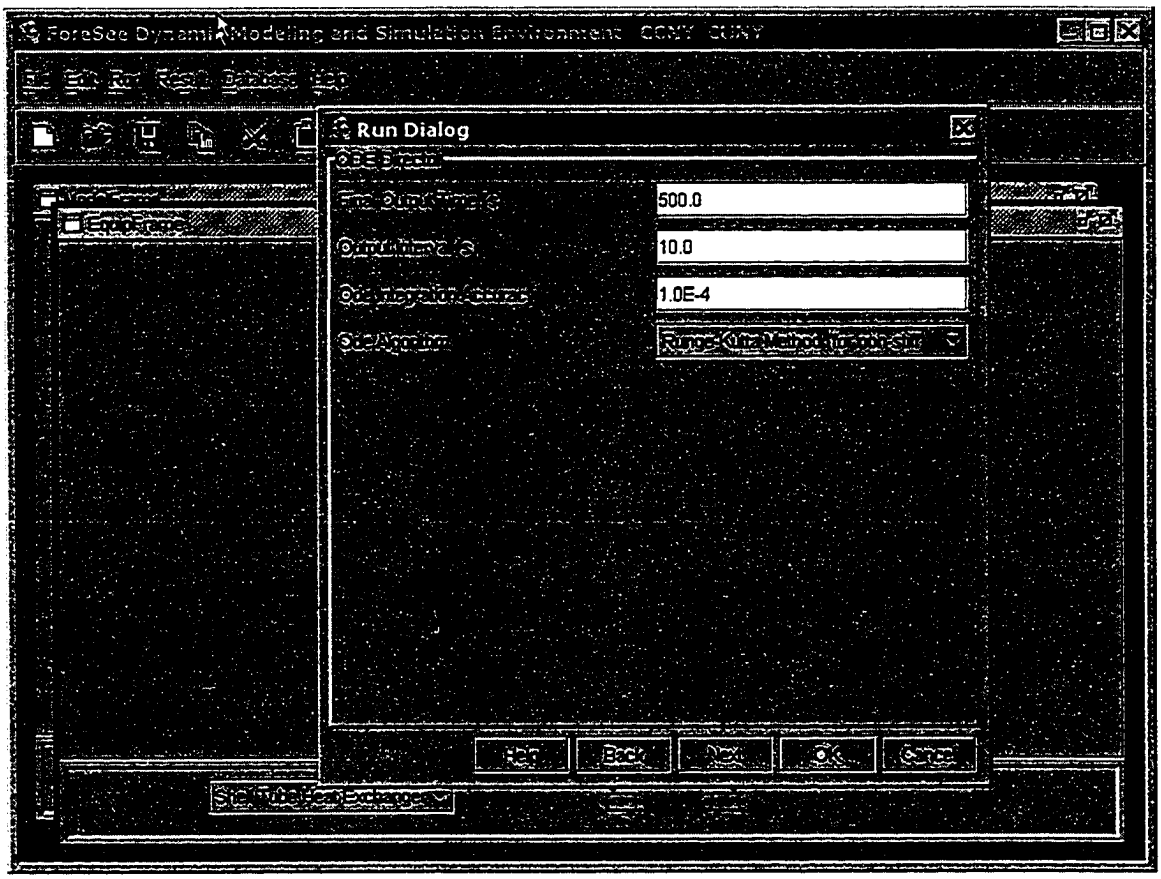



Figure 3-5 ODEs Solver Level in ForeSee

### 3.2.3 Flowsheet Level

Flowsheet is a network that comprises a set of equipment items and a set of spool pieces. The primary function of the Flowsheet routine is to invoke the designated equipment models and spool pieces in a specified order. Flowsheets are intended to represent and explain chemical processes. To make them easy to understand, they are constructed with a consistent set of symbols for equipment, piping, and operating conditions [Walas 1988].

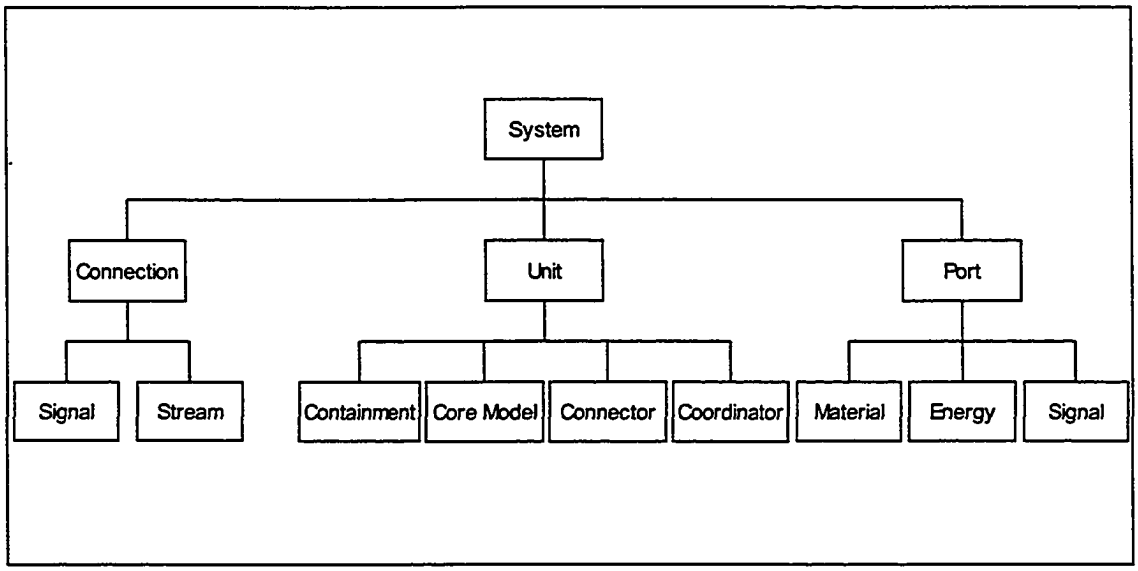



Figure 3-6 Building Block Hierarchy of Flowsheet Level in ForeSee

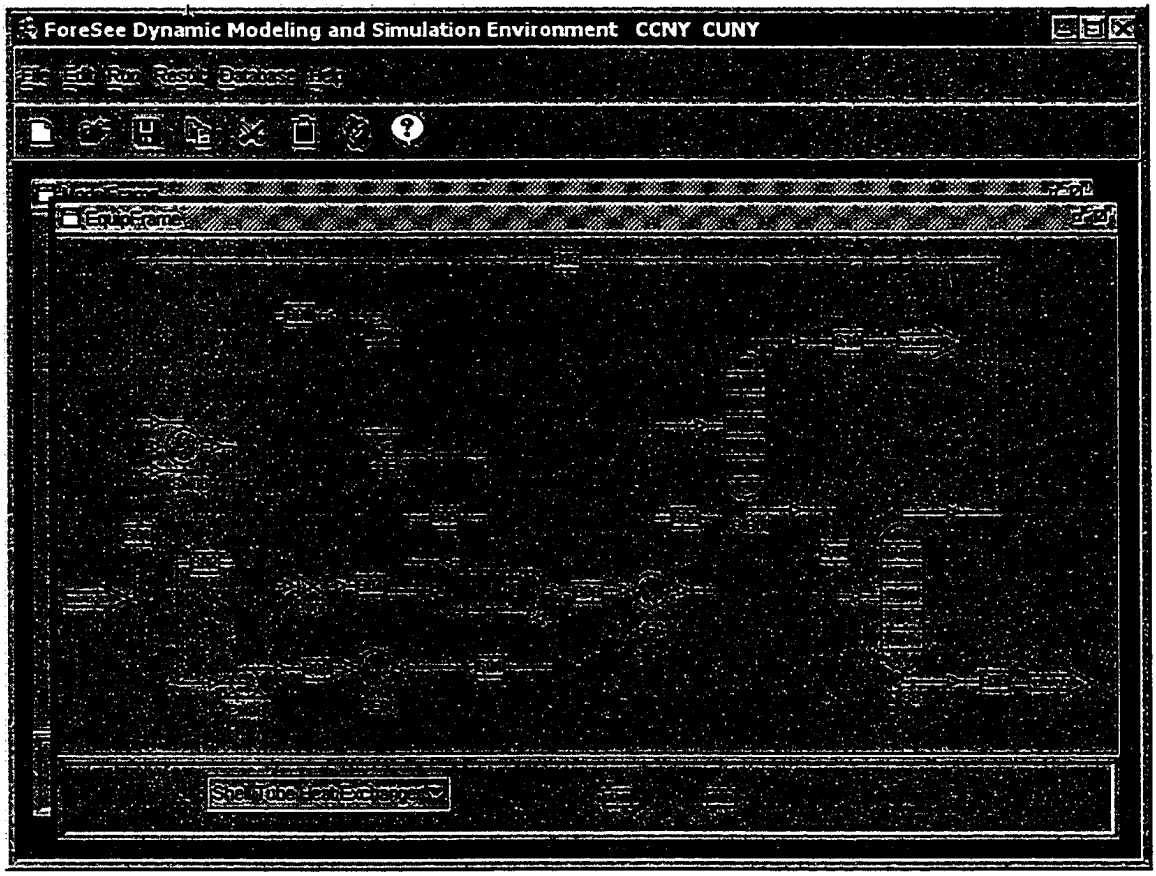



Figure 3-7 Flowsheet Level in ForeSee

### 3.2.4 Equipment Model Level

The function of Equipment Model is to evaluate the derivatives associated with its model equations. An equipment model can be represented in terms of the four types of modeling components, namely, *containments*, *core models*, *connectors* and *coordinators* as shown in Figure 3-8.

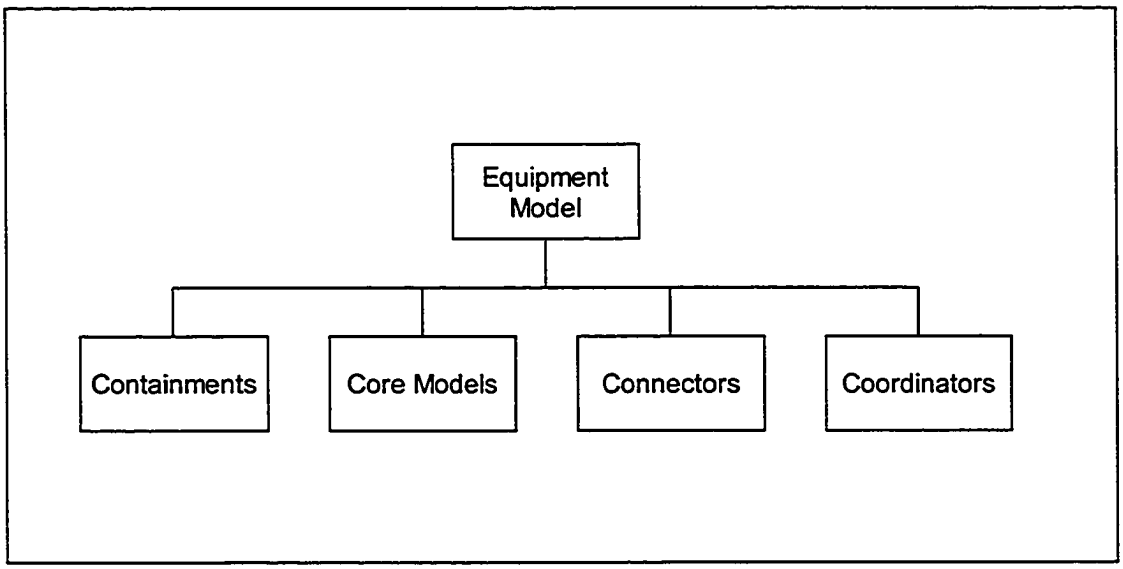



Figure 3-8 Modeling Components Hierarchy of Equipment Model

If the structural diagram for an equipment model resembles that for a flowsheet, the resemblance is not entirely coincidental. It is a way of visualizing how an equipment model functions just as a flowsheet diagram allows a process engineer to understand how a process works without having to refer to the underlying mathematical equations. This provides a powerful tool both for creating new equipment models and for understanding existing ones.

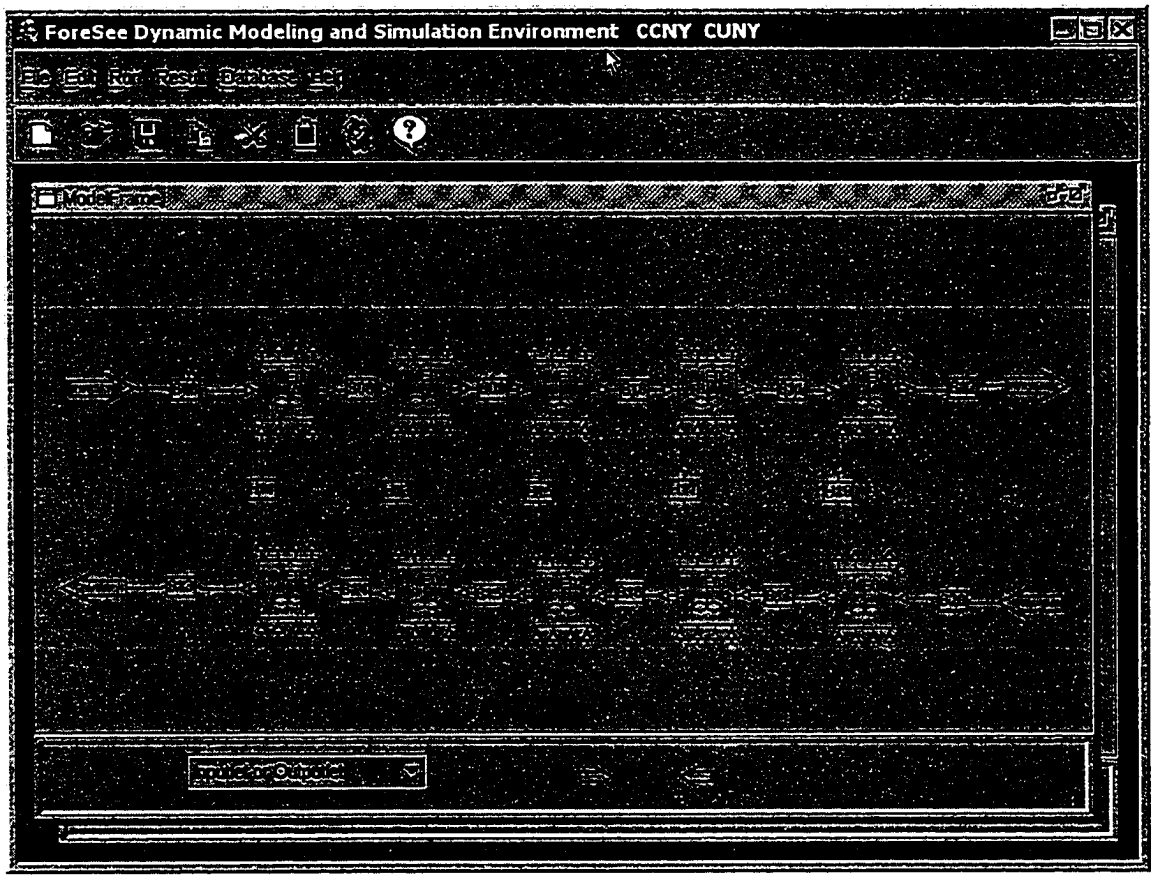



Figure 3-9 Equipment Model Level in ForeSee

### 3.2.5 Spool Piece Level

The function of a spool piece is to determine the flow rate of a stream through a specified configuration of piping. The simplest spool piece is one for which the flow rate is specified as a function of time. This would be appropriate if the piping configuration includes a flow meter, a control valve, and a flow controller. More realistic spool pieces determine the flow rate based on upstream and downstream nozzle pressures and the length and diameter of the piping. Complex spool pieces involve more than one pipe segment. One example is a flow splitter in which the inlet segment branches into two or more outlet segments. Another is an in-line mixer in which two or

more inlet segments come together resulting in one outlet segment.

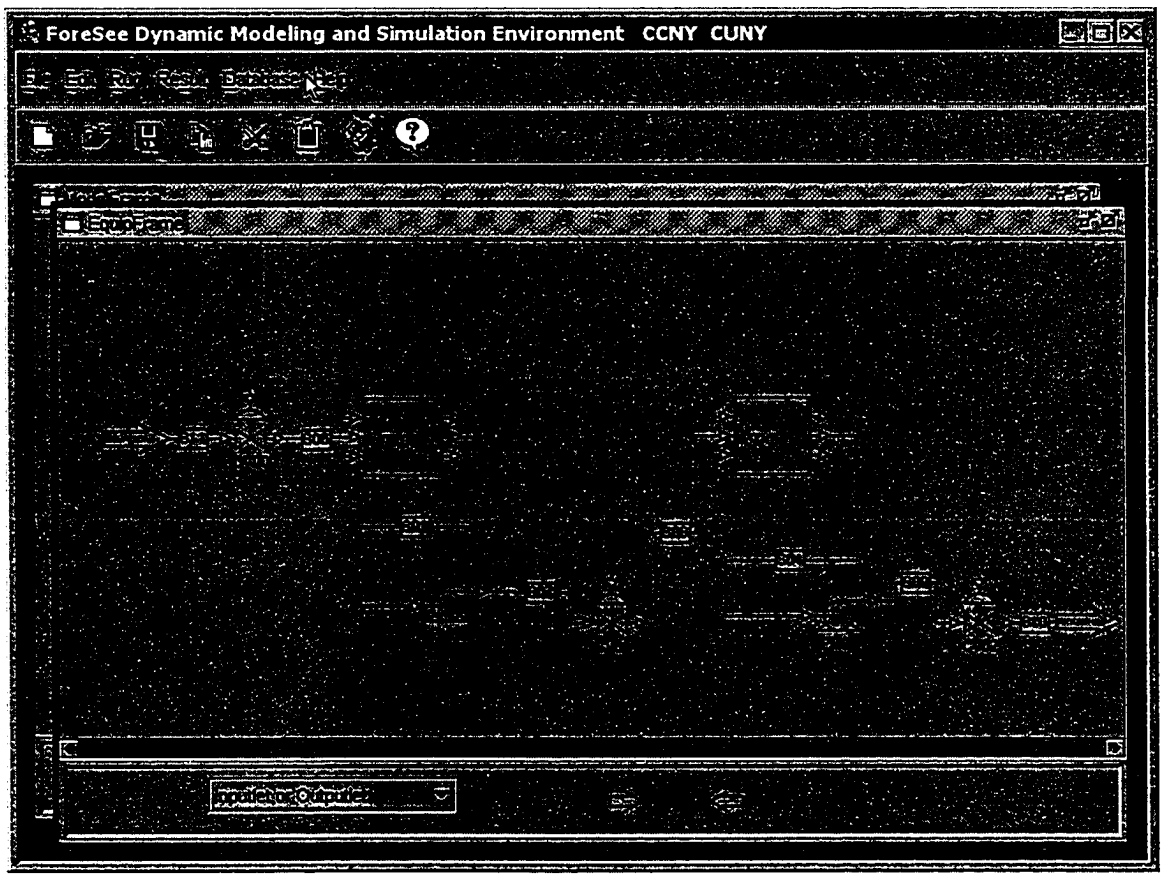



Figure 3-10 Spool Piece Level in ForeSee

The number of piping configurations one may encounter are numerous and varied. Not only are there various piping configurations, but also each may contain components other than just simple pipe. Some of these components are fittings, control valves, block valves, check valves, reducers and expanders, pumps and compressors. It is not practical to develop a library of spool pieces to cover every possible situation that may arise in practice. Instead, we will adopt an approach similar to that for equipment models. We will represent a complex spool piece as a configuration of piping segments, each characterized by a combination of piping components.

### 3.2.6 Process Control Level

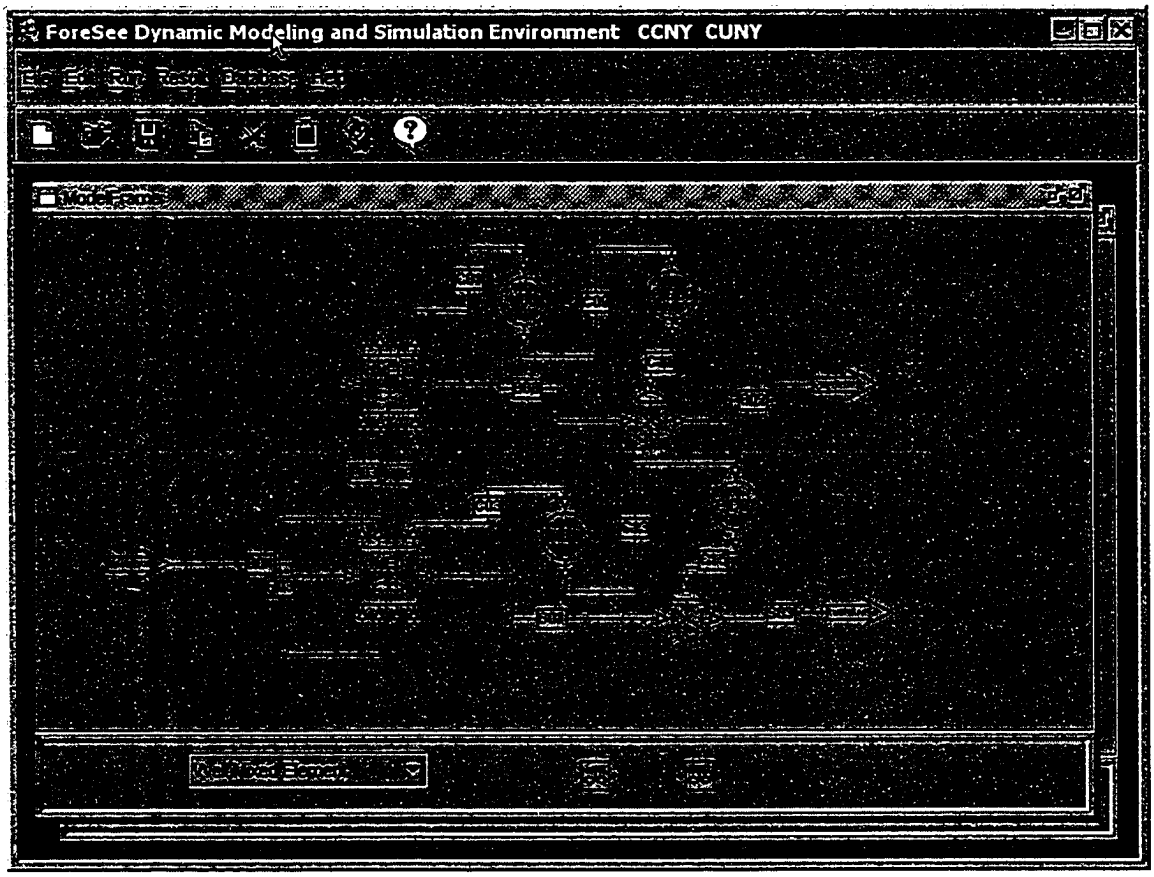


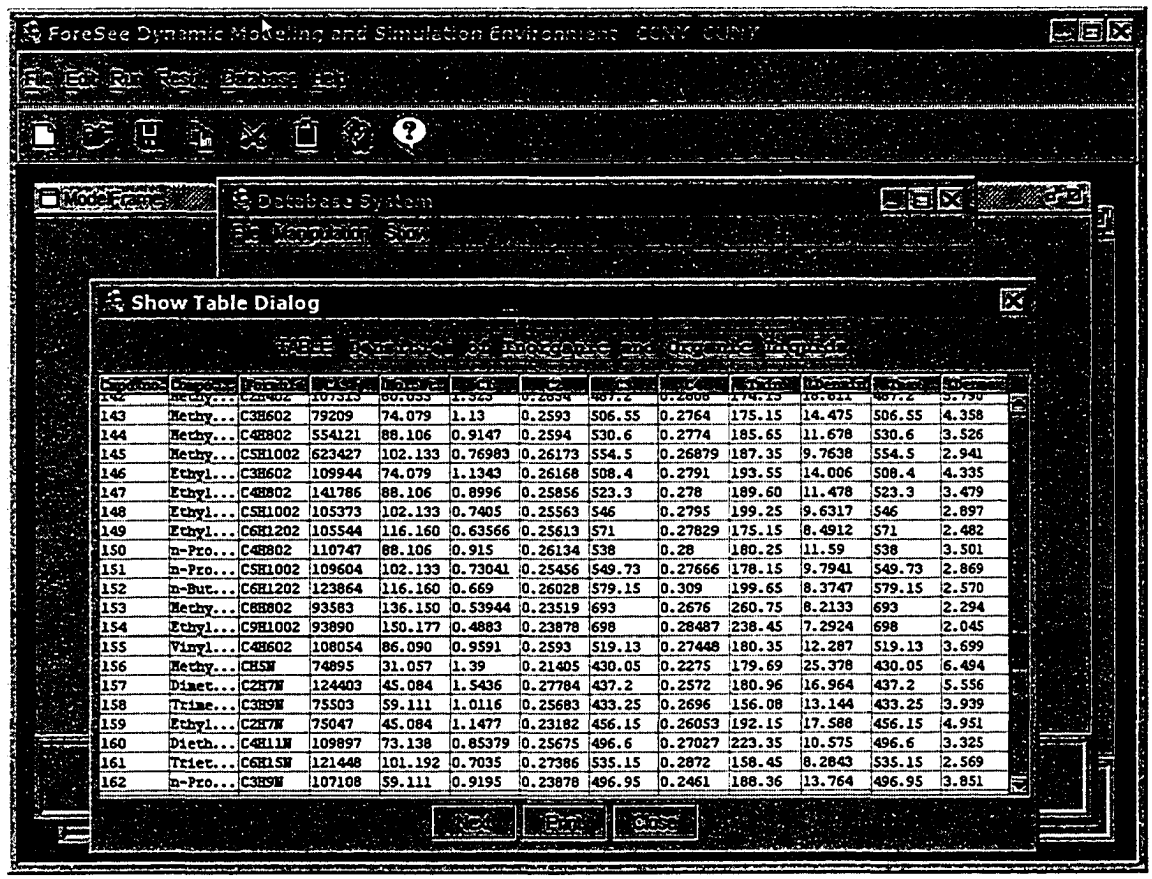

Figure 3-11 Process Control Level in ForeSee

In feedback control, after an offset of the controlled variable from a preset value has been generated, the controller acts to eliminate or reduce the offset. Usually there is produced an oscillation in the value of the controlled variable whose amplitude, period, damping and permanent offset depend on the nature of the system and the mode of the action of the controller. The usual controllers provide one, two, or three of these modes of corrective actions (proportional, integral, and derivative).

### 3.2.7 Component Model Level

Component models can be categorized into four types (*containments, core models, connectors,* and *coordinators*), each of which describes a different aspect of system behavior. Together they provide a comprehensive framework for describing the dynamic behavior of actual items of equipment. The detailed description about component models will be presented in Chapter 4.

### 3.2.8 Physical Property Level




| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 143 | Methy... | C3H6O2 | 79209 | 74.079 | 1.13 | 0.2593 | 506.55 | 0.2764 | 175.15 | 14.475 | 506.55 | 4.358 |
| 144 | Methy... | C4H8O2 | 554121 | 88.106 | 0.9147 | 0.2594 | 530.6 | 0.2774 | 185.65 | 11.678 | 530.6 | 3.526 |
| 145 | Methy... | C5H10O2 | 623427 | 102.133 | 0.76983 | 0.26173 | 554.5 | 0.26879 | 187.35 | 9.7638 | 554.5 | 2.941 |
| 146 | Ethyl... | C3H6O2 | 109944 | 74.079 | 1.1343 | 0.26168 | 508.4 | 0.2791 | 193.55 | 14.006 | 508.4 | 4.335 |
| 147 | Ethyl... | C4H8O2 | 141786 | 88.106 | 0.8996 | 0.25856 | 523.3 | 0.278 | 189.60 | 11.478 | 523.3 | 3.479 |
| 148 | Ethyl... | C5H10O2 | 105373 | 102.133 | 0.7405 | 0.25563 | 546 | 0.2795 | 199.25 | 9.6317 | 546 | 2.897 |
| 149 | Ethyl... | C6H12O2 | 105544 | 116.160 | 0.63566 | 0.25613 | 571 | 0.27829 | 175.15 | 8.4912 | 571 | 2.482 |
| 150 | n-Pro... | C4H8O2 | 110747 | 88.106 | 0.915 | 0.26134 | 538 | 0.28 | 180.25 | 11.59 | 538 | 3.501 |
| 151 | n-Pro... | C5H10O2 | 109604 | 102.133 | 0.73041 | 0.25456 | 549.73 | 0.27666 | 178.15 | 9.7941 | 549.73 | 2.869 |
| 152 | n-But... | C6H12O2 | 123864 | 116.160 | 0.669 | 0.26028 | 579.15 | 0.309 | 199.65 | 8.3747 | 579.15 | 2.570 |
| 153 | Methy... | C6H8O2 | 93583 | 136.150 | 0.53944 | 0.23519 | 693 | 0.2676 | 260.75 | 8.2133 | 693 | 2.294 |
| 154 | Ethyl... | C9H10O2 | 93890 | 150.177 | 0.4883 | 0.23878 | 698 | 0.28487 | 238.45 | 7.2924 | 698 | 2.045 |
| 155 | Vinyl... | C4H6O2 | 108054 | 86.090 | 0.9591 | 0.2593 | 519.13 | 0.27448 | 180.35 | 12.287 | 519.13 | 3.699 |
| 156 | Methy... | CH5N | 74895 | 31.057 | 1.39 | 0.21405 | 430.05 | 0.2275 | 179.69 | 25.378 | 430.05 | 6.494 |
| 157 | Dimet... | C2H7N | 124403 | 45.084 | 1.5436 | 0.27784 | 437.2 | 0.2572 | 180.96 | 16.964 | 437.2 | 5.556 |
| 158 | Trime... | C3H9N | 75503 | 59.111 | 1.0116 | 0.25683 | 433.25 | 0.2696 | 156.08 | 13.144 | 433.25 | 3.939 |
| 159 | Ethyl... | C2H7N | 75047 | 45.084 | 1.1477 | 0.23182 | 456.15 | 0.26053 | 192.15 | 17.588 | 456.15 | 4.951 |
| 160 | Dieth... | C4H11N | 109897 | 73.138 | 0.85379 | 0.25675 | 496.6 | 0.27027 | 223.35 | 10.575 | 496.6 | 3.325 |
| 161 | Triet... | C6H15N | 121448 | 101.192 | 0.7035 | 0.27386 | 535.15 | 0.2872 | 158.45 | 8.2843 | 535.15 | 2.569 |
| 162 | n-Pro... | C3H9N | 107108 | 59.111 | 0.9195 | 0.23878 | 496.95 | 0.2461 | 188.36 | 13.764 | 496.95 | 3.851 |

Figure 3-12 Database System Level in ForeSee

The equipment models and, in some cases, the spool piece models

requires physical properties values in order to evaluate their respective model equations. In order to eliminate duplication of effort and to provide flexibility, the properties are evaluated on demand by another level in the hierarchy as shown in Figure 3-2, namely, *Physical Property System*.

## 3.3 Object-Oriented Design

Major requirements to be met by any advanced simulator architecture are flexibility, ease of use, maintenance and extension, portability to even advanced software and hardware platforms as well as full integration of different functionalities. However, simulator users should not bother about internal details of a simulation environment as long as they are hidden by sophisticated graphical-user-interfaces; these internals are of essential importance if far-reaching strategic decisions on the choice of particular software must be made. Ease of use, maintenance and potential extensions of an architecture as well as portability to different computer platform with possibly advanced structures become a major concern, since they will determine largely the ability of the vendor to adapt the product to the rapidly changing conditions on the software and hardware market [Rumbaugh 1991].

Object-oriented programming (OOP) took the best ideas of structured programming and combined them with several new concepts. The result was a different way of organizing a program. In the most general sense, a program can be organized in one of two ways: around its code (what is happening) or around its data (who is being affected). Using only structured programming techniques, programs are typically organized around code. This approach can be thought of as *code acting on data*. For example, a program written in a structured language such as C language is defined by its

functions, any of which may operate on any type of data used by the program.

Object-oriented programs work the other way around. They are organized around data, with the key principle being *data controlling access to code*. In an object-oriented language, the data and functions are permitted to act on that data. Thus, a data type defines precisely what sort of operations can be applied to that data.

To support the principles of object-oriented programming, all OOP languages have three traits in common: *encapsulation*, *polymorphism* and *inheritance*. Java is an object-oriented language developed in recent years and contains expected object-oriented features such as inheritance (the ability of a class to extend another class), polymorphism (the ability to separate an interface from its implementation), and encapsulation (the ability of objects to hide certain methods and instance variables from other objects) [Warren & Bishop 1999].

### 3.3.1 Encapsulation

Encapsulation is the mechanism that binds together the data and the methods that manipulate that data, and keep both safe from outside interference and misuse. If an object-oriented programming paradigm is employed, each application is encapsulated in a self-contained entity, a so-called object, which consists of its own private data (attributes) and a set of operations or methods (functions). Both may be stored in an object-oriented database. The interior properties of the object such as the type of data representation and the actual data values or the details of the methods are hidden to other objects. All the interaction between the application and the user interface (itself an object) is accomplished by synchronous and

asynchronous message passing handled by standardized communication interface. The basic structure, which is self-contained objects communicating by message passing, can be replicated within each object on lower hierarchical levels. Each object can be instantiated multiply, inheriting all the methods of the generic object, but operating on different data.

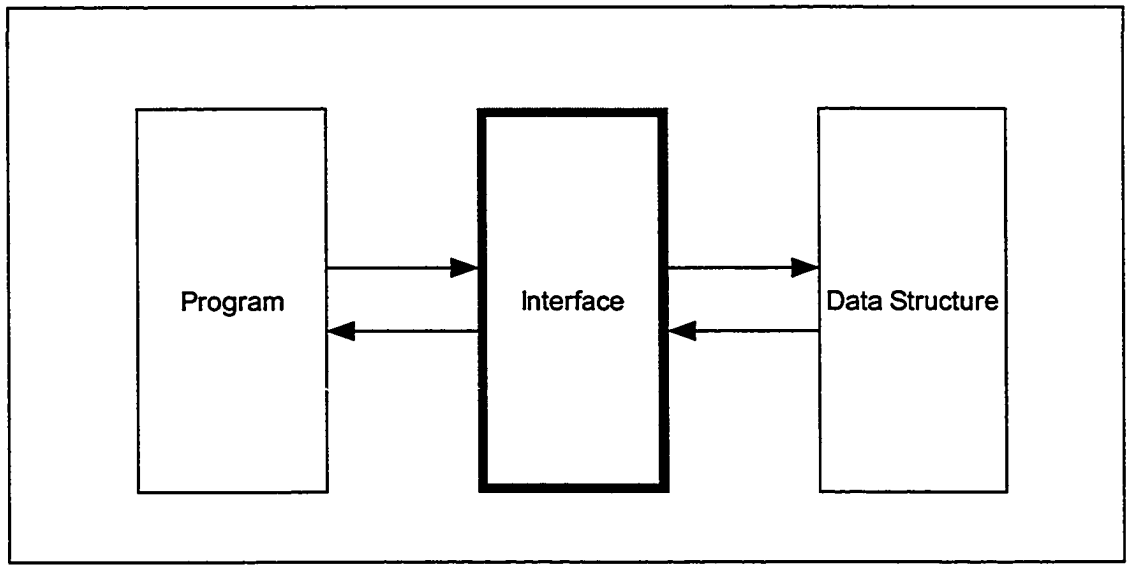



Figure 3-13 Encapsulation Structure

### 3.3.2 Inheritance

Inheritance is the characteristic that attributes and operations among object classes can be shared in a hierarchical relationship. Each class can be divided into subclasses; each subclass inherits all the properties of the superclass in addition to defining its own unique properties. Inheritance is also the process by which one object can acquire the properties of another object. This is important because it supports the concept of classification. In general, most knowledge is made manageable by hierarchical classification.

For instance, the method **getName()** in the base class **CoreModel** is inherited in the derived classes (**PlugFlowElement**, **WellMixedElement** and **StreamSplitter**).

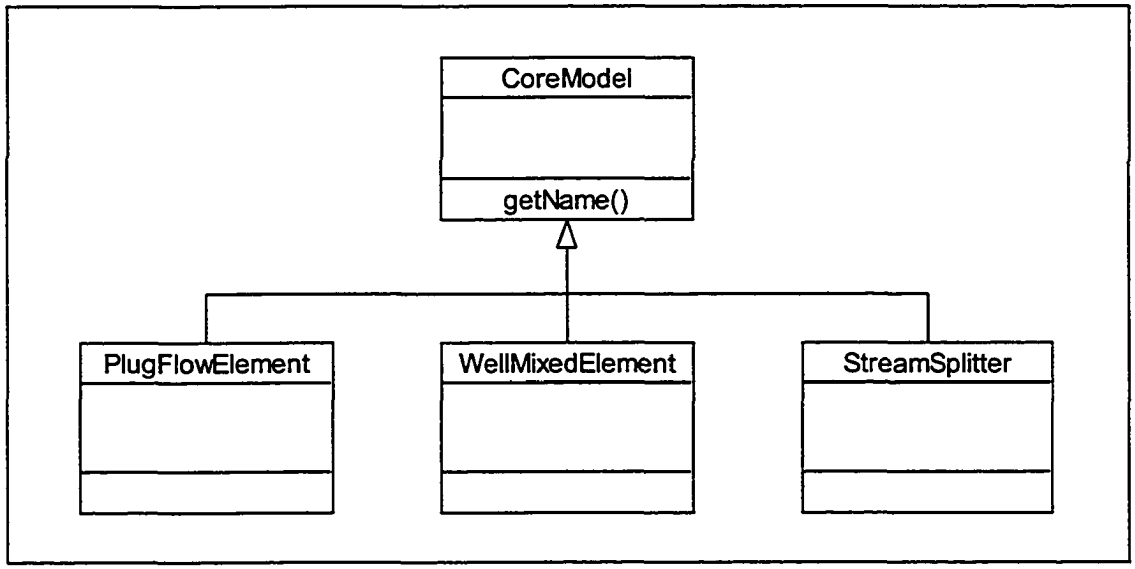



Figure 3-14 Inheritance Structure

### 3.3.3 Polymorphism

Polymorphism is the attribute that allows one interface to control access to a general class of actions, i.e. *one interface, multiple methods*. Polymorphism is the characteristic that a given operation may behave differently on different object classes. For instance, a copy command has different implementations for buffers, disk files, and tape files; a **getTransferRate()** method has different implementations for **HeatTransferConnector** class and **MassTransferConnector** class.

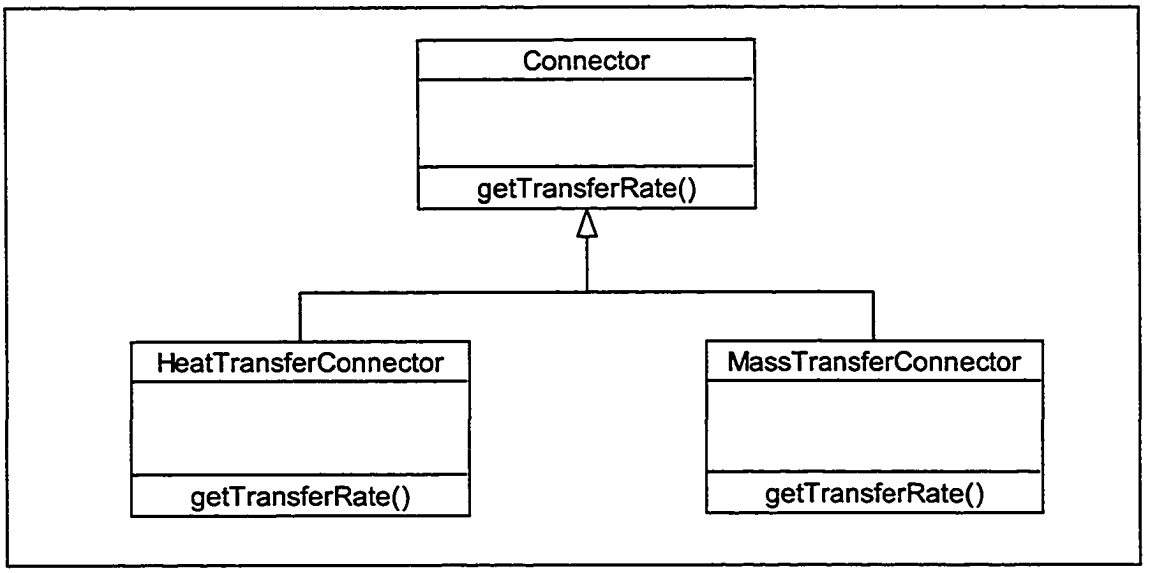



Figure 3-15 Polymorphism Structure

### 3.3.4 Java

Notions of encapsulation, inheritance and polymorphism encourage the user to group similar objects according to common attributes and define specifications with added local detail. Concepts existing in previously written code can be reused in the definition of new ones.

Examples of object-oriented languages are C++, Java and C#. Java is an object-oriented programming language developed in recent years [Gosling et al. 2000]. Java is also an interpreted language. The Java compiler generates byte-codes for the Java Virtual Machine (the Java interpreter and run-time system), rather than native machine code. To actually run a Java program, we use the Java interpreter to execute the compiled byte-codes. Because Java byte-codes are platform-independent, Java programs can run on any platform as long as that system implements the Java Virtual Machine. This is a particularly important for applications distributed over

different platforms or Internet.

Object-oriented modeling facilitates modeling through decomposition, reuse and high-level description. The decomposition of large and complex models into smaller submodels increases readability and understanding of the model. A submodel in a composite model can itself be a composite model, creating a structure hierarchy of submodels. Therefore, we choose object-oriented and platform-independent programming language Java to implement our current dynamic modeling and simulation system (ForeSee).

## 3.4 Conclusions

There are a number of conclusions that can be drawn from the work reported in this chapter.

- Combining the advantages of both, the flexibility of the equation-oriented modeling tools and the ease-of-use of the block-oriented modeling tools, a dynamic modeling and simulation environment called ForeSee has been developed.
- The structure of ForeSee is hierarchical. The process model is decomposed the equipment models, and the equipment models can be decomposed into a set of four types of modeling components. The modeling components are aggregated into the equipment models, and finally assembled into a complete process model.
- ForeSee is object-oriented and implemented by the object-oriented and platform-independent programming language Java.

# Chapter 4 Modeling Components of the Simulator

## 4.1 Introduction

In general, the model equations in chemical engineering could be divided into the three categories, that is, *conservation equations*, *constitutive equations*, and *constraint equations*. Based on these three types of model equations, the corresponding three types of modeling components (*core models*, *connectors*, and *coordinators*) have been developed. The core models usually describe conservation equations. What the connectors can describe are usually constitutive equations except state equations, which is handled by the physical property system. The coordinators are used to handle the constraint equations.

## 4.2 Model Equations

A model is an abstraction of reality that can be used to represent some aspects of the real process, which is considered important for the modeler. Aspects that are important for the modeler define the goals or objectives of the modeler. If the main objective of a model is to predict the behavior of a system with reasonable accuracy and within a reasonable period of time, a model should encapsulate all the information, which is considered necessary to describe the reality within the required level of accuracy. Usually the information includes such as [Presig 1995; Jensen & Gani 1998]:

- State variable transformations provide the link between the

internal state and the state variables used in transfer laws, physical property models, kinetic laws and other such relations.

- Transfer laws describe how the extensive quantities are being transferred between phases.
- Production term generally describes mass conversation in a system by the means of a chemical or biological reaction.
- Physical properties.
- Geometrical properties.

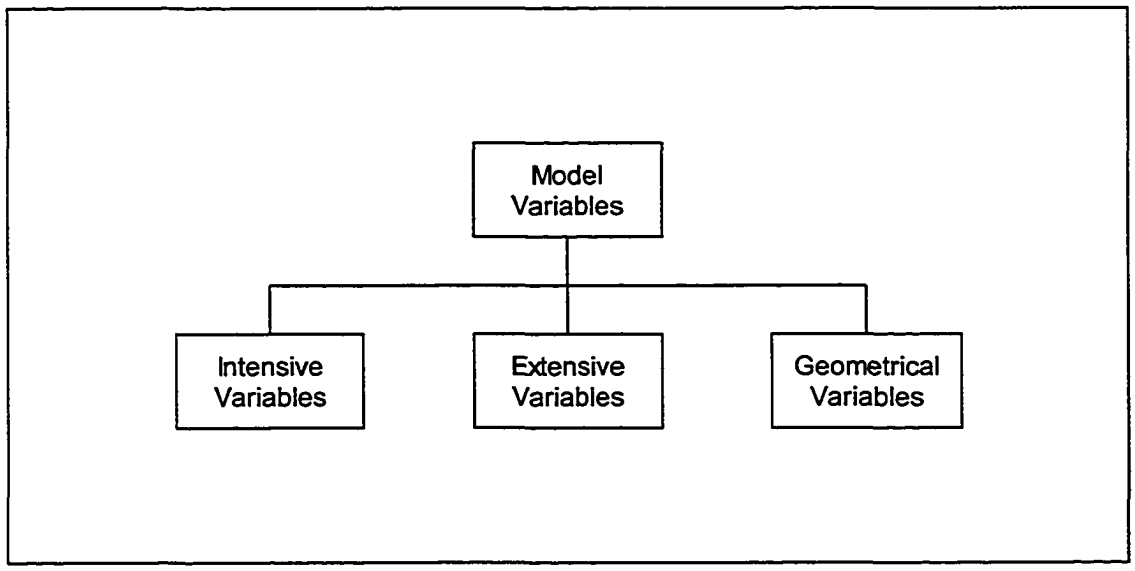



Figure 4-1 Classification of Model Variables

Some of this information is of empirical nature. Examples are kinetic laws and physical property relations. Others are strictly mathematical such as state variable transformations. Others are somewhere in-between such as transfer laws.

There might be three types of variables available in the dynamic modeling and simulation. All of them are:

- Extensive variables

- Intensive variables
- Geometrical variables

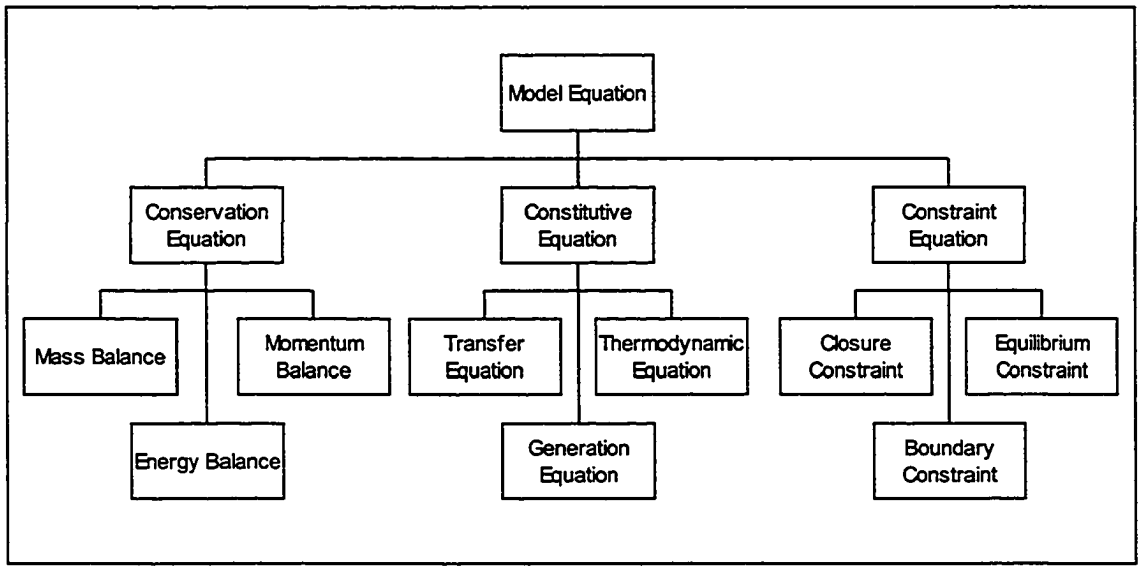



Figure 4-2 Taxonomy of Process Model Equations

The classification of the various model equations is essential for getting an overview of what type of model equations a modeling tool is able to handle. Classifying the model equations typically encountered in chemical processes according to the physical foundations reveals three types of equations:

- Conservation Equations (Balance Equations)
- Constitutive Equations
- Constraint Equations

### 4.2.1 Conservation Equations

Conservation equations or balance equations define the conservation of extensive variables over a specified region (control shell or control volume).

Extensive variables (for instance, mass, energy and momentum) characterize the process and satisfy a conversation principle that allows associating to each extensive variable and each elementary subsystem a *balance equation* of the form [Drengstig et al. 1997; Weiss & Preisig 1997]:

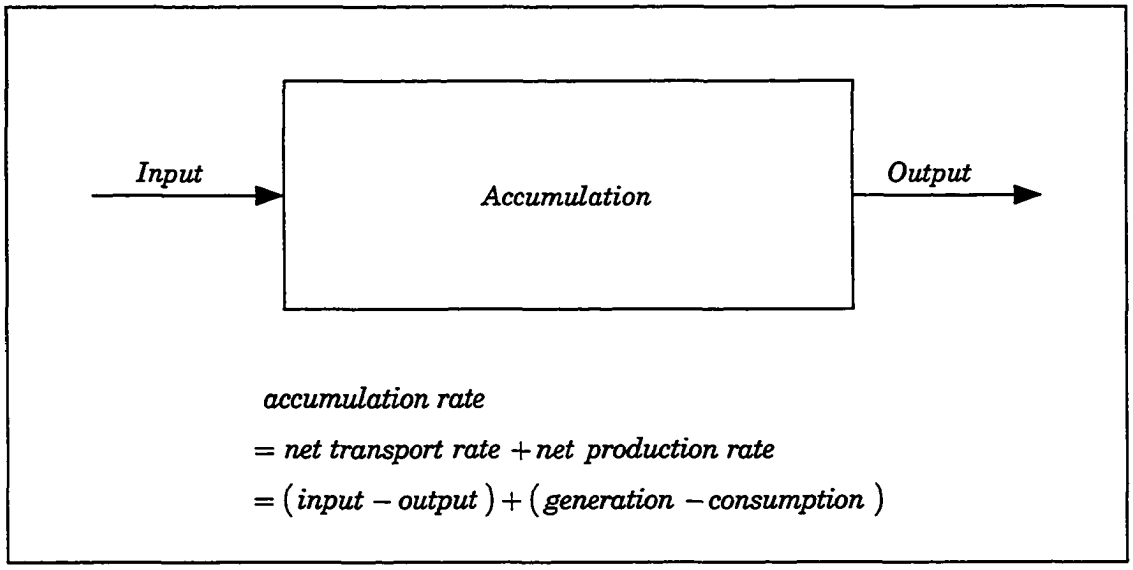



Figure 4-3 Derivation of the Generic Form of the Balance Equation

Obviously, the net flow rate consists of the sum of all inflow rates minus the sum of all outflow rates and a similar remark is true for the net production rate, which consists of the effects of interior production and consumption phenomena (the appearing or disappearing of chemical species or energy within the system boundaries).

The balance equations express the overall extensive structure of a chemical process model. Balance equations for the species, for the energy, and for the momentum should serve as a sufficient number of equations to determine the physical condition of a given region in equilibrium. According to Duhem's theorem [Smith & Van Ness 1987], the number of independent variables in a closed region in equilibrium is $n_c + 2$ ($n_c$ is equal to the

number of components). Because the species balances, energy balance, and the momentum balance are $n_c + 2$ independent equations, the physical state of a closed region in equilibrium must be fully determined by this set of balance equations.

Table 4-1 The Components of Balance Equations

| Accumulation | Transport | Production |
|---|---|---|
| Mass<br>(Chemical Species) | Diffusion<br>Convection | Surface Reaction<br>Volume Reaction |
| Energy | Conduction<br>Convection<br>Radiation | Electrical Generation<br>Mechanical Generation |
| Momentum | Convection | Surface Force<br>Volume Force |

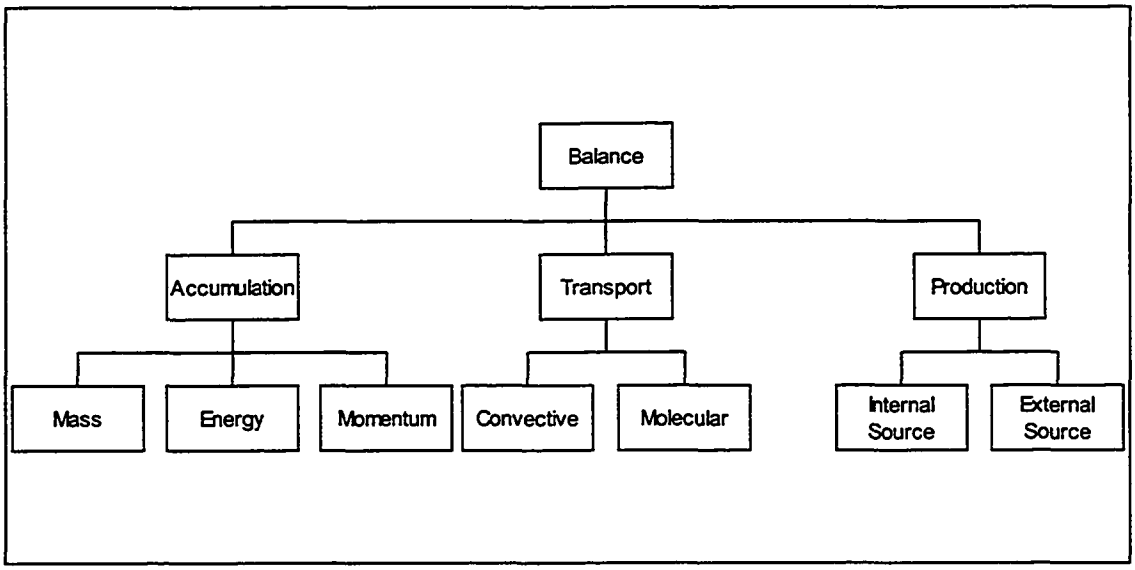



Figure 4-4 Decomposition of Balance Equations

Figure 4-4 represents the various terms (accumulation, transport, and production) in the balance equations. These terms are normally described by constitutive equations, i.e., the quantitative size of these terms are usually

related to intensive characterizing properties through constitutive relations.

- **Mass Balance**

If the fluid consists of more than one component, the principle of mass conservation applies to each individual component in the mixture as well as to the mixture as a whole. For each component $i$, the principle of mass balance results in an equation of the form [Bird et al. 1960; Slattery 1999]

$$\frac{\partial}{\partial t}\rho_i = -\nabla \cdot (\rho_i \boldsymbol{v} + \boldsymbol{j}_i) + r_i \qquad i = 1,2,\cdots,n_c \tag{4-1}$$

where $\rho_i$ is the mass density of species $i$, $\boldsymbol{j}_i$ is the diffusive mass flux, $r_i$ is the rate of production of component $i$ due to chemical reaction. Addition of all $n_c$ equations of this kind gives the equation of continuity for the mixture, described in Equation (4-2).

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \boldsymbol{v} = 0 \tag{4-2}$$

For system in which molecular diffusion occurs, the mass flux is the sum of the bulk flow of species $i$ and the diffusional flux of that species.

$$\boldsymbol{n}_i = \rho_i \boldsymbol{v} + \boldsymbol{j}_i \qquad i = 1,2,\cdots,n_c \tag{4-3}$$

Equation (4-1) may be rewritten as

$$\frac{\partial}{\partial t}\rho_i = -\nabla \cdot \boldsymbol{n}_i + r_i \qquad i = 1,2,\cdots,n_c \tag{4-4}$$

- **Momentum Balance**

Momentum flows into and out of the volume element by two mechanisms: by *convection* (due to the bulk flow of the fluid) and by

*molecular transfer* (due to the velocity gradients in the system). The principle of conservation of momentum for the mixture dictates as follows:

$$\frac{\partial}{\partial t}\rho \boldsymbol{v} = -\nabla \cdot (\rho \boldsymbol{v}\boldsymbol{v} + \boldsymbol{\tau} + p\boldsymbol{\delta}) + \rho \boldsymbol{g} \tag{4-5}$$

where $\boldsymbol{\tau}$ is shear stress tensor, $p$ is the static pressure, and $\boldsymbol{\delta}$ is the unit tensor. We define the following momentum flux:

$$\boldsymbol{\phi} = \rho \boldsymbol{v}\boldsymbol{v} + \boldsymbol{\tau} + p\boldsymbol{\delta} \tag{4-6}$$

Equation (4-5) may be rewritten as

$$\frac{\partial}{\partial t}\rho \boldsymbol{v} = -\nabla \cdot \boldsymbol{\phi} + \rho \boldsymbol{g} \tag{4-7}$$

- **Energy Balance**

The principle of conservation of energy for the mixture is:

$$\frac{\partial}{\partial t}\rho\left(U + \tfrac{1}{2}v^2\right) = -\nabla \cdot \left[\rho\left(U + \tfrac{1}{2}v^2\right)\boldsymbol{v} + \boldsymbol{q} + (\boldsymbol{\tau} + p\boldsymbol{\delta}) \cdot \boldsymbol{v}\right] + (\rho \boldsymbol{v} \cdot \boldsymbol{g}) \tag{4-8}$$

where $\boldsymbol{q}$ is the multicomponent energy flux relative to the mass average velocity $\boldsymbol{v}$. We define the following energy flux:

$$\boldsymbol{e} = \rho\left(U + \tfrac{1}{2}v^2\right)\boldsymbol{v} + \boldsymbol{q} + (\boldsymbol{\tau} + p\boldsymbol{\delta}) \cdot \boldsymbol{v} \tag{4-9}$$

Equation (4-8) may be rewritten as follows:

$$\frac{\partial}{\partial t}\rho\left(U + \tfrac{1}{2}v^2\right) = -\nabla \cdot \boldsymbol{e} + \rho \boldsymbol{v} \cdot \boldsymbol{g} \tag{4-10}$$

### 4.2.2 Constitutive Equations

Constitutive equations define extensive variables in terms of intensive and geometric variables (for example, flux and thermodynamic equations). The *constitutive equation* has the following form:

$$\mathbf{y} = f(\mathbf{x},\mathbf{z}) \tag{4-11}$$

where $\mathbf{y}$ is a set of extensive variables (e.g., internal energy, enthalpy), $\mathbf{x}$ a set of intensive variables (e.g., composition, temperature, and pressure) and $\mathbf{z}$ a set of geometric variables (e.g., volume, void fraction), respectively.

The geometric variables help to define the basic entity of the model that is the elementary subsystem. Its main property is that physical properties are assumed constant across the spatial extent of an elementary subsystem.

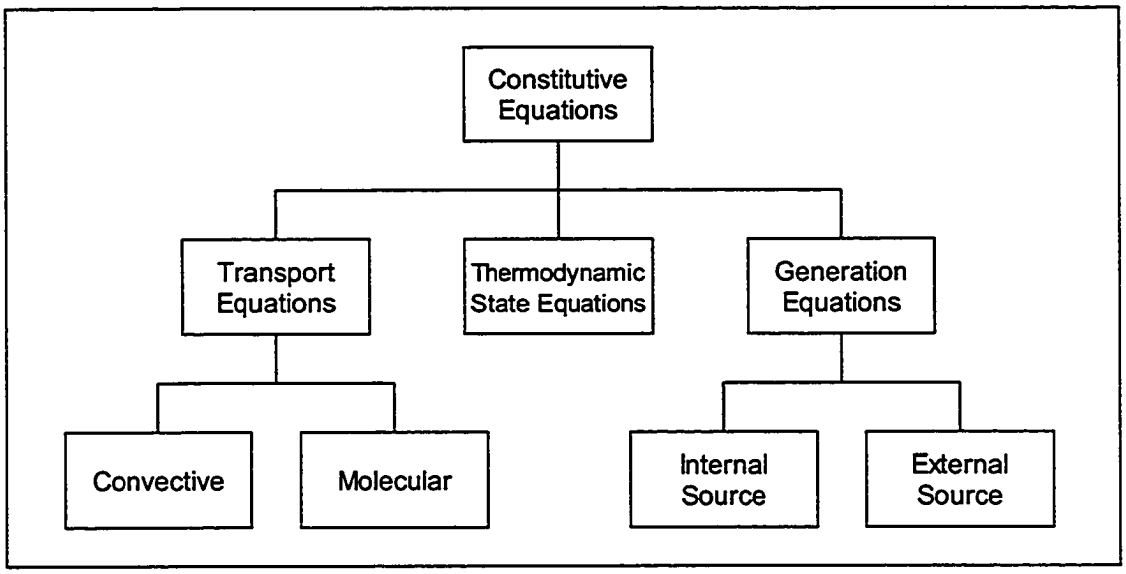



Figure 4-5 Decomposition of Constitutive Equations

In principle, to each connection, we might associate an intensive

variable that is the driving force for the flow through that intensive variable.

- **Transfer Equations**

Transfer equations are also referred to as flux equations. Transfer equations define extensive rates of transfer in terms of the driving forces of the intensive variables. The *transfer equation* has the following form [Weiss & Preisig 1997]:

$$\textit{transfer rate of } y = \textit{transfer coefficient} \times \textit{driving force of } x \qquad (4\text{-}12)$$

where $y$ is a generic extensive variable, $x$ a generic intensive variable. The following table shows the rough relations between fluxes and driving forces.

Table 4-2 The Rough Relations between Fluxes and Driving Forces

| Fluxes | Driving Forces |
|---|---|
| Mass | Concentration Gradient (Fick's Law) |
| Energy | Temperature Gradient (Fourier's Law) |
| Momentum | Velocity Gradients (Newton's Law) |

The model is almost complete if we specify the transfer rates by relations often called transfer laws and the production rates. These, however, are determined by intensive variables, which characterize the process locally. In most practical cases, we can distinguish among variables, which are involved in the transfer law acting as a driving force, or potential for the flow in the sense that the flow rate is zero if the difference across the boundary of this potential is zero, and the direction of the flow is such that the difference decreases. However, the intensive variables have a far more important role to play in the modeling procedure. They are usually directly measurable variables. Mass and energy are almost never directly measured, but are

deduced from densities, temperature, pressure, etc., combined with geometric data (e.g., volume). It is always possible to choose a minimal set of intensive variables that, together with the geometric variables, uniquely determine the extensive variable.

- **Generation Equations**

The rate of a homogeneous reaction is determined by the composition of the reaction mixture, the temperature, and the pressure.

$$r = f(tempeature, pressure, composition) \tag{4-13}$$

Consider the reactions of $n_c$ chemical species participating in $n_r$ independent chemical reactions. By independent it is meant that no one of the stoichiometric equations can be derived from the others by a linear combination [Froment & Bischoff 1990; Levenspiel 1972].

$$\sum_{i=1}^{n_c} \alpha_{ij} A_i = 0 \qquad j = 1,2,\cdots,n_r \tag{4-14}$$

where the stoichiometric coefficients $\alpha_{ij}$ are taken positive for products and negative for reactants, $A_i$ represents chemical species $i$.

The following equalities exist between the different rates:

$$-\frac{1}{\alpha_{1j}}\frac{dN_{1j}}{dt} = -\frac{1}{\alpha_{2j}}\frac{dN_{2j}}{dt} = -\frac{1}{\alpha_{ij}}\frac{dN_{ij}}{dt} \tag{4-15}$$

where $N_{ij}$ represents the molar amount of one of the chemical species $A_i$ in the $j^{th}$ reaction.

The rate of reaction is generally expressed on an intensive basis as follows:

$$r_j \equiv \frac{1}{V}\frac{1}{\alpha_{ij}}\frac{dN_{ij}}{dt} \tag{4-16}$$

- **Thermodynamic State Equations**

Thermodynamic state equations introduce the coupling between intensive characterizing properties (composition, temperature, and pressure) and derived thermodynamic properties such as enthalpy, density, and viscosity. Any equation that defines such as a relationship is referred to as an equation of state. Because the macroscopic properties of homogeneous *PVT* systems in equilibrium states can be expressed as functions of pressure, temperature, and composition only, the most general form of an equation of state is as follows:

$$f(y,p,T,x_1,x_2\cdots,x_n) = 0 \tag{4-17}$$

where $y$ is any derived thermodynamic property.

### 4.2.3 Constraint Equations

Constraint equations are the equations where some constraint conditions are applied to the system. Constraint equations can be roughly grouped into closure, equilibrium, and boundary. The decomposition of constraint equations is shown in Figure 4-6.

- **Closure Constraints**

Closure equations are equations where one variable is fixed via others. These equations are usually related or based on summation over component properties. Typical example of closure equations is:

$$\sum_{i=1}^{n_c} x_i = 1 \tag{4-18}$$

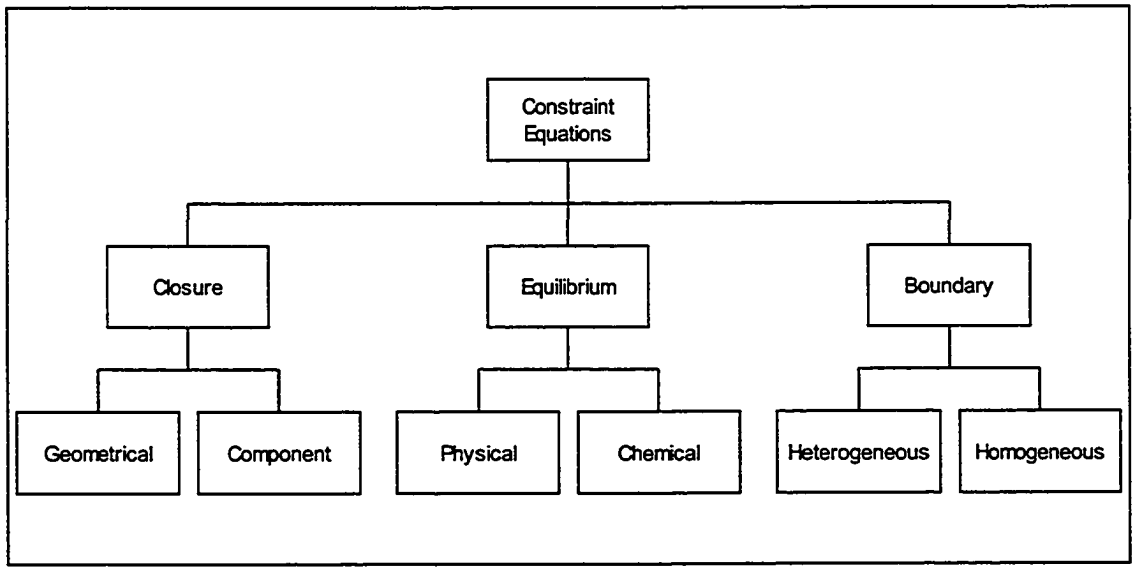



Figure 4-6 Decomposition of Constraint Equations

- **Equilibrium Constraints**

Equilibrium is defined as the limiting state where the net change with respect to all independent coordinates in the system is equal to zero. It is well known that this condition is fulfilled when the Gibbs energy is at the global minimum. Therefore, we obtain a required condition for multiphase equilibrium:

$$dG = VdP - SdT + \sum_{i=1}^{n_c} \mu_i dx_i = 0 \tag{4-19}$$

$$\mu_i \equiv \left[\frac{\partial G}{\partial x_i}\right]_{P,T} \tag{4-20}$$

It can be shown that multiple phases at the same T and P are in

equilibrium when the chemical potential $\mu_i$ of each species is the same in all phases [Smith & Van Ness 1987]:

$$\begin{aligned} T^{\alpha} &= T^{\beta} = \cdots = T^{\pi} \\ P^{\alpha} &= P^{\beta} = \cdots = P^{\pi} \\ \mu_i^{\alpha} &= \mu_i^{\beta} = \cdots = \mu_i^{\pi} \end{aligned} \tag{4-21}$$

- **Boundary Constraints**

Boundary constraints can be either heterogeneous or homogeneous. While heterogeneous boundary constraints describe the transfer of matter between phase boundaries, homogeneous boundary constraints describe limiting conditions within phase boundaries.

## 4.3 Modeling Components

While each of chemical processes is functionally quite different from the others, if we look at the fundamental components of each, these are quite similar in their physical behavior. This means that simulation models of complex systems can be built up hierarchically starting with relatively simple models of these fundamental components. These fundamental modeling components can be categorized into four types (*containments, core models, connectors,* and *coordinators*), each of which describes a different aspect of system behavior. Together they provide a comprehensive framework for describing the dynamic behavior of actual items of equipment [Rinard 1998, 1996].

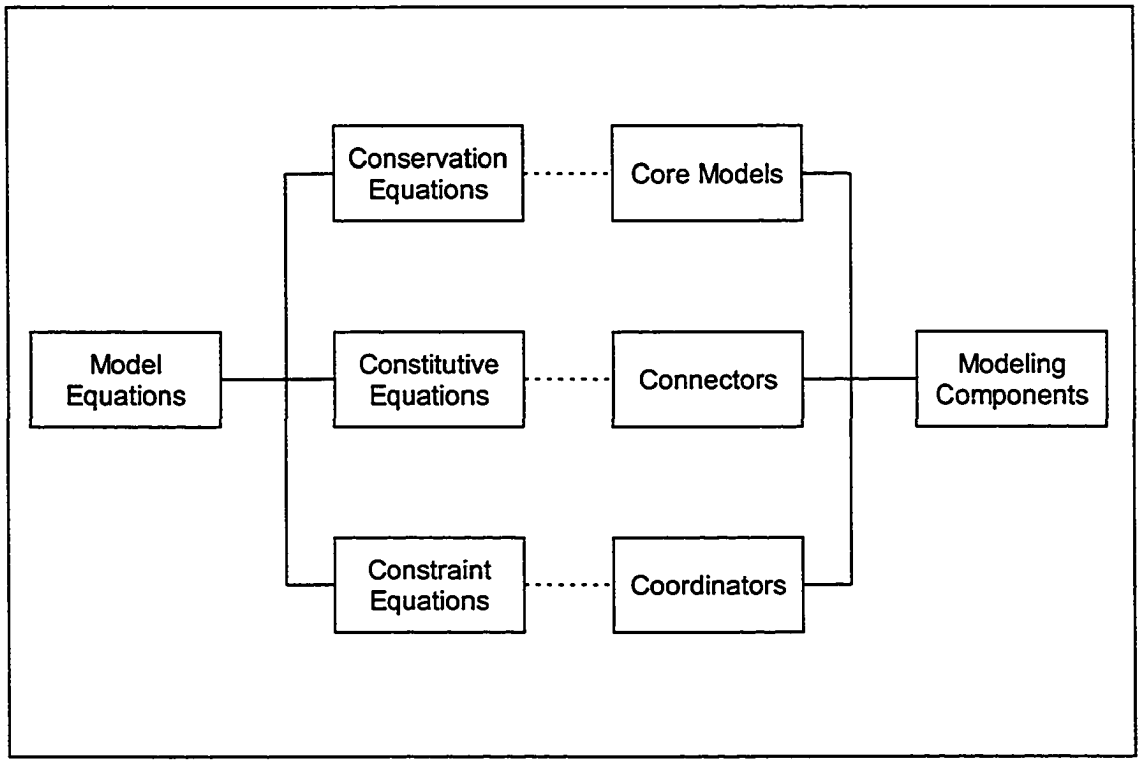



Figure 4-7 Relationship between Model Equations and Modeling Components

### 4.3.1 Containments

Containment is the *physical topology* of an equipment item. A containment is also the fixed space in which a given set of dynamic processes involving one or more phases taken place. The containment component provides for the physical description of the piece of equipment being modeled. For instance, if it is a mixing tank that is of interest, the physical description might include the tank orientation, tank diameter, tank height or length, the elevation of the tank above the datum level, the elevations of the piping nozzles for the inlet and outlet streams, and the pad pressure, etc. Not all of this information is needed if we are in the early stages of developing a

flowsheet simulation model. However, as the plant design progresses, more of this information might be required.

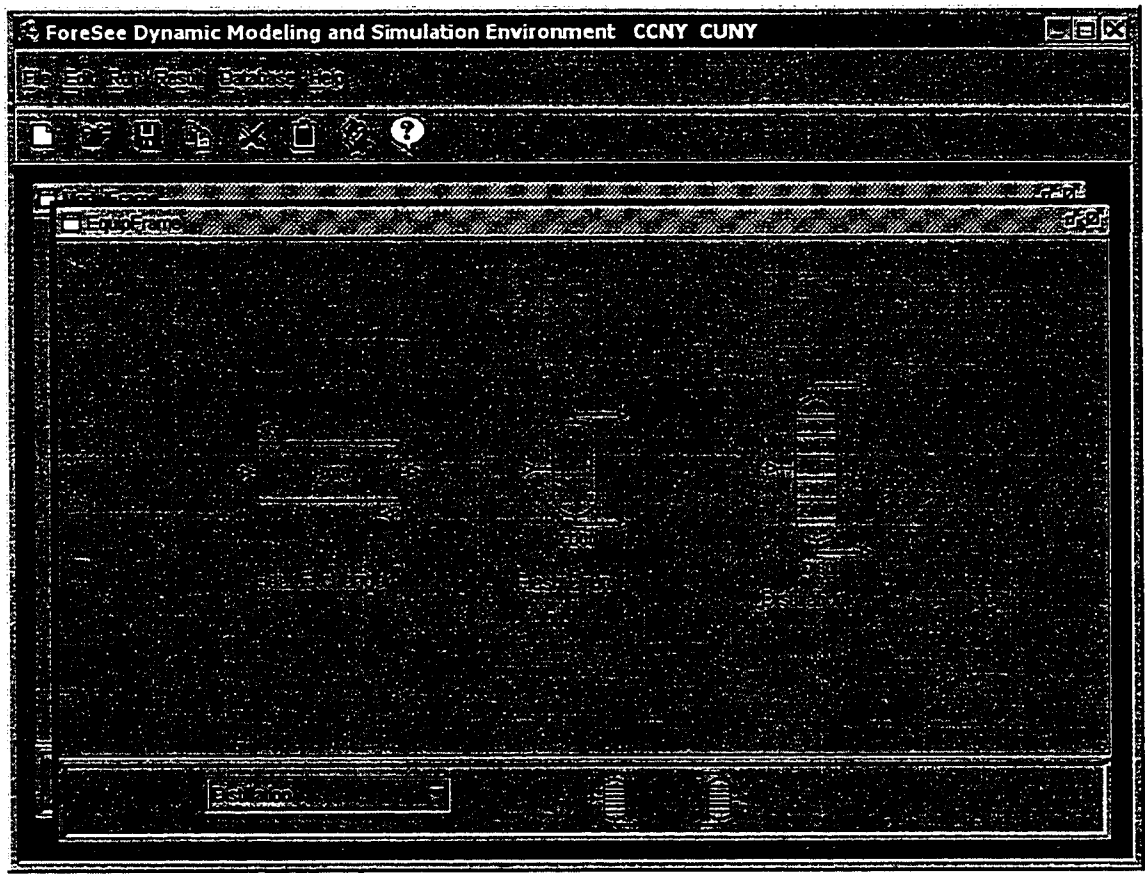



Figure 4-8 Containment Icon Samples in ForeSee Prototype

Another example is a shell-tube heat exchanger. Here the containment information might include the number of tubes, tube diameter, tube wall thickness, tube length, shell diameter, shell wall thickness, and the number of tube passes, etc. For a very detailed model, it might also include baffle type, baffle spacing, and baffle cut, etc.

### 4.3.2 Core Models

A core model describes the dynamic behavior of the *intensive variables*

(e.g. composition, temperature, and pressure, etc.) which characterize a specific phase within its phase volume. In general, conservative equations are determined by core models.

An important characteristic of core model is the mixing pattern that the phase is assumed to follow. The following are the core models that have been developed so far.

- **Well-Mixed Element (WME)**

  This core model is also referred to as a *continuous stirred tank* (CST) when there is bulk flow in and out. There is only one feed stream and only one exit stream. As its name suggests, the contents are well stirred and uniform throughout in a tank. Thus, the exit stream from this tank has the same composition, pressure, and temperature as the fluid within the tank. The WME model can, in principle, be used to model any system in which each dynamic volume of phase can be assumed to be well mixed.

- **Plug Flow Element (PFE)**

  This model differs from the others previously introduced in that it allows for variation of the process variables such as temperature, pressure, and composition along a spatial dimension. The basic assumptions for PFE model are: (1) There is no fluid mixing in the axial direction of flow. (2) The fluid is well-mixed in the radial direction; there are no radial gradients of pressure, temperature, or composition. (3) The only resistance to heat or mass transfer is at wall or phase boundary. The necessary and sufficient condition for plug flow is for the residence time in the reactor to be the same for

all elements of fluid.

- **Stream Mixer (SM)**

  It is a zero-holdup mixer and provides for mixing two or more multi-phase streams into one multi-phase stream that has the same phase state as the input streams instantaneously. It is also called *zero volume mixer* (ZVM). By zero volume, we mean that it has no dynamics. A stream mixer can have many input streams but, of course, only one output stream. It might be used, for instance, to model an in-line mixer that has a very small residence time compared other equipment items in the process.

- **Stream Splitter (SS)**

  It is a zero-holdup splitter and separates one multi-phase stream into two or more different multi-phase streams that have the same composition, pressure, and temperature as the input stream. The mass flow rates cannot be determined from the input stream. A stream splitter can have many output streams but, of course, only one input stream.

- **Substream Mixer (SSM)**

  It is a zero-holdup mixer and provides for mixing two or more different single-phase streams into one multi-phase stream instantaneously.

- **Substream Splitter (SSS)**

  It is a zero-holdup splitter and separates one multi-phase stream

into two or more different single-phase streams that might have the different composition. The mass flow rates can be determined from the input stream.

Other core models are possible but many of these can be suitably represented by combinations of those already defined above. Keep in mind that these are all based on macro mixing models. In the future, as both the computer power and the computational techniques improve, core models based on computational fluid dynamics formulations may become practical for process dynamic simulation.

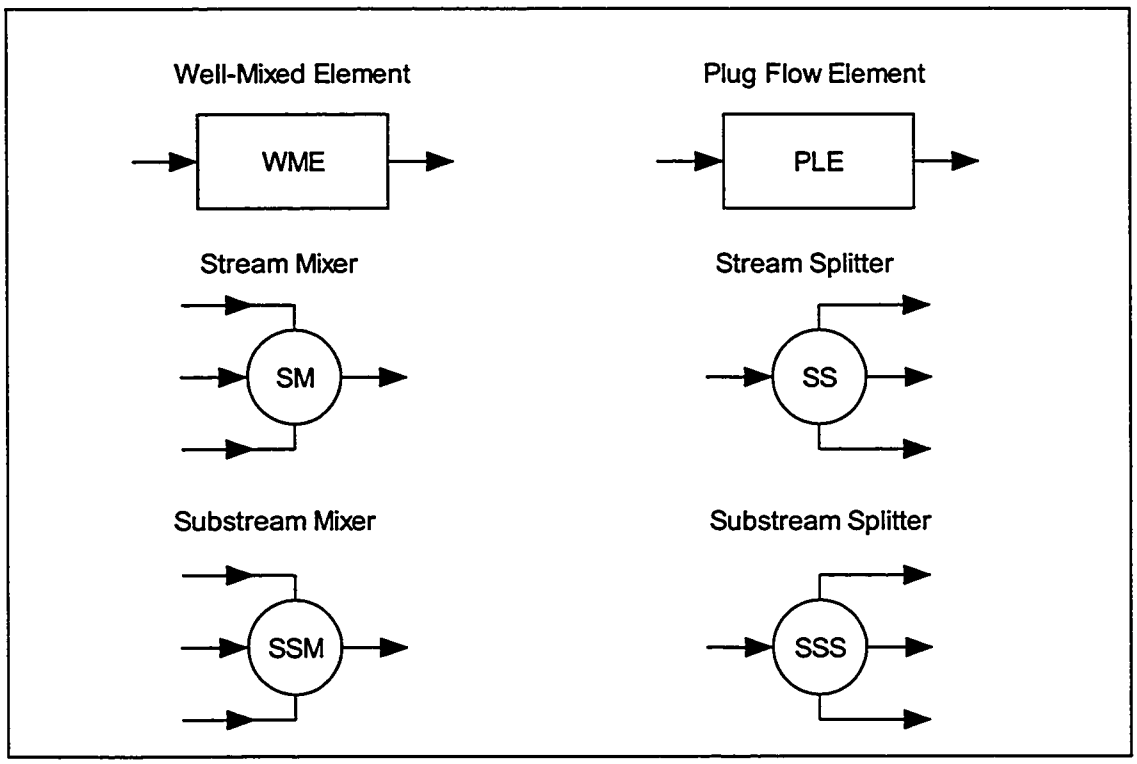



Figure 4-9 Core Model Icon Samples

### 4.3.3 Connectors

A connector allows for the transfer of heat or mass between two phases or the transformation of material within a phase or between two or more phases by either chemical or biochemical reactions. If two or more phases are present in an equipment item, then there will probably be the transfer of the *extensive variables* (e.g. mass, energy, etc.) between phases. The transfer rates of the extensive variables between phases are determined by another type of component model known as a connector. Many of these will be quite simple. However, some, such as those required for heterogeneous catalysis, might be rather complex. In general, constitutive equations are determined by connectors.

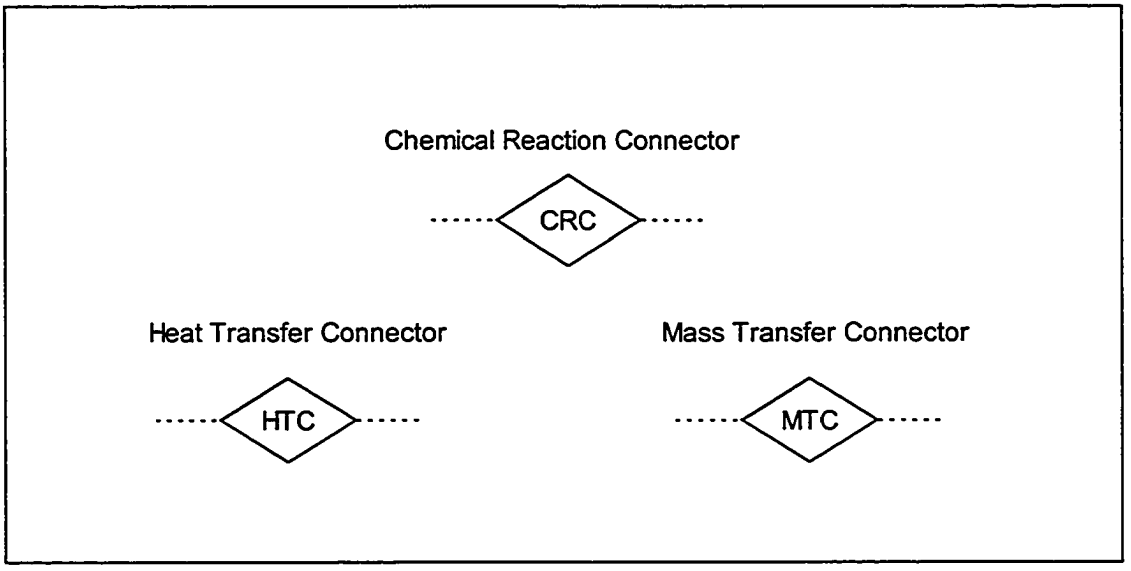



Figure 4-10 Connector Icon Samples

- **Heat Transport Connector (HTC)**

  This connector is used to determine the heat transfer rate between

phases due to the phase temperature difference.

- **Mass Transport Connector (MTC)**

  This connector is used to determine the flux of mass transfer between phases due to the partial fugacity difference of each component.

- **Chemical Reaction Connector (CRC)**

  This connector is used to determine the chemical reaction rate between or within phases due to the chemical potential difference.

### 4.3.4 Coordinators

The function of a coordinator is to determine the dynamic behavior of the *geometric variables* (e.g. volume, surface, etc.) possessed by the various material phases within a containment. A phase is a distinct material characterized by its thermodynamic intensive variables, i.e. composition, temperature, and pressure. It may be a vapor, liquid, or a solid. A simple example is a coordinator to determine liquid volume within a tank. A more complex example is a coordinator that determines the void fraction (porosity) in a fluidized bed. In general, constraint equations are determined by coordinators.

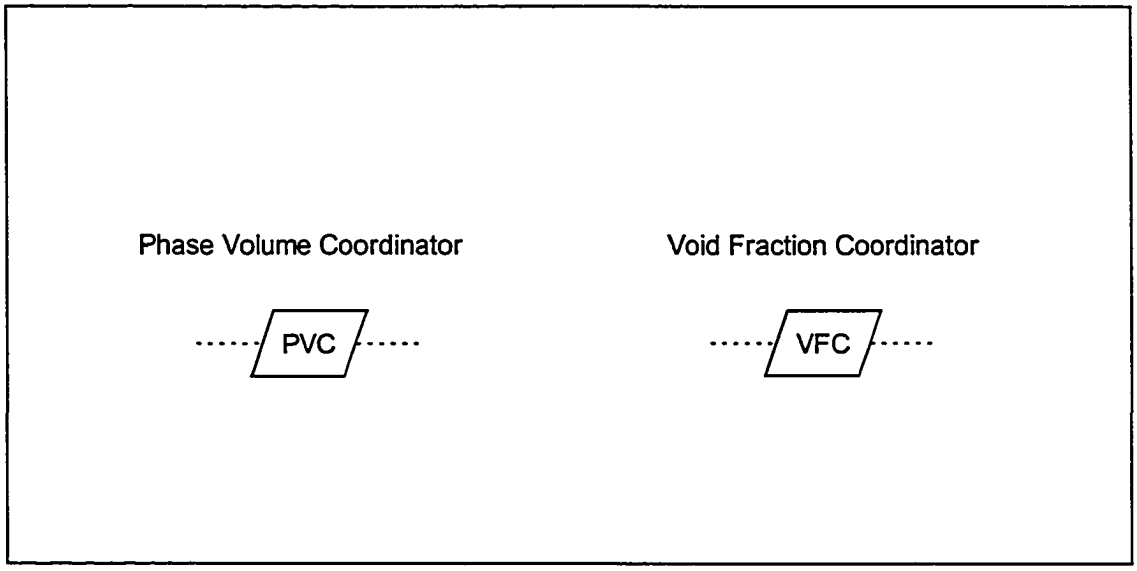



Figure 4-11 Coordinator Icon Samples

- **Phase Volume Coordinator (PVC)**

In many types of equipment, the entire equipment volume is not occupied by a single phase. For instance, in a decanter, there are two liquid phases and, depending upon how the liquid levels are controlled, the volume occupied by each phase may vary with time. In a heat exchanger, there are potentially any number of phases, the minimum being the flowing phase contained in the tubes and, similarly, the flowing phase contained in the shell. If the thermal capacities of the tube wall or the shell wall are considered significant, the one or both of the solid phases may also be included in the model.

Two types of coordinators are suggested by these two examples. In the case of the decanter, the coordinator is dynamic since the phase volumes can vary with time. In the case of the heat exchanger, the coordinator is static since the phase volumes are fixed.

- **Void Fraction Coordinator (VFC)**

  This coordinator is used to determine the void fraction of the phase. In a fluidized bed, the coordinator is dynamic since the porosity can vary with time.

### 4.3.5 Relationships

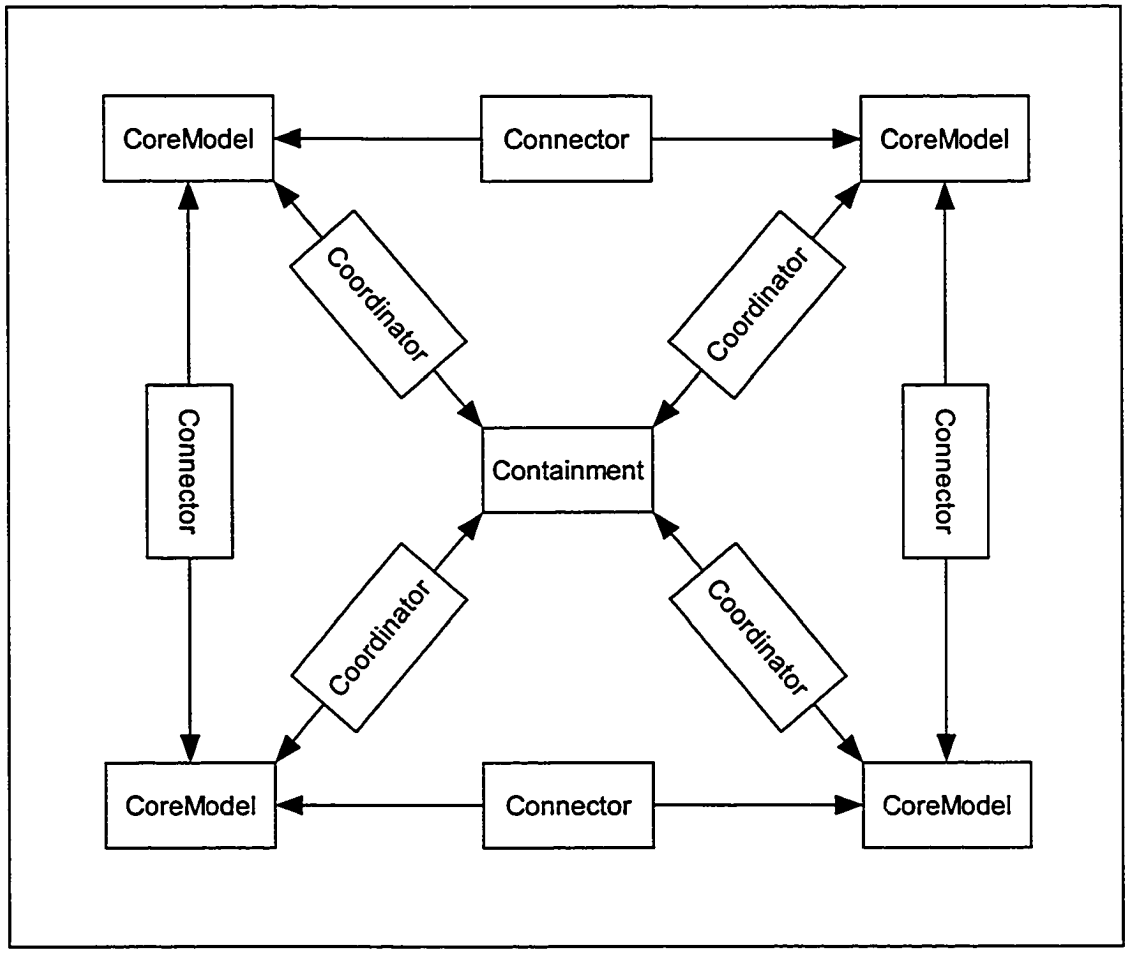



Figure 4-12 Relationships among Modeling Components

The relationships among modeling components (containments, core

models, connectors, and coordinators ) is shown in Figure 4-12.

## 4.4 Conclusions

- The majority of chemical engineering model equations could be divided into the three categories (conservation, constitutive, and constraint) with corresponding types of variables (intensive, extensive, and geometrical).
- Based on three categories of model equations (conservation, constitutive, and constraint), the corresponding three types of modeling components (Core Models, Connectors, and Coordinators) are constructed.
- Since library of modeling components is compact, creating new equipment models is straightforward and relatively quick. It will be shown in the following chapter that a lot of equipment models can be established from the small and compact library of modeling components.
- All core models are developed based on macro mixing models so far. In the future, as both the computer power and the computational techniques improve, core models based on computational fluid dynamics formulations may become practical for process dynamic simulation.

# Chapter 5 Illustrative Example

## 5.1 Introduction

We choose flash drum as an illustrative example to demonstrate how equipment model can be visually assembled from an interconnected set of modeling components (containments, core models, connectors, and coordinators).

## 5.2 Flash Drum

### 5.2.1 Containment

Flash distillation is a continuous and single-stage partial vaporization without reflux. Flash distillation consists of vaporizing a define fraction of the liquid in such a way that the evolved vapor is almost in equilibrium with the residual liquid, separating the vapor from the liquid, and condensing the vapor. An intimate mixture of vapor and liquid enters the vapor separator of flash drum, in which sufficient time is allowed for the vapor and liquid portions to separate. Because of the intimacy of contact of liquid and vapor before separation, the separated streams are almost in equilibrium [McCabe et al. 1993]. Containment describes the physical size of flash drum, for example, the height and the diameter of flash drum.

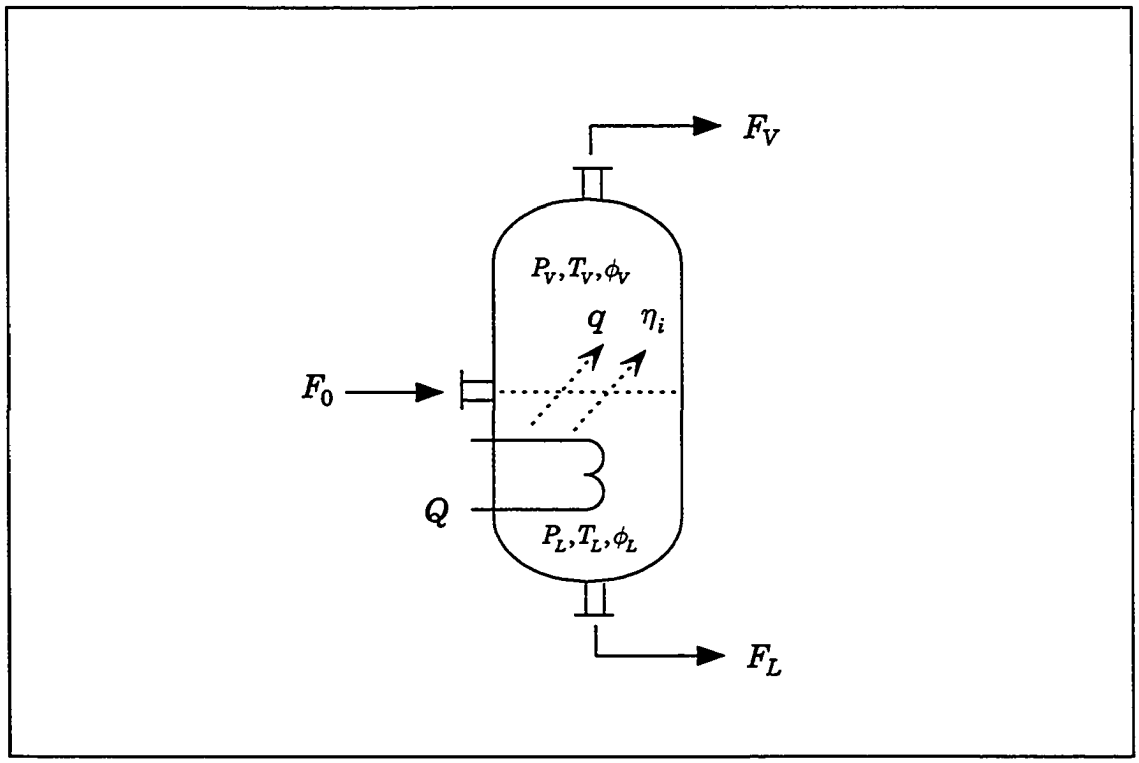



Figure 5-1 Flash Drum

The notation in this example is as follows:

$F_0$ = molar flow rate of input [kmol/s]

$F_L$ = molar flow rate of liquid output [kmol/s]

$F_V$ = molar flow rate of vapor output [kmol/s]

$\hat{f}_i^L$ = partial fugacity of component $i$ in the liquid phase [kPa]

$\hat{f}_i^V$ = partial fugacity of component $i$ in the vapor phase [kPa]

$f_i^L$ = fugacity of pure component $i$ at the liquid temperature [kPa]

$f_i^V$ = fugacity of pure component $i$ at the vapor temperature [kPa]

$\bar{H}_i$ = partial molar enthalpy of the $i^{th}$ component [kJ/kmol]

$\bar{H}_i^*$ = partial molar enthalpy of the $i^{th}$ component crossing the interface [kJ/kmol]

$K_i$ = mass transfer coefficient between phases [$kmol/m^2 \cdot s \cdot kPa$]

$P_L$ = pressure of the liquid phase [kPa]

$P_V$ = real pressure of the vapor phase [kPa]

$q$ = heat transfer rate between phases [$kJ/m^2 \cdot s$]

$Q$ = heat input [$kJ/m^2 \cdot s$]

$T_L$ = temperature of the liquid phase [K]

$T_V$ = temperature of the vapor phase [K]

$x_i$ = liquid molar fraction.

$y_i$ = vapor molar fraction.

$\gamma_i$ = activity coefficient of component $i$ in the liquid mixture

$\hat{\phi}_i$ = fugacity coefficient of component $i$ in the vapor mixture

$\phi_V$ = vapor phase volume [$m^3$]

$\phi_L$ = liquid phase volume [$m^3$]

$\eta_i$ = mass transfer rate of $i^{th}$ component [$kmol/m^2 \cdot s$]

### 5.2.2 Core Model

The decomposition of flash drum modeling components is summarized in Figure 5-2 and Figure 5-3. Based on the decomposition diagram, the dynamic equipment model of flash drum can be established from the interconnected set of core models, connectors, and coordinators. There are three core models in flash drum. One is substream splitter, the others are vapor well-mixed element and liquid well-mixed element.

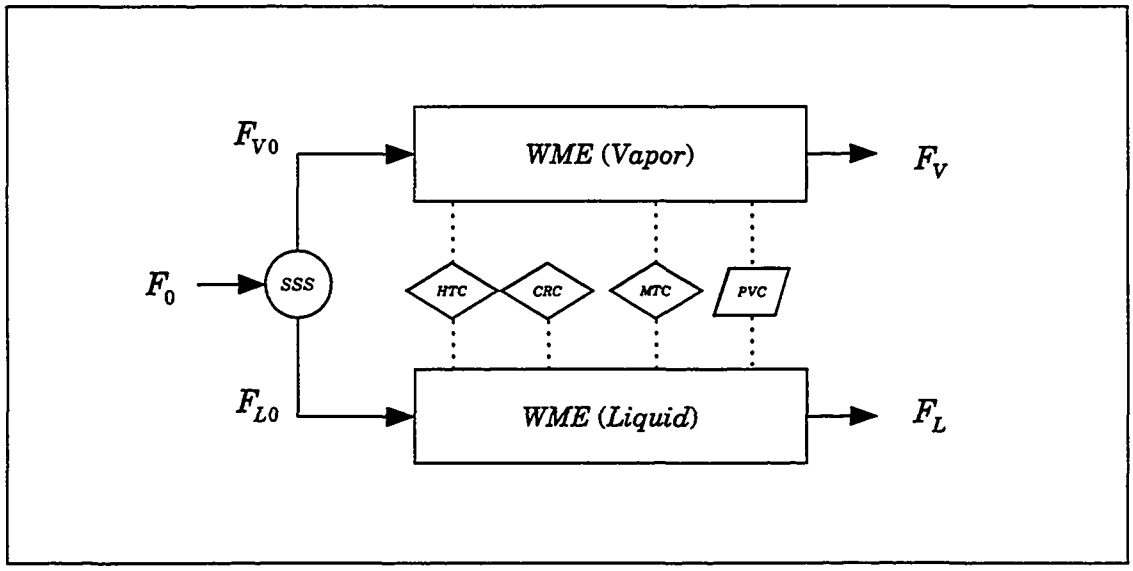



Figure 5-2 Modeling Decomposition of Flash Drum

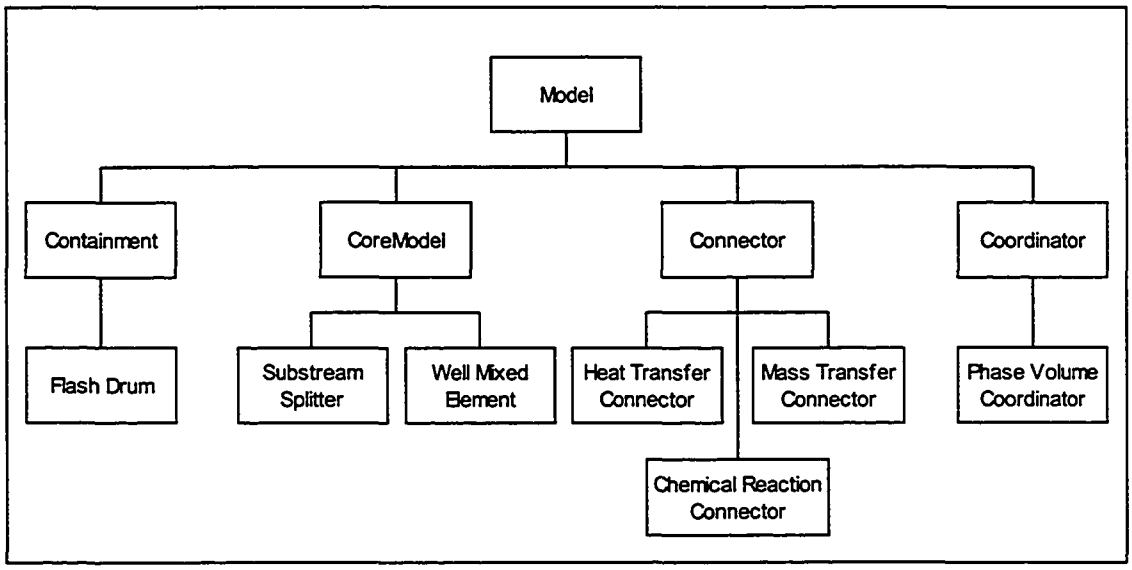



Figure 5-3 Decomposition of Flash Drum Model

- component mass balance for liquid phase

The following equations are for mass transfer rates between vapor and liquid phases.

$$\eta_{i,L} = K_i\left(\hat{f}_i^V - \hat{f}_i^L\right) \tag{5-1}$$

$$\hat{f}_i^L = \gamma_i x_i f_i^L \tag{5-2}$$

$$\hat{f}_i^V = \hat{\phi}_i y_i P_V \tag{5-3}$$

$$\eta_L \equiv \sum_{i=1}^{n_c} \eta_{i,L} \tag{5-4}$$

Suppose that components A (Methanol) and B (Acetic Acid) react in the liquid phase of the flash drum to produce components C (Methyl Acetate) and D (Water), the reversible reaction being defined by the stoichiometric equation

$$CH_4O(A) + C_2H_4O_2(B) \underset{k_2}{\overset{k_1}{\rightleftharpoons}} C_3H_6O_2(C) + H_2O(D) \tag{5-5}$$

The reaction rates now become

$$r_1 = k_1 x_A x_B \tag{5-6}$$

$$r_2 = k_2 x_C x_D \tag{5-7}$$

$$R_1 = R_A = -r_1 + r_2 \tag{5-8}$$

$$R_2 = R_B = -r_1 + r_2 \tag{5-9}$$

$$R_3 = R_C = r_1 - r_2 \tag{5-10}$$

$$R_4 = R_D = r_1 - r_2 \tag{5-11}$$

$$R = \sum_{i=1}^{n_c} R_i = 0 \tag{5-12}$$

Based on Equations (C-47) and (C-39) in Appendix C, we get

$$\phi_L \rho_L \frac{dx_i}{dt} = F_{ci,L} \tag{5-13}$$

$$F_{ci,L} \equiv F_{L0}(x_{i0} - x_i) + A(\eta_{i,L} - x_i \eta_L) + \phi_L R_i \tag{5-14}$$

- overall mass balance for liquid phase

Based on Equations (C-49), (C-41), and (C-44), we get

$$\frac{d\phi_L}{dt} = \frac{1}{\rho_L} F_{m,L} - \frac{\frac{\partial \rho_L}{\partial T_L}}{\rho_L^2 C_{P,L}} Q_{e,L} \tag{5-15}$$

$$F_{m,L} \equiv \rho_L \sum_{i=1}^{n_c} \left( x_{i0} F_{L0} - x_i F_L + A\eta_{i,L} + \phi_L R_i \right) \bar{V}_{i,L} \tag{5-16}$$

$$\begin{aligned} Q_{e,L} = {} & F_{L0} \sum_{i=1}^{n_c} x_{i0} \left( \bar{H}_{i0,L} - \bar{H}_{i,L} \right) + A \sum_{i=1}^{n_c} \eta_{i,L} \lambda_{i,L} \\ & - \phi_L \sum_{j=1}^{n_r} r_j H_{rj} + A(Q + q_L) \end{aligned} \tag{5-17}$$

$$q_L = U_h (T_V - T_L) \tag{5-18}$$

$$\lambda_{i,L} \equiv \bar{H}_i^* - \bar{H}_{i,L} \tag{5-19}$$

$$H_{rj} \equiv \sum_{i=1}^{n_c} \alpha_{ij} \bar{H}_{i,L} \tag{5-20}$$

- energy balance for liquid phase

Based on Equations (C-48) and (C-44) in Appendix C, we get

$$\frac{dT_L}{dt} = \frac{1}{\phi_L \rho_L C_{P,L}} Q_{e,L} \tag{5-21}$$

- component mass balance for gas phase

Based on Equations (C-50) and (C-39), we get

$$\phi_L \rho_L \frac{dy_i}{dt} = F_{ci,V} \tag{5-22}$$

$$F_{ci,V} \equiv F_{V0} (y_{i0} - y_i) + A(\eta_{i,V} - y_i \eta_V) \tag{5-23}$$

$$\eta_{i,V} = K_i \left( \hat{f}_i^L - \hat{f}_i^V \right) \tag{5-24}$$

$$\eta_V \equiv \sum_{i=1}^{n_c} \eta_{i,V} \tag{5-25}$$

- overall mass balance for gas phase

We suppose that the total volume of gas and liquid phases is constant. Based on Equations (C-53), (C-52), and (C-44) in Appendix C, we get

$$\frac{d\phi_V}{dt} = \frac{d\phi_T}{dt} - \frac{d\phi_L}{dt} = -\frac{d\phi_L}{dt} \tag{5-26}$$

$$\frac{dP_V}{dt} = \frac{1}{\phi_V}\left[\frac{\rho C_P F_m - \frac{\partial \rho}{\partial T} Q_e - \left(\rho^2 C_P + P\frac{\partial \rho}{\partial T}\right)\frac{d\phi}{dt}}{\rho C_P \frac{\partial \rho}{\partial P} + \left(1 - \rho\frac{\partial H}{\partial P}\right)\frac{\partial \rho}{\partial T}}\right]_V \tag{5-27}$$

$$F_{m,V} \equiv \rho_V \sum_{i=1}^{n_c} \left(y_{i0} F_{V0} - y_i F_V + A\eta_{i,V}\right)\bar{V}_{i,V} \tag{5-28}$$

$$Q_{e,V} = F_{V0}\sum_{i=1}^{n_c} y_{i0}\left(\bar{H}_{i0,V} - \bar{H}_{i,V}\right) + A\sum_{i=1}^{n_c}\eta_{i,V}\lambda_{i,V} + Aq_V \tag{5-29}$$

$$\lambda_{i,V} \equiv \bar{H}_i^* - \bar{H}_{i,V} \tag{5-30}$$

$$q_V = U_h\left(T_L - T_V\right) \tag{5-31}$$

- energy balance for gas phase

Based on Equations (C-51), (C-41) and (C-44), we get

$$\frac{dT_V}{dt} = \frac{1}{\phi_V}\left[\frac{\left(1 - \rho\frac{\partial H}{\partial P}\right)F_m + \frac{\partial \rho}{\partial P} Q_e + \left(\rho^2\frac{\partial H}{\partial P} - \rho + P\frac{\partial \rho}{\partial P}\right)\frac{d\phi}{dt}}{\rho C_P \frac{\partial \rho}{\partial P} + \left(1 - \rho\frac{\partial H}{\partial P}\right)\frac{\partial \rho}{\partial T}}\right]_V \tag{5-32}$$

- flow equations

For the open loop option, we assume that the flow rates of the streams leaving the flash drum are specified as functions of time, generally determined by pressure drop and the flow resistance of spool piece. This completes the development of the basic model equations for the flash drum.

### 5.2.3 Connector

There are three connectors in flash drum. One is chemical reaction connector within liquid phase, the others are heat and mass transfer connectors between vapor and liquid phases.

### 5.2.4 Coordinator

There are two coordinators in flash drum. One is phase volume coordinator for liquid phase, the other is also phase volume coordinator for vapor phase.

### 5.2.5 Correlation

The model equations described by the respective modeling components are summarized in Table 5-1. The visual decomposition of flash drum model in ForeSee is shown in Figure 5-4.

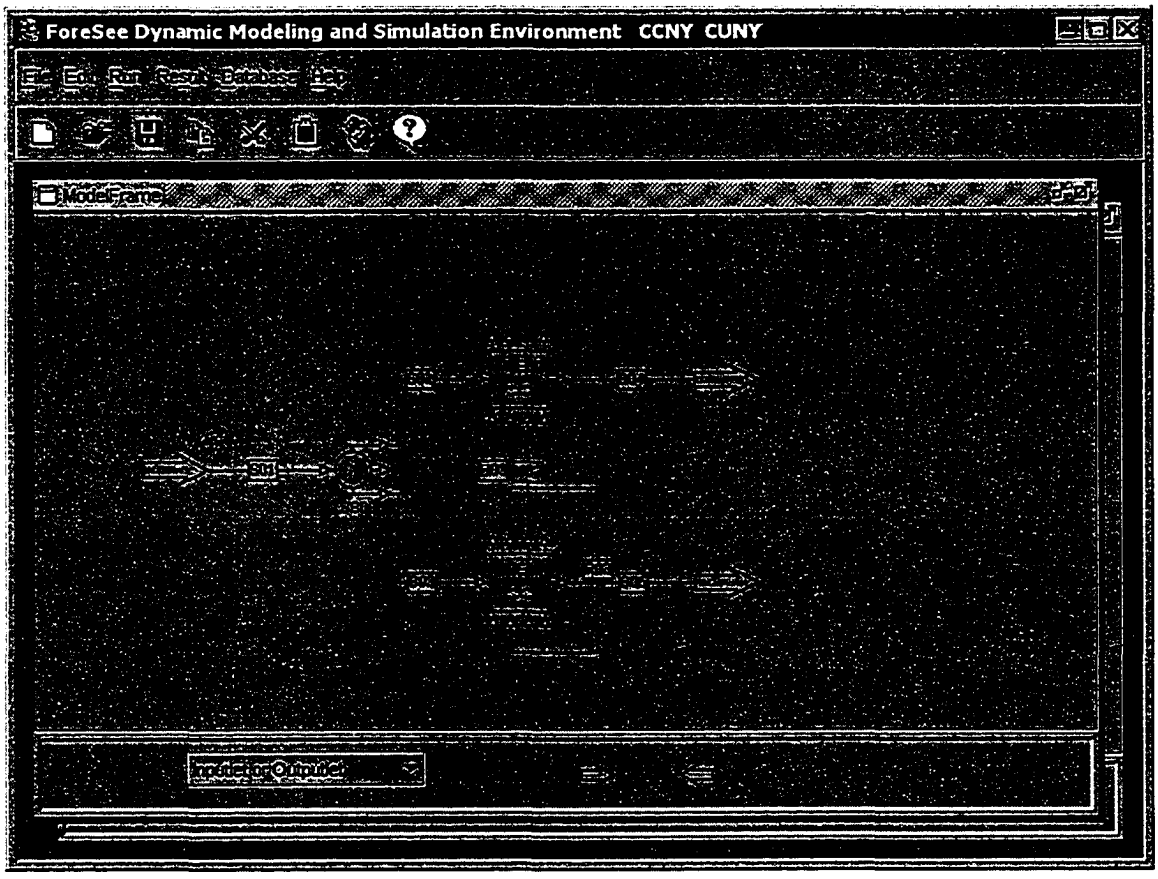



Figure 5-4 Modeling Decomposition of Flash Drum in ForeSee

Table 5-1 Relationship between Model Equations and Modeling Components in Flash Drum

| Modeling Components | Model Equations |
|---|---|
| CoreModels (Conservation Equations) | Equations (5-13), (5-21), (5-22) and (5-32) |
| Connectors (Constitutive Equations) | Equations (5-6), (5-7), (5-1), (5-24), (5-18) and (5-31) |
| Coordinators (Constraint Equations) | Equations (5-15), (5-26) and (5-27) |

### 5.2.6 Process Control

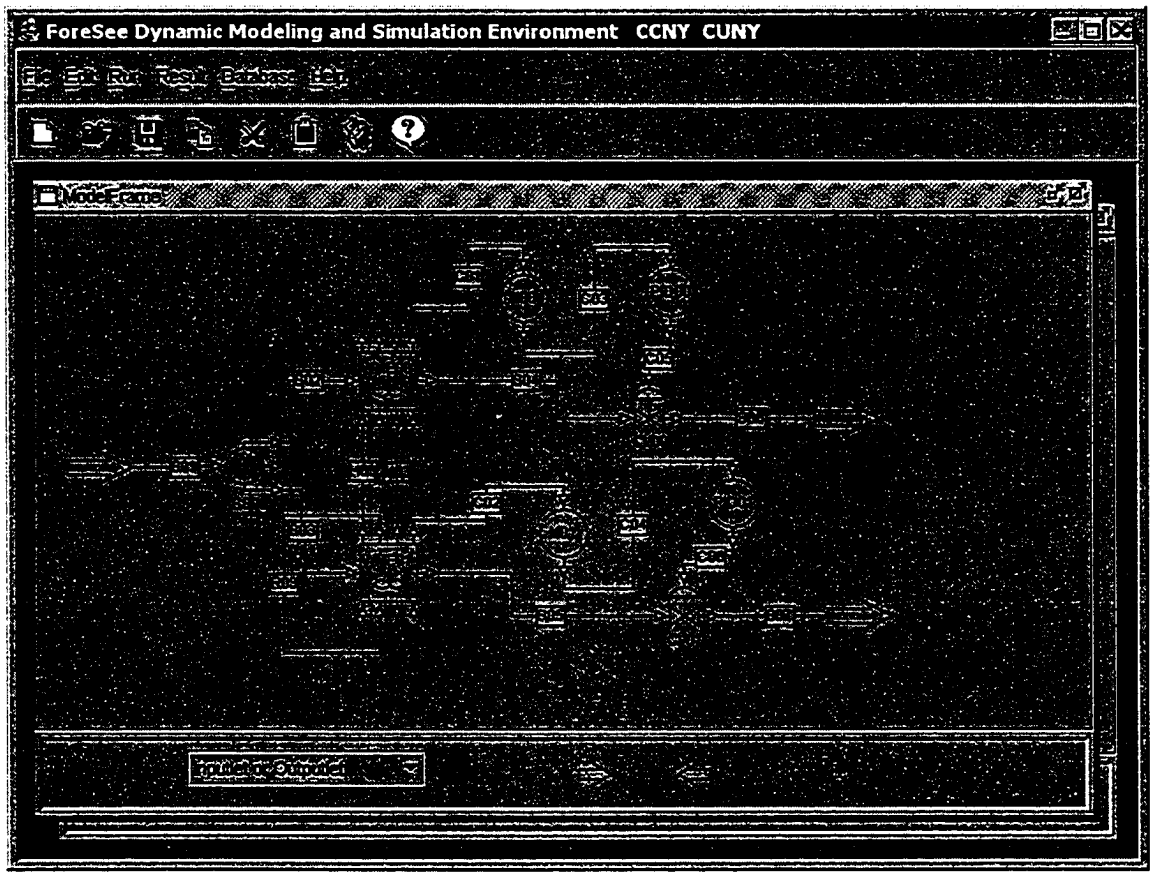



Figure 5-5 Instrumentation and Control in Flash Drum

Two proportional controllers are used to keep vapor phase pressure and liquid level steady. One is used to adjust the vapor flow rate, the other for liquid flow rate.

### 5.2.7 Dynamics

The composition profiles of vapor and liquid phases (with reaction) are shown in Figure 5-6 and Figure 5-7, respectively.

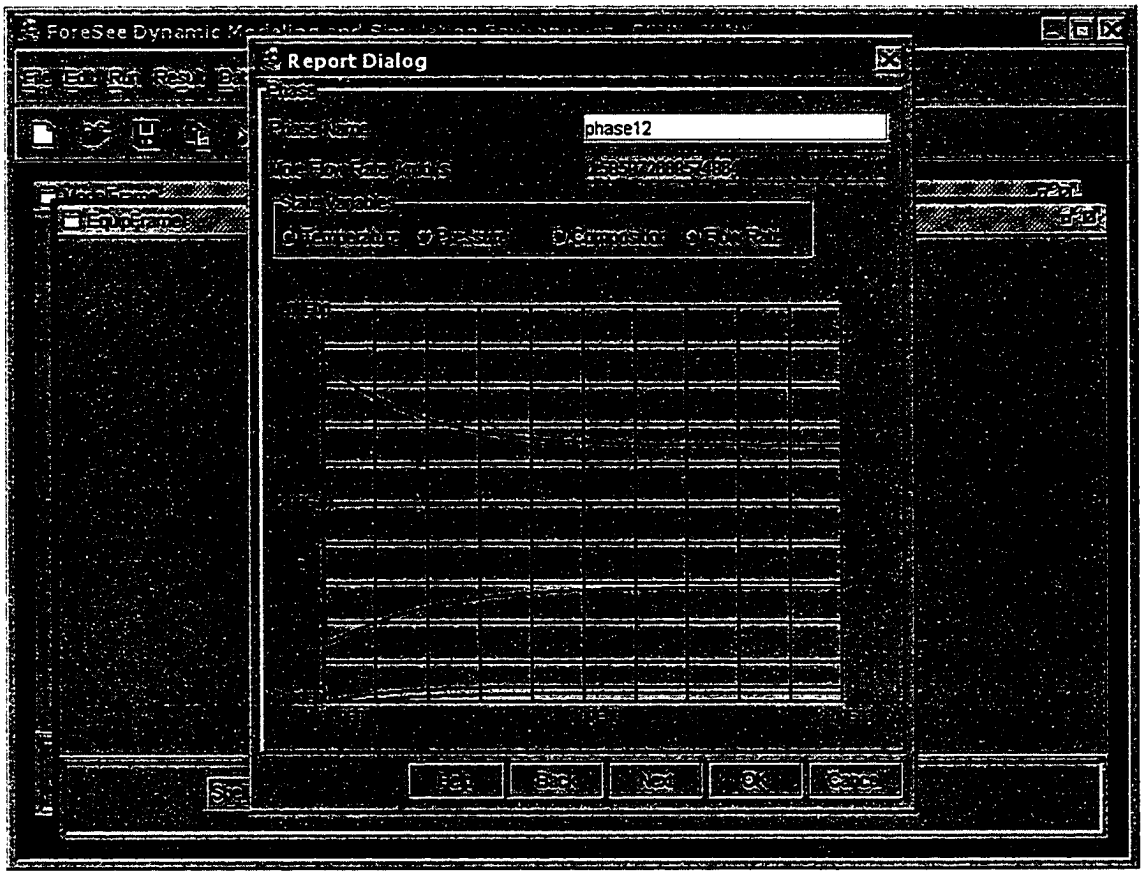



Figure 5-6 Vapor Composition Profile in Flash Drum

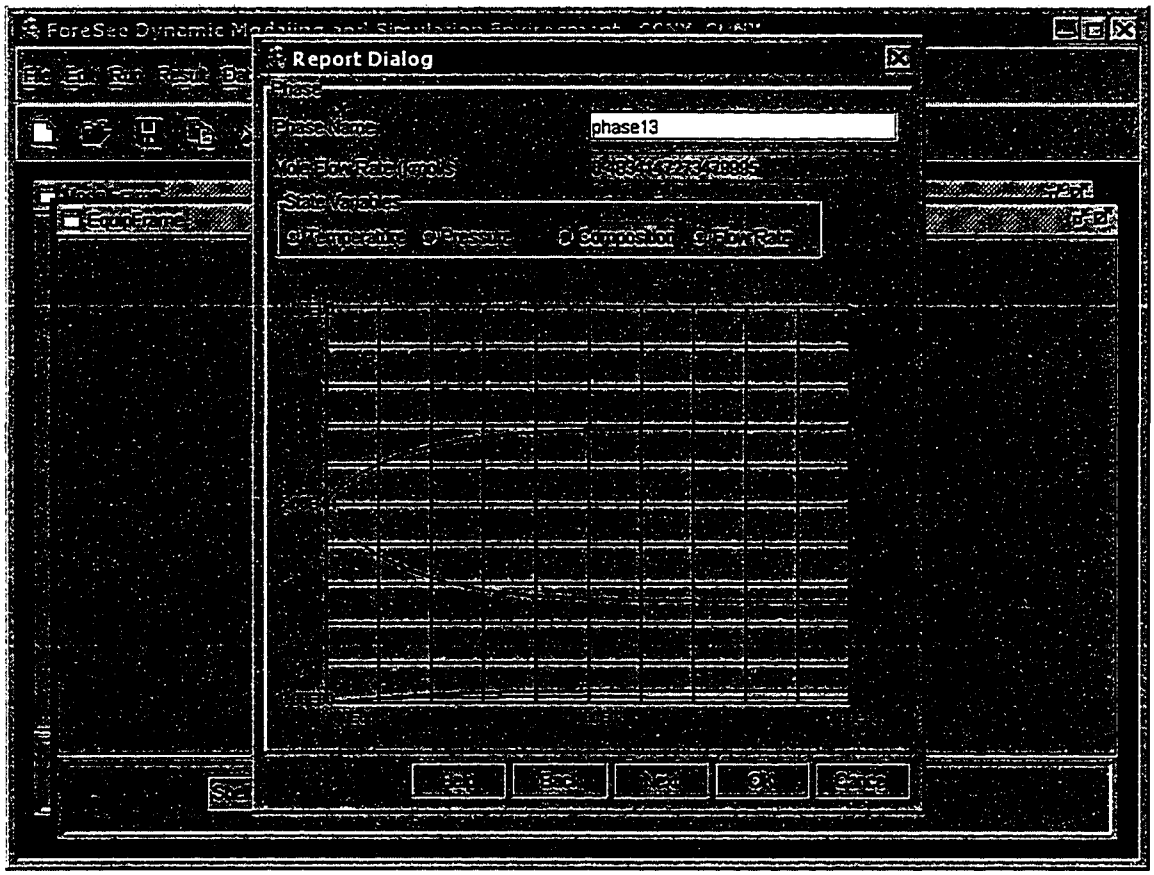



Figure 5-7 Liquid Composition Profile in Flash Drum

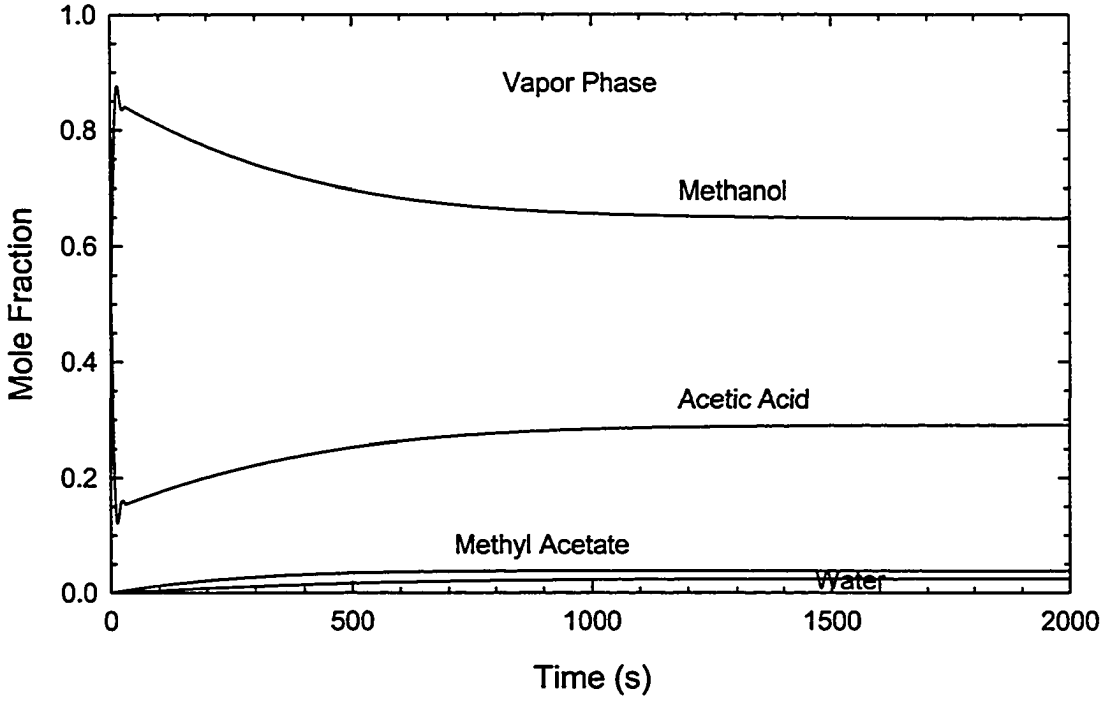



Figure 5-8 Dynamic Response of Flash Drum Vapor Composition

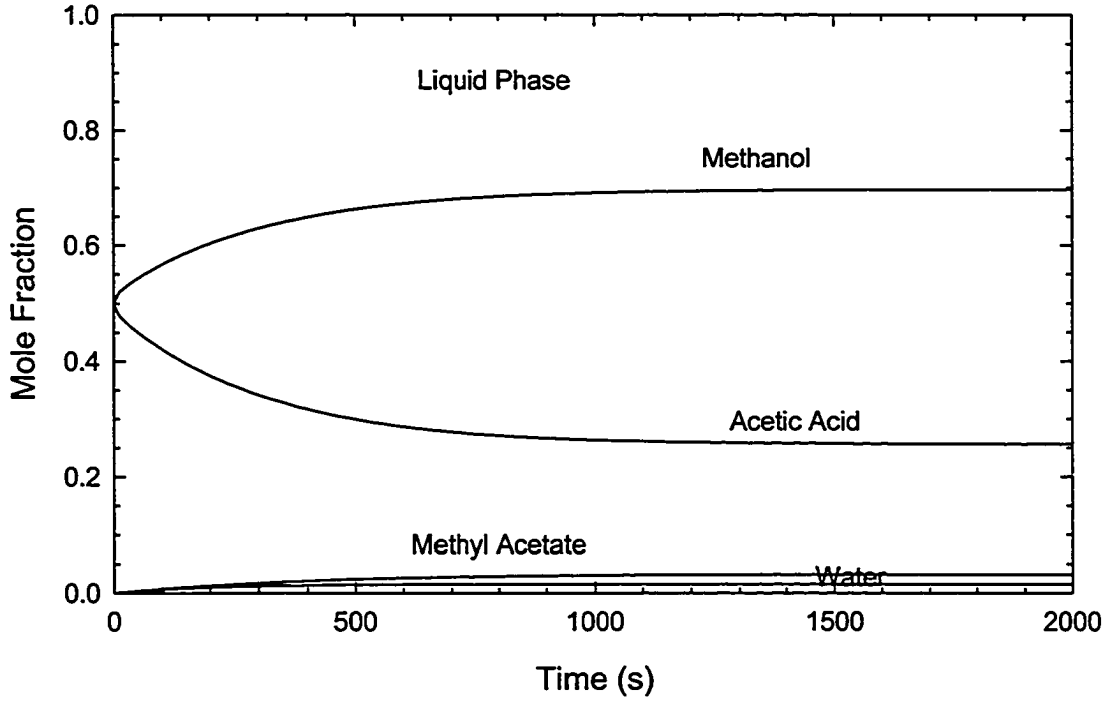



Figure 5-9 Dynamic Response of Flash Drum Liquid Composition

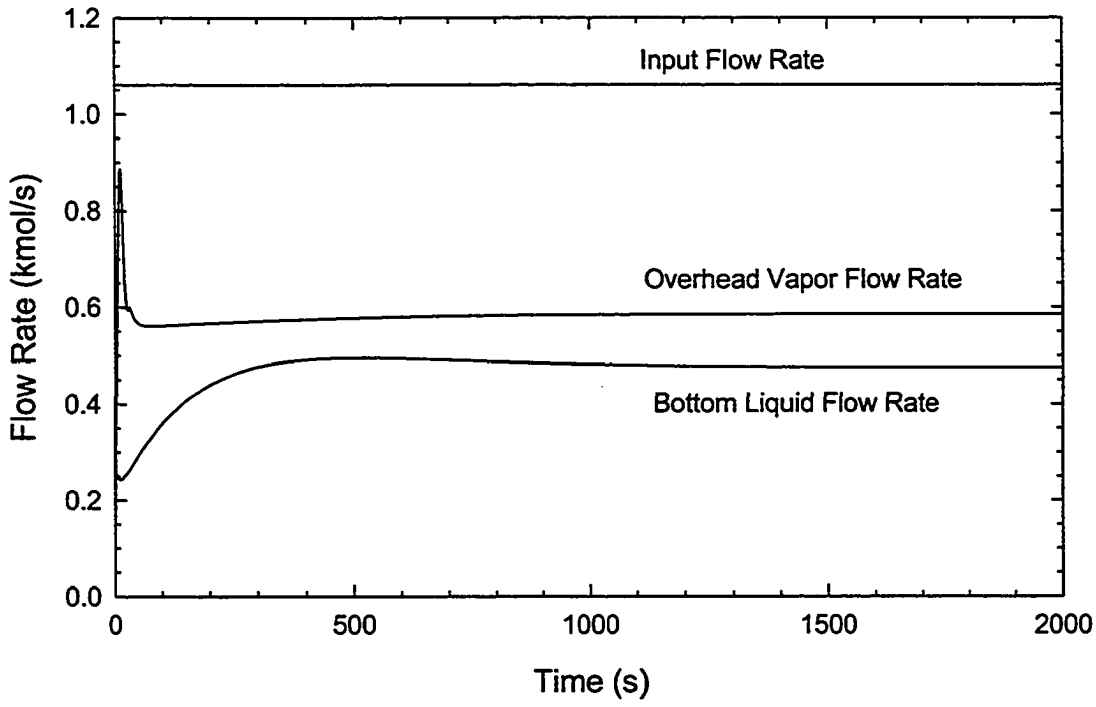



Figure 5-10 Dynamic Response of Flash Drum Flow Rates

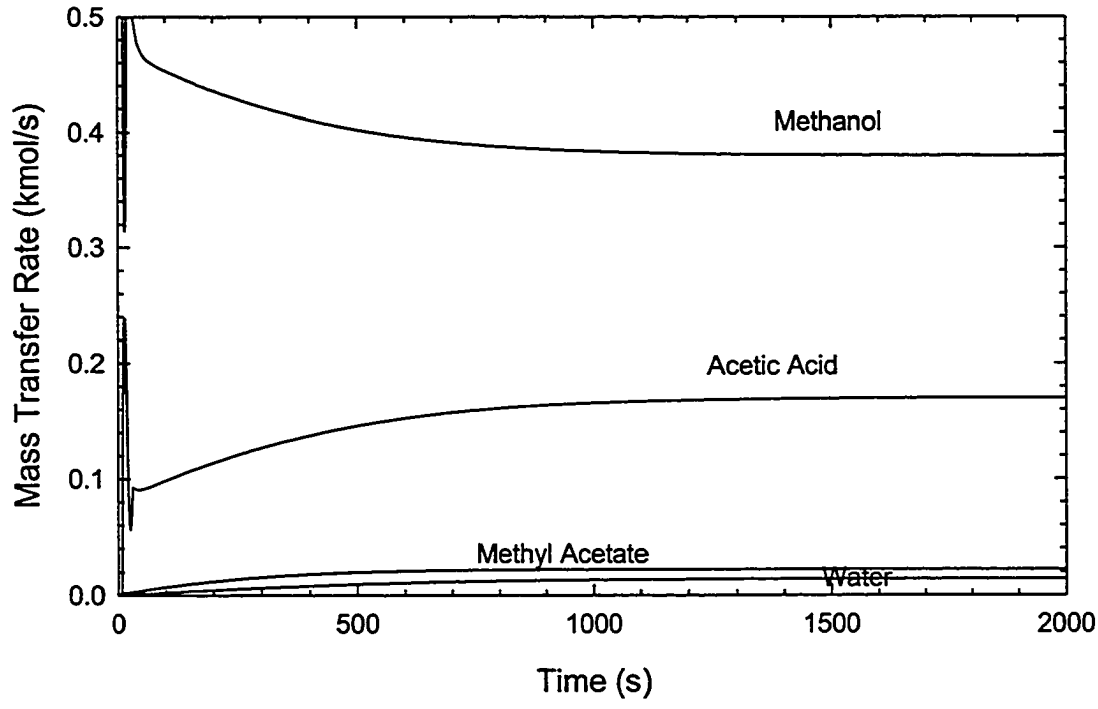



Figure 5-11 Dynamic Response of Mass Transfer Rates

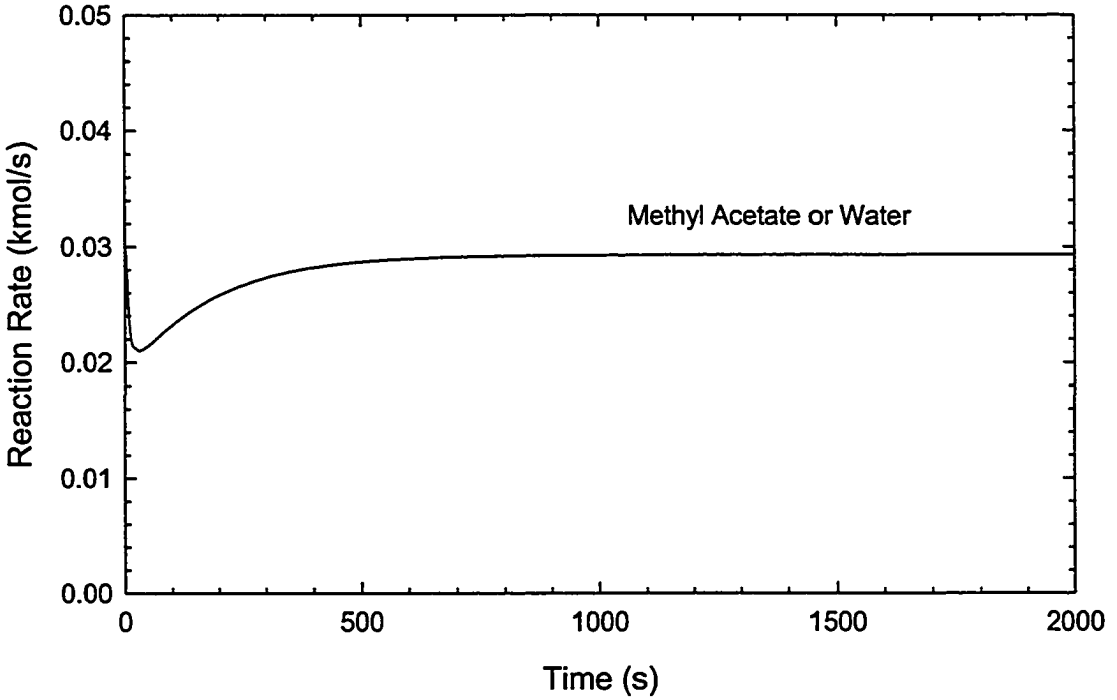



Figure 5-12 Dynamic Response of Reaction Rates in Liquid Phase

The composition profiles of vapor and liquid phases (without reaction) are shown in Figure 5-13 and Figure 5-14, respectively. The steady-state comparison for flash drum without reaction between ForeSee and ASPEN is summarized in Table 5-2.

Table 5-2 Steady-State Comparison for Flash Drum between ForeSee and ASPEN

| Variables | ForeSee | ASPEN |
|---|---|---|
| Feed Flow Rate [kmol/s] | 1.0607 | 1.0607 |
| Feed Composition [%] (Methanol, Acetic Acid) | (50.00, 50.00) | (50.00, 50.00) |
| Overhead Flow Rate [kmol/s] | 0.5621 | 0.5708 |
| Overhead Composition [%] (Methanol, Acetic Acid) | (69.93, 30.07) | (69.54, 30.46) |
| Bottom Flow Rate [kmol/s] | 0.4986 | 0.4899 |
| Bottom Composition [%] (Methanol, Acetic Acid) | (27.54, 72.46) | (27.23, 72.77) |

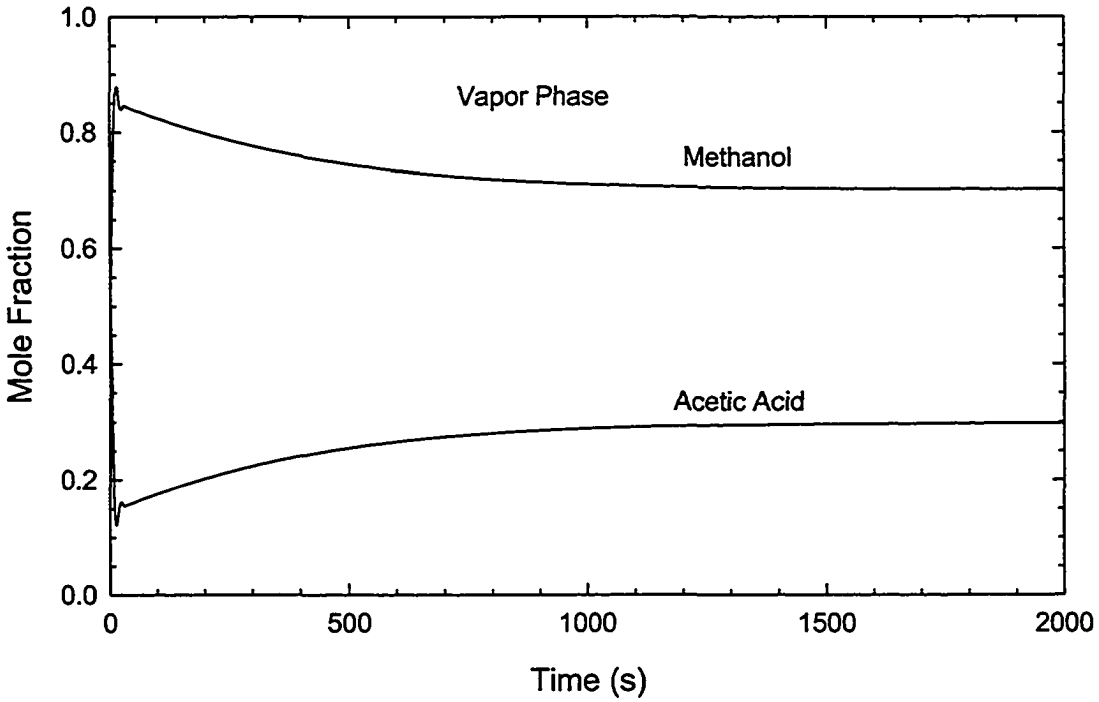



Figure 5-13 Dynamic Response of Flash Drum Vapor Composition

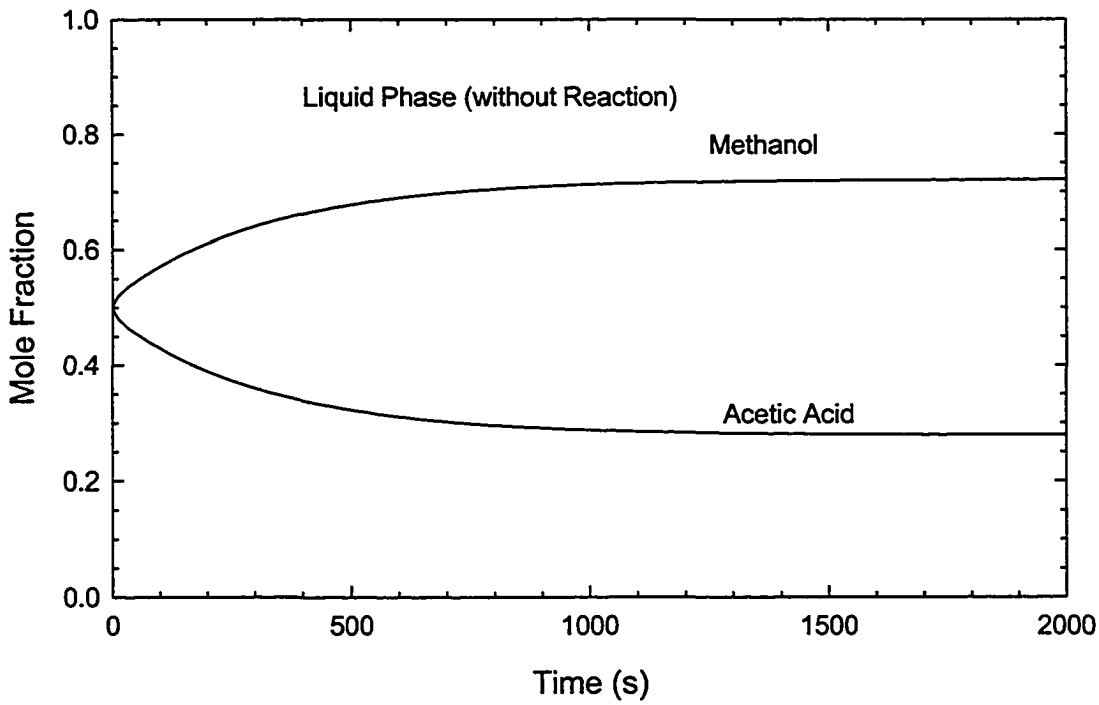



Figure 5-14 Dynamic Response of Flash Drum Liquid Composition

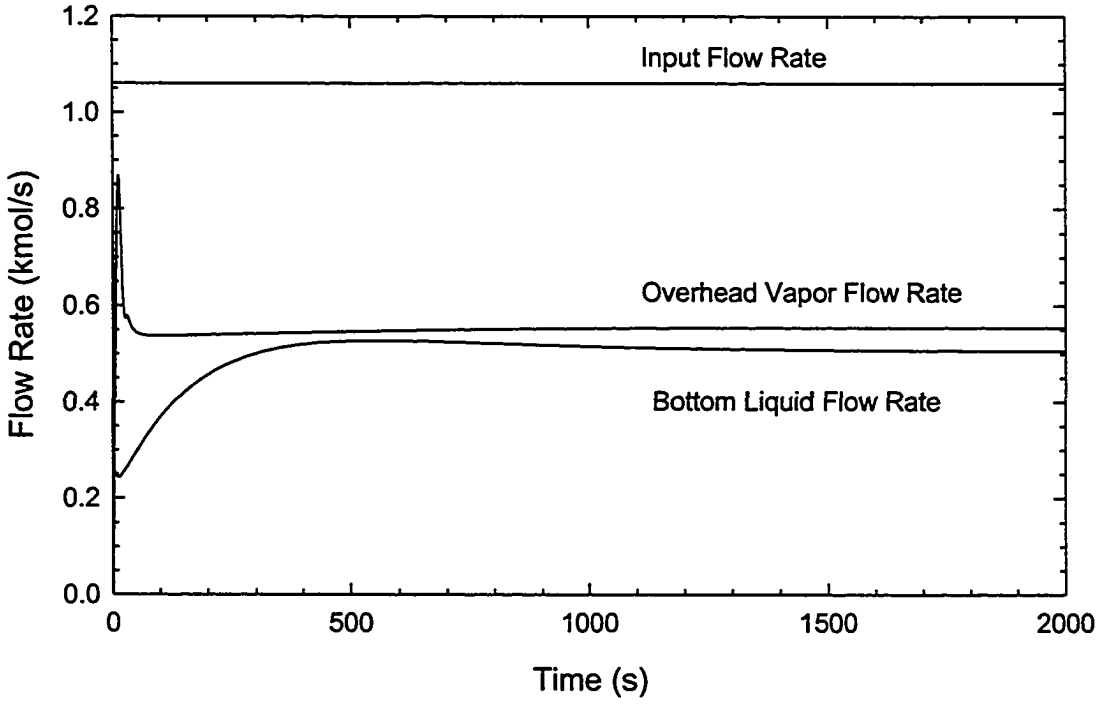



Figure 5-15 Dynamic Response of Flash Drum Flow Rates

## 5.3 Conclusions

- Equipment model can be visually assembled from an interconnected set of modeling components (containments, core models, connectors, and coordinators).
- The development of equipment model is easy to understand. There are no black box models in model configuration.

# Chapter 6 Proof of Concept

## 6.1 Introduction

While each of chemical processes is functionally quite different from the others, if we look at the fundamental components of each, these are quite similar in their physical behavior. This means that simulation models of complex processes can be built up hierarchically starting with relatively simple models of these fundamental components.

The basic premise in this approach to modeling is that a relatively small set of modeling components can be used to generate a variety of equipment models.

Table 6-1 Equipment Model Samples

| Modeling Components | Equipment Models |
|---|---|
| Well-Mixed Element Core Model<br>Plug Flow Element Core Model<br>Heat Transfer Connector<br>Mass Transfer Connector<br>Chemical Reaction Connector<br>Phase Volume Coordinator<br>Void Fraction Coordinator | Continuous Stirred Tank Reactor<br>Shell-Tube Heat Exchanger<br>Reboiler<br>Condenser<br>Flash Drum<br>Distillation<br>Packed Bed Gas Absorption<br>Mixer-Settler<br>Evaporator<br>Gas Membrane Separator<br>Decanter<br>Cyclone Separator<br>Tubular Flow Reactor<br>Bubble Bed Reactor<br>Fixed Bed Reactor<br>Spray Tower Extractor |

It is in the representation of equipment models that ForeSee differs most strongly from all the other simulation systems. In these systems an equipment model consists of a set equations and the associated code for their execution. In ForeSee an equipment model is specified as an interconnected set of modeling components (*containments*, *core models*, *connectors*, and *coordinators*).

The following equipment models may roughly be classified into two groups: *WME-based* and *PFE-based* equipment models.

## 6.2 WME-Based Equipment Models

The core models of this type of equipment models are mainly well-mixed elements.

### 6.2.1 Continuous Stirred Tank Reactor

The continuous stirred tank reactor is schematically shown in Figure 6-1. Stirring is used to mix the ingredients initially, to maintain homogeneity during reaction, and to enhance heat transfer at a jacket wall or internal surfaces. Knowledge of the extent to which a stirred tank does approach complete mixing is essential to being able to predict its performance as a reactor. The ideal stirred tank reactor is a reactor in which the contents are well stirred and uniform throughout. Thus the exit stream from this reactor has the same composition, pressure, and temperature as the fluid within the reactor.

The decomposition of the continuous stirred tank reactor modeling components is summarized as follows:

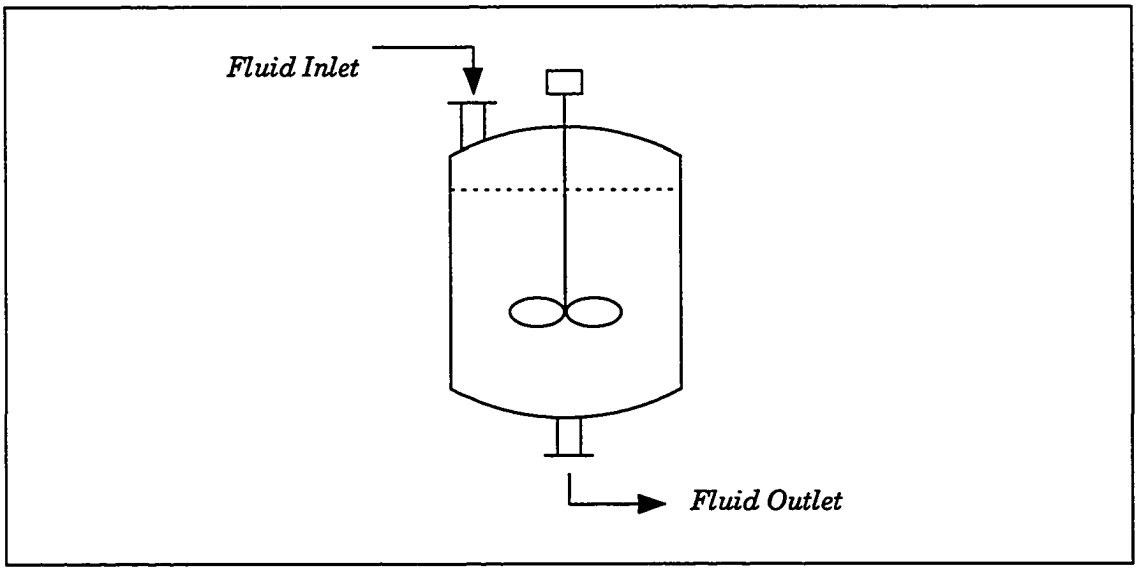



Figure 6-1 Continuous Stirred Tank Reactor

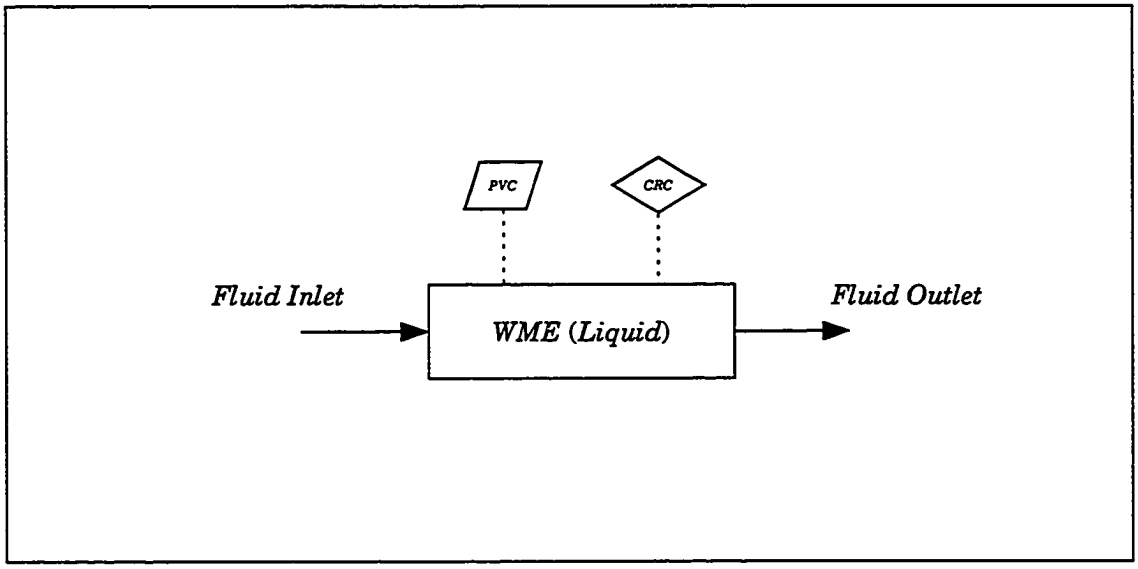



Figure 6-2 Modeling Decomposition of Stirred Tank Reactor

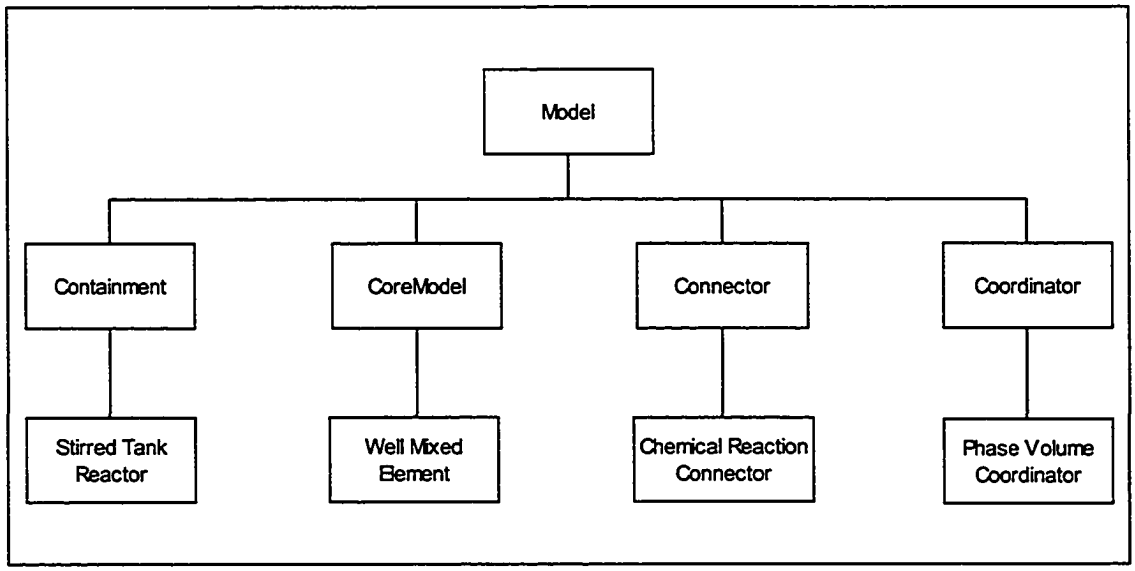



Figure 6-3 Decomposition of Stirred Tank Reactor Model

### 6.2.2 Shell-Tube Heat Exchanger

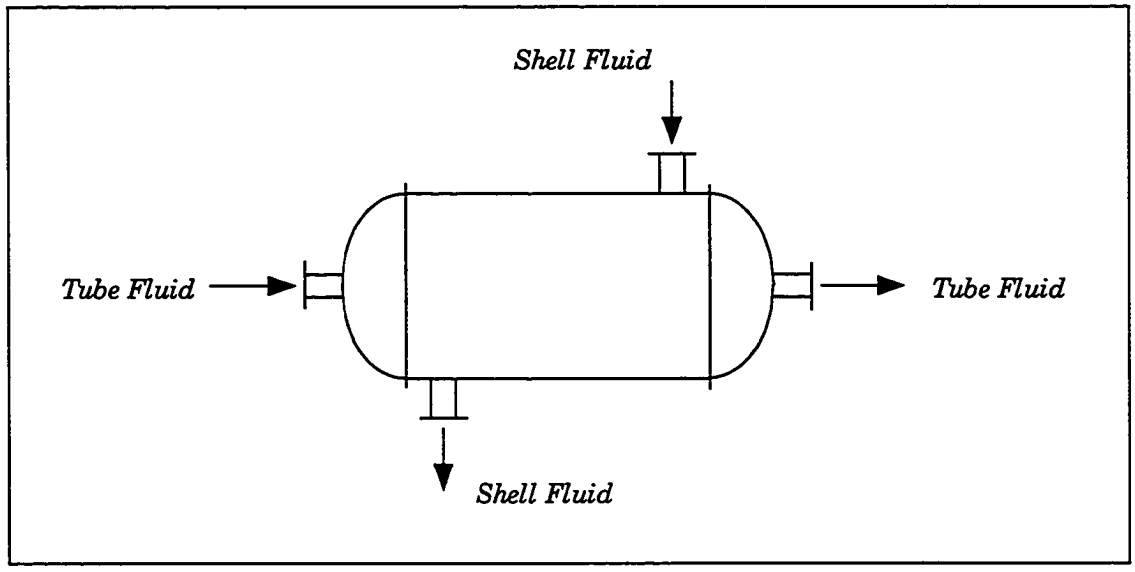



Figure 6-4 Shell-Tube Heat Exchanger

Heat exchangers are used to transfer heat between hot and cold

streams. They have separate passages for the two streams and operate continuously. Shell-tube heat exchangers are made up of a number of tubes in parallel and series through which one fluid travels and are enclosed in a shell through which the other fluid flows. We are approximating the heat exchanger by a series of four well-mixed stages as illustrated in Figure 6-5. Multiple stages can be used, if desired. For a shell-tube heat exchanger, we require one containment, one coordinator, one core model, and one connector. The containment in this case is the shell-tube heat exchanger vessel with various nozzles for connecting spool pieces for flows in and out of the equipment. The core model is the well-mixed element (WME) in which perfect mixing is assumed. The connector is the heat transfer connector (HTC). The coordinator is the static phase volume coordinator (PVC). Both the shell and the tube wall thermal capacitance have been neglected in this case.

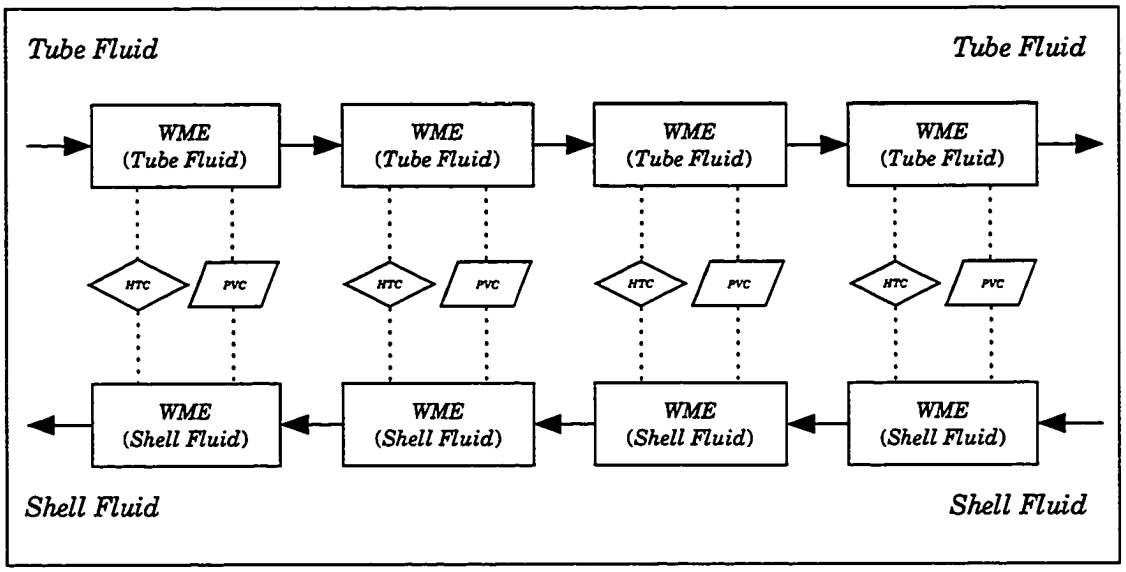



Figure 6-5 Modeling Decomposition of Shell-Tube Heat Exchanger

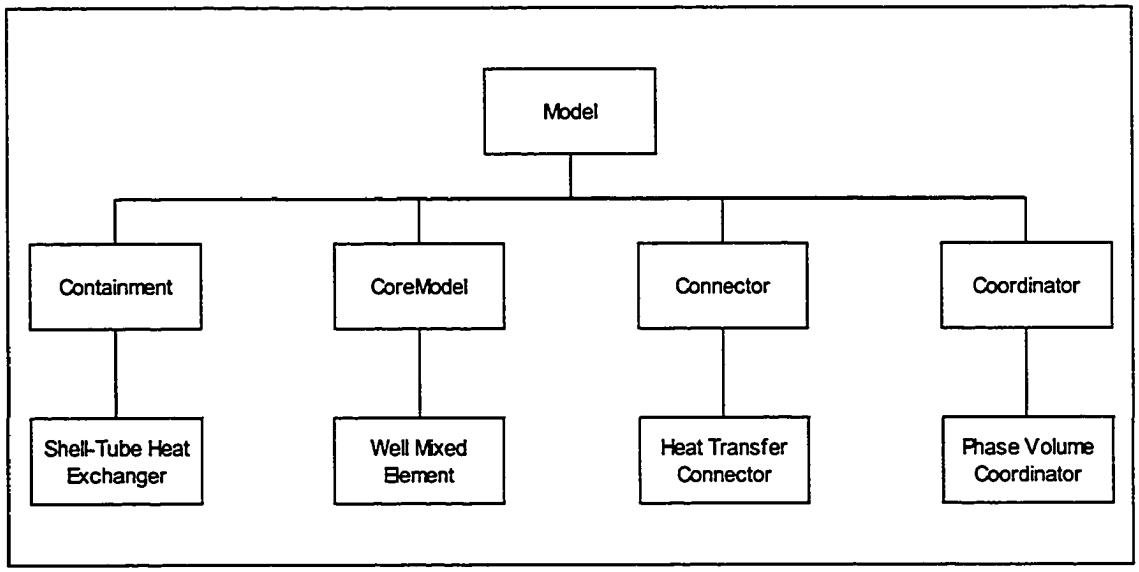



Figure 6-6 Decomposition of Shell-Tube Heat Exchanger Model

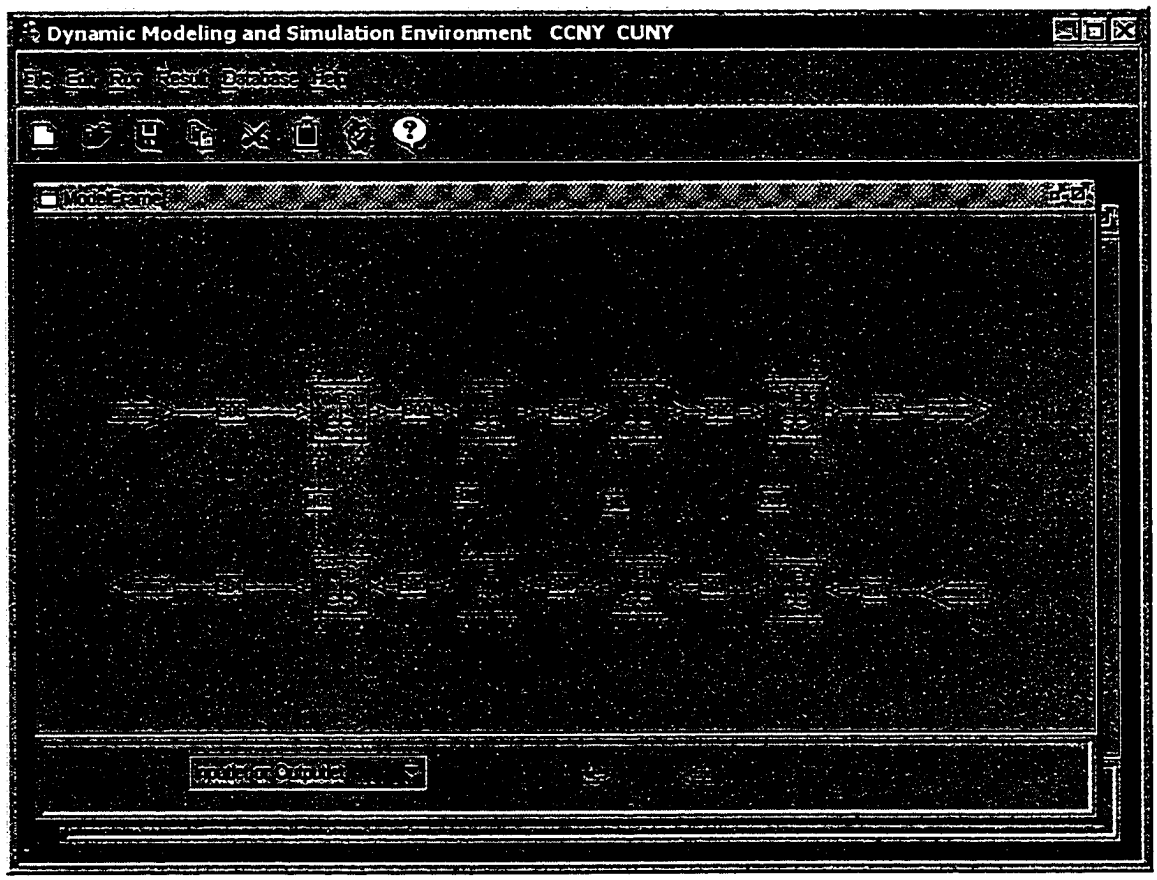



Figure 6-7 Modeling Decomposition of Heat Exchanger in ForeSee

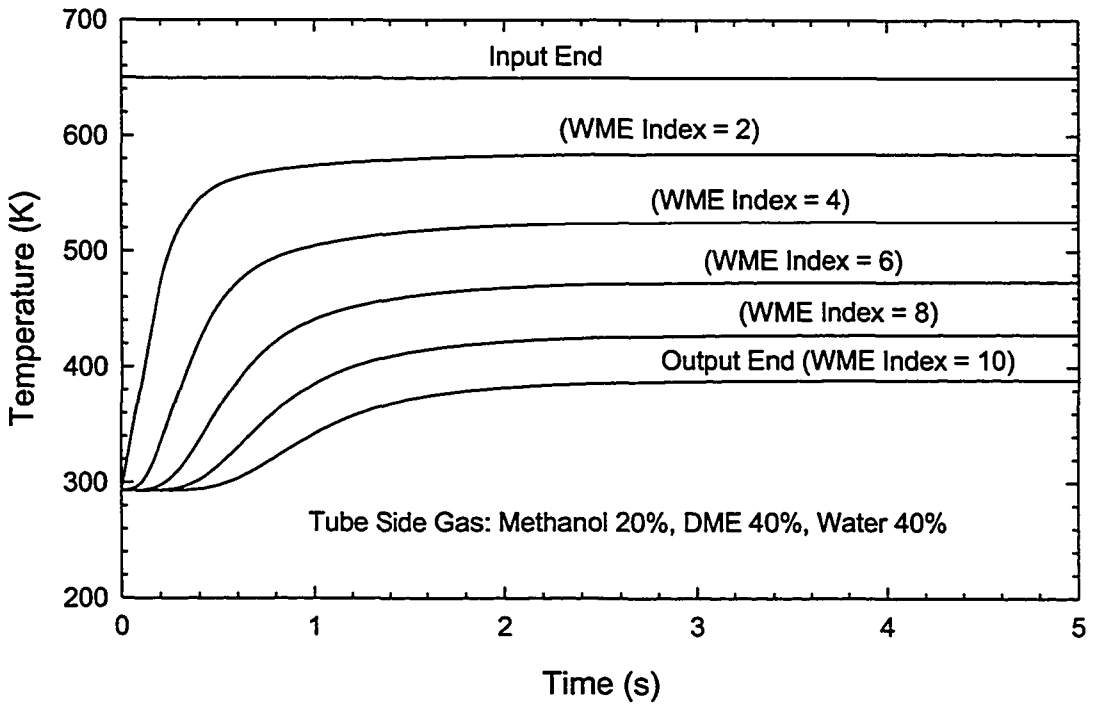



Figure 6-8 Dynamic Response of Tube Side Temperature

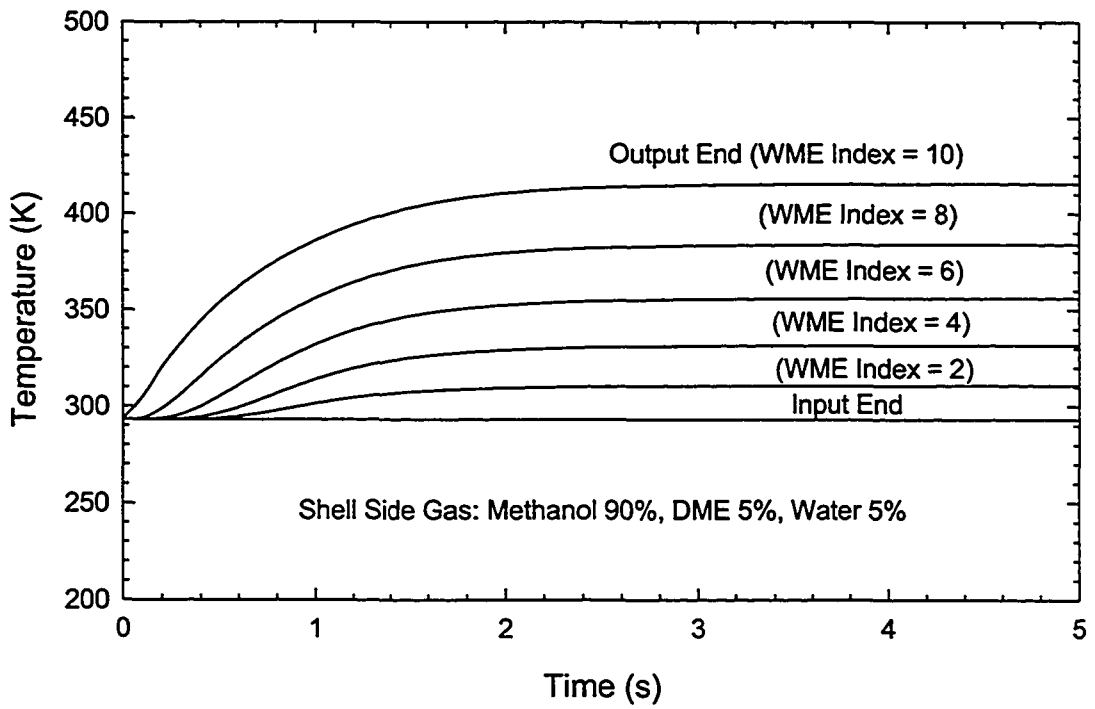



Figure 6-9 Dynamic Response of Shell Side Temperature

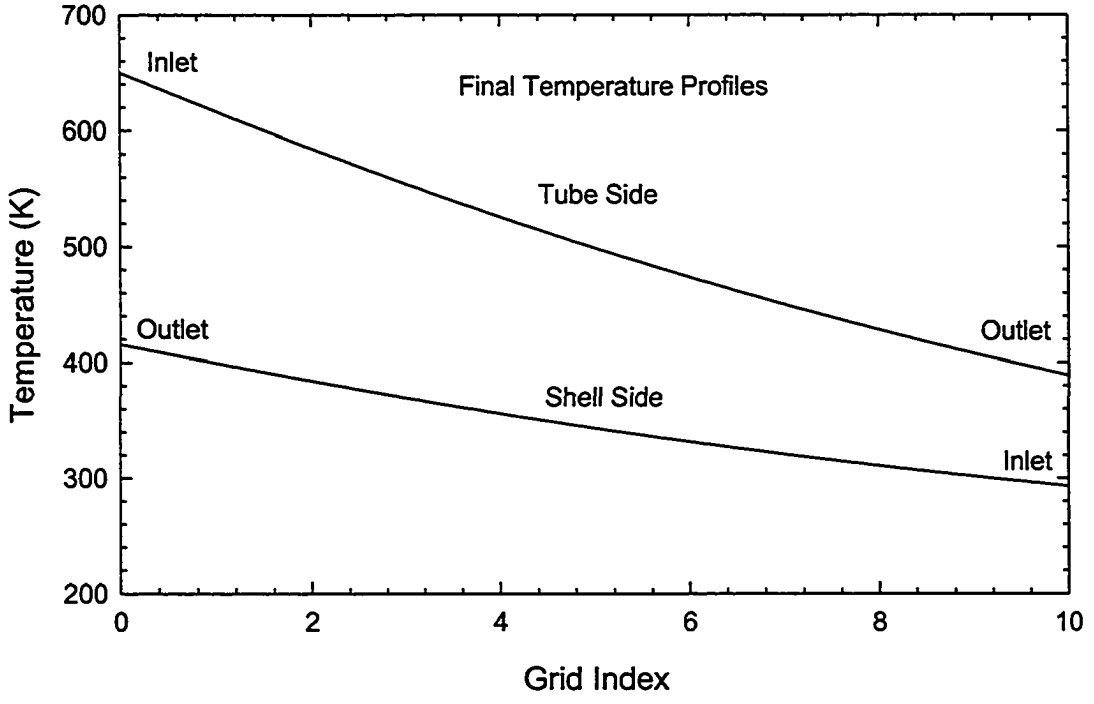



Figure 6-10 Final Temperature Profiles of Shell-Tube Exchanger

The dynamic responses of tube and shell side temperatures are shown in Figure 6-8 and Figure 6-9, respectively. What is shown in Figure 6-10 is the steady-state temperature profiles of shell-tube heat exchanger.

### 6.2.3 Reboiler

Most industrial reboilers are enclosed vessels that are used primarily to provide boilup for distillation and similar towers. Figure 6-11 shows the feed flow supplied in liquid form and the exit flow withdraw as vapor. We have used a single well-mixed stage to model the tube side of the reboiler. Multiple stages can be used, if desired, similar for the heat exchanger presented in the previous section.

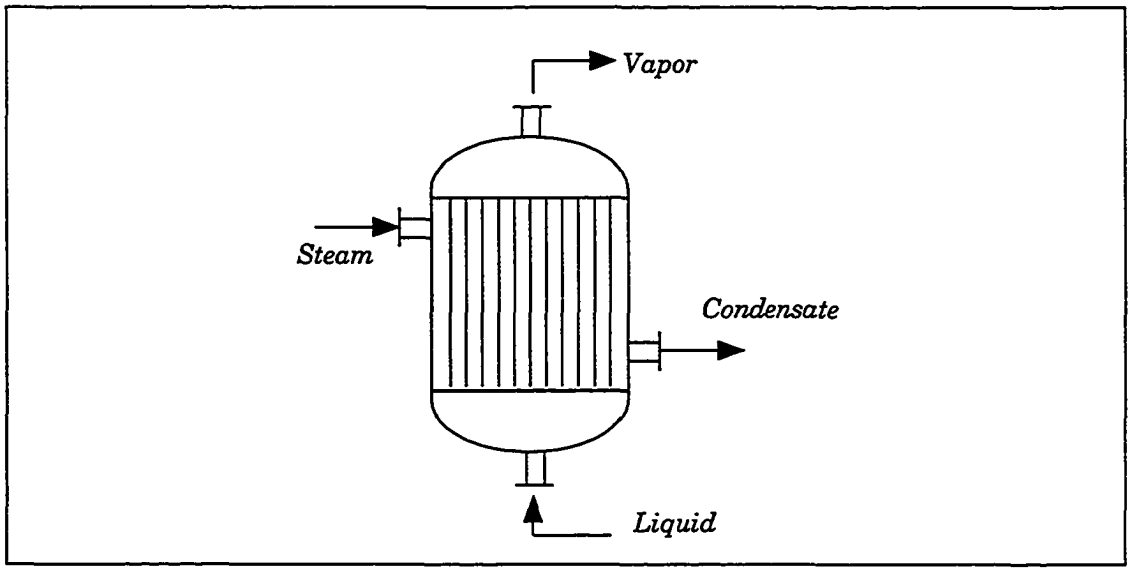



Figure 6-11 Reboiler

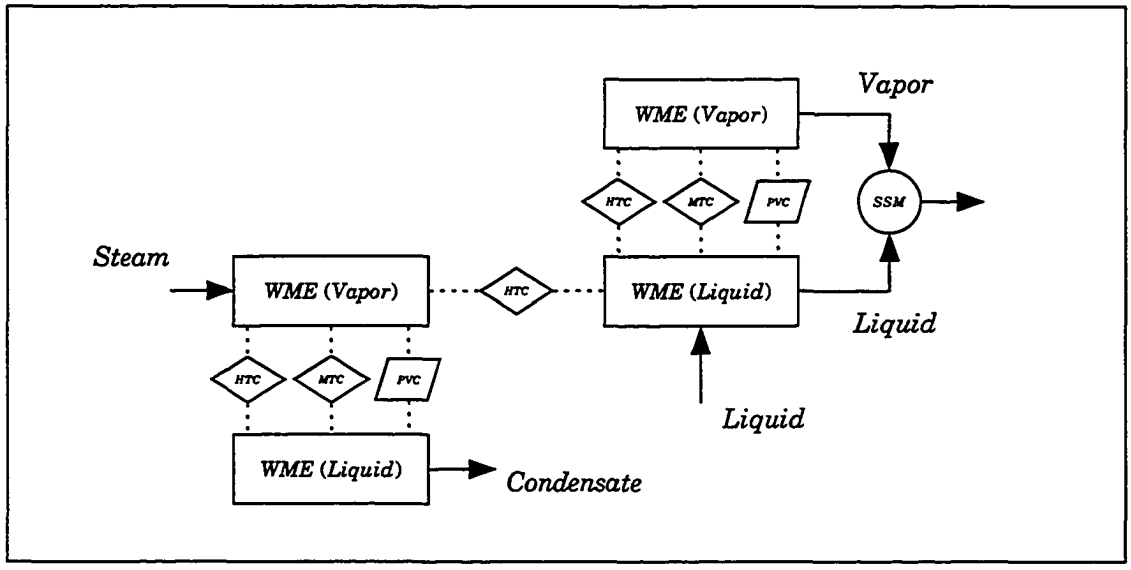



Figure 6-12 Modeling Decomposition of Reboiler

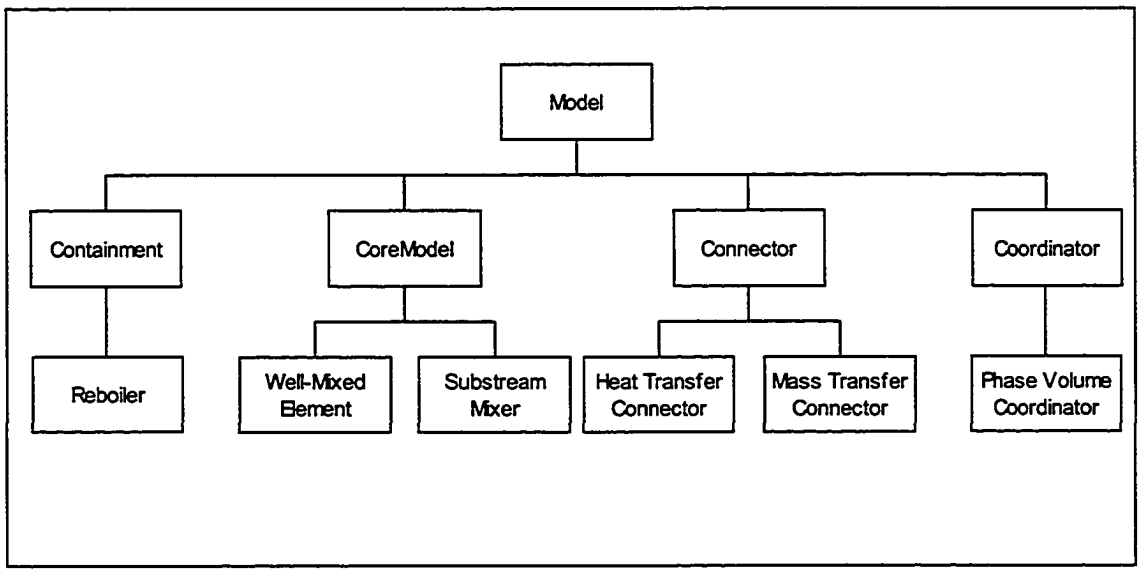



Figure 6-13 Decomposition of Reboiler Model

### 6.2.4 Condenser

Most industrial condensers are enclosed vessels that are used primarily to provide reflux for distillation and similar towers. Figure 6-11 shows the feed flow supplied in vapor form and the exit flow withdraw as liquid. Here we have used three well-mixed stages to approximate the shell side of the condenser. For a condenser, we require one containment, two core models (well-mixed element and substream mixer), two connectors (heat transfer connector and mass transfer connector), and one coordinator (phase volume coordinator).

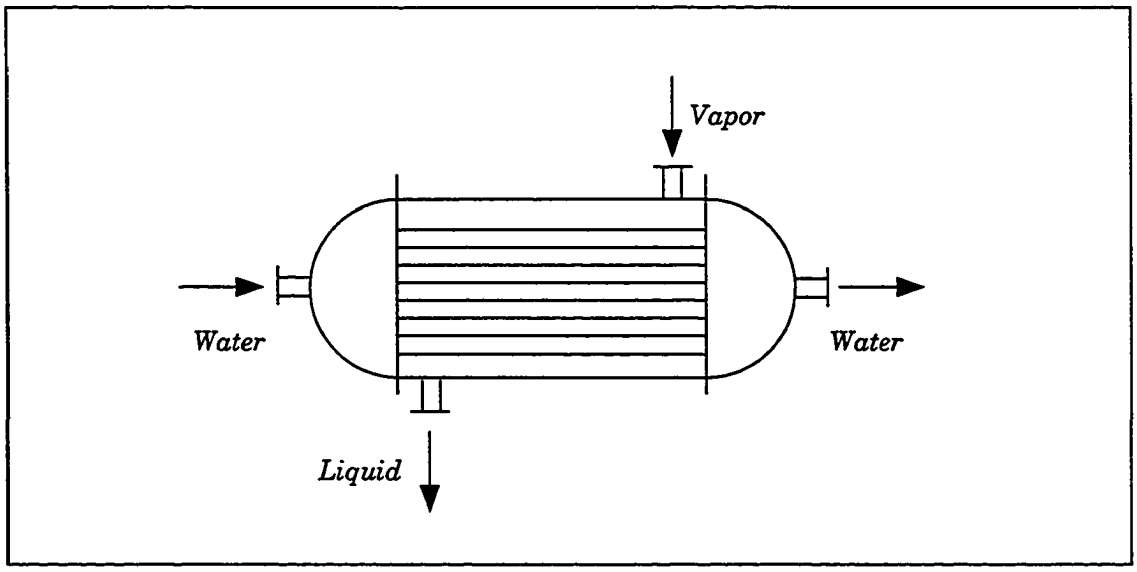



Figure 6-14 Condenser

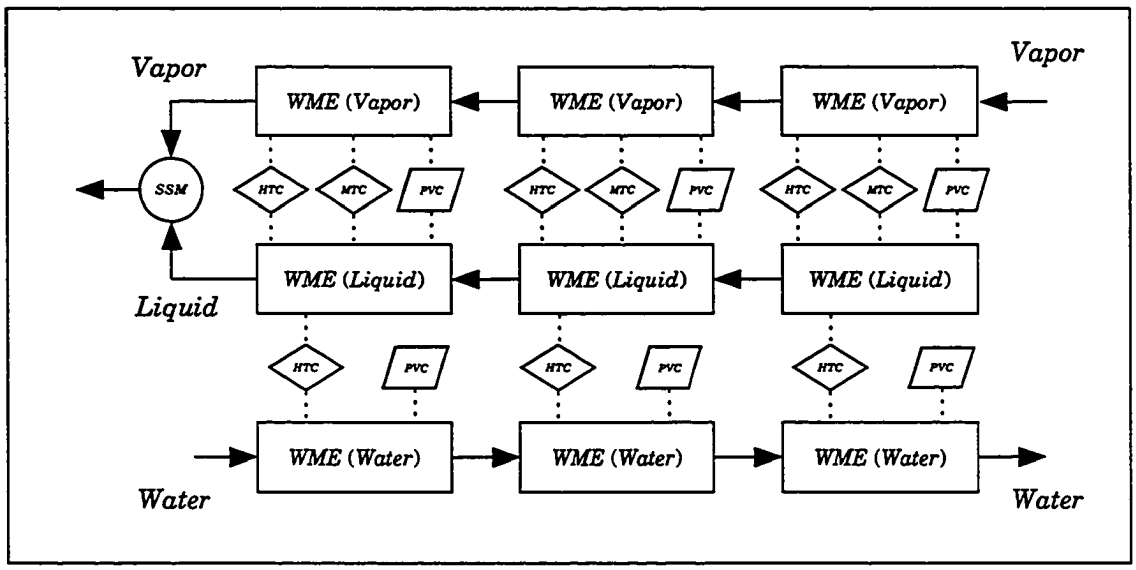



Figure 6-15 Modeling Decomposition of Condenser

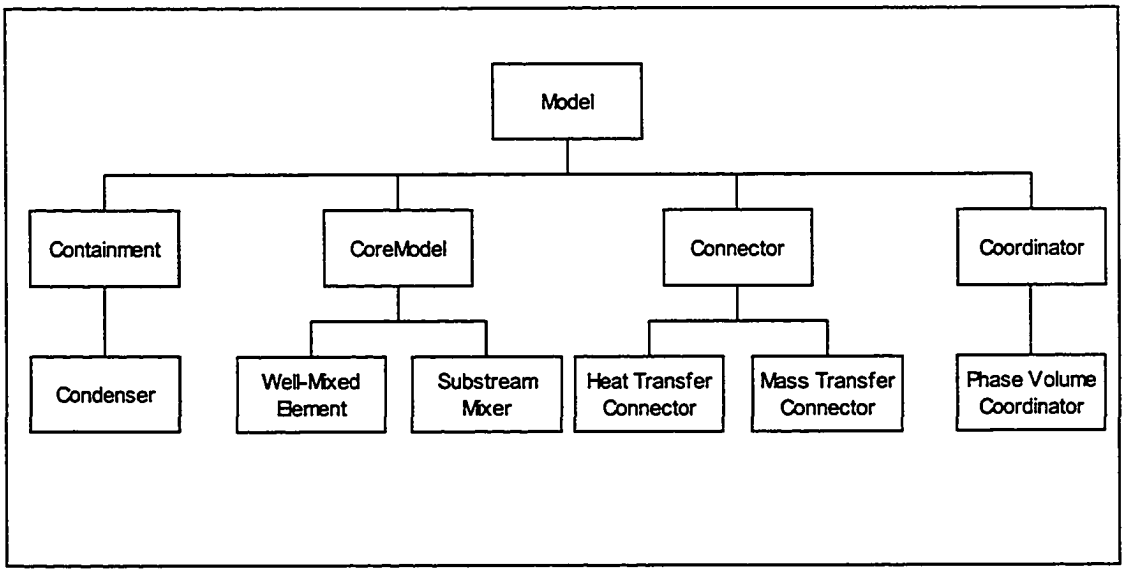



Figure 6-16 Decomposition of Condenser Model

### 6.2.5 Distillation

A typical continuous distillation column includes both rectifying and stripping sections. All plates above the feed plate constitute the rectifying section, and all plates below the feed, including the feed plate itself, constitute the stripping section. The feasibility of separation of mixtures by distillation, absorption, or stripping depends on the fact that the composition of vapor and liquid phases are different from each other at equilibrium.

The modeling of the distillation is accomplished by the individual analysis of each stage and then the lumping together of all the stages for an overall and simultaneous solution. These solution equations are integrated with respect to time for the dynamic response of the system. Stage element such as shown in Figure 6-18 is idealized by the assumptions:

- The composition of the liquid leaving the stage is the same as that on the stage

- There are no gradients across the stage

These assumptions yield an ideal stage; that is, one with perfect mixing of the liquid.

What is shown in Figure 6-22 is a five-tray column with a one-stage condenser and a one-stage reboiler.

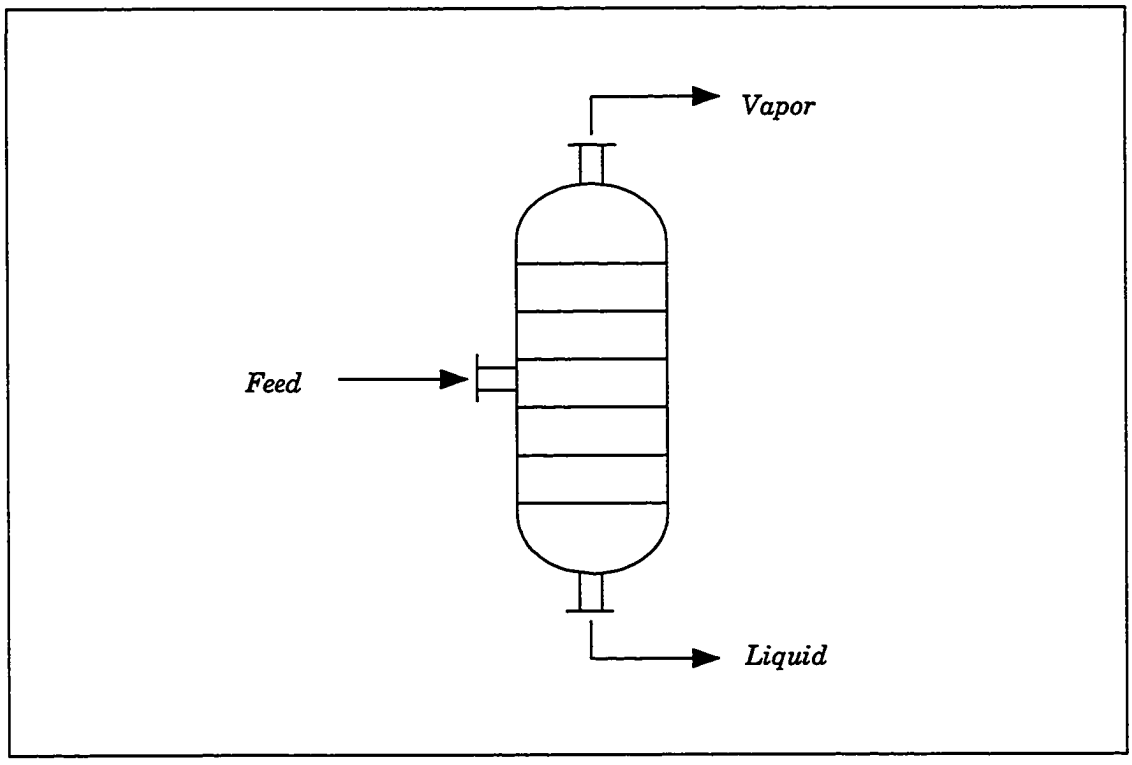



Figure 6-17 Continuous Distillation

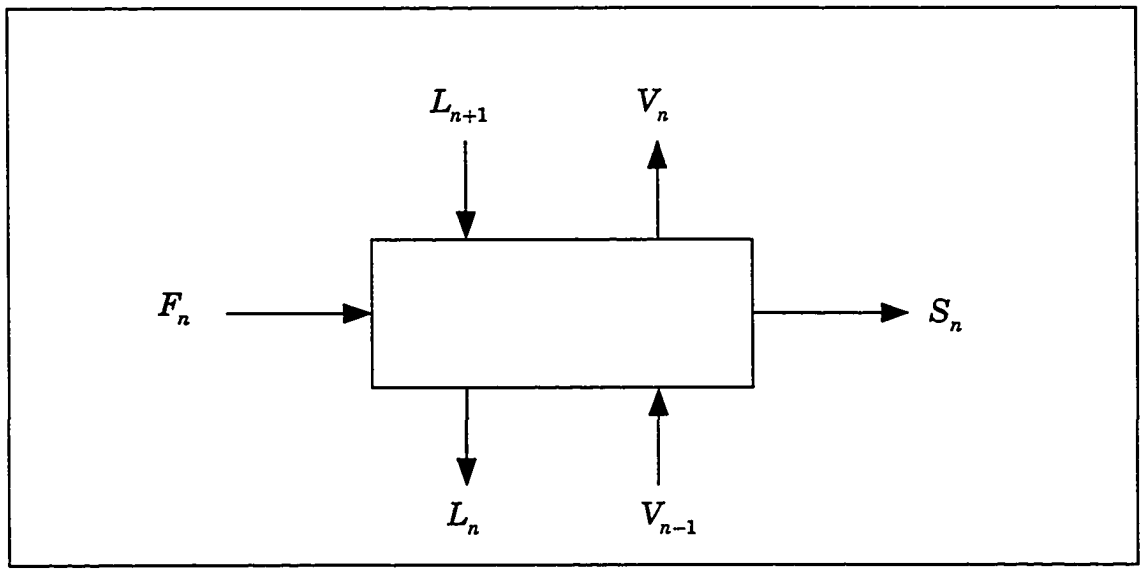



Figure 6-18 Distillation Stage

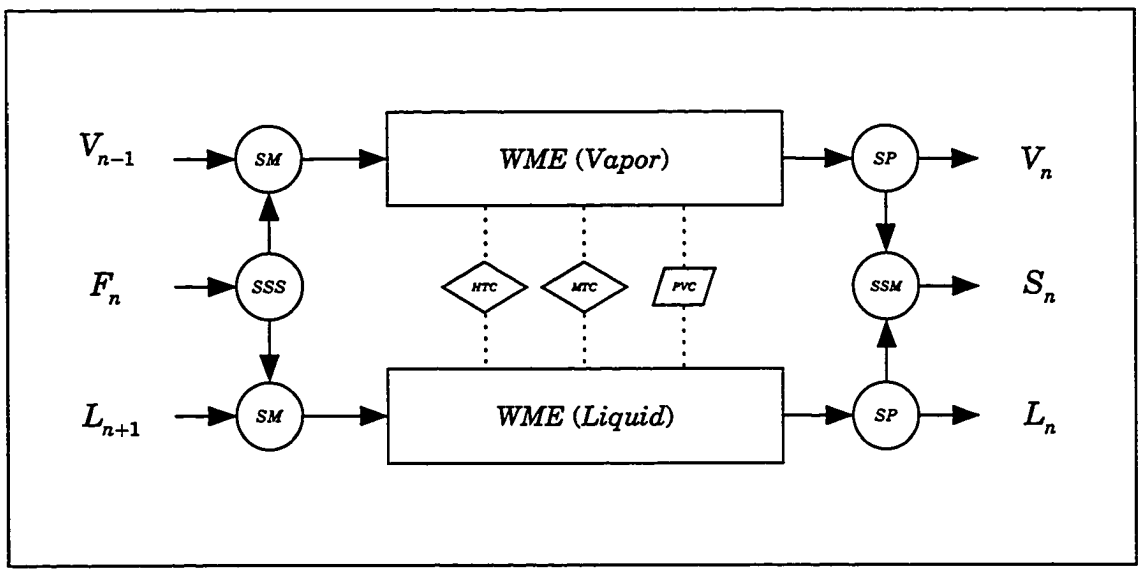



Figure 6-19 Decomposition of Distillation Stage Model

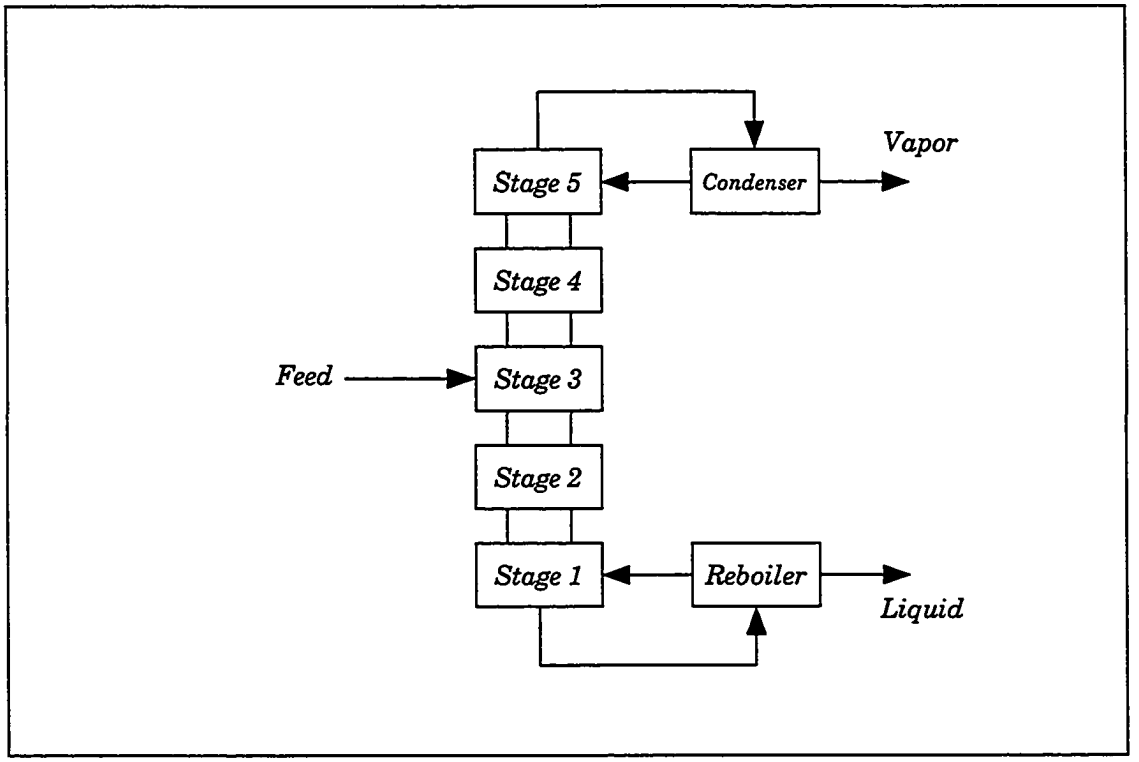



Figure 6-20 Modeling Decomposition of Distillation

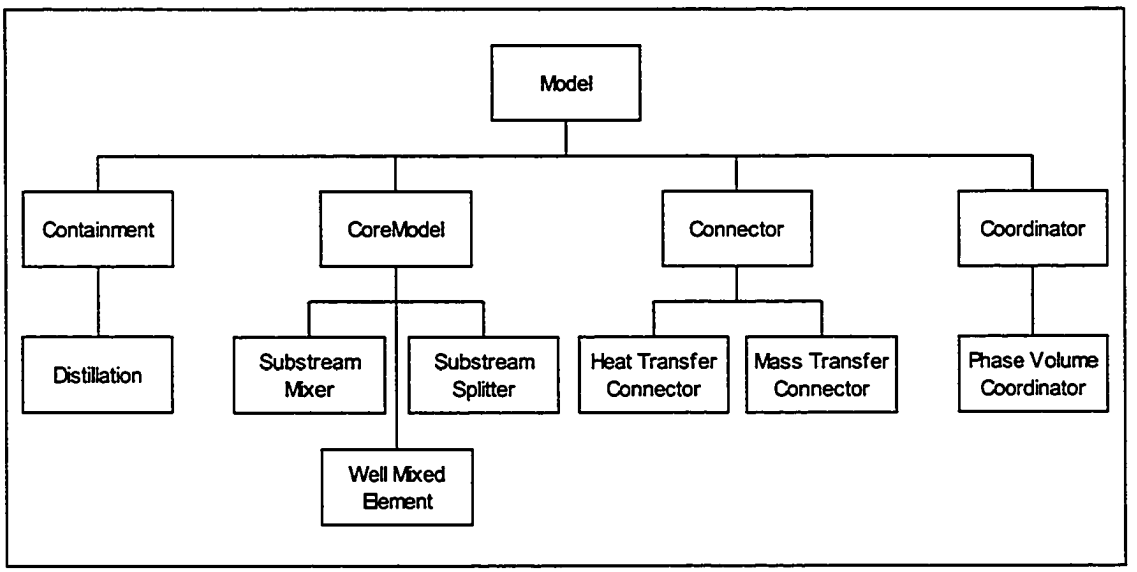



Figure 6-21 Decomposition of Distillation Model

The dynamic responses of the distillation are shown as follows. The

steady-state comparison for distillation column between ForeSee and ASPEN are shown in Table 6-2.

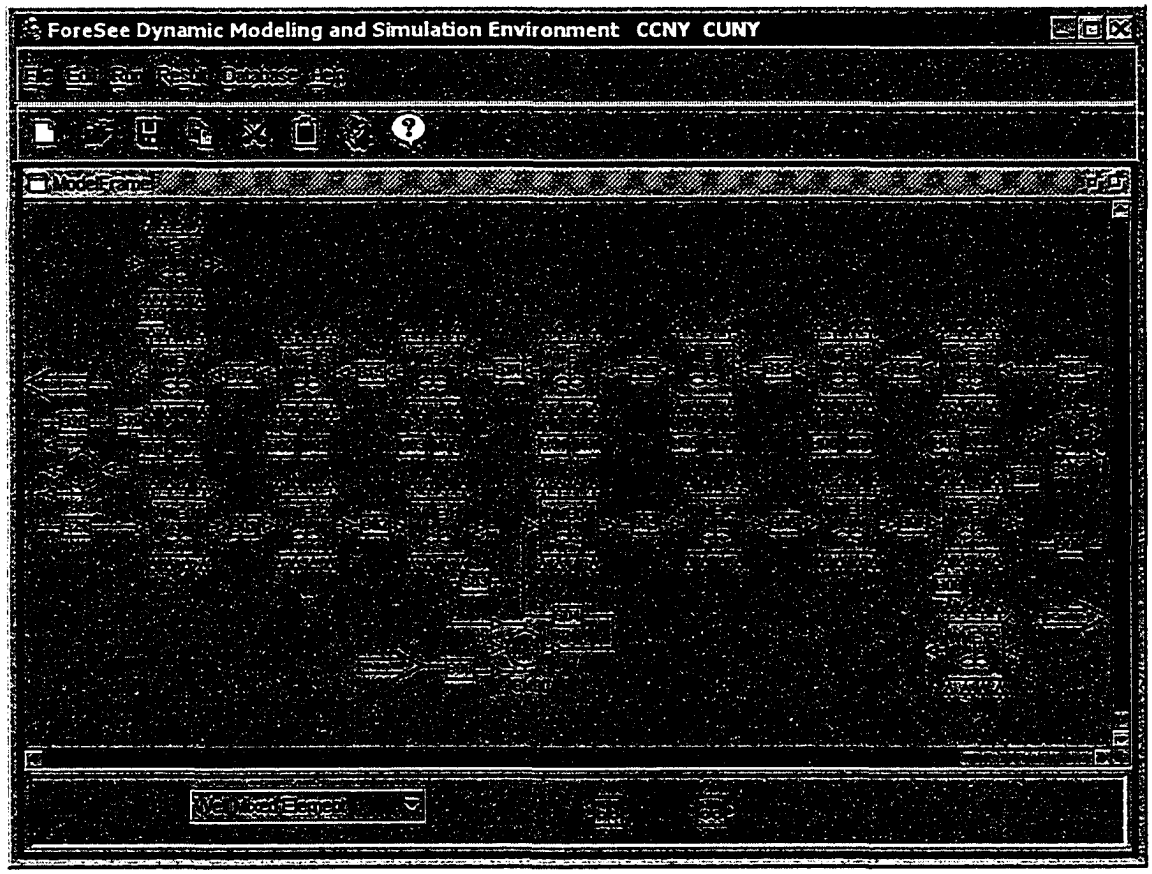



Figure 6-22 Modeling Decomposition of Distillation in ForeSee

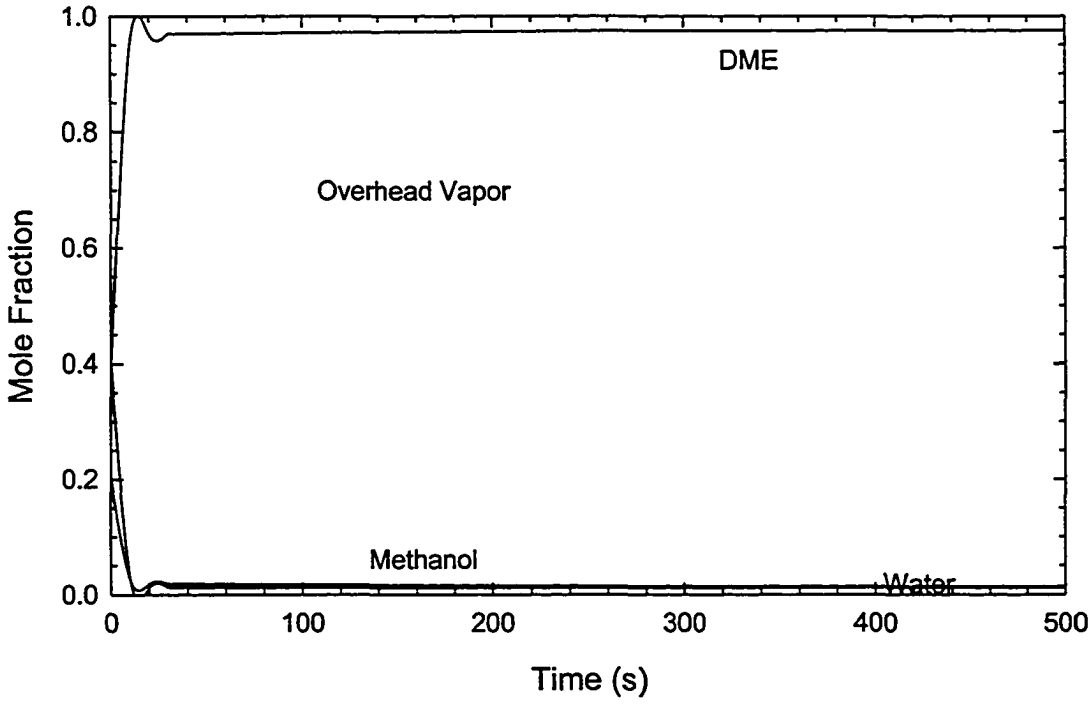



Figure 6-23 Dynamic Response of Overhead Vapor Composition

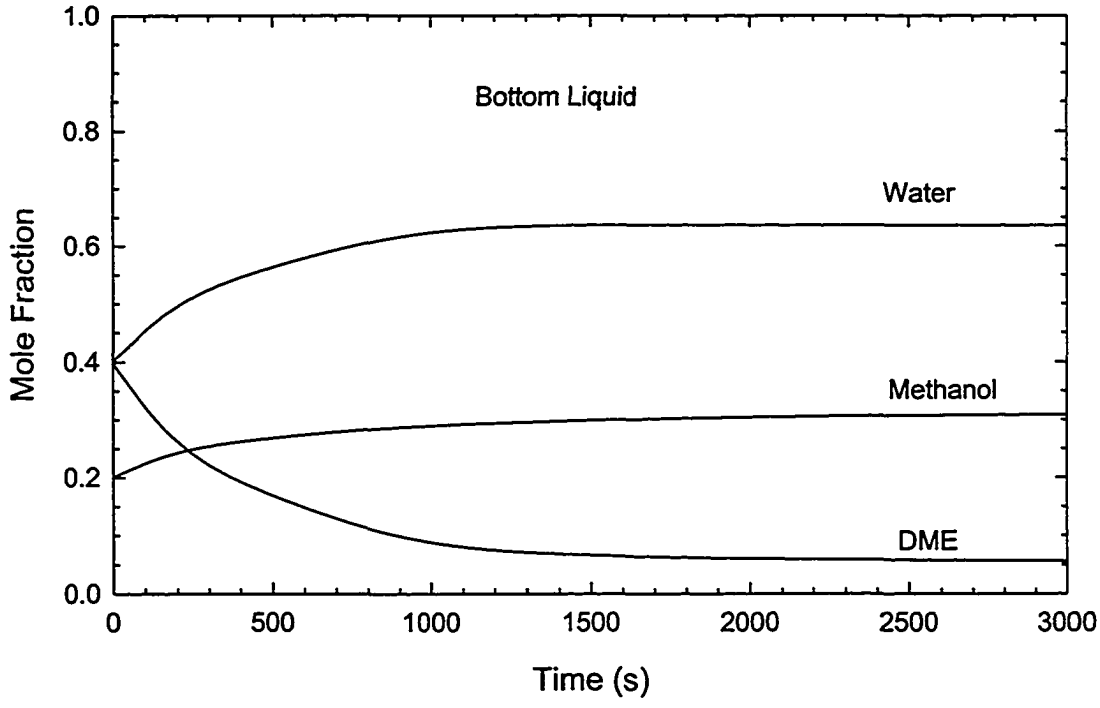



Figure 6-24 Dynamic Response of Bottom Liquid Composition

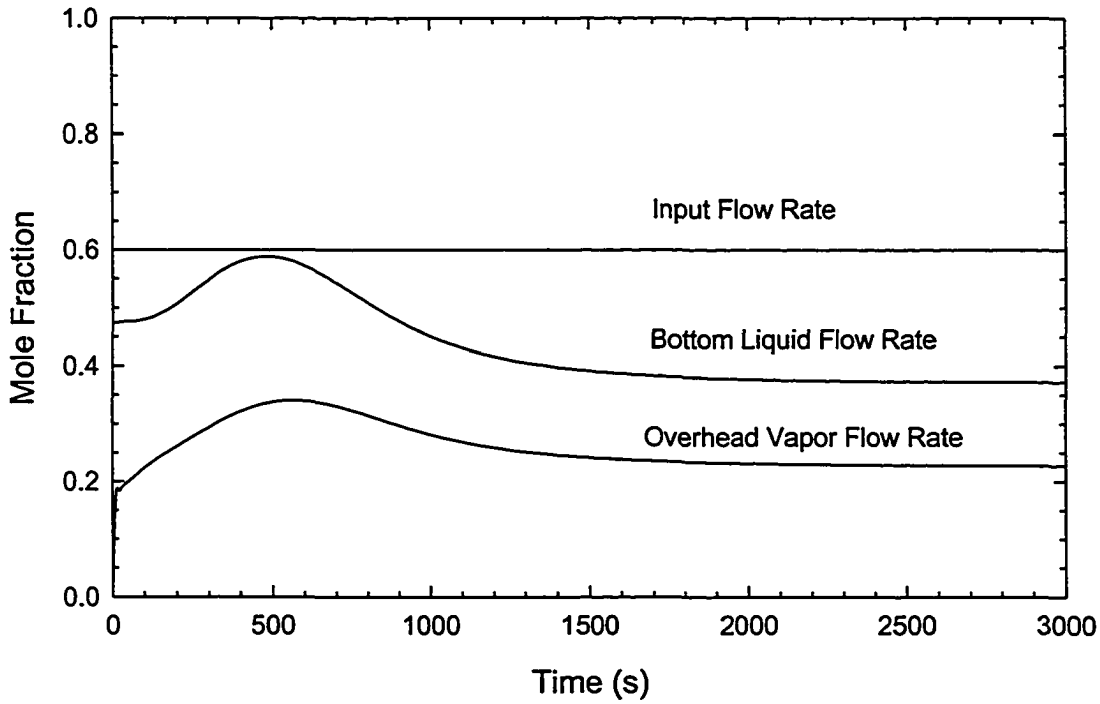



Figure 6-25 Dynamic Response of Flow Rates in Distillation

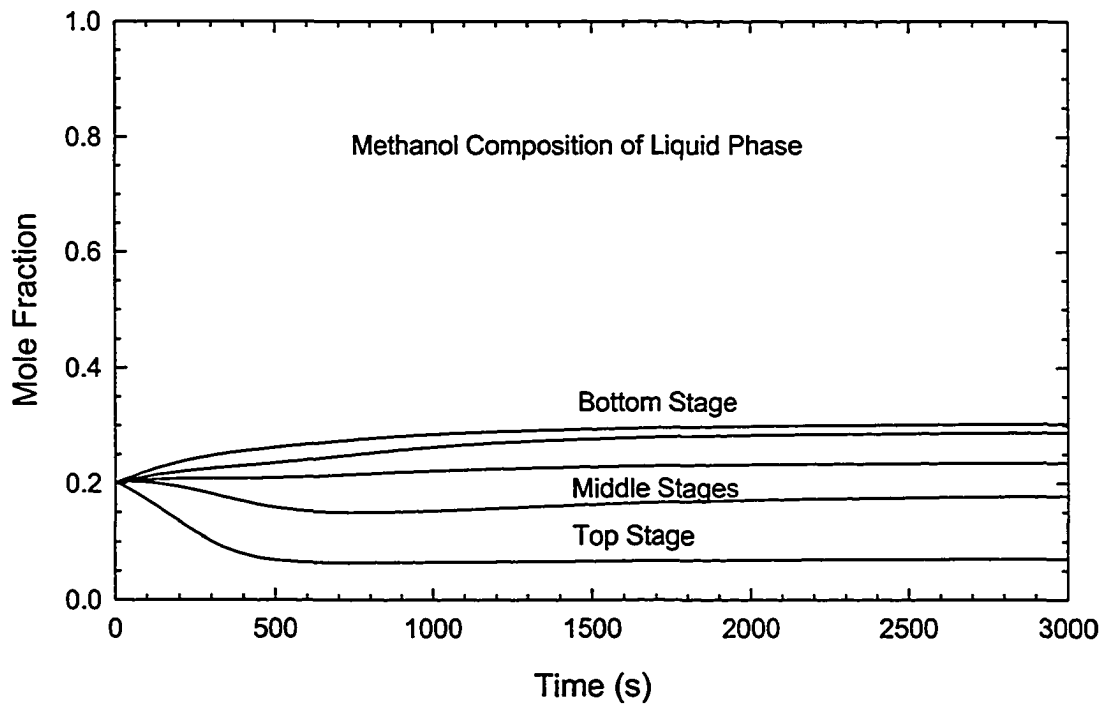



Figure 6-26 Dynamic Responses of Methanol Composition of Liquid Phases

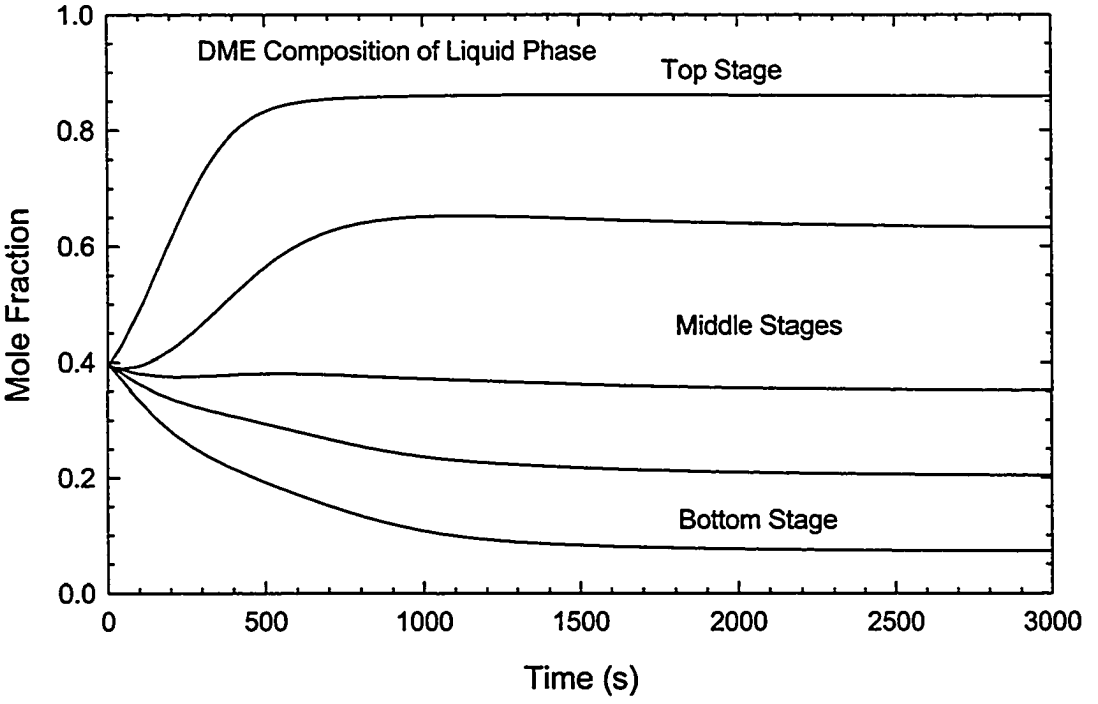



Figure 6-27 Dynamic Responses of DME Composition of Liquid Phases

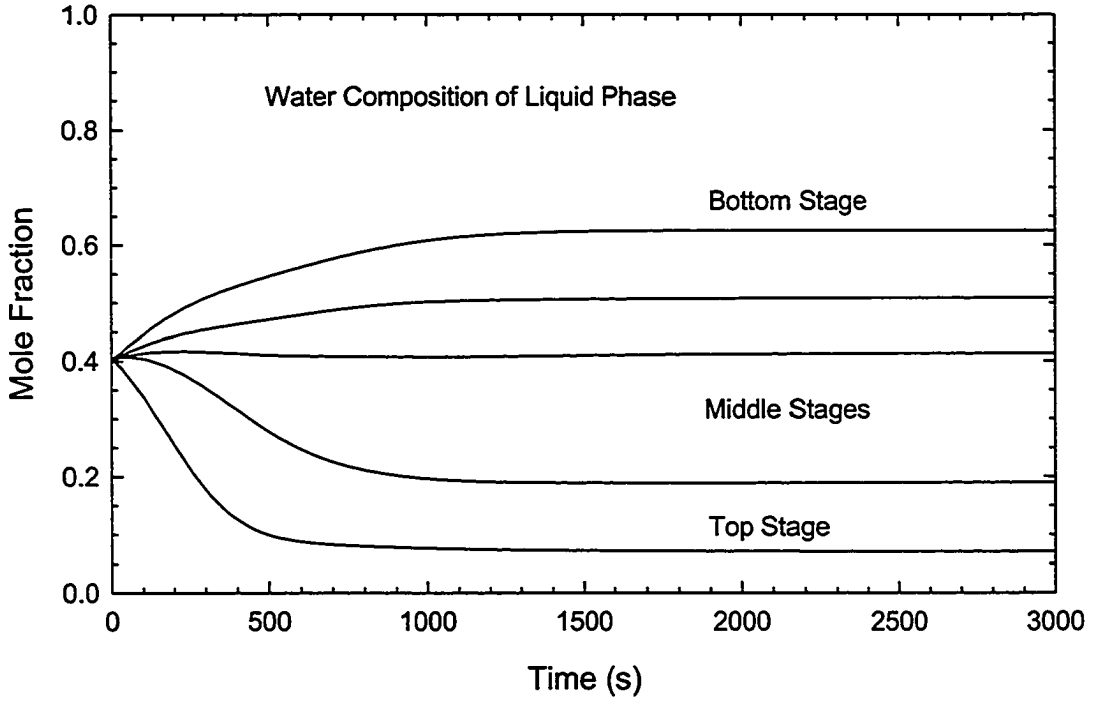



Figure 6-28 Dynamic Responses of Water Composition of Liquid Phases

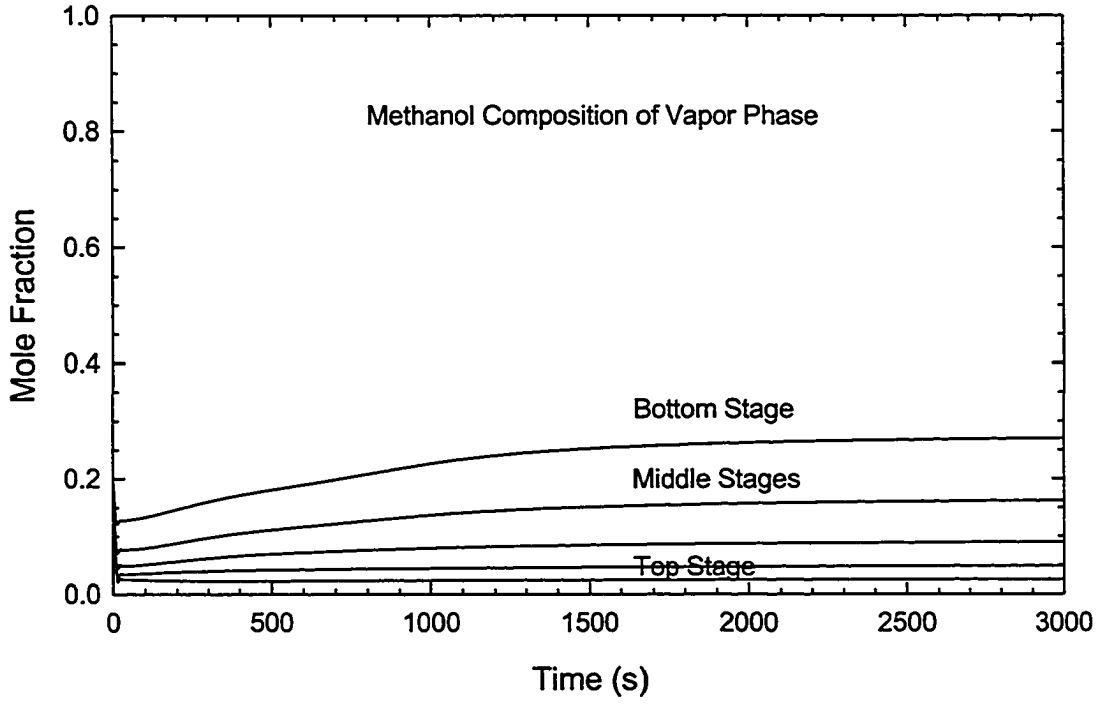



Figure 6-29 Dynamic Responses of Methanol Composition of Vapor Phases

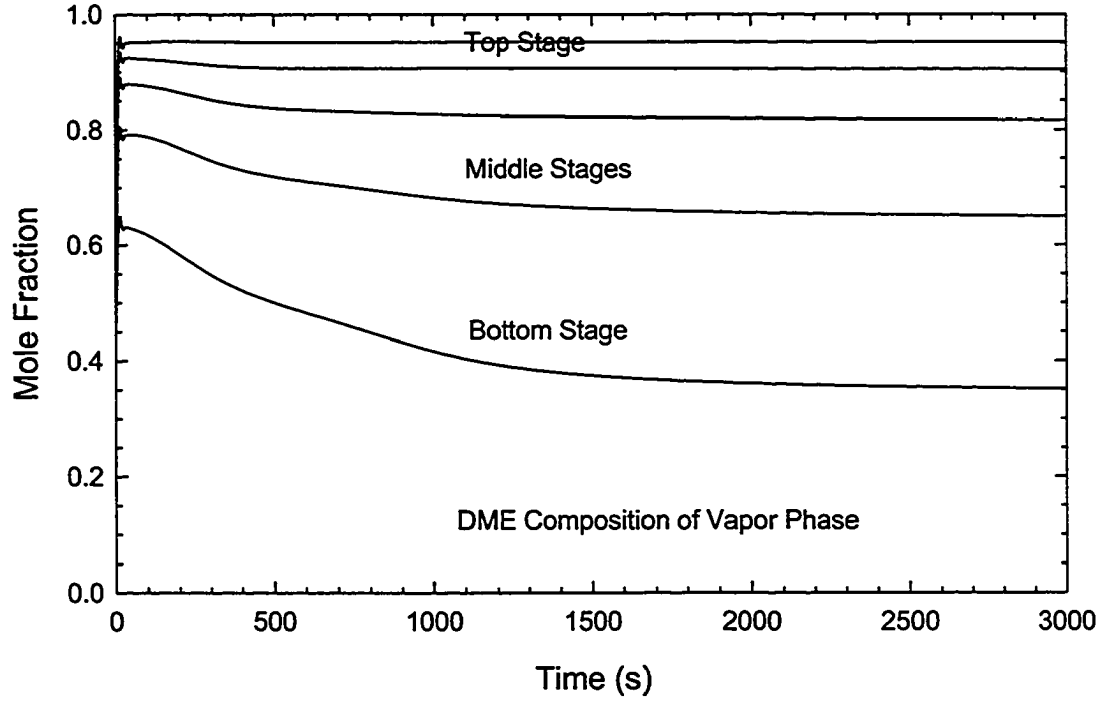



Figure 6-30 Dynamic Responses of DME Composition of Vapor Phases

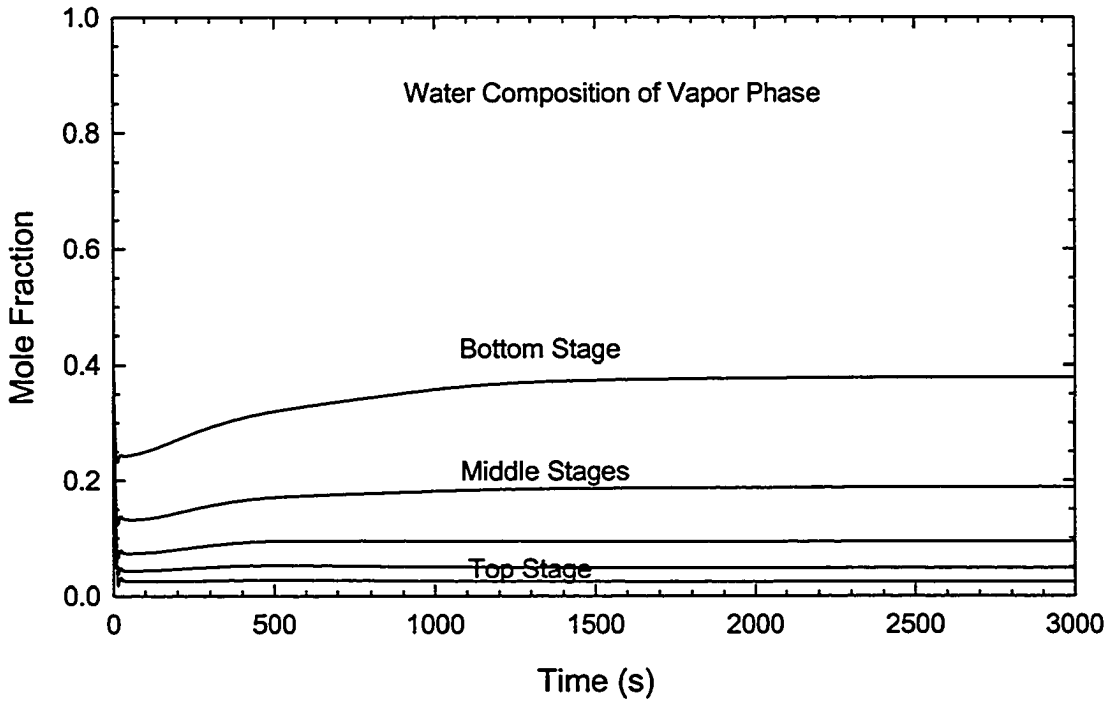



Figure 6-31 Dynamic Responses of Water Composition of Vapor Phases

Table 6-2 Steady-State Comparison for Distillation between ForeSee and ASPEN (Reflux = 2/3)

| Variables | ForeSee | ASPEN |
|---|---|---|
| Feed Flow Rate [kmol/s] | 0.6000 | 0.6000 |
| Feed Composition [%] (DME, Methanol, Water) | (40.00, 20.00, 40.00) | (40.00, 20.00, 40.00) |
| Overhead Flow Rate [kmol/s] | 0.2270 | 0.2270 |
| Overhead Composition [%] (DME, Methanol, Water) | (97.37, 1.54, 1.09) | (97.45, 2.04, 0.51) |
| Bottom Flow Rate [kmol/s] | 0.3725 | 0.3730 |
| Bottom Composition [%] (DME, Methanol, Water) | (5.37, 31.03, 63.60) | (5.04, 30.93, 64.03) |

## 6.2.6 Packed Bed Gas Absorption

The gas absorption is equipped with a gas inlet and distributing space

at the bottom; a liquid inlet and distributor at the top; gas and liquid outlets at the top and bottom, respectively; and a supported mass of inert solid tower packing. The inlet liquid is distributed over the top of the packing by the distributor. The gas enters the distributing space below the packing and flows upward through the interstices in the packing countercurrent to the flow of the liquid. The packing provides a large area of contact between the liquid and gas and encourages intimate contact between phases. The solute in the rich gas is absorbed by the fresh liquid entering the tower and lean gas leaves the top. The liquid is enriched in solute as it flows down the tower, and concentrated liquid leaves the bottom of the tower through the liquid outlet.

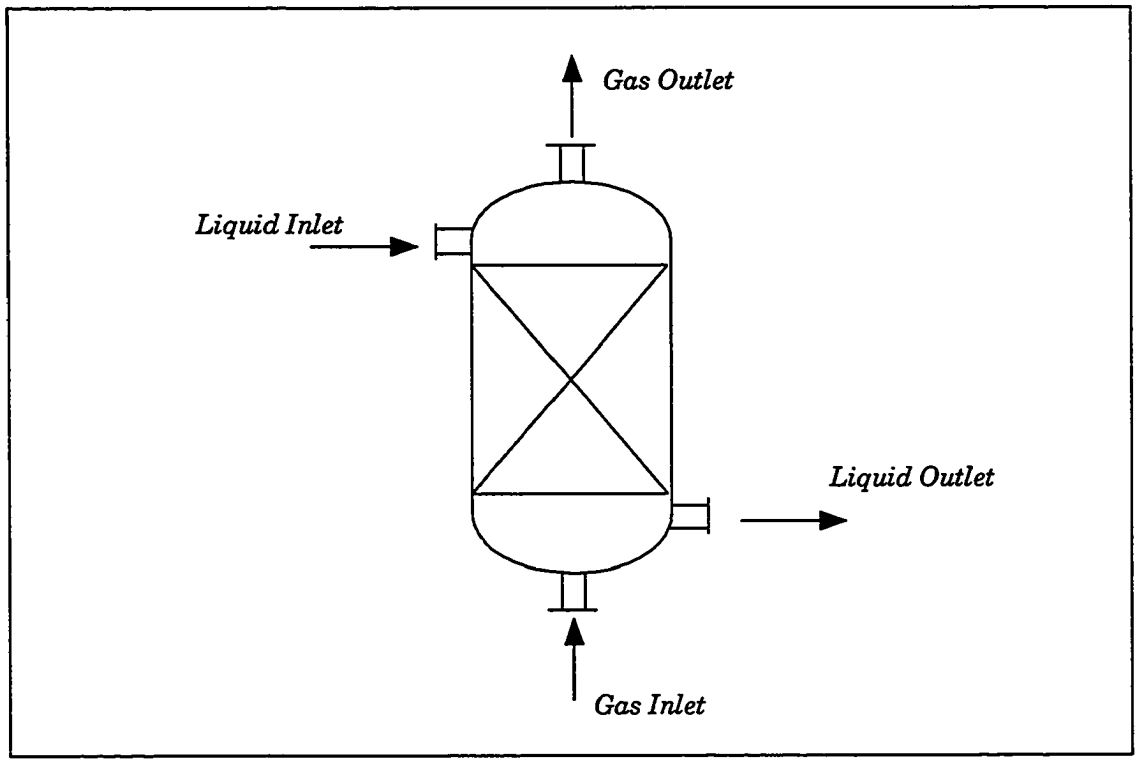



Figure 6-32 Packed Bed Gas Absorption

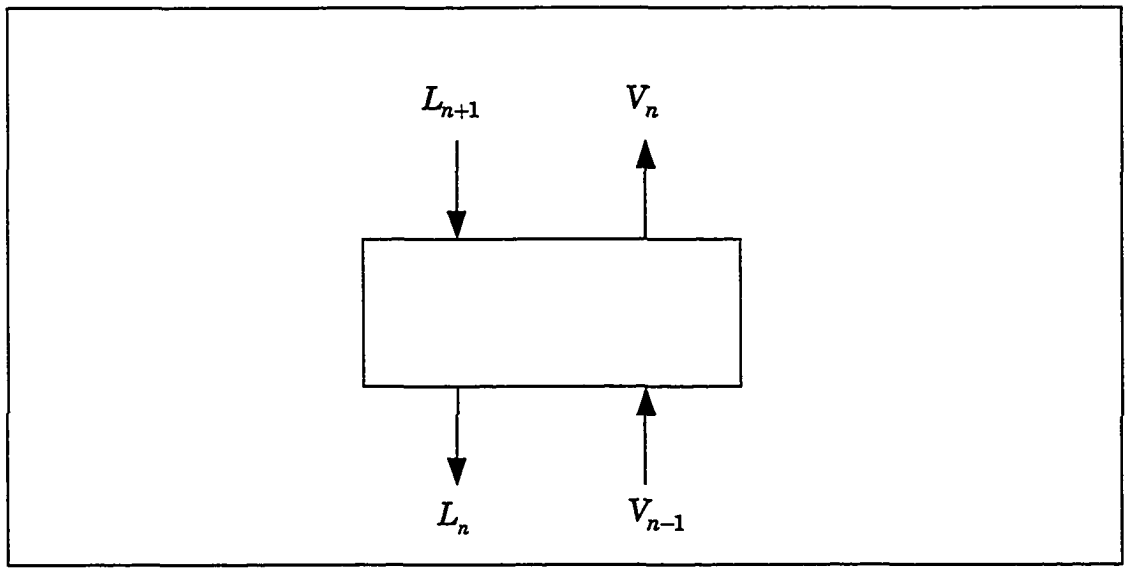



Figure 6-33 Packed Bed Gas Absorption Section

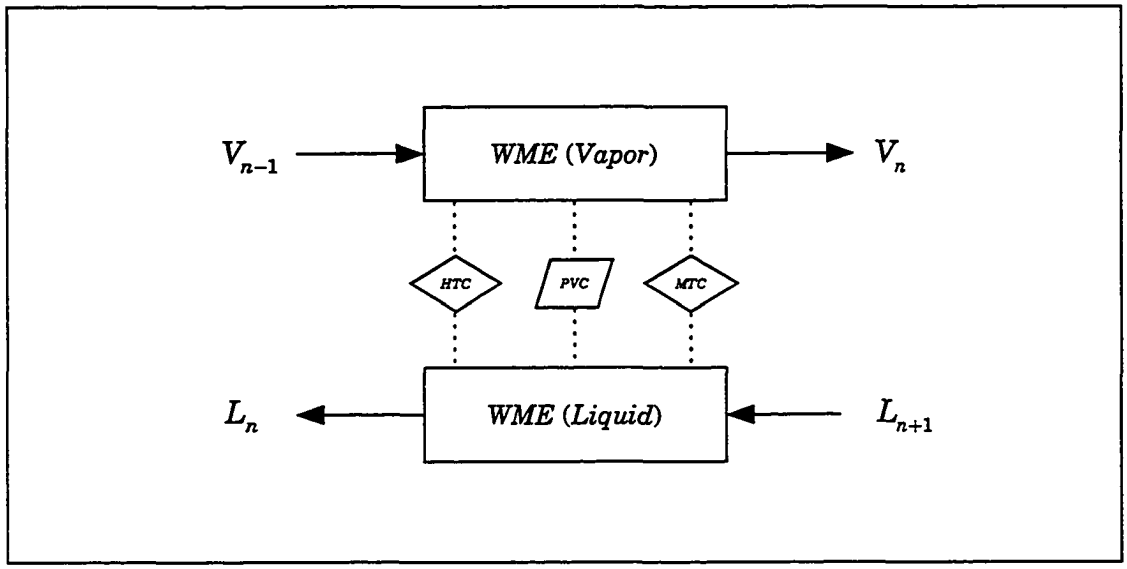



Figure 6-34 Decomposition of Gas Absorption Section Model

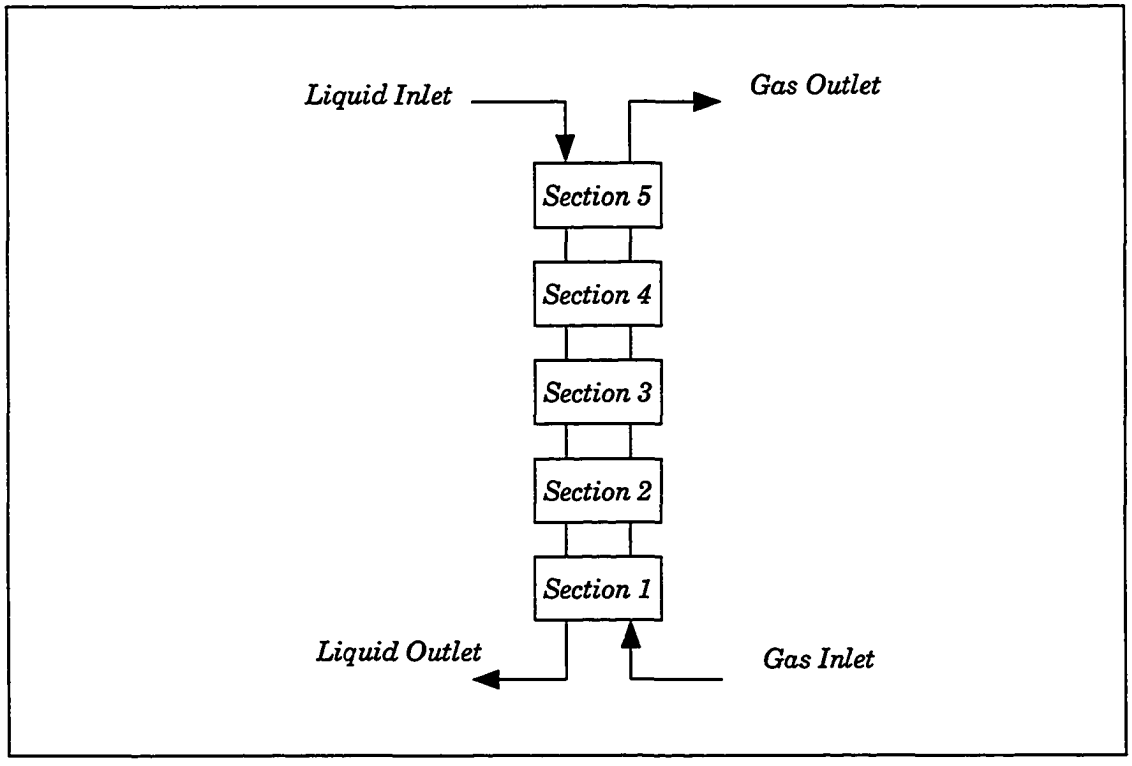



Figure 6-35 Modeling Decomposition of Gas Absorption

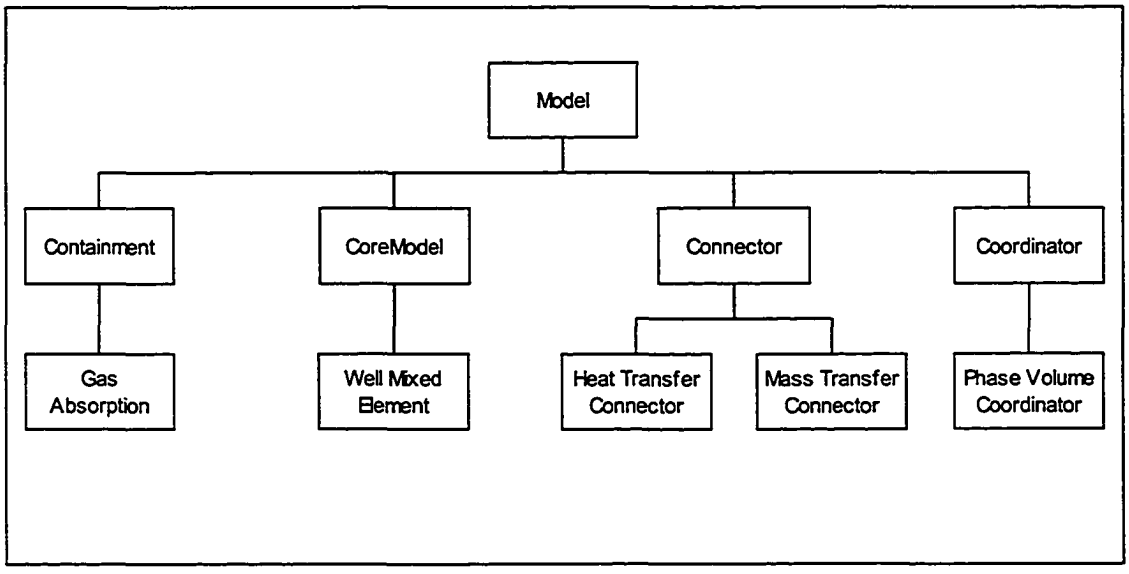



Figure 6-36 Decomposition of Gas Absorption Model

The packed bed gas absorption can also be approximated by using plug

flow element model, which is illustrated in Figure 6-37.

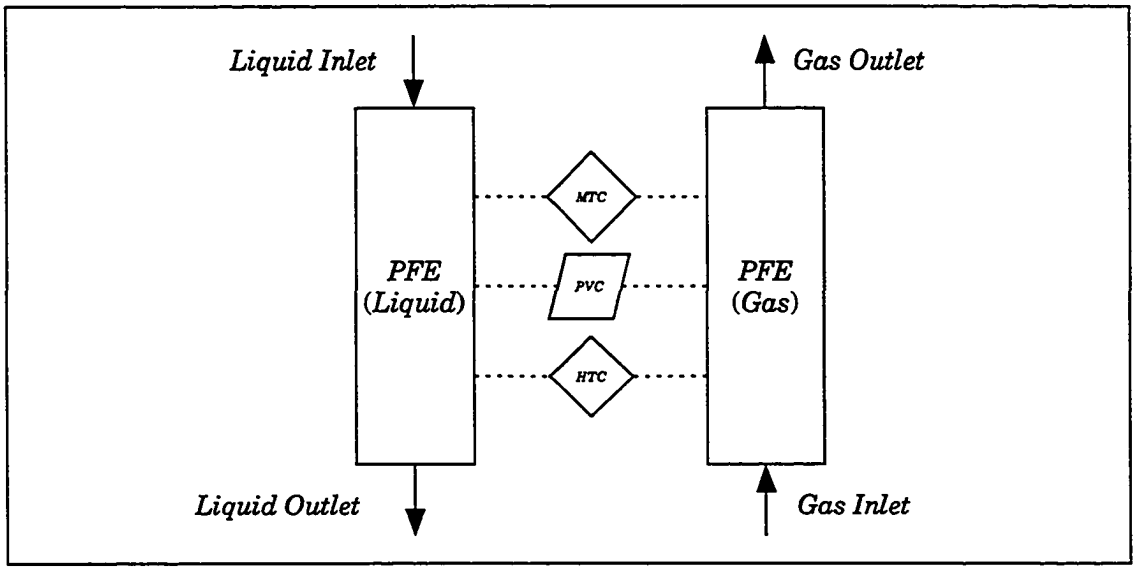



Figure 6-37 Modeling Decomposition of Packed Bed Gas Absorption

## 6.2.7 Mixer-Settler

The original and in concept the simplest way of accomplishing extractions is to mix the two phases thoroughly in one vessel and then to allow the phases to separate in another vessel. A series of such operations performed with series or countercurrent flows of the phases can accomplish any desired degree of separation. A schematic of such a unit is in Figure 6-38.

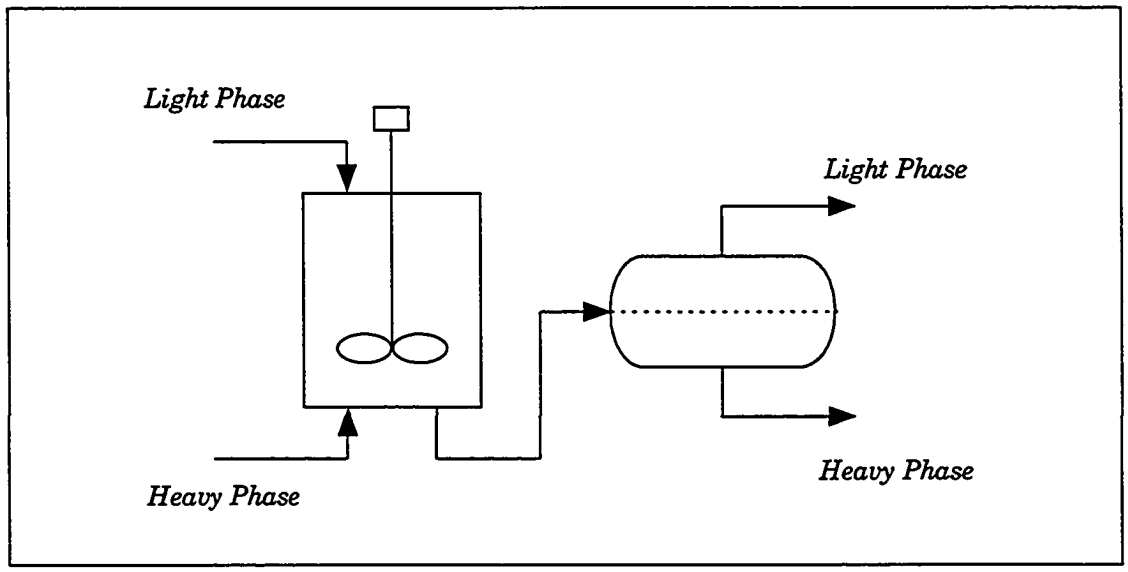



Figure 6-38 Mixer-Settler

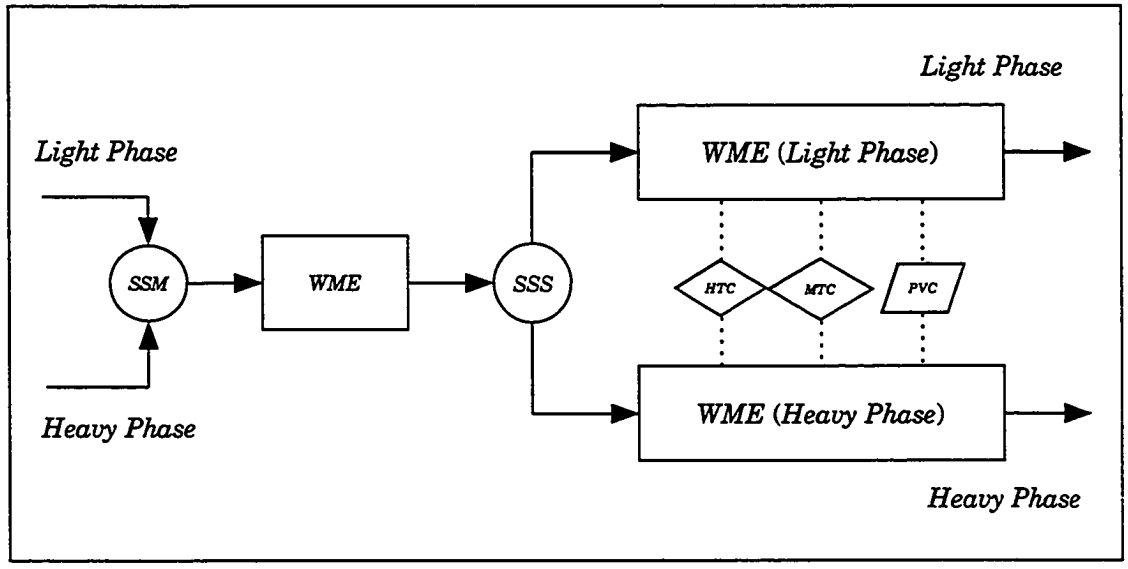



Figure 6-39 Modeling Decomposition of Mixer-Settler

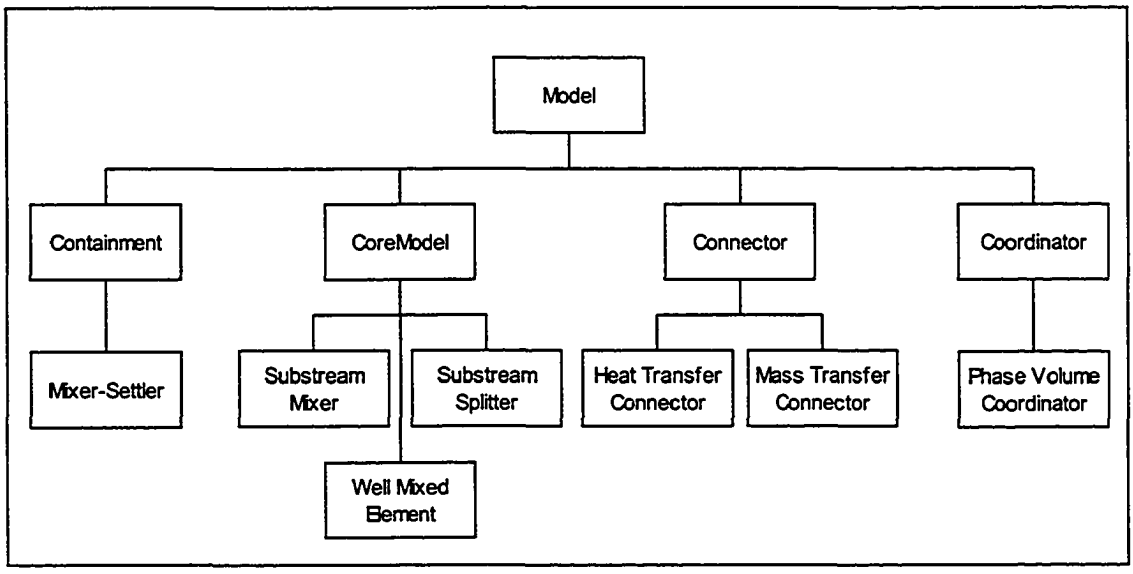



Figure 6-40 Decomposition of Mixer-Settler Model

## 6.2.8 Evaporator

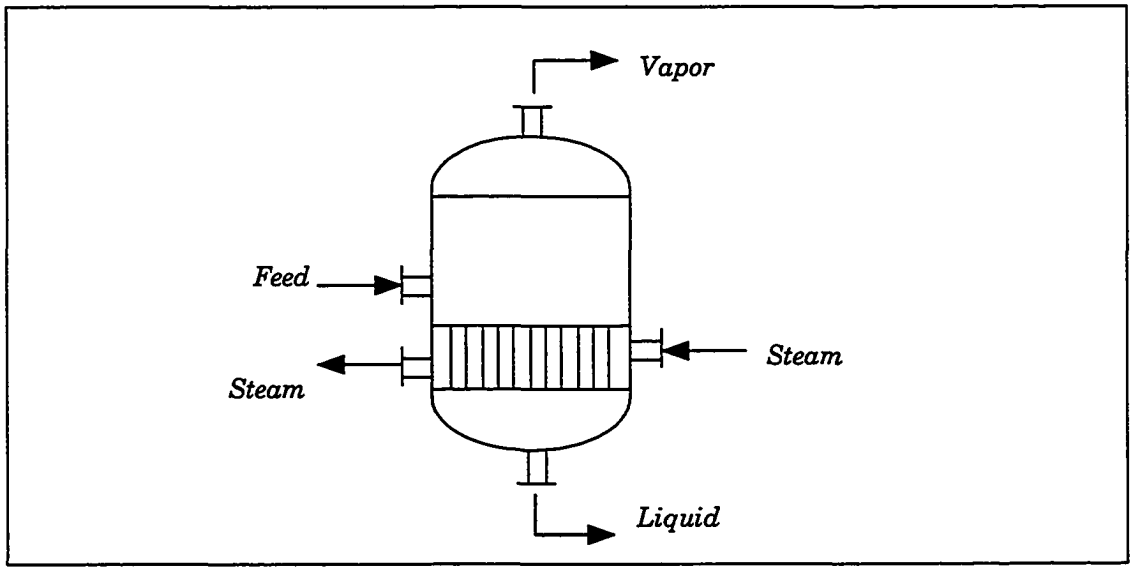



Figure 6-41 Evaporator

Evaporators employ heat to concentrate solutions. They are reboilers

with special provisions for separating liquid and vapor phases. A schematic of such a unit is represented in Figure 6-41.

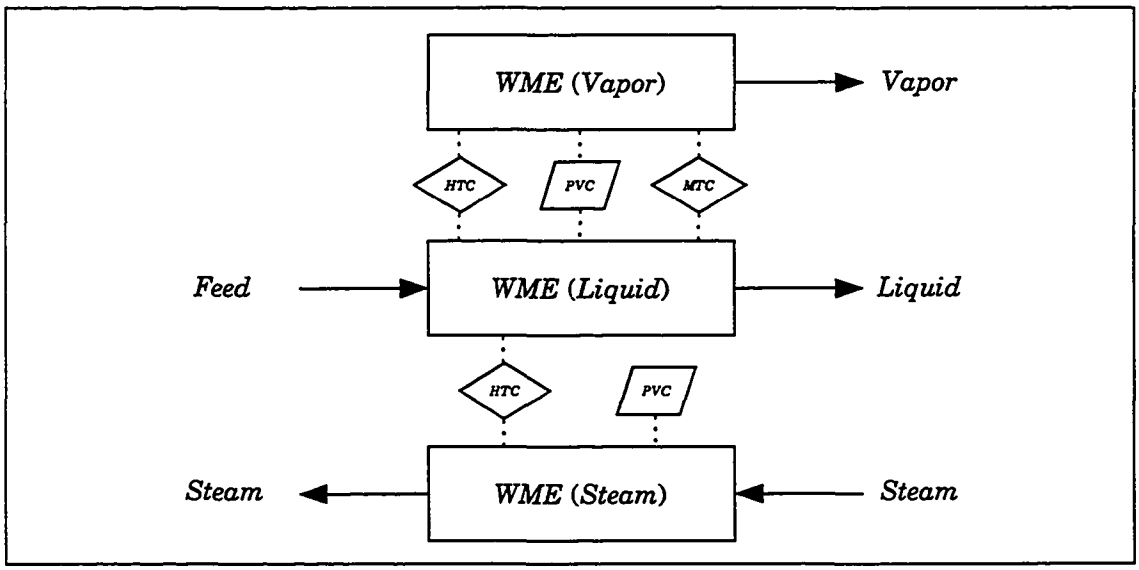



Figure 6-42 Modeling Decomposition of Evaporator

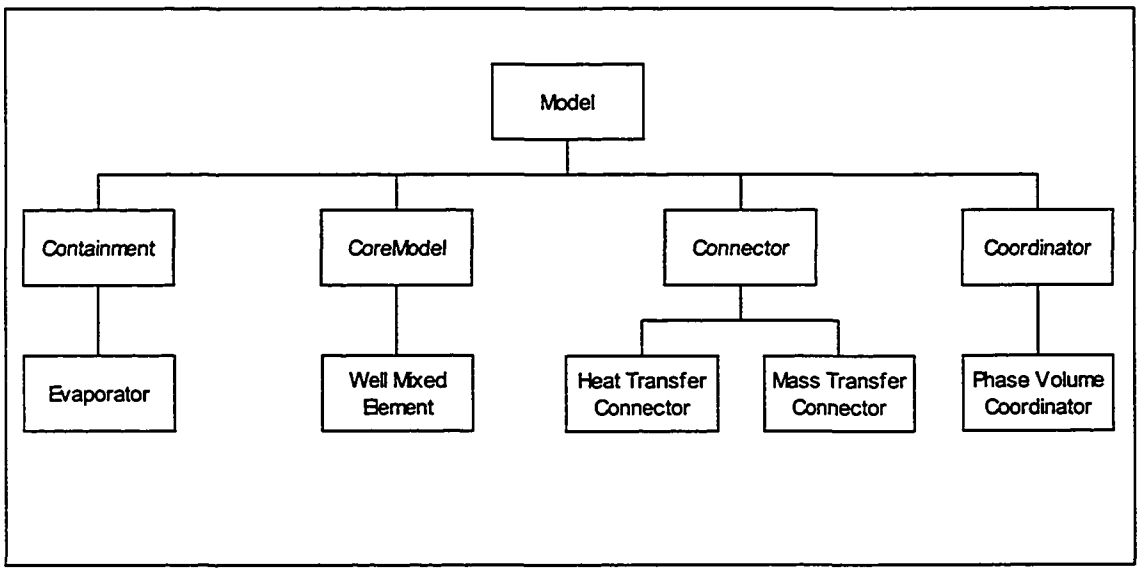



Figure 6-43 Decomposition of Evaporator Model

### 6.2.9 Gas Membrane Separator

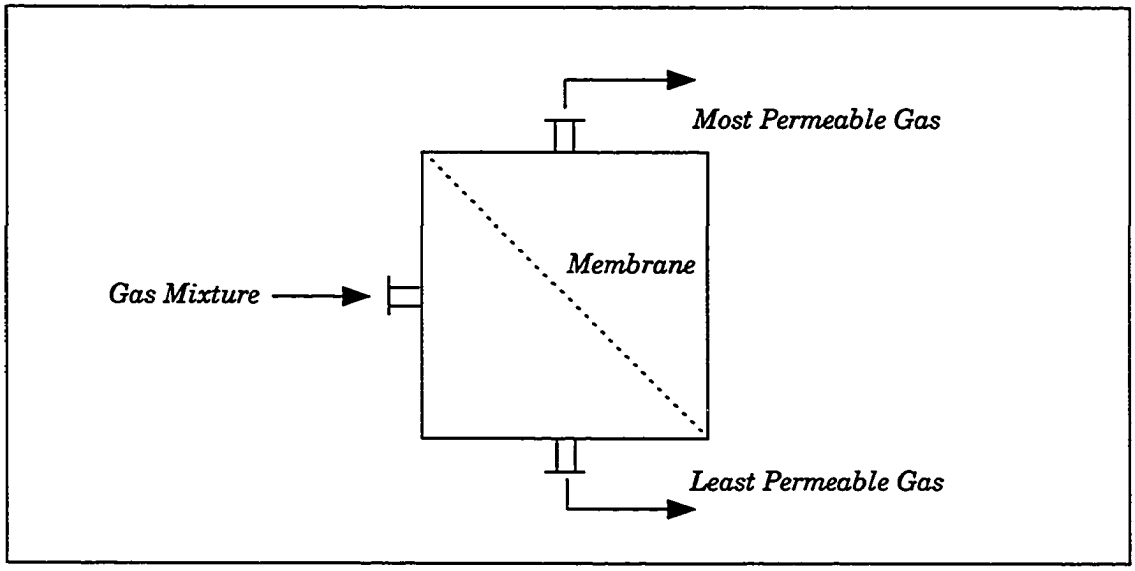



Figure 6-44 Gas Membrane Separator

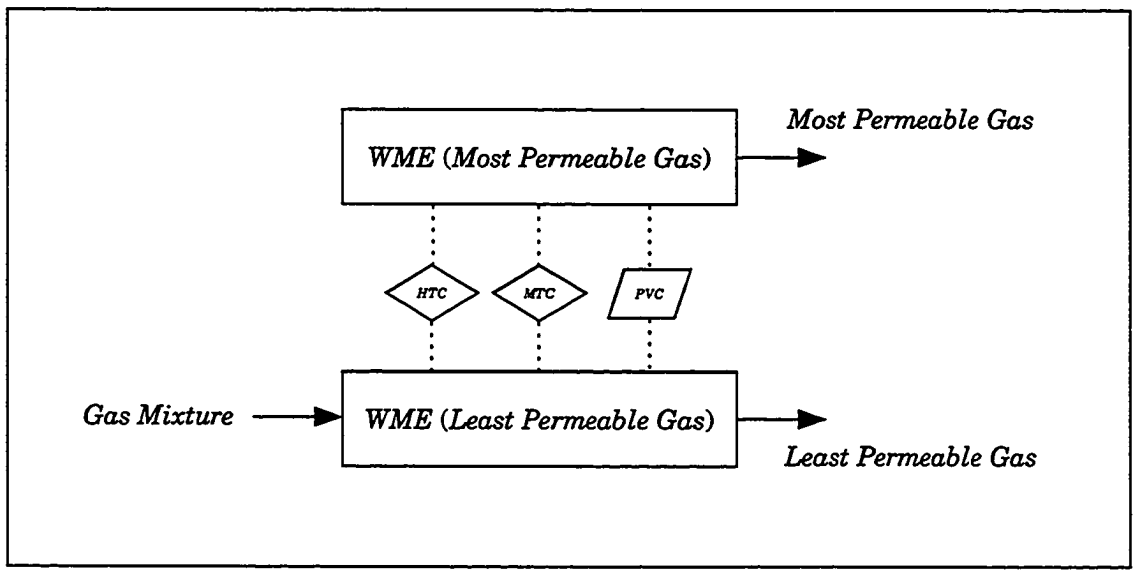



Figure 6-45 Modeling Decomposition of Gas Membrane Separator

Differences in rates of permeation of membranes by various gases are

utilized for the separation of mixtures, for instance, of hydrogen from ammonia plant gas, of carbon dioxide from natural gas, and of helium from natural gas. A schematic of the simplest modeling of such a unit is shown in Figure 6-44.

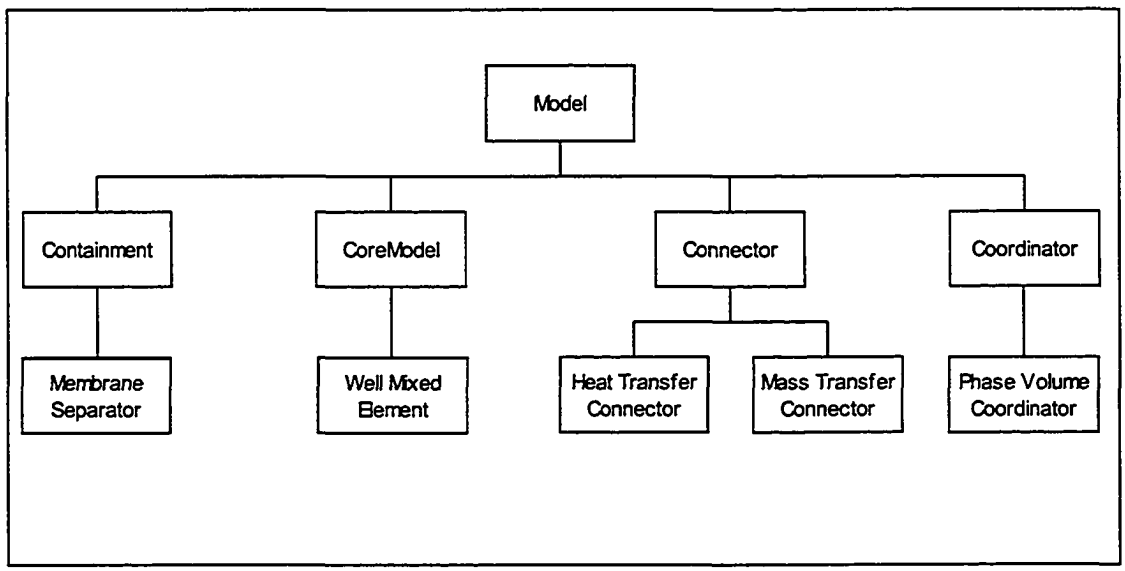



Figure 6-46 Decomposition of Gas Membrane Separator Model

## 6.2.10 Decanter

A gravity decanter is used for the continuous separation of two immiscible liquids of differing densities. The feed mixture enters at one end of the separator. The two liquids flow slowly through the vessel, separate into two layers, and discharge at the other end of the separator. A schematic of such a unit is in Figure 6-47.

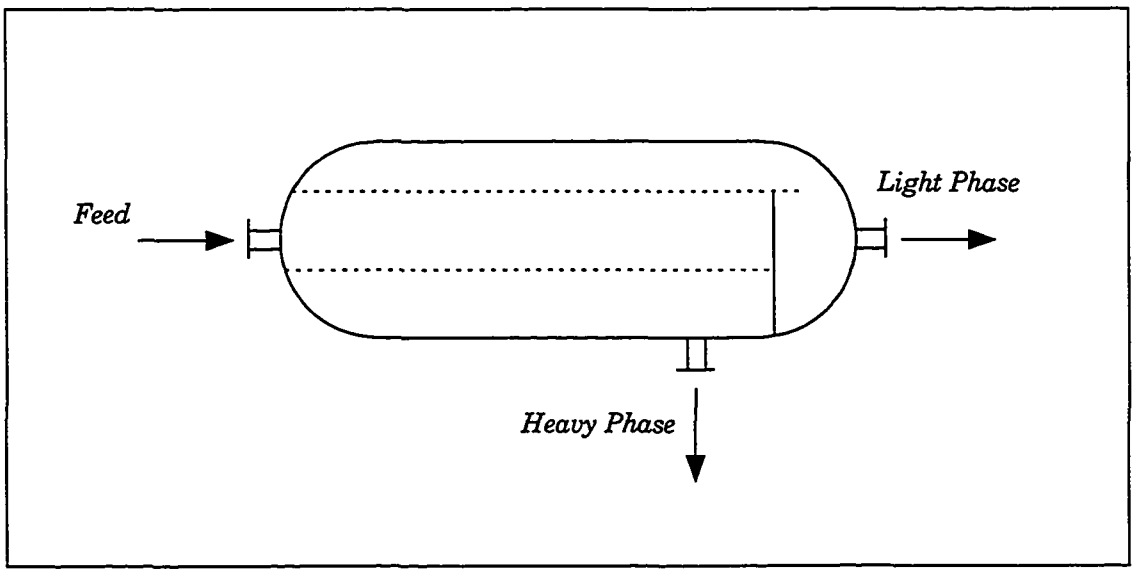



Figure 6-47 Decanter

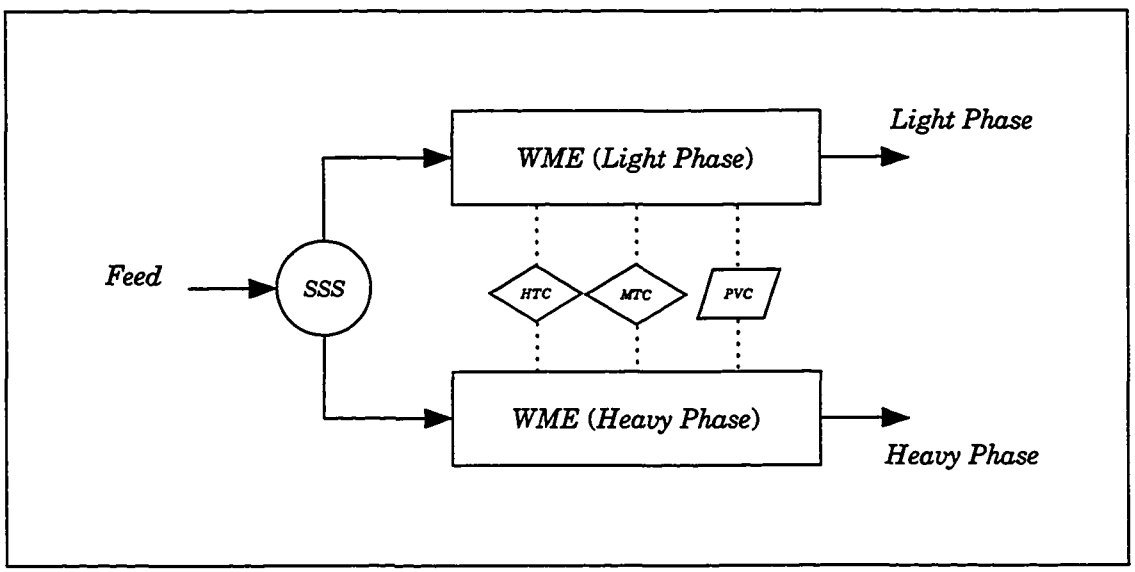



Figure 6-48 Modeling Decomposition of Decanter

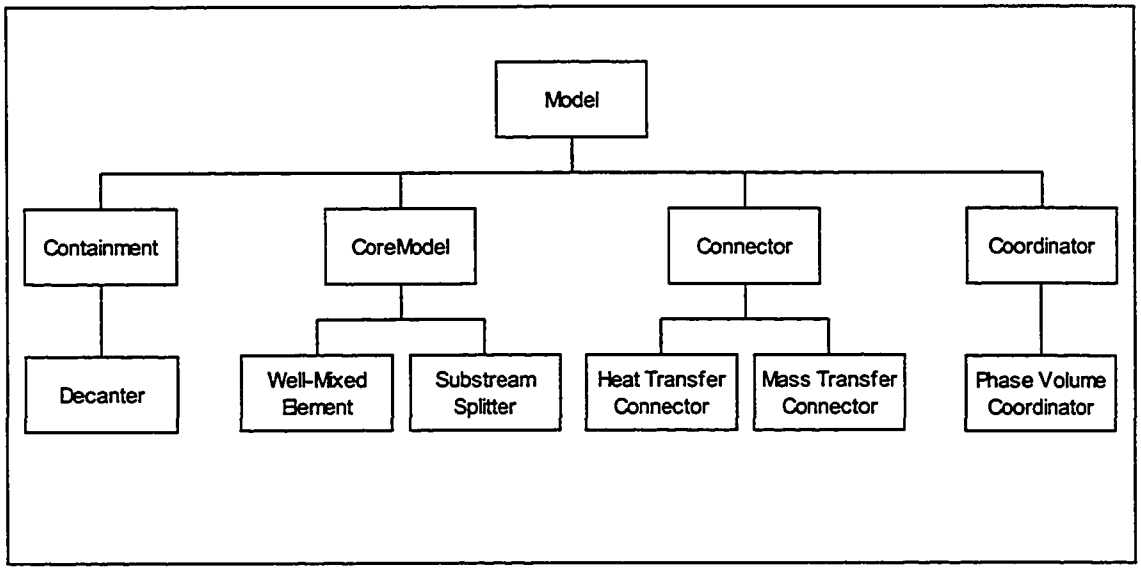



Figure 6-49 Decomposition of Decanter Model

## 6.2.11 Cyclone Separator

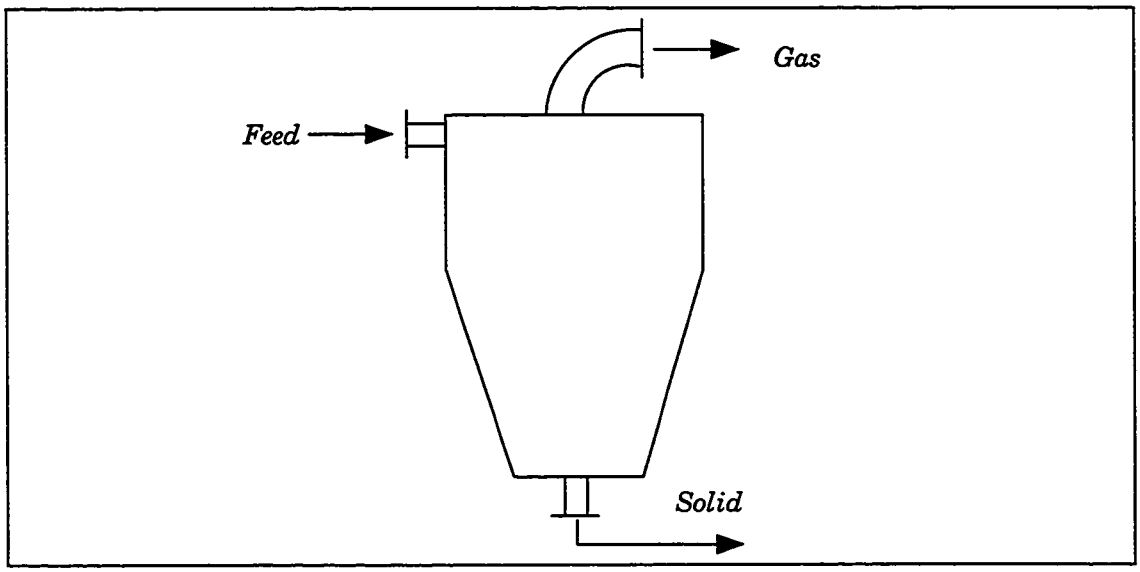



Figure 6-50 Cyclone Separator

In cyclone separators the gas enters tangentially at a high velocity,

rotates several times, and leaves through a central pipe. Such equipment has been used widely for the removal of dusts. A schematic of such a unit is represented in Figure 6-50.

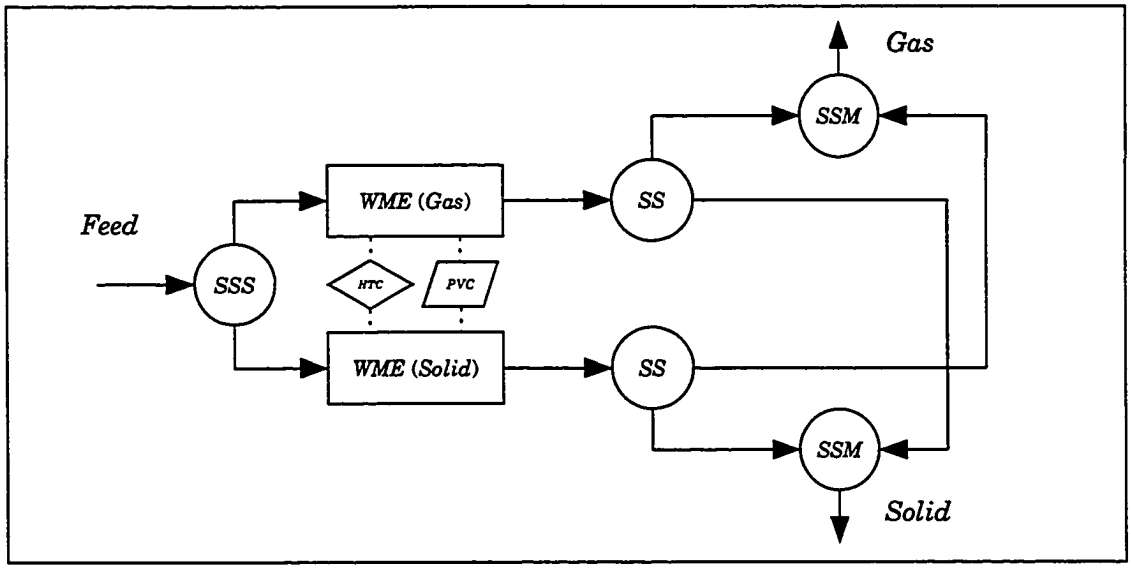



Figure 6-51 Modeling Decomposition of Cyclone Separator

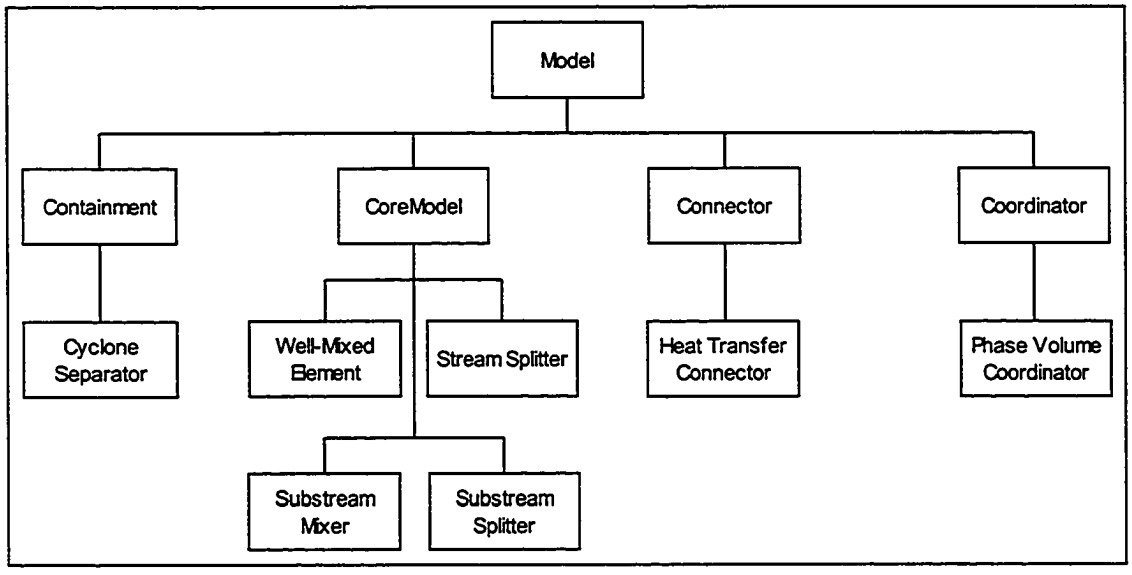



Figure 6-52 Decomposition of Cyclone Separator Model

## 6.3 PFE-Based Equipment Models

The core models of this type of equipment models are mainly plug flow elements.

### 6.3.1 Tubular Flow Reactor

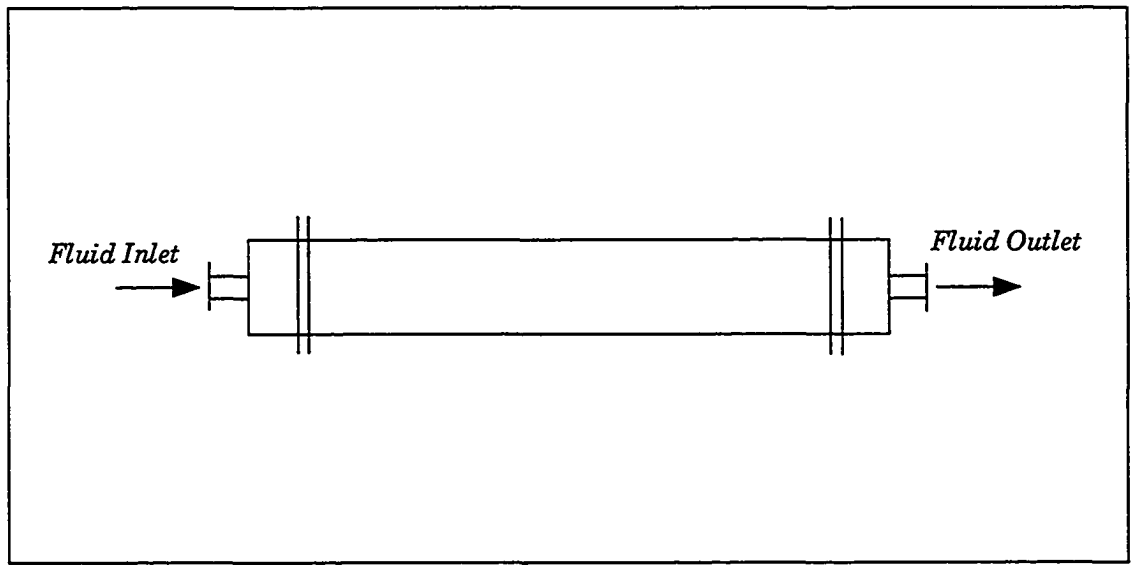



Figure 6-53 Tubular Flow Reactor

The ideal behavior of tubular flow reactor (TFR) is plug flow, in which all non-reacting molecules have equal residence times. Any backmixing that occurs is incidental, the result of natural turbulence or that induced by obstructions to flow by catalyst granules or tower packing or necessary internals of the vessels. The action of such obstructions can be two-edged, however, in that some local back-mixing may occur, but on the whole a good approach to plug flow is developed because large scale turbulence is inhibited. Any required initial blending of reactants is accomplished in

mixing nozzles. As a result of chemical reaction, gradients of concentration and temperature are developed in the axial direction of tubular flow reactor.

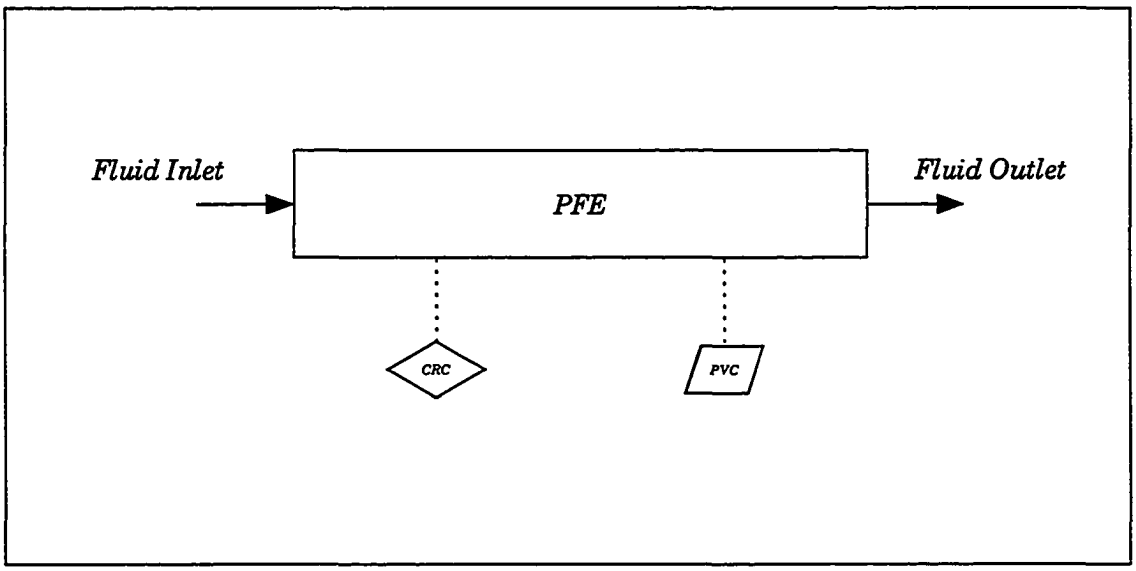



Figure 6-54 Modeling Decomposition of Tubular Flow Reactor

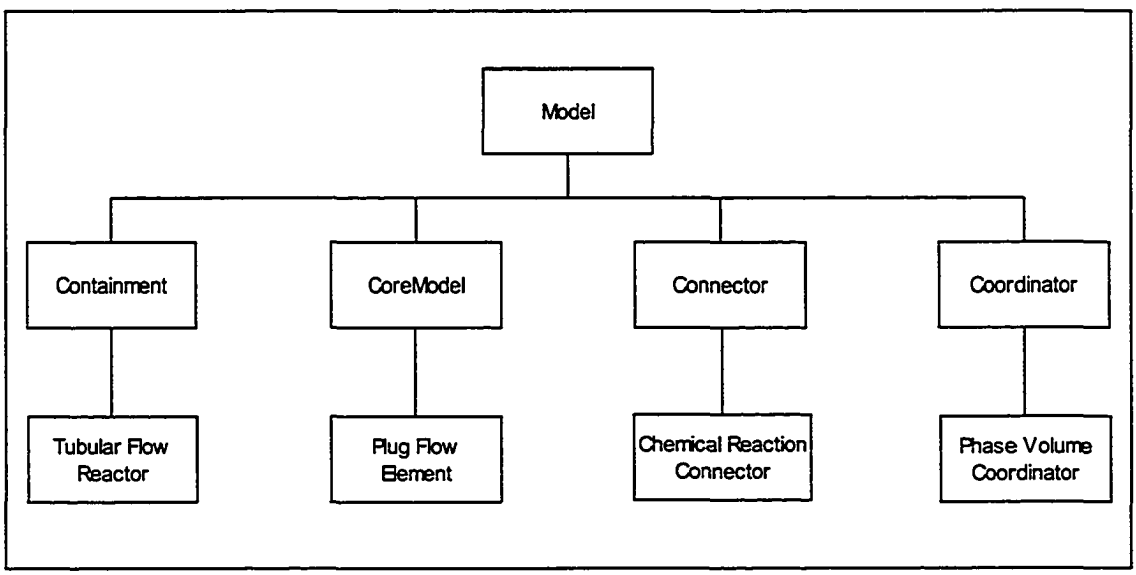



Figure 6-55 Decomposition of Tubular Flow Reactor Model

The dynamic responses of temperature, composition, and conversion in

the reactor are shown as follows.

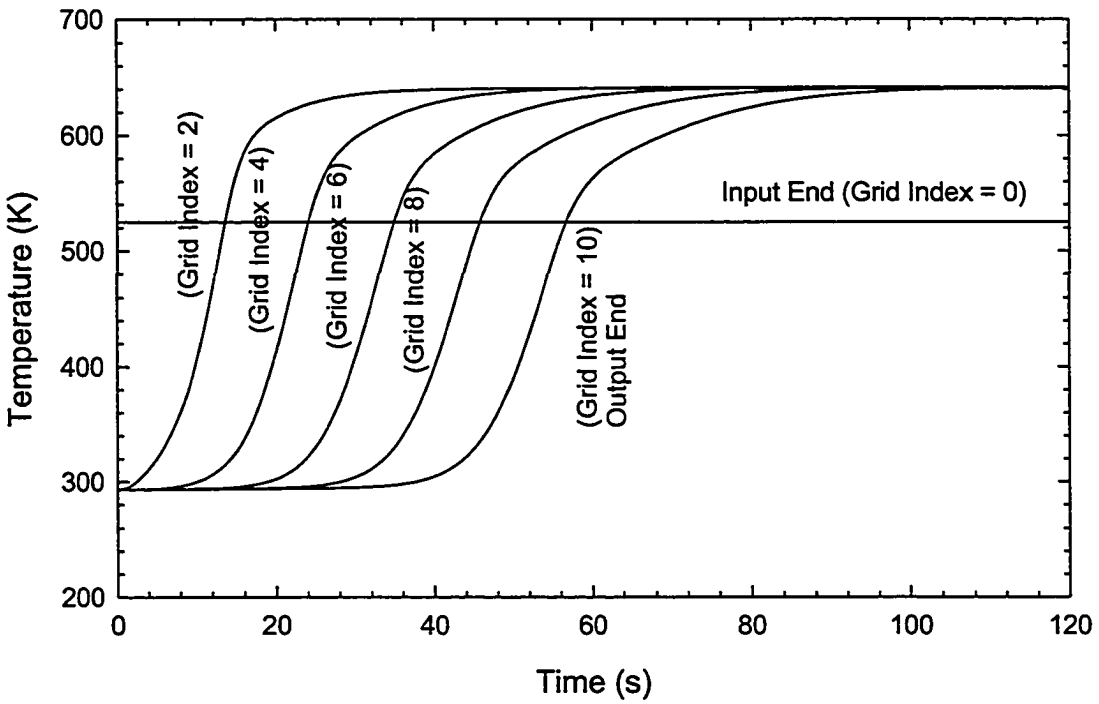



Figure 6-56 Dynamic Response of Temperature

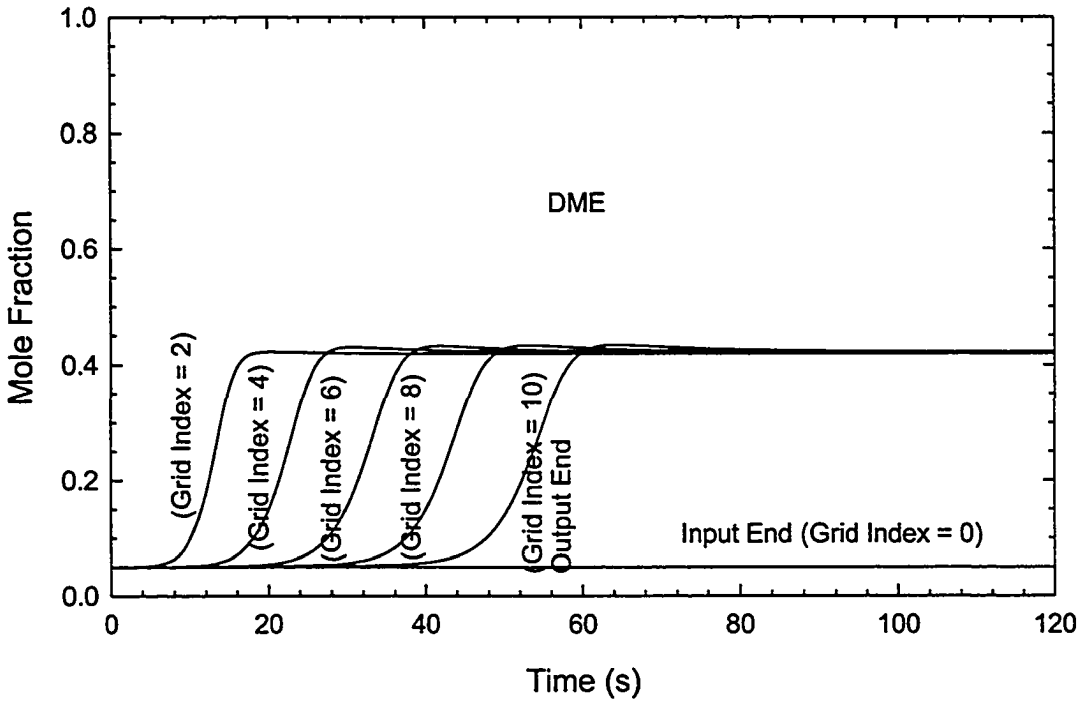



Figure 6-57 Dynamic Response of DME Composition

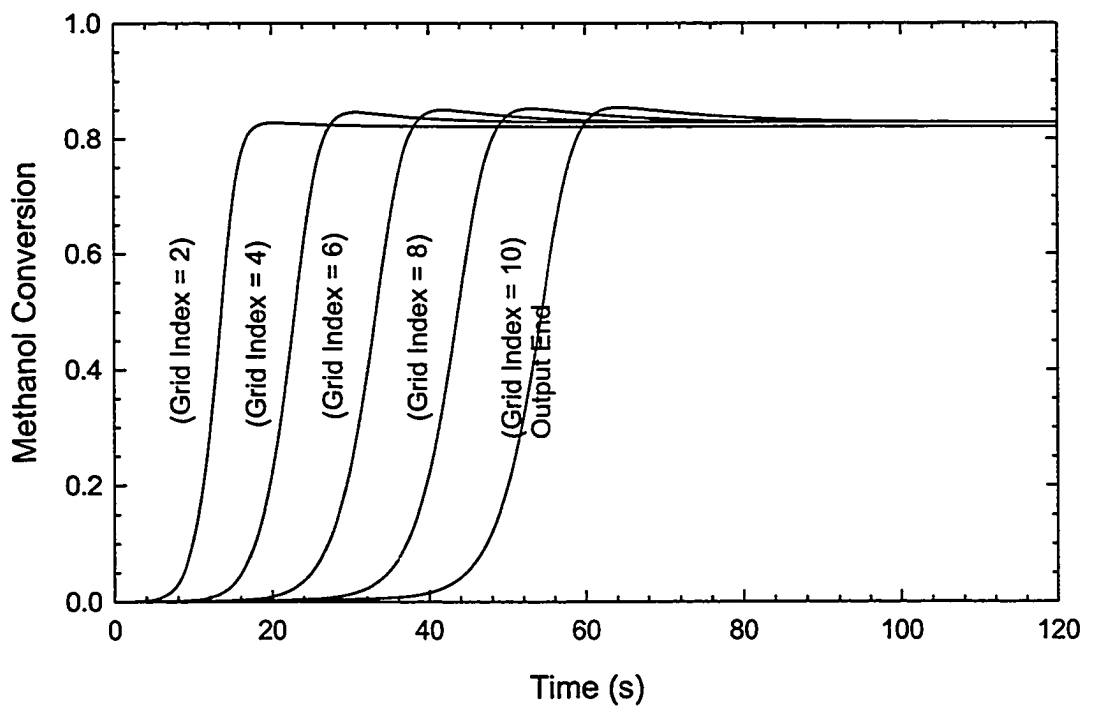



Figure 6-58 Dynamic Response of Methanol Conversion

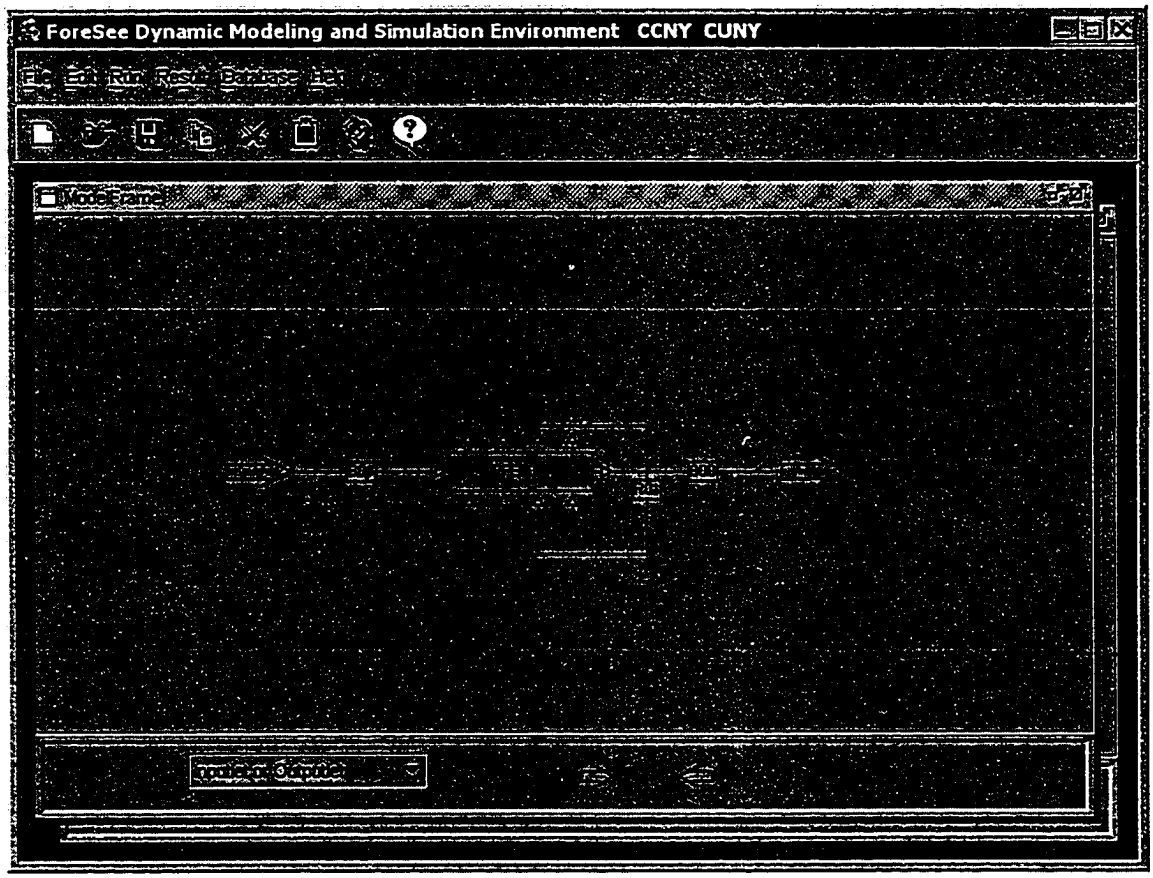



Figure 6-59 Modeling Decomposition of Tubular Flow Reactor

## 6.3.2 Bubble Bed Reactor

Figure 6-60 schematically represents basic two-phase model. A fraction of the total flow rate through the bed is considered to be in the bubble phase, the rest in the emulsion phase. Because of coalescence or bubble growth some interchange of gas was postulated between the bubble phase and the dense phase surrounding it, which is also called *emulsion phase*. Between the bubble phase and the emulsion phase there is a certain interchange of gas, or cross flow. At the outlet, both streams, with their respective conversations, are hypothetically mixed to give the exit stream, with its mean conversion or concentration. Since there is no reaction in the bubble phase and because of

its high velocity, the flow through that phase is usually taken to be of the plug flow type. In the emulsion phase either various degrees of deviation from plug flow or complete mixing can be postulated [Froment & Bischoff 1990].

The decomposition of bubble bed reactor model is shown as follows:

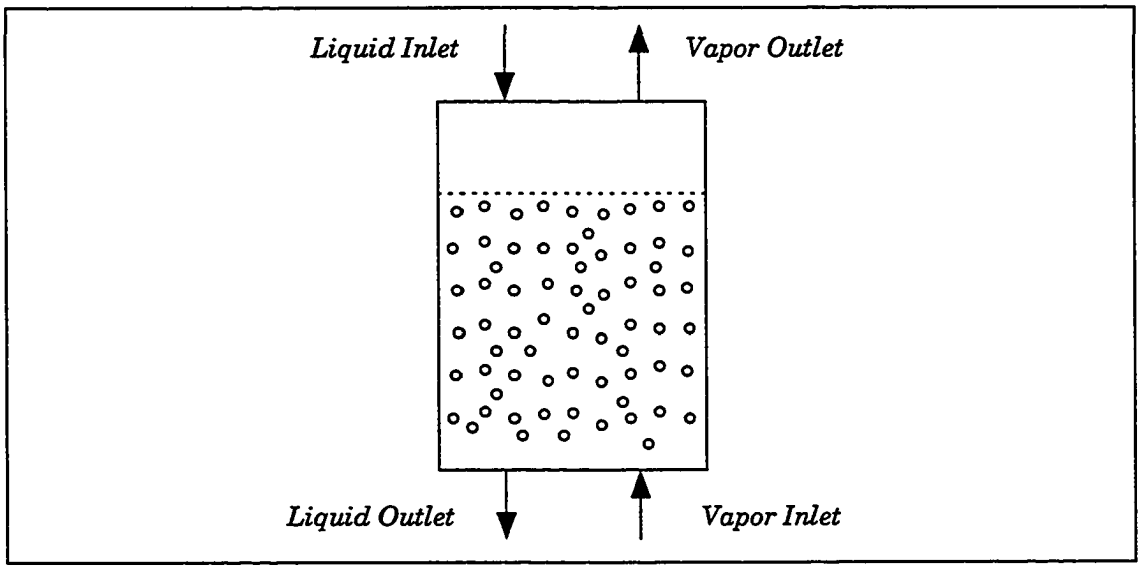



Figure 6-60 Bubble Bed Reactor

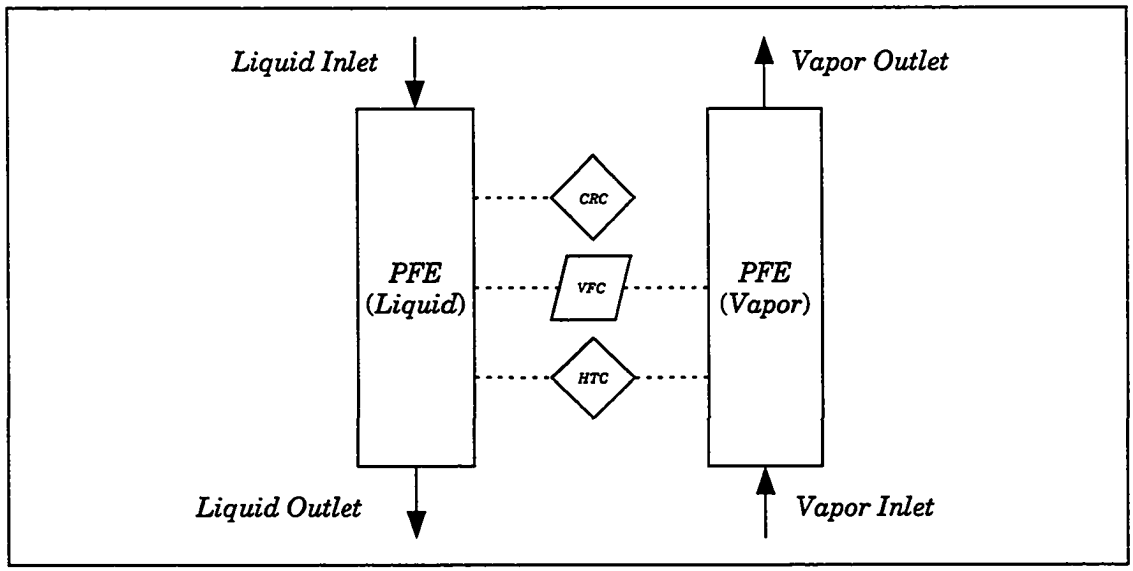



Figure 6-61 Decomposition of Bubble Bed Reactor

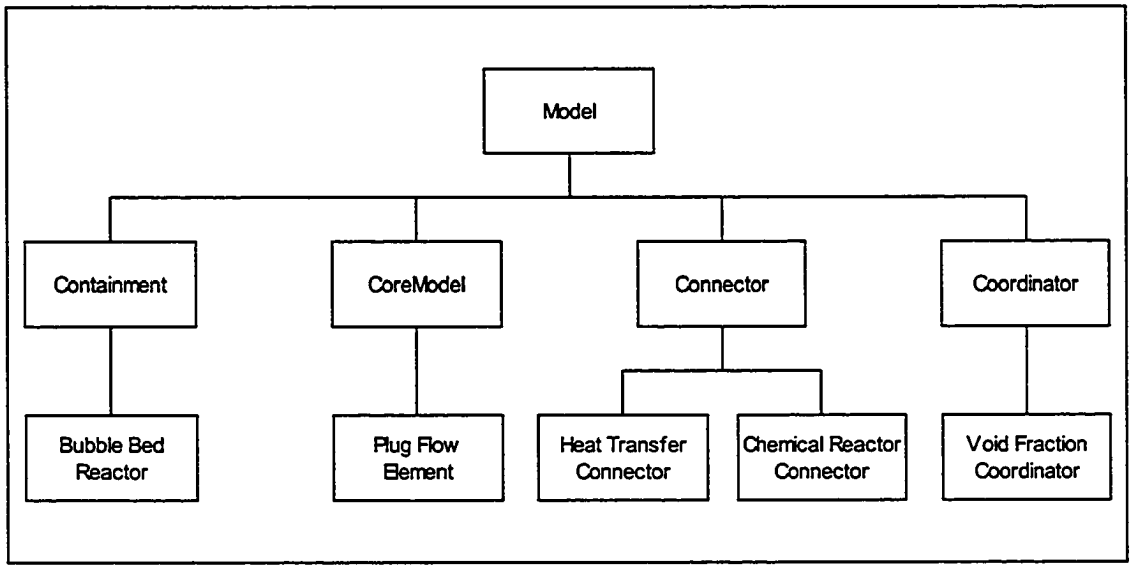



Figure 6-62 Decomposition of Bubble Bed Reactor Model

## 6.3.3 Fixed Bed Reactor

The decomposition of fixed bed reactor model is shown as follows:

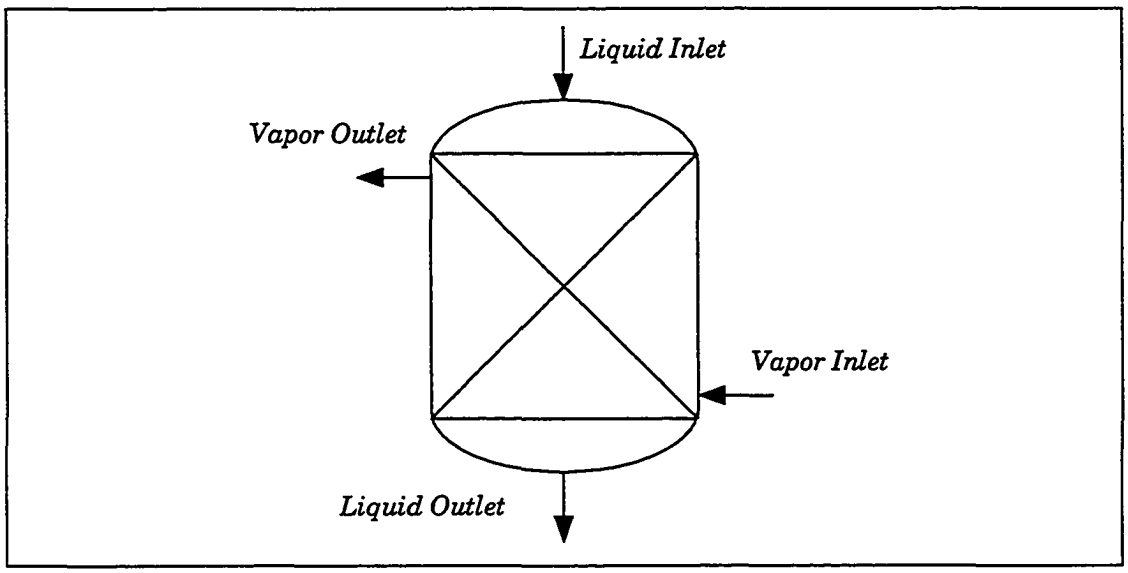



Figure 6-63 Fixed Bed Reactor

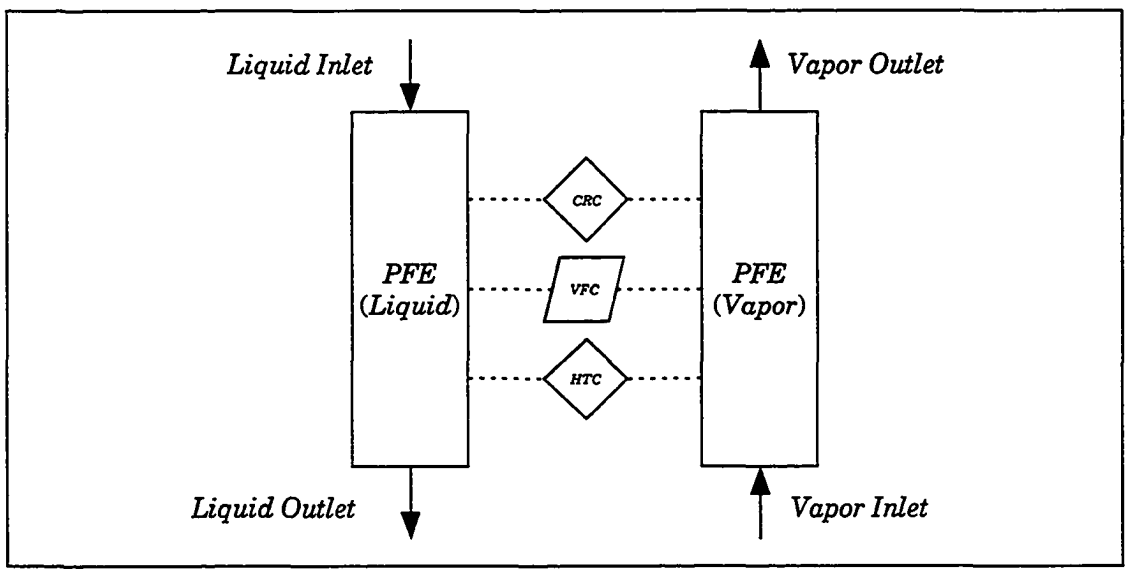



Figure 6-64 Modeling Decomposition of Fixed Bed Reactor

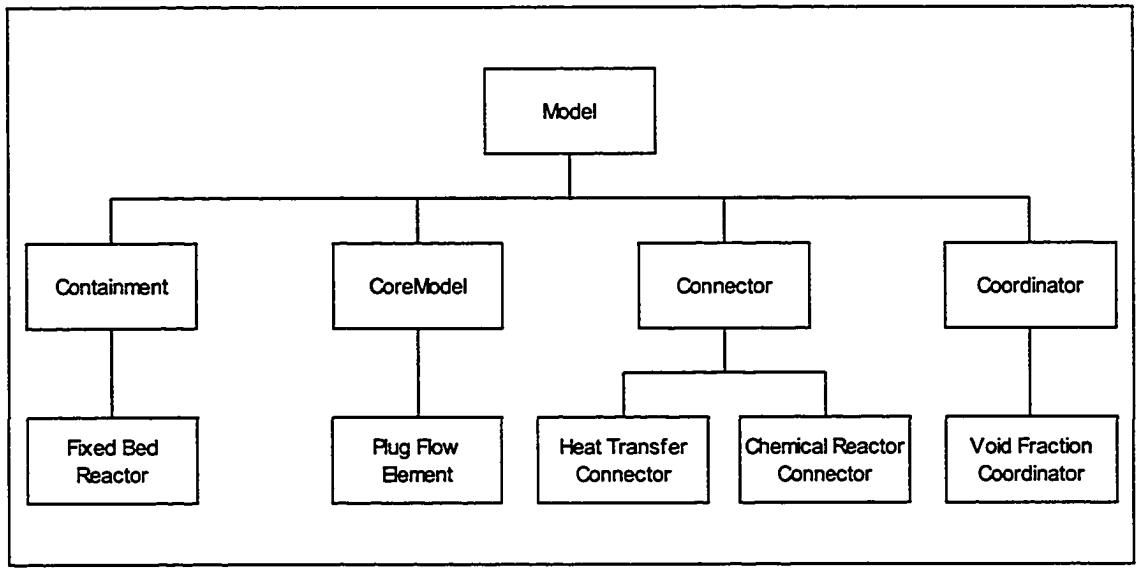



Figure 6-65 Decomposition of Fixed Bed Reactor Model

### 6.3.4 Spray Tower Extractor

Spray tower extractors give differential contacts, not stage contacts, and mixing and settling proceed simultaneously and continuously. In the spray tower shown in Figure 6-66, the lighter liquid is introduced at the bottom and distributed as small drops by the nozzles. The drops are collected at the top and form the stream of light liquid leaving the top of the tower. The heavy liquid flows downward as a continuous stream and leaves the bottom of the tower.

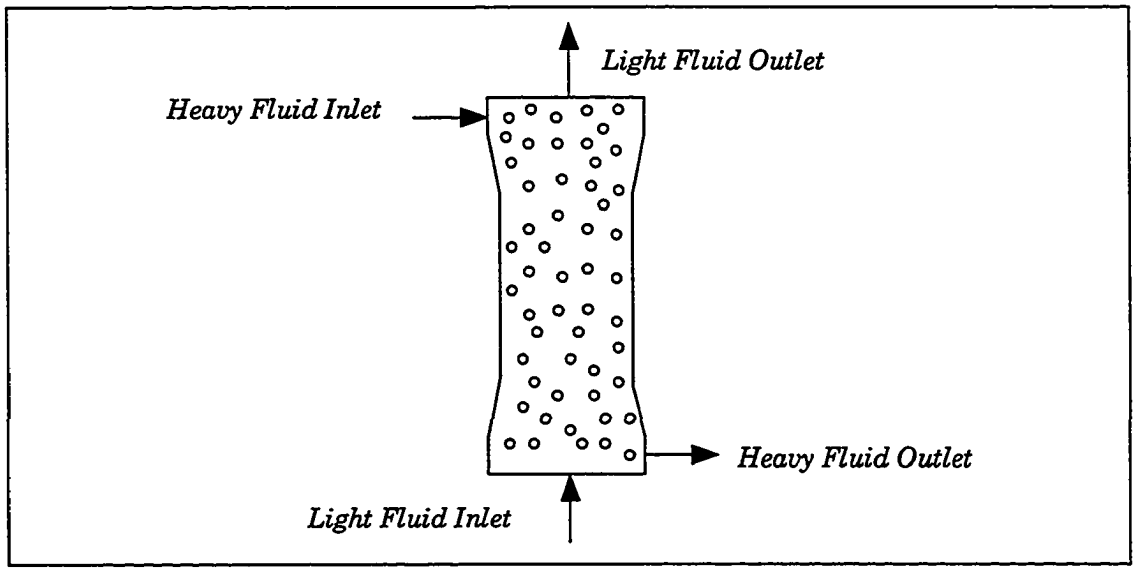



Figure 6-66 Spray Tower Extractor

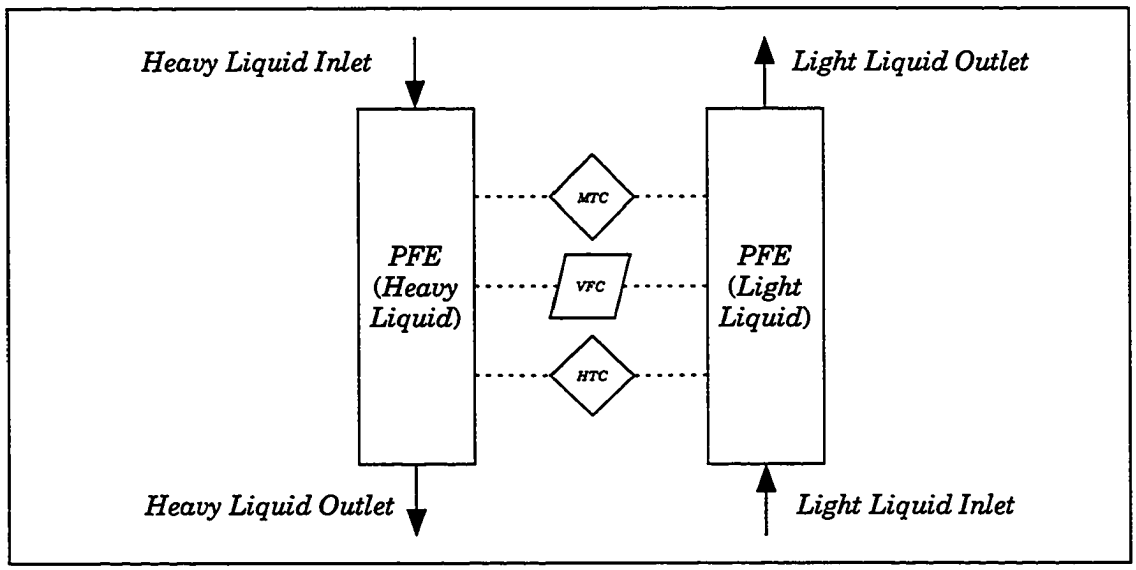



Figure 6-67 Modeling Decomposition of Spray Tower Extractor

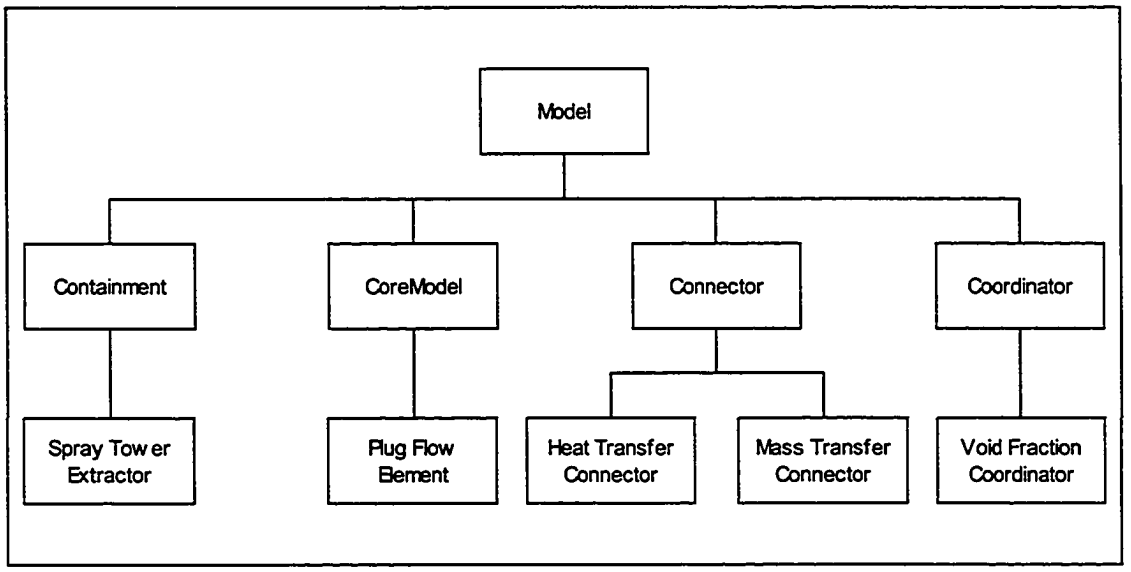



Figure 6-68 Decomposition of Spray Tower Extractor Model

## 6.4 Dimethyl Ether Process

Figure 6-69 is a preliminary process flow diagram for the dimethyl ether (DME) production process. The raw material is methanol. The feed plus recycle is pumped in P-201, heated and vaporized in a heat exchanger (E-201), and superheated in a heat exchanger (E-202), and then sent to the reactor (R-201) in which dimethyl ether is formed. The reactor effluent is cooled in E-202 and condensed in a heat exchanger (E-203), and it is then sent to the separation section. In a distillation column (T-201), DME is produced in the top stream, with methanol and water in the bottom stream. In a distillation column (T-202), the top stream contains methanol for recycle, and the bottom stream contains waste water.

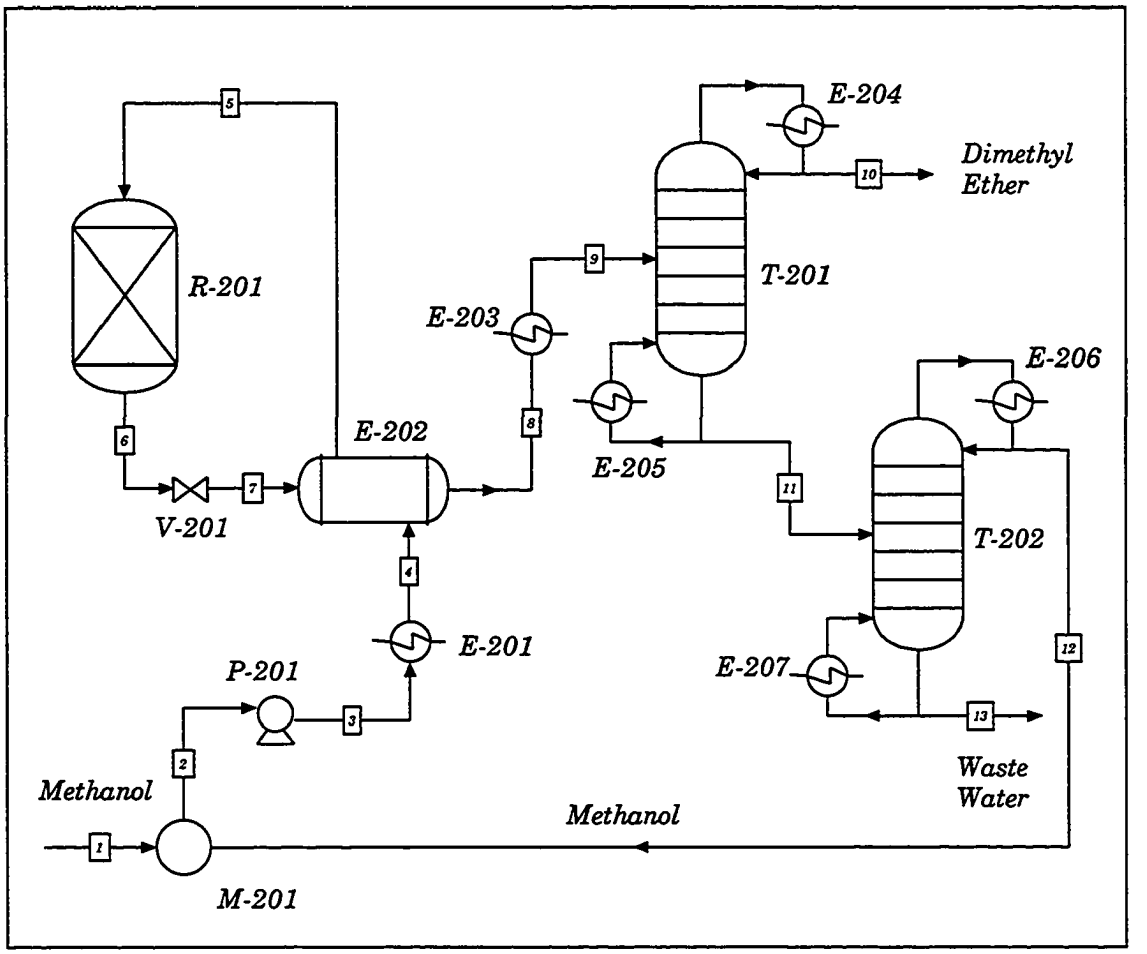



Figure 6-69 Dimethyl Ether Process

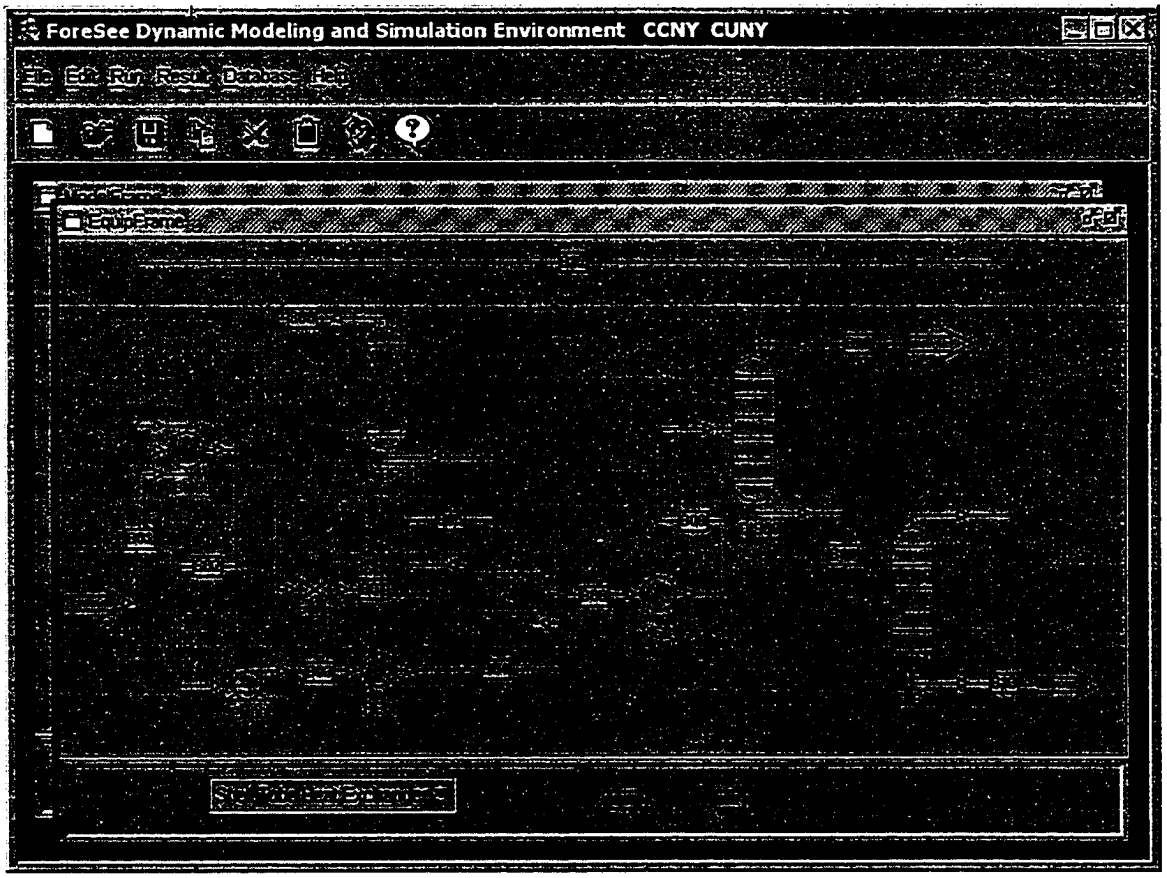



Figure 6-70 Dimethyl Ether Process in ForeSee

The DME process is a network that comprises a set of equipment items and a set of spool pieces. The equipment items in this example are *Fixed Bed Reactor*, *Distillation Columns*, *Shell-Tube Heat Exchanger*, *Reboiler* and *Condenser*, in which modeling components have been configured in the proceeding sections. The spool pieces in this example are pump and valve. The pump increase the pressure of the feed plus recycle to a minimum of 15 bar. The valve is used to accomplish pressure reduction. Therefore, based on the aggregation of all modeling components, we can accomplish the dynamic simulation of a complete DME process. The modeling decomposition of DME process is shown in Figure 6-71.

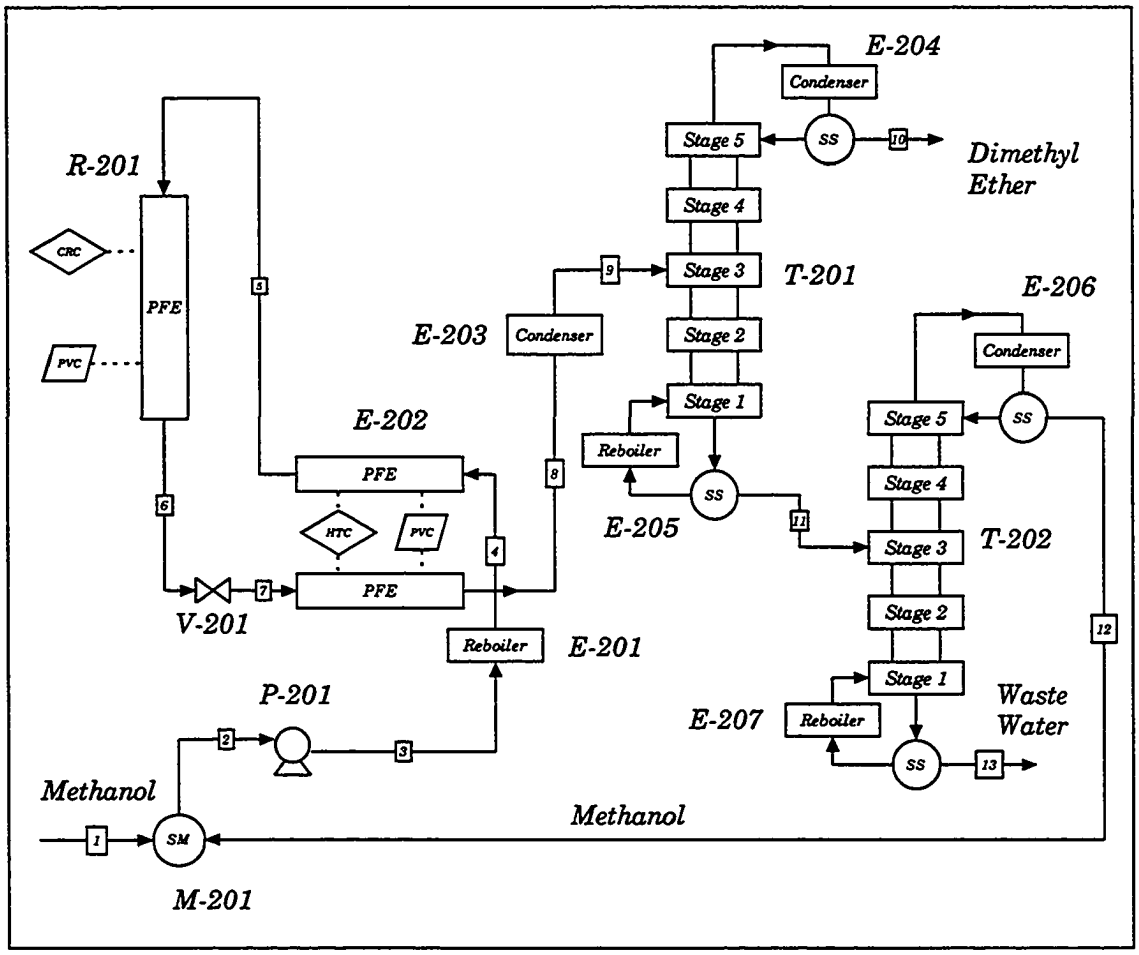



Figure 6-71 Modeling Decomposition of Dimethyl Ether Process

## 6.5 Conclusions

- A relatively small set of modeling components can be used to generate a variety of equipment models.
- A lot of complex model of chemical process equipment can be visually configured just by selection, parameterization and aggregation of modeling objects taken from a component model library.

# Chapter 7 Conclusions and Recommendations

A number of conclusions can be drawn from this work that should be useful to process engineers responsible for the dynamic modeling and simulation of process systems.

- A ForeSee prototype has been developed which demonstrates some of the key aspects of four sets of modeling components to represent the dynamic modeling and simulation of chemical processes. It is based on an object-oriented design of the software and programmed in Java. This prototype supports the graphical drag-and-drop interface for creating both equipment models and assembling them into a flowsheet.
- The heart of any dynamic flowsheet simulator is the unit operation model library. ForeSee equipment models are built on the well-establishing 4C modeling components (*containments, core models, connectors,* and *coordinators*). The user can construct new equipment model by the selection, aggregation, and parameterization of 4C modeling components graphically (no need for programming).
- ForeSee is open and customizable. There are no black box models in ForeSee. The user has full access to all models and can customize models if necessary to fully represent the details of the process.
- ForeSee uses DIPPR (Design Institute for Physical Properties) data for rigorous and reliable physical property calculations. This provides a totally consistent basis for steady-state and dynamic simulations.

- ForeSee benefits from the proven implicit integration and numerical solution methods for robust, stable, and accurate dynamic flowsheet simulation.

It is in the representation of equipment models that ForeSee differs most strongly from all the other simulation systems. In these systems an equipment model consists of a set equations and the associated code for their execution. In ForeSee an equipment model is specified as an interconnected set of 4C modeling components.

The basic premise in this approach to modeling is that a relatively small set of 4C modeling components can be used to generate a variety of equipment models, and in turn a lot of higher level models of complex processes can be assembled from the equipment models. This approach has several advantages:

- If the libraries of 4C modeling components can be kept reasonably compact, then creation and maintenance of a dynamic modeling and simulation environment is greatly simplified. The amount of code that must be written, debugged, maintained, and documented is kept to a bare minimum. All of the model codes resides in the set of 4C modeling components. All model development from derivation of equations to coding and debugging has already been done if the appropriate 4C modeling components are available. Equipment models do not exist as codes but as definition files indicating how the model is constructed using the 4C modeling components.
- If the appropriate 4C modeling components are available, creating new equipment models is straightforward and relatively quick. This will greatly reduce the lead-time required to set up a process model that requires new equipment models. This approach is well adapted

to the modeling and simulation of multiphase systems.

- The compactness of the libraries of 4C modeling components should simplify the understanding of the entire simulation system. Equipment models will no longer be black boxes to the user, but will be fully knowable from the descriptions of 4C modeling components and the equipment model configuration diagrams. The equipment model is represented graphically to the user rather as a set of equations. All that the user has to do is to configure a new equipment model by the selection, parameterization and aggregation of the four sets of component models using a graphical drag-and-drop interface.

There are a number of aspects of dynamic modeling and simulation that need to be explored in depth to demonstrate the feasibility of the hierarchical modeling approach to the dynamic simulation of complex systems.

- The number of 4C modeling components that are required to generate a library of equipment models of sufficient richness to permit the dynamic modeling and simulation of a range of typical chemical processes should be further extended in the future.
- The corresponding modeling components for spool piece system should be further developed.
- The corresponding modeling components for process control system should be further developed.
- The physical property and database system should be further improved in the future.
- Model Complexity (e.g., DAEs index, stiffness ratio, and discontinuities) in the dynamic modeling and simulation should be

further evaluated in the future.

- As both the computer power and the computational techniques improve, the model formulation of the modeling components based on computational fluid dynamics may become practical for chemical process dynamic simulation.

# Appendix A Numerical Solution of the Simulator

## A.1 Introduction

Throughout most of its history, dynamic simulation has been limited to the simplest possible models. This limitation was imposed by the finite hardware capabilities of the analog computers used in the 1950s and 1960s and by the lack of memory and speed of the digital computers of the 1970s and 1980s. The situation is changing rapidly. Great progress has been made during the last decade in the dynamic simulation of chemical processes. The need for tighter and more sophisticated process control and the growing dependence upon simulation for safety system validation and operator training and certification has led to demands for high fidelity models and replica simulators.

## A.2 Measures of Model Complexity

Several measures can be used to characterize model complexity. These can be used to make a choice of the appropriate model to use in a given modeling and simulation.

### A.2.1 Classification of Model Equations

The simplest and most straightforward measure of model complexity is merely the types of equations required to describe the model. In general

chemical engineering model equations can be classified in four categories: algebraic, ordinary differential, differential algebraic and partial differential. A general classification for chemical engineering model equations is shown in Table A-1 [Seinfeld & Lapidus 1974; Franks 1972; Friedly 1972].

Table A-1 Classification for Chemical Engineering Model Equations

| Model Type | Characteristics |
|---|---|
| Algebraic Equations (AEs) | Linear<br>Nonlinear |
| Ordinary Differential Equations (ODEs) | Linear<br>Nonlinear<br>Stiff<br>Nonstiff |
| Differential Algebraic Equations (DAEs) | Index = 0<br>Index = 1<br>Index > 1 |
| Partial Differential Equations (PDEs) | Parabolic<br>Hyperbolic<br>Elliptical |

## A.2.2 DAEs Index

An adequate representation of the transient behavior of a complex chemical process usually leads to a large number of differential-algebraic equations (DAEs). A system of initial value differential-algebraic equations can be written in its general implicit form as in Equation (A-1), subject to initial conditions in Equation (A-2), where $t \in \Re, \mathbf{y} \in \Re^n$ and $\mathbf{F} \in \Re^n$ is a sufficiently differential function [Lefkopoulos & Stadtherr 1993a, 1993b].

$$\mathbf{F}(t, \mathbf{y}(t), \mathbf{y}'(t)) = 0 \tag{A-1}$$

$$\mathbf{y}(t_0) = \mathbf{y}_0 \tag{A-2}$$

The existence of algebraic constraints on the variables is expressed by the singularity of the Jacobian matrix $\partial \mathbf{F} / \partial \mathbf{y}'$. Such sets of differential and algebraic equations are frequently restated and solved as ordinary differential equations (ODEs) by differentiation and/or extensive algebraic manipulation. The minimum number of times this differentiating must be performed to completely reduce the DAEs to ODEs is referred to as the index of DAEs.

### A.2.3 Stiffness Ratio

Stiffness is a major problem in dynamic simulation, particularly when gas phase flow rate transients must be included in the model. Stiffness is defined as the ratio of the absolute value of the real part of the largest eigenvalue of the model to the absolute value of the real part of the smallest. The larger this ratio, the stiffer the system. The reciprocal of the largest eigenvalue determines the maximum step size that can be employed by the explicit ODEs solver while the smallest eigenvalue governs how long a perturbed system will take to come to a new steady-state. Thus, the higher the stiffness ratio, the longer the computer time required to simulate a given amount of process time.

### A.2.4 Real Time Factor

A measure of model complexity that is related to stiffness is the real time factor (RTF). This is defined as the ratio of process time interval being stimulated to the amount of computer time required to carry out the simulation. Obviously, the RTF will depend upon the computing speed of the

computer being used. The faster the computer speed, the higher the RTF. RTF is an important measure for both engineering simulators and training simulators. For the former, a high RFT is required so that the engineering can evaluate a reasonable number of case studies within a reasonable period of time. For the latter, the RTF does not have to be large but still must be greater than 1.0 if the simulator is to meet the requirement of responding in real time [Huang 1996].

### A.2.5 Discontinuities

Most ODEs solvers require that the right-hand sides and their first derivatives be continuous for the methods to work properly. These have mostly been designed to solve initial value problems. Most process simulations are subject to various discontinuities. Some are introduced externally through the actions of the function generator; others are internal and result from model branching. For a boiler in heat up mode, the temperature increases and the vapor production rate is zero. When the boiling point is reached, the temperature time derivative goes to zero and the vapor production rate is positive. This model branching discontinuity will cause all but the most robust ODEs solvers to fail [Preston & Berzins 1991].

### A.2.6 Classification of Simulation Strategies

Basically two different classes of simulation strategies are distinguished:

- the simultaneous solution of all plant equations with one single numerical algorithm which we will call *direct integration*;

- the solution of a suitably partitioned system by means of different numerical algorithms applied to each subsystem, which will be referred to as *modular integration.*

The simultaneous solution of the plant equations with one single algorithm requires one homogeneous set of equations for which automatic integration methods are available. The simultaneous solution of distributed and lumped parameter models is obtained by discretization of the spatial independent variables known as the *method of lines* [Schiesser 1991]. In this method, the spatial derivatives are approximated by finite differences on a grid of spatial coordinates.

## A.3 Integration of Ordinary Differential Equations

Problems involving ordinary differential equations (ODEs) can always be reduced to the study of sets of first-order differential equations. For example, the third-order equation [Gear 1971]

$$\frac{d^3y}{dt^3} + p(t)\frac{d^2y}{dt^2} + q(t)\frac{dy}{dt} = r(t) \tag{A-3}$$

can be rewritten as three first-order equations

$$\frac{dy}{dt} = x(t) \tag{A-4}$$

$$\frac{dx}{dt} = z(t) \tag{A-5}$$

$$\frac{dz}{dt} = r(t) - q(t)x(t) - p(t)z(t) \tag{A-6}$$

where x and z are new variables. This exemplifies the procedure for an

arbitrary ODE. The usual choice for the new variables is to let them be just derivatives of each other.

The generic problem in ordinary differential equations is thus reduced to the study of a set of $N$ coupled first-order differential equations for the functions $y_i$, having the general form

$$\frac{dy_i(t)}{dt} = f_i(t, y_1, y_2, \cdots, y_N) \qquad i = 1,2,\cdots,N \tag{A-7}$$

where the functions $f_i$ on the right-hand side are known.

A problem involving ODEs is not completely specified by its equations. Even more crucial in determining how to attack the problem numerically is the nature of the problem's boundary conditions. Boundary conditions are algebraic conditions on the values of the functions $y_i$ in Equation (A-7).

Usually, it is the nature of the boundary conditions that determines which numerical methods will be feasible. Boundary conditions divide into two broad categories.

- *Initial value problems*, in which all the $y_i$ are given at some value $t_0$, and it is desired to find the $y_i$ at some final point $t_f$, or at some discrete list of points (for example, at tabulated intervals).
- *Two-point boundary value problems*, in which , on the other hands, boundary conditions are specified at more than one $t$. Typically, some of the conditions will be specified at $t_0$ and the remainder at $t_f$.

We will consider exclusively the initial value problem in this thesis. The underlying idea of any routine for solving the initial value problem is always this: rewrite the $dy$ and $dt$ in Equation (A-7) as finite steps $\Delta y$ and $\Delta t$, and multiply the equations by $\Delta t$. This gives algebraic formulas for the

change in the functions when the independent variable $t$ is stepped by one step-size $\Delta t$. In the limit of making the step-size very small, a good approximation to the underlying differential equation is achieved.

### A.3.1 Runge-Kutta Method

Many methods are available for performing numerical integration of differential equations. Experience with a wide range of typical chemical engineering problems has shown that, with almost no exceptions, Runge-Kutta method is quite adequate for any situation. By far the most often used is the classical *fourth-order Runge-Kutta formula* [Press et al. 1992]:

$$k_1 = hf(t_n, y_n) \tag{A-8}$$

$$k_2 = hf(t_n + \frac{h}{2}, y_n + \frac{k_1}{2}) \tag{A-9}$$

$$k_3 = hf(t_n + \frac{h}{2}, y_n + \frac{k_2}{2}) \tag{A-10}$$

$$k_4 = hf(t_n + h, y_n + k_3) \tag{A-11}$$

$$y_{n+1} = y_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5) \tag{A-12}$$

A good ODEs integrator should exert some adaptive control over its own progress, making frequent changes in its step-size. Usually the purpose of this adaptive step-size control is to achieve some predetermined accuracy in the solution with minimum computational effort. With fourth-order Runge-Kutta, the most straightforward technique by far is *step doubling*. We take each step twice, once as a full step, then, independently, as two half steps.

The general form of a *fifth-order Runge-Kutta formula* is

$$k_1 = hf(t_n, y_n) \tag{A-13}$$

$$k_2 = hf(t_n + a_2h, y_n + b_{21}k_1) \tag{A-14}$$

$$k_3 = hf(t_n + a_3h, y_n + b_{31}k_1 + b_{32}k_2) \tag{A-15}$$

$$k_4 = hf(t_n + a_4h, y_n + b_{41}k_1 + b_{42}k_2 + b_{43}k_3) \tag{A-16}$$

$$k_5 = hf(t_n + a_5h, y_n + b_{51}k_1 + b_{52}k_2 + b_{53}k_3 + b_{54}k_4) \tag{A-17}$$

$$k_6 = hf(t_n + a_6h, y_n + b_{61}k_1 + b_{62}k_2 + b_{63}k_3 + b_{64}k_4 + b_{65}k_5) \tag{A-18}$$

$$y_{n+1} = y_n + c_1k_1 + c_2k_2 + c_3k_3 + c_4k_4 + c_5k_5 + c_6k_6 + O(h^6) \tag{A-19}$$

The embedded fourth-order formula is

$$y^*_{n+1} = y_n + c^*_1k_1 + c^*_2k_2 + c^*_3k_3 + c^*_4k_4 + c^*_5k_5 + c^*_6k_6 + O(h^5) \tag{A-20}$$

and so the error estimate is

$$\Delta \equiv y_{n+1} - y^*_{n+1} = \sum_{i=1}^{6}\left(c_i - c^*_i\right)k_i \tag{A-21}$$

The particular values of the various constants are given in the following table [Press et al. 1992].

Table A-2 The Parameters for Embedded Runge-Kutta Method

| $i$ | $a_i$ | $b_{ij}$ | | | | | $c_i$ | $c^*_i$ |
|---|---|---|---|---|---|---|---|---|
| 1 | | | | | | | $^{37}/_{378}$ | $^{2825}/_{27648}$ |
| 2 | $^1/_5$ | $^1/_5$ | | | | | 0 | 0 |
| 3 | $^3/_{10}$ | $^3/_{40}$ | $^9/_{40}$ | | | | $^{250}/_{621}$ | $^{18575}/_{48384}$ |
| 4 | $^3/_5$ | $^3/_{10}$ | $-^9/_{10}$ | $^6/_5$ | | | $^{125}/_{594}$ | $^{13525}/_{55296}$ |
| 5 | 1 | $-^{11}/_{54}$ | $^5/_2$ | $-^{70}/_{27}$ | $^{35}/_{27}$ | | 0 | $^{277}/_{14336}$ |
| 6 | $^7/_8$ | $^{1631}/_{55296}$ | $^{175}/_{512}$ | $^{575}/_{13824}$ | $^{44275}/_{110592}$ | $^{253}/_{4096}$ | $^{512}/_{1771}$ | $^1/_4$ |
| $j$ | | 1 | 2 | 3 | 4 | 5 | | |

## A.3.2 Bulirsch-Stoer Method

There are three key ideas in the *Bulirsch-Stoer* method [Press et al. 1992]. The first idea is to consider the final answer of a numerical calculation as itself being an analytic function of an adjustable parameter like the step-size $h$. That analytic function can be probed by performing the calculation with various values of $h$, none of them being necessarily small enough to yield the accuracy that we desire. The second idea has to do with what kind of fitting function is used. In general, the polynomial or rational function extrapolation is used. The third idea is to use a method whose error function is strictly even, allowing the rational function or polynomial approximation to be in terms of the variable $h^2$ instead of just $h$.

A single Bulirsch-Stoer step takes us from $x$ to $x+H$, where $H$ is supposed to be quite a large distance. That single step is a grand leap consisting of many substeps of modified midpoint method, which are then extrapolated to zero step size.

The sequence of separate attempts to cross the interval $H$ is made with increasing values of $n$, the number of substeps. The sequence is:

$$n = 2,4,6,8,10,12,14,\cdots \tag{A-22}$$

For each step, we do not know in advance how far up this sequence will go. After each successive $n$ is tried, a polynomial or rational function extrapolation is attempted. That extrapolation gives both extrapolated values and error estimates. If the errors are not satisfactory, we go higher in $n$. If they are satisfactory, we go on to the next step and begin anew with $n = 2$. In general, the Bulirsch-Stoer method is the best-known way to obtain high-accuracy solutions to ordinary differential equations with minimal

computational effort.

### A.3.3 Bader-Deufhard Method

The Bulirsch-Stoer method, which discretizes the differential equation using the modified midpoint rule, does not work for stiff problems. Stiff systems are sets of differential equations that contain a mixture of very fast dynamic equations and very slow dynamic equations. Single-step and multiple-step algorithms are known to have trouble with this type of equation system. To obtain completely accurate answers, the integration step-size must be very small and this usually means prohibitively long computation times in order to observe the slow dynamic response. On the other hand, if the integration step-size is too big, the resulting inaccuracies in the fast dynamic parts can lead to general stability problems and inaccurate result.

For the system

$$\mathbf{y}' = \boldsymbol{f}(\mathbf{y}) \tag{A-23}$$

implicit differencing gives

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\boldsymbol{f}(\mathbf{y}_{n+1}) \tag{A-24}$$

In general, this is some nasty set of nonlinear equations that has to be solved iteratively at each step. Suppose we try linearizing the equations, as in Newton's method:

$$\boldsymbol{f}(\mathbf{y}_{n+1}) = \boldsymbol{f}(\mathbf{y}_n) + \left(\frac{\partial \boldsymbol{f}}{\partial \mathbf{y}}\right)_n (\mathbf{y}_{n+1} - \mathbf{y}_n) \tag{A-25}$$

Here $\partial \boldsymbol{f}/\partial \mathbf{y}$ is the Jacobian matrix of the partial derivatives of the right-

hand side. Rearranging Equation (A-24) we get the *semi-implicit Euler method.*

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\left[\mathbf{1} - h\frac{\partial f}{\partial \mathbf{y}}\right]^{-1} f(\mathbf{y}_n) \tag{A-26}$$

For the Bader-Deuflhard method [Press et al. 1992], the starting point is an implicit form of the midpoint rule:

$$\mathbf{y}_{n+1} - \mathbf{y}_{n-1} = 2hf(\frac{\mathbf{y}_{n+1} + \mathbf{y}_{n-1}}{2}) \tag{A-27}$$

Linealizing the right-hand side of Equation (A-23) about $\mathbf{f}(\mathbf{y}_n)$

$$f(\frac{\mathbf{y}_{n+1} + \mathbf{y}_{n-1}}{2}) = f(\mathbf{y}_n) + \left(\frac{\partial f}{\partial \mathbf{y}}\right)_n \left(\frac{\mathbf{y}_{n+1} + \mathbf{y}_{n-1}}{2} - \mathbf{y}_n\right) \tag{A-28}$$

and rearranging Equation (A-27) we get the *semi-implicit midpoint method.*

$$\left(\mathbf{1} - h\frac{\partial f}{\partial \mathbf{y}}\right)\mathbf{y}_{n+1} = \left(\mathbf{1} + h\frac{\partial f}{\partial \mathbf{y}}\right)\mathbf{y}_{n-1} + 2h\left(f(\mathbf{y}_n) - \frac{\partial f}{\partial \mathbf{y}}\mathbf{y}_n\right) \tag{A-29}$$

For the practical implementation, it is better to rewrite the equations using $\Delta_k \equiv \mathbf{y}_{k+1} - \mathbf{y}_k$. With $h = H/m$, start with calculating a special first step (the semi-implicit Euler step)

$$\Delta_0 = \left[\mathbf{1} - h\frac{\partial f}{\partial \mathbf{y}}\right]^{-1} hf(\mathbf{y}_0) \tag{A-30}$$

$$\mathbf{y}_1 = \mathbf{y}_0 + \Delta_0 \tag{A-31}$$

Then for $k = 1,\cdots,m-1$, set

$$\Delta_k = \Delta_{k-1} + 2\left[\mathbf{1} - h\frac{\partial f}{\partial \mathbf{y}}\right]^{-1} [hf(\mathbf{y}_k) - \Delta_{k-1}] \tag{A-32}$$

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \Delta_k \tag{A-33}$$

Finally compute a special last step

$$\Delta_m = \left[\mathbf{1} - h\frac{\partial f}{\partial \mathbf{y}}\right]^{-1} [hf(\mathbf{y}_m) - \Delta_{m-1}] \tag{A-34}$$

$$\bar{\mathbf{y}}_m \equiv \tfrac{1}{2}(\mathbf{y}_{m+1} + \mathbf{y}_{m-1}) = \mathbf{y}_m + \Delta_m \tag{A-35}$$

The error series for the Bader-Deuflhard method once again involves only even powers of $h$. This method works very well for almost all stiff problems and lends itself to extrapolation exactly as in the Bulirsch-Stoer method.

## A.4 Solution of Nonlinear Algebraic Equations

### A.4.1 Newton-Raphson Method

Whereas the procedures for solving systems of linear equations are straightforward, those for solving sets of nonlinear equations are not nearly so well formulated. Newton-Raphson is a general method used for the solution of $n$ simultaneous nonlinear equations in $n$ unknown variables. This technique tries to improve an initial guess for the solution vector via a linearization procedure.

A set of $N$ nonlinear equations in $N$ unknowns may be written:

$$f_i(x_1, x_2, \cdots, x_N) = 0 \qquad i = 1,2,\cdots,N \tag{A-36}$$

or in vector notation,

$$f(\mathbf{x}) = \mathbf{0} \tag{A-37}$$

If there exists a present set of guesses, $\mathbf{x}^j$, for the solution of Equation (A-37), the function $\boldsymbol{f}$ may be rewritten for any other $\mathbf{x}$ as a Taylor series expansion about $\mathbf{x}^j$ as follows:

$$\boldsymbol{f}(\mathbf{x}) = \boldsymbol{f}(\mathbf{x}^j) + \frac{\partial \boldsymbol{f}(\mathbf{x}^j)}{\partial \mathbf{x}}(\mathbf{x} - \mathbf{x}^j) + O\left((\mathbf{x} - \mathbf{x}^j)^2\right) \tag{A-38}$$

The partial derivative in Equation (A-38) is called the Jacobian matrix. Although the Jacobian can be evaluated analytically, it is usually estimated numerically. Ideally, the next set of guesses, $\mathbf{x}^{j+1}$, would be the solution to Equation (A-37). If this were the case, truncating the nonlinear higher-order terms of Equation (A-38) would become [Ramirez 1989]

$$\boldsymbol{f}(\mathbf{x}^{j+1}) = \boldsymbol{f}(\mathbf{x}^j) + \frac{\partial \boldsymbol{f}(\mathbf{x}^j)}{\partial \mathbf{x}}(\mathbf{x}^{j+1} - \mathbf{x}^j) = \mathbf{0} \tag{A-39}$$

Rearranging Equation (A-39) gives

$$\mathbf{x}^{j+1} = \mathbf{x}^j + \left[\frac{\partial \boldsymbol{f}(\mathbf{x}^j)}{\partial \mathbf{x}}\right]^{-1} \boldsymbol{f}(\mathbf{x}^j) \tag{A-40}$$

Because of the linear approximation of Equation (A-38), $\mathbf{x}^{j+1}$ will not be the exact solution to Equation (A-37). However, Equation (A-40) may be used iteratively to coverage on a solution to Equation (A-37).

Systems of nonlinear equations may have many solutions. Depending on the initial guess $\mathbf{x}^0$, the Newton-Raphson method may converge to different solutions. In that case, it is wise to make the best initial guesses possible and use physical reasoning in interpreting the solution. Also, the Jacobian matrix may become singular as the solution is approached. If this occurs, solution by the Newton-Raphson technique may be impossible, and other nonderivative methods should be used.

## A.5 Implementation of ForeSee Prototype

We have organized the routines into three nested levels. The lowest level is the piece we call the *ODEs Algorithm routine*. This implements the basic formulas of the method, starts with dependent variables $y_i$ at $t$, and calculates new values of the dependent variables at the value $t + h$. The algorithm routine also yields up some information about the quality of the solution after the step. The routine is dumb, however, and it is unable to make any adaptive decision about whether the solution is of acceptable or not.

That quality-control decision is the piece we call the *ODEs Stepper routine*. The stepper routine calls the algorithm routine. It may reject the result, set a smaller step-size, and call the algorithm routine again, until compatibility with a predetermined accuracy criterion has been achieved. The stepper routine's fundamental task is to take the largest step size consistent with specified performance. Only when this is accomplished does the true power of an algorithm come to light.

Above the stepper routine is the *ODEs Driver routine*, which starts and stops the integration, stores intermediate results, and generally acts as an interface with the user.

We summarize the above descriptions as follows:

- Runge-Kuta Method

  Algorithm: the fifth-order Runge-Kutta method

  Stepper: the fifth-order Runge-Kutta step

  Driver: driver with adaptive step-size control

- Bulirsch-Stoer Method

  Algorithm: modified midpoint method

Stepper: Bulirsch-Stoer step

Driver: driver with adaptive step-size control

- Bader-Deufhard Method

Algorithm: semi-implicit extrapolation method

Stepper: semi-implicit extrapolation

Driver: driver with adaptive step-size control

## A.6 Conclusions

- In general, for applications not demanding high precision, and where convenience is paramount, Runge-Kutta with adaptive step-size control is recommended. For higher precision applications, the Bulirsch-Stoer method dominates. Stiff differential equations require special methods. The Bader-Deufhard method can be used for almost all stiff problems.
- The numerical integration of DAEs of Index 2 or greater is fraught with difficulties. What is usually done is to convert the DAEs to ODEs by differentiating the AEs with respect to time convert them to ODEs.

# Appendix B Physical Property and Database System of the Simulator

## B.1 Introduction

It is well known that the use of realistic physical properties representations such as equations of state, activity coefficient models, and complex mixing rules greatly increases the amount of computer time required to carry out a given simulation. In the years of dynamic simulation, only the simplest models (perfect gas, Raoult's Law, ideal mixing, etc.) could be used without grossly exceeding the available computing resources. More recently, the widespread availability of sophisticated physical properties systems in steady-state simulators such as ASPEN, have led to similar expectations for dynamic simulators. If for no other reason, the steady-state accuracy requires it. Even with today's greatly improved computing resources, the indiscriminate use of complex physical properties models can result in unacceptably low real time factors.

## B.2 Physical Property System

### B.2.1 Ideal Solution Properties

For any ideal solution, a mixture property depends only on the

properties of the pure constituent species which comprise the mixture. No information about the mixture other than its composition is required. Ideal gases are the special cases of ideal solutions. In ideal solutions all molecules are of the same size and all forces between molecules (like and unlike) are equal. Ideal solution behavior is often approximated by solutions comprised of molecules not too different in size and of the same chemical nature. Thus the equation written for the ideal solution model provides an essential relation:

$$V^{id} = \sum_{i=1}^{n_c} x_i V_i \tag{B-1}$$

where $V^{id}$ is the molar volume of the ideal solution formed from pure species with actual molar volume $V_i$ at the temperature and pressure of the mixture and in the same physical state as the mixture, $x_i$ is the molar fraction of the mixture. The volume change of mixing is zero for ideal solutions. Other extensive thermodynamic properties (for instance, $U$ and $H$) can be given by the similar equations [Smith & Van Ness 1987]:

An ideal solution is also one for which the partial fugacity of each component in solution is given by

$$\hat{f}_i^{id} = x_i f_i \tag{B-2}$$

at all pressures, temperatures, and compositions, where $f_i$ is the fugacity of pure species $i$ at the mixture $T$ and $P$ and in the same physical state as the mixture.

## B.2.2 Non-Ideal Solution Properties

Although the ideal solution models approximate the behavior of certain

fluid mixtures, they do not adequately represent the behavior of most solutions of interests to chemical engineers. However, these models of ideal behavior provide convenient references to which the behavior of non-ideal solutions may be compared.

The derivation of non-ideal thermodynamic properties starts with the observation that the thermodynamic properties of a homogeneous phase are functions of temperature, pressure, and the numbers of moles of the individual species that comprise the phase.

$$dM = \left(\frac{\partial M}{\partial P}\right)_{T,x} dP + \left(\frac{\partial M}{\partial T}\right)_{P,x} dT + \sum_{i=1}^{n_c} \bar{M}_i dx_i \tag{B-3}$$

$$M = \sum_{i=1}^{n_c} x_i \bar{M}_i \tag{B-4}$$

A non-ideal solution is also one for which the partial fugacity of each component in solution is given by

$$\hat{f}_i = \gamma_i x_i f_i \tag{B-5}$$

at all pressures, temperatures, and compositions, where $\gamma_i$ is the activity coefficient of each component in solution.

## B.3 Relational Database

A database is a means of storing information in such a way that information can be retrieved from it. In simplest terms, a *relational* database is one that presents information in tables with rows and columns. A table is referred to as a *relation*, which explains the term *relational database*.

The relational database model is based on branches of mathematics

called set theory and predicate logic. The basic idea behind the relational model is that a database consists of a series of unordered tables (or relations) that can be manipulated using non-procedural operations that return tables. This model was in vast contrast to the more traditional database theories of the time that were much more complicated, less flexible and dependent on the physical storage methods of the data.

Usually we use the terms *relations* (*tables*, *files*), *attributes* (*columns*, *fields*) and *tuples* (*rows*, *records*) to describe the relational database.

### B.3.1 Relationships

The relational model dictates that each row in a table be unique. If we allow duplicate rows in a table, then there is no way to uniquely address a given row via programming. We guarantee uniqueness for a table by designating a *primary key*. A primary key is a key made up of one column or a composite key made up of two or more columns that contain unique values for a table. Although primary keys are a function of individual tables, if we created databases that consisted of only independent and unrelated tables, we would have little need for them. Primary keys become essential, however, when we start to create relationships that join together multiple tables in a database.

A distinguishing feature of relational databases is that it is possible to get data from more than one table in what is called a *join*. There must be one column that appears in both tables in order to relate them to each other. This column, which must be the primary key in one table, is called the *foreign key* in the other table. A foreign key must be either null or equal to an existing primary key value of the table to which it refers.

Relationships between real word entities can be quite complex, involving numerous entities each having multiple relationships with each other. For example, a family has multiple relationships between multiple people – all at the same time. In a relational database, however, we consider only relationships between pairs of tables. These tables can be related in one of three different ways [Date 1995]:

- **One-to-One Relationships**

  Two tables are related in a one-to-one (1–1) relationship if, for every row in the first table, there is at most one row in the second table. True one-to-one relationships seldom occur in the real world. This type of relationship is often created to get around some limitation of the database management software rather than to model a real-world situation. Tables that are related in one-to-one relationship should always have the same primary key, which will serve as the join column.

- **One-to-Many Relationships**

  Two tables are related in a one-to-many (1–M) relationship if, for every row in the first table, there can be zero, one, or many rows in the second table, but for every row in the second table there is exactly one row in the first table. The one-to-many relationship is also referred to as a parent-children relationship. One-to-many relationships are the most commonly modeled relationships.

- **Many-to-Many Relationships**

  Two tables are related in a many-to-many (M–M) relationship when

for every row in the first table, there can be many rows in the second table, and for every row in the second table, there can be many rows in the first table. Many-to-many relationships can not be directly modeled in relational database programs. These types of relationships must be broken into multiple one-to-many relationships.

## B.3.2 Normalization

When designing databases we are faced with a series of choices. How many tables will there be and what will they represent? Which columns will go in which tables? What will the relationships between the tables be? The answers each to these questions lies in something called normalization. Normalization is the process of simplifying the design of a database so that it achieves the optimum structure.

Normalization theory gives us the concept of normal forms to assist in achieving the optimum structure. The normal forms are a linear progression of rules that are applied to database, with each higher normal form achieving a better and more efficient deign. In general, the normal forms are:

- **First Normal Form**

  A relation is in First Normal Form (1NF) if and only if all column values must be atomic. 1NF dictates that, for every row-by-column position in a given table, there exists only one value, not an array or list of values. The benefits from this rule should be fairly obvious. If lists of values are stored in a single column, there is no simple way to manipulate those values. Retrieval of data becomes much more

laborious and difficult to generalize.

- **Second Normal Form**

  A relation is in Second Normal Form (2NF) if and only if it is in 1NF and every non-key column is fully dependent on the primary key. Put another way, tables should only store data relating to one entity and that entity should be described by its primary key.

- **Third Normal Form**

  A relation is in Third Normal Form (3NF) if and only if it is in 2NF and all non-key columns are mutually independent. Dependencies cause problems when we add, update, or delete records.

### B.3.3 Integrity Rules

Relational tables follow certain integrity rules to ensure that the data stay accurate and are always accessible. There are two types of integrity rules:

- **General Integrity Rules**

  The relational model specifies two general integrity rules: *entity integrity* and *referential integrity*. They are referred to as general rules, because they apply to all databases. The entity integrity rule says that the primary keys cannot contain null data. The reason for this rule should be obvious. We cannot uniquely identify or reference a row in a table, if the primary key of that table can be null. It is important to note that this rule applies to both simple

and composite keys. For composite keys, none of the individual columns can be null. A null value does not equate to a blank or zero. A blank is considered equal to another blank, a zero is equal to another zero, but two null values are not considered equal. The referential integrity rule says that the foreign key must be either null or equal to an existing primary key value of the table to which it refers. This is different from a primary key, which may not be null.

- **Database-Specific Integrity Rules**

  All integrity constraints that do not fall under entity integrity or referential integrity are termed database-specific rules or business rules. These types of rules are specific to each database and come from the rules of the business being modeled by the database. It is important to note that the enforcement of business rules is as important as the enforcement of the general integrity rules. Without the specification and enforcement of business rules, bad data will get in the database.

## B.4 Database and XML

XML (eXtensible Markup Language) is a formal recommendation from the World Wide Web Consortium (W3C). XML is all about metadata and the idea that certain groups of people have similar needs for describing and organizing the data they use. Like HTML, XML is a set of tags and declarations, but rather than being concerned with formatting information on a page, XML focuses on providing information about the data itself and how

it relates to other data. XML is a flexible way to create common information formats and share both the format and the data. The power of XML is transmitting data from system to system, application to application, and business to business [McCarty & McCarthy 1997].

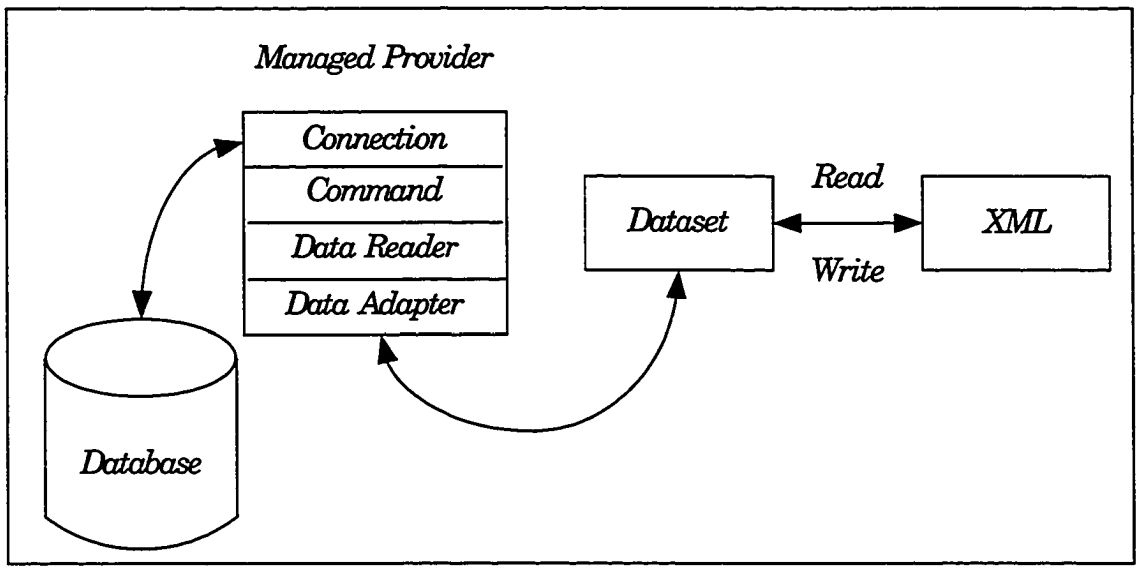



Figure B-1 Relational Database and XML

As a general rule, it is pretty safe to say there are two basic scenarios where XML is being used. The first, and still most prevalent, is to use XML as a data exchange mechanism. Data is not stored natively as XML rather than as relational table in this scenario. The second application for XML is to apply XML to various document management functions. This is largely where native XML databases come into play. Developers of these systems will relish the easy search, query, and transformation enabled by storing data as XML directly within a database.

Whether or not to characterize documents as data-centric or document-centric will depend on what kind of database to use. As a general rule, data is stored in a traditional database, such as a relational, object-oriented, or

hierarchical database, and documents are stored in a *native XML database* (a database designed especially for storing XML).

If data-centric (simple and regular), relational database will be chosen; if document-centric (complex and irregular), XML will be chosen. *Native XML databases* are databases designed especially to store XML documents. Like other databases, they support features like transactions, security, multi-user access, query languages, and so on. The only difference from other databases is that their internal model is based on XML and not something else, such as the relational model.

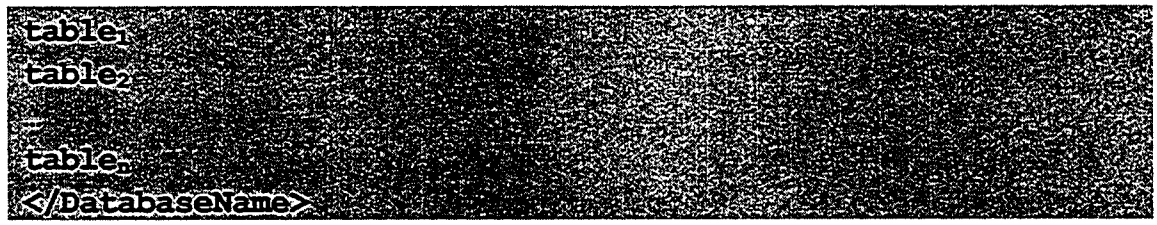
Native XML databases are most clearly useful for storing document-centric documents. This is because native XML databases preserve things like document order, processing instructions, comments, and entity usage, while XML-enabled databases do not.

On the plus side, XML provides many of the things found in databases: storage, schemas, query languages, programming interfaces, and so on. On the minus side, it lacks many of the things found in real databases: efficient storage, indexes, security, transactions and data integrity, multi-user access, triggers, queries across multiple documents, and so on.

A relational database consists of a set of *tables*, where each table is a set of *records*. A record in turn is a set of *fields* and each field is a pair field-name/field-value. All records in a particular table have the same number of fields with the same field-names.

- **Database**

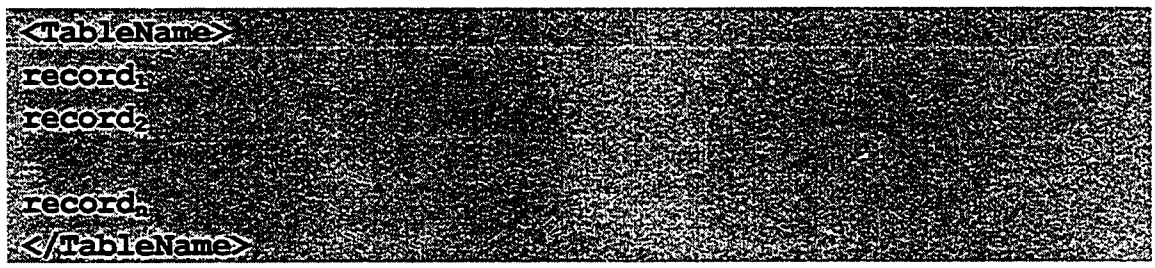
We can model the database with a *document node* and its associated *element node*:




The *DatabaseName* is the name of the database. The order of the tables is arbitrary, since a relational database defines no ordering on them.

- **Table**

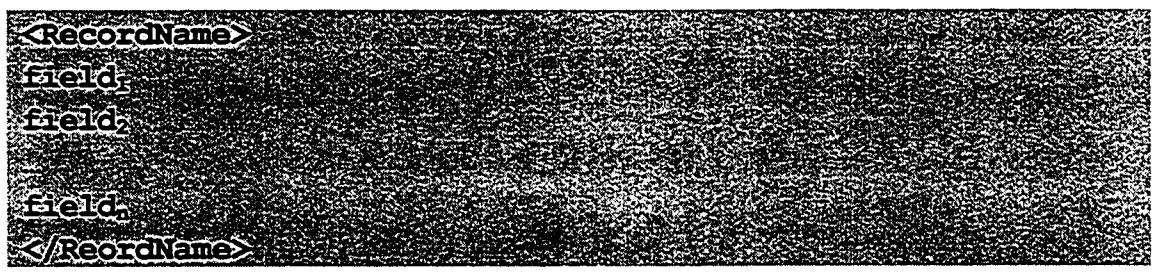
Each table of the database is represented by an element node with the records as its children:




The *TableName* is the name of the table. The order of the records is arbitrary, since the relational data model defines no ordering on them.

- **Record**

A record is also represented by an element node, with its fields as children:




The *RecordName* is arbitrary, since the relational data model does not define a name for a record type. However, in XML it cannot be omitted. The order of the fields is again immaterial.

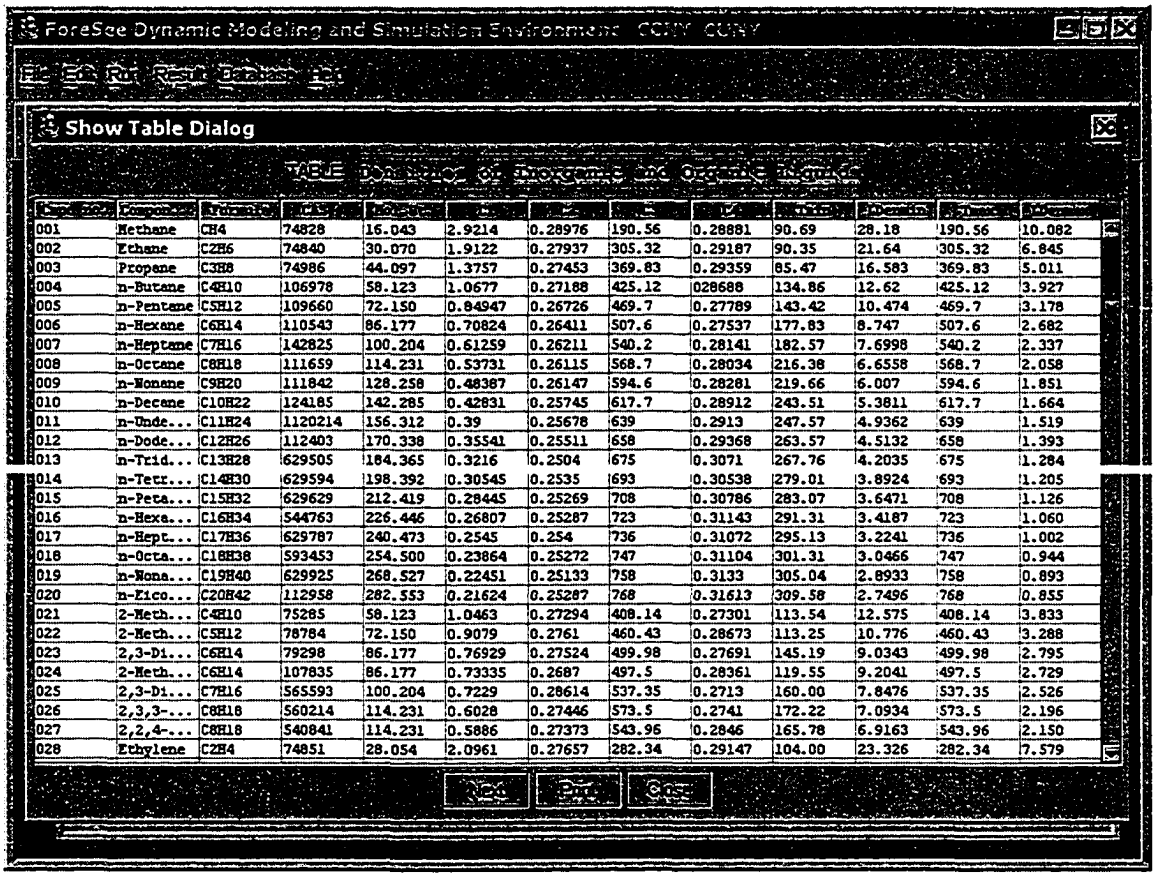
- **Field**

A field is represented as an element node with a data node as its only child:

```
<FieldName type="some_type">
some_value
</FieldName>
```

The value of **some_type** indicates the type of the value (such as *string, number, boolean, date*).

## B.5 Implementation of ForeSee Prototype




| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 001 | Methane | CH4 | 74828 | 16.043 | 2.9214 | 0.28976 | 190.56 | 0.28881 | 90.69 | 28.18 | 190.56 | 10.082 |
| 002 | Ethane | C2H6 | 74840 | 30.070 | 1.9122 | 0.27937 | 305.32 | 0.29187 | 90.35 | 21.64 | 305.32 | 6.845 |
| 003 | Propane | C3H8 | 74986 | 44.097 | 1.3757 | 0.27453 | 369.83 | 0.29359 | 85.47 | 16.583 | 369.83 | 5.011 |
| 004 | n-Butane | C4H10 | 106978 | 58.123 | 1.0677 | 0.27188 | 425.12 | 028688 | 134.86 | 12.62 | 425.12 | 3.927 |
| 005 | n-Pentane | C5H12 | 109660 | 72.150 | 0.84947 | 0.26726 | 469.7 | 0.27789 | 143.42 | 10.474 | 469.7 | 3.178 |
| 006 | n-Hexane | C6H14 | 110543 | 86.177 | 0.70824 | 0.26411 | 507.6 | 0.27537 | 177.83 | 8.747 | 507.6 | 2.682 |
| 007 | n-Heptane | C7H16 | 142825 | 100.204 | 0.61259 | 0.26211 | 540.2 | 0.28141 | 182.57 | 7.6998 | 540.2 | 2.337 |
| 008 | n-Octane | C8H18 | 111659 | 114.231 | 0.53731 | 0.26115 | 568.7 | 0.28034 | 216.38 | 6.6558 | 568.7 | 2.058 |
| 009 | n-Nonane | C9H20 | 111842 | 128.258 | 0.48387 | 0.26147 | 594.6 | 0.28281 | 219.66 | 6.007 | 594.6 | 1.851 |
| 010 | n-Decane | C10H22 | 124185 | 142.285 | 0.42831 | 0.25745 | 617.7 | 0.28912 | 243.51 | 5.3811 | 617.7 | 1.664 |
| 011 | n-Unde... | C11H24 | 1120214 | 156.312 | 0.39 | 0.25678 | 639 | 0.2913 | 247.57 | 4.9362 | 639 | 1.519 |
| 012 | n-Dode... | C12H26 | 112403 | 170.338 | 0.35541 | 0.25511 | 658 | 0.29368 | 263.57 | 4.5132 | 658 | 1.393 |
| 013 | n-Trid... | C13H28 | 629505 | 184.365 | 0.3216 | 0.2504 | 675 | 0.3071 | 267.76 | 4.2035 | 675 | 1.284 |
| 014 | n-Tetr... | C14H30 | 629594 | 198.392 | 0.30545 | 0.2535 | 693 | 0.30538 | 279.01 | 3.8924 | 693 | 1.205 |
| 015 | n-Peta... | C15H32 | 629629 | 212.419 | 0.28445 | 0.25269 | 708 | 0.30786 | 283.07 | 3.6471 | 708 | 1.126 |
| 016 | n-Hexa... | C16H34 | 544763 | 226.446 | 0.26807 | 0.25287 | 723 | 0.31143 | 291.31 | 3.4187 | 723 | 1.060 |
| 017 | n-Hept... | C17H36 | 629787 | 240.473 | 0.2545 | 0.254 | 736 | 0.31072 | 295.13 | 3.2241 | 736 | 1.002 |
| 018 | n-Octa... | C18H38 | 593453 | 254.500 | 0.23864 | 0.25272 | 747 | 0.31104 | 301.31 | 3.0466 | 747 | 0.944 |
| 019 | n-Nona... | C19H40 | 629925 | 268.527 | 0.22451 | 0.25133 | 758 | 0.3133 | 305.04 | 2.8933 | 758 | 0.893 |
| 020 | n-Eico... | C20H42 | 112958 | 282.553 | 0.21624 | 0.25287 | 768 | 0.31613 | 309.58 | 2.7496 | 768 | 0.855 |
| 021 | 2-Meth... | C4H10 | 75285 | 58.123 | 1.0463 | 0.27294 | 408.14 | 0.27301 | 113.54 | 12.575 | 408.14 | 3.833 |
| 022 | 2-Meth... | C5H12 | 78784 | 72.150 | 0.9079 | 0.2761 | 460.43 | 0.28673 | 113.25 | 10.776 | 460.43 | 3.288 |
| 023 | 2,3-Di... | C6H14 | 79298 | 86.177 | 0.76929 | 0.27524 | 499.98 | 0.27691 | 145.19 | 9.0343 | 499.98 | 2.795 |
| 024 | 2-Meth... | C6H14 | 107835 | 86.177 | 0.73335 | 0.2687 | 497.5 | 0.28361 | 119.55 | 9.2041 | 497.5 | 2.729 |
| 025 | 2,3-Di... | C7H16 | 565593 | 100.204 | 0.7229 | 0.28614 | 537.35 | 0.2713 | 160.00 | 7.8476 | 537.35 | 2.526 |
| 026 | 2,3,3-... | C8H18 | 560214 | 114.231 | 0.6028 | 0.27446 | 573.5 | 0.2741 | 172.22 | 7.0934 | 573.5 | 2.196 |
| 027 | 2,2,4-... | C8H18 | 540841 | 114.231 | 0.5886 | 0.27373 | 543.96 | 0.2846 | 165.78 | 6.9163 | 543.96 | 2.150 |
| 028 | Ethylene | C2H4 | 74851 | 28.054 | 2.0961 | 0.27657 | 282.34 | 0.29147 | 104.00 | 23.326 | 282.34 | 7.579 |

Figure B-2 Table Structure of Physical Property Database in ForeSee

Table structure interface is used in ForeSee prototype in order to search, modify, delete, and create a database. The table structure in ForeSee prototype is shown in Figure B-2.

We use XML structure to store physical property data in ForeSee prototype. The XML structure in ForeSee prototype is shown in Figure B-3.

DIPPR data are used in ForeSee for rigorous and reliable physical property calculations [Perry et al. 1996]. This provides a totally consistent basis for steady-state and dynamic simulations.

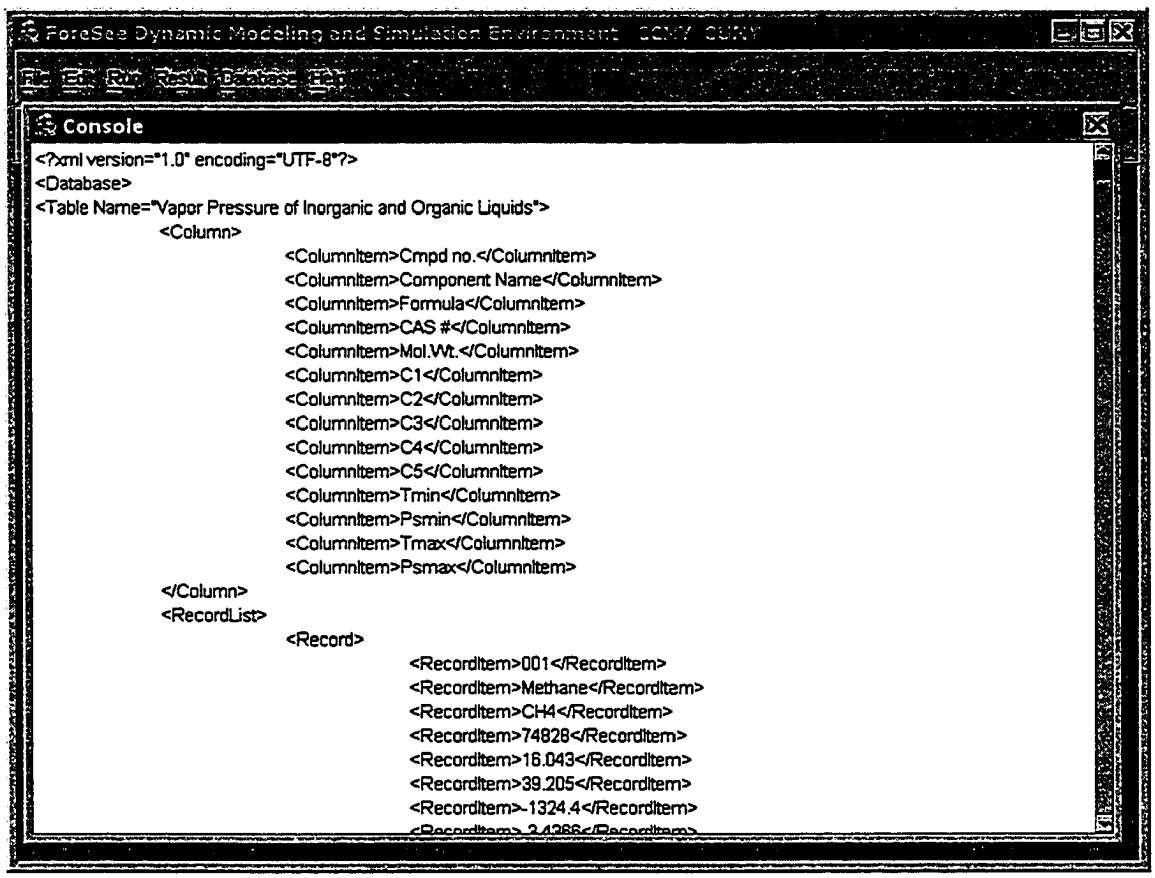



Figure B-3 XML Structure of Physical Property Database in ForeSee

## B.6 Conclusions

- Combining the advantages of table and XML structures, Database system in ForeSee prototype is easy to modify and search.
- Because Java is portable code and XML is portable data, we choose Java to implement ForeSee prototype and XML to store physical property data. Therefore, ForeSee prototype is platform-independent and can run on any platform computer.

# Appendix C Model Formulation of Modeling Components

## C.1 Introduction

The numerical solution of DAEs for the purposes of simulation is much more difficult than for ODEs. DAEs are not ODEs. A different method might be used to solve DAEs. Unfortunately, there are few solvers available for DAEs, DASSL being one of the few [Brenan et al. 1989]. What is usually done is to convert the DAEs to ODEs by differentiating the AEs with respect to time to convert them to ODEs. The minimum number of times this differentiating must be performed to completely reduce the AEs to ODEs is referred to as the index of DAEs. System whose index is equal to one can be reliably integrated using ODEs solvers, although care must be taken in specifying a consistent set of initial conditions that satisfy the original AEs. Systems with an index higher than one cannot be reliably integrated. Fortunately, with sufficient care in model formulation (e.g., re-examine some other assumption to remove the index problem), the DAEs arising in process simulation will be of Index 1 or even 0. The approach used in ForeSee prototype is designed to insure that models being used are of Index 1 or less.

# C.2 Core Models

## C.2.1 Well-Mixed Element (WME)

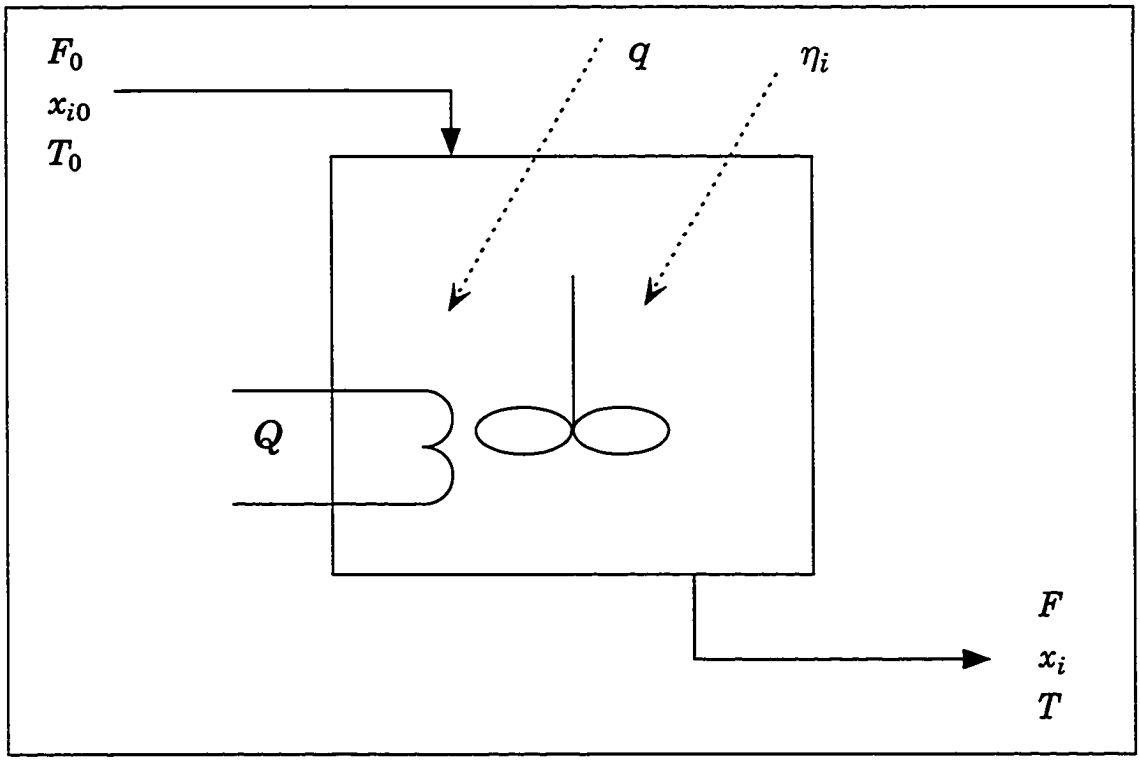



Figure C-1 Well-Mixed Element (WME)

The following is a development of the dynamic model equations for a well-mixed element such as that shown in Figure C-1. Well-mixed element is idealized by the assumptions:

- The composition, temperature and pressure in the tank are the same as in the output

Based on the conservation of mass for each component in a multi-component phase and the conservation of energy, these equations have the

general form:

Let us introduce the following notation:

$A$ = mass or heat transfer surface area [$m^2$]

$C_P$ = heat capacity at constant pressure [kJ/kmol · K]

$F$ = molar output flow rate [kmol/s]

$F_0$ = molar input flow rate [kmol/s]

$H$ = molar enthalpy of the material leaving the element [kJ/kmol]

$H_0$ = molar enthalpy of the material entering the element [kJ/kmol]

$\bar{H}_i$ = partial molar enthalpy of the $i^{th}$ component [kJ/kmol]

$\bar{H}_i^*$ = partial molar enthalpy of the $i^{th}$ component crossing the interface [kJ/kmol]

$n_c$ = number of components

$n_r$ = number of reactions occurred within the element

$P$ = phase pressure [kPa]

$Q$ = rate of heat addition to the element [$kJ/m^2 \cdot s$]

$q$ = heat transfer rate crossing the interface [$kJ/m^2 \cdot s$]

$R$ = rate of reaction of the system [$kmol/m^3 \cdot s$]

$R_i$ = rate of reaction of the $i^{th}$ component into the system [$kmol/m^3 \cdot s$]

$r_j$ = reaction rate of the $j^{th}$ chemical reaction [$kmol/m^3 \cdot s$]

$T$ = phase temperature [K]

$U$ = molar internal energy of the material leaving the element [kJ/kmol]

$V$ = molar volume of the element [$m^3$/kmol]

$\bar{V}_i$ = partial molar volume of the $i^{th}$ component [$m^3$/kmol]

$x_i$ = phase molar fraction.

$\alpha_{ij}$ = stoichiometric coefficient

$\phi$ = phase volume [$m^3$]

$\eta$ = rate of mass transfer crossing the interface [$kmol/m^2 \cdot s$]

$\eta_i$ = rate of mass transfer of the $i^{th}$ component into the system [$kmol/m^2 \cdot s$]

$\rho$ = molar density [$kmol/m^3$]

$$\frac{d\Psi}{dt} = \mathbf{F}(\mathbf{Y},\mathbf{Z},\mathbf{M}) \tag{C-1}$$

$$\mathbf{G}(\mathbf{Y},\mathbf{Z},\mathbf{M}) = 0 \tag{C-2}$$

where $\Psi$ is a vector of holdups of conservative quantities, $\mathbf{Y}$ is a vector of state variables, $\mathbf{Z}$ a vector of algebraic variables, and $\mathbf{M}$ a vector of input variables. Note that these equations form a set of differential-algebraic equations (DAEs). Specifically $\Psi$ and $\mathbf{F}(\mathbf{Y}, \mathbf{Z}, \mathbf{M})$ are defined as [Rinard 1998]:

$$\Psi = \left[\phi\rho x_1, \phi\rho x_2, \cdots, \phi\rho x_{n_c}, \phi\rho U\right]^T \tag{C-3}$$

$$\mathbf{F} = \left[f_1, f_2, \cdots, f_{n_c}, f_{n_c+1}\right]^T \tag{C-4}$$

The right-hand-side of Equation (C-1), namely $\mathbf{F}(\mathbf{Y}, \mathbf{Z}, \mathbf{M})$, has the form:

$$f_i = F_0 x_{i0} - F x_i + A\eta_i + \phi R_i \qquad i = 1,2,\cdots,n_c \tag{C-5}$$

$$R_i \equiv \sum_{j=1}^{n_r} \alpha_{ij} r_j \tag{C-6}$$

$$f_{nc+1} = F_0 H_0 - FH + A\sum_{i=1}^{n_c} \eta_i \bar{H}_i^* + A(Q+q) \tag{C-7}$$

While there may be more, there is at least one algebraic equation in the set of Equation (C-2), namely,

$$g_1(x_1,x_2,\cdots,x_{nc}) = \sum_{i=1}^{n_c} x_i - 1 = 0 \tag{C-8}$$

Equation (C-8) along with Equation (C-1) form a set of DAEs of index one. If Equation (C-8) is differentiated with respect to time, we get the following equation:

$$\frac{dg_1}{dt} = \sum_{i=1}^{n_c} \frac{dx_i}{dt} = 0 \tag{C-9}$$

Summing the first $n_c$ elements of Equation (C-1) over $i$ and applying Equations (C-8) and (C-9) gives an overall mass balance of the form:

$$\frac{d\phi\rho}{dt} = F_0 - F + A\eta + \phi R \tag{C-10}$$

$$\eta \equiv \sum_{i=1}^{n_c} \eta_i \tag{C-11}$$

$$R \equiv \sum_{i=1}^{n_c} R_i \tag{C-12}$$

Equation (C-10) can be used to replace Equation (C-8) giving an argumented holdup of the form

$$\Xi = \left[\phi\rho x_1, \phi\rho x_2, \cdots, \phi\rho x_{n_c}, \phi\rho, \phi\rho U\right]^T \tag{C-13}$$

$$\Gamma = \left[\gamma_1, \gamma_2, \cdots, \gamma_{n_c}, \gamma_{n_c+1}, \gamma_{n_c+2}\right]^T \tag{C-14}$$

and Equations (C-1) and (C-2) become

$$\frac{d\Xi}{dt} = \Gamma(\mathbf{Y},\mathbf{Z},\mathbf{M}) \tag{C-15}$$

$$\Omega(\mathbf{Y},\mathbf{Z},\mathbf{M}) = 0 \tag{C-16}$$

- **Conservation Equations**

For component material balance

$$\frac{d\phi\rho x_i}{dt} = \gamma_i = F_0 x_{i0} - F x_i + A\eta_i + \phi R_i \tag{C-17}$$

For overall mass balance

$$\frac{d\phi\rho}{dt} = \gamma_{n_c+1} = F_0 - F + A\eta + \phi R \tag{C-18}$$

For energy balance

$$\frac{d\phi\rho U}{dt} = \gamma_{n_c+2} = F_0 H_0 - FH + A\sum_{i=1}^{n_c}\eta_i \bar{H}_i^* + A(Q+q) \tag{C-19}$$

Equation (C-15) can be expanded as follows:

$$J_\gamma \frac{d\mathbf{Y}}{dt} = \Gamma(\mathbf{Y},\mathbf{Z},\mathbf{M}) \tag{C-20}$$

$$J_\gamma \equiv \left(\frac{\partial \Xi_i}{\partial y_j}\right)^T \tag{C-21}$$

$J_\gamma$ will be referred to as the holdup Jacobean. To reduce Equation (C-20) to standard first-order form we requires inverting the holdup Jacobean to give

$$\frac{d\mathbf{Y}}{dt} = J_\gamma^{-1}\Gamma(\mathbf{Y},\mathbf{Z},\mathbf{M}) \tag{C-22}$$

We note, however, that the inversion of $J_\gamma$ is not a trivial matter. For this problem if we use the chain rule of differentiation, we can convert the above balance equations into the first-order form:

$$\phi\rho\frac{dx_i}{dt} = F_0(x_{i0} - x_i) + A(\eta_i - x_i\eta) + \phi(R_i - x_i R) \tag{C-23}$$

The overall material balance is

$$\frac{d\rho\phi}{dt} = \rho\frac{d\phi}{dt} + \phi\frac{d\rho}{dt} = F_0 - F + A\eta + \phi R \tag{C-24}$$

Substituting Equation (C-24) into the energy balance Equation (C-19) gives

$$\begin{aligned}\phi\rho\frac{dU}{dt} &= F_0(H_0 - U) - F(H - U) \\ &\quad + A\left(\sum_{i=1}^{n_c}\eta_i\bar{H}_i^* - \eta U\right) + A(Q + q) - \phi RU\end{aligned} \tag{C-25}$$

- **Constitutive Equations**

The final step in the derivation is to expand the time derivatives of the thermodynamic state functions $H$ and $V$ ($V = 1/\rho$) in terms of $T$, $P$ and $x_i$. Doing so gives:

$$\frac{dH}{dt} = \frac{\partial H}{\partial T}\frac{dT}{dt} + \frac{\partial H}{\partial P}\frac{dP}{dt} + \sum_{i=1}^{n_c}\left(\bar{H}_i\frac{dx_i}{dt}\right) \tag{C-26}$$

$$\bar{H}_i \equiv \left(\frac{\partial H}{\partial x_i}\right)_{P,T} \tag{C-27}$$

$$C_P \equiv \left(\frac{\partial H}{\partial T}\right)_{P,x} \tag{C-28}$$

$$H = \sum_{i=1}^{n_c} x_i\bar{H}_i \tag{C-29}$$

$$\frac{dV}{dt} = \frac{\partial V}{\partial T}\frac{dT}{dt} + \frac{\partial V}{\partial P}\frac{dP}{dt} + \sum_{i=1}^{n_c}\left(\bar{V}_i\frac{dx_i}{dt}\right) \tag{C-30}$$

$$\bar{V}_i \equiv \left(\frac{\partial V}{\partial x_i}\right)_{P,T} \tag{C-31}$$

$$V = \sum_{i=1}^{n_c} x_i\bar{V}_i \tag{C-32}$$

$$\frac{d\rho}{dt} = -\rho^2\frac{dV}{dt} \tag{C-33}$$

Substituting Equation (C-30) into Equation (C-33) gives

$$\frac{d\rho}{dt} = \frac{\partial \rho}{\partial T}\frac{dT}{dt} + \frac{\partial \rho}{\partial P}\frac{dP}{dt} - \rho^2 \sum_{i=1}^{n_c}\left(\bar{V}_i \frac{dx_i}{dt}\right) \tag{C-34}$$

Let us define $H \equiv U + \xi PV$ where $\xi \approx 0$ for incompressible fluid (e.g., liquid and solid) and $\xi = 1$ for compressible fluid (e.g., gas). Therefore, we get the following equation:

$$\frac{dU}{dt} = \frac{dH}{dt} - \xi\left(P\frac{dV}{dt} + V\frac{dP}{dt}\right) \tag{C-35}$$

Substituting Equation (C-34) into the overall mass balance Equation (C-24) gives

$$\begin{aligned}\rho\frac{d\phi}{dt} + \phi\frac{\partial \rho}{\partial T}\frac{dT}{dt} + \phi\frac{\partial \rho}{\partial P}\frac{dP}{dt} &= F_0 - F + A\eta + \phi R \\ &\quad + \phi\rho^2 \sum_{i=1}^{n_c}\left(\bar{V}_i \frac{dx_i}{dt}\right)\end{aligned} \tag{C-36}$$

Substituting Equations (C-24), (C-33), and (C-35) into the energy balance Equation (C-25) gives

$$\begin{aligned}\phi\rho\frac{dH}{dt} - \xi\left(P\frac{d\phi}{dt} + \phi\frac{dP}{dt}\right) &= F_0(H_0 - H) + A\left(\sum_{i=1}^{n_c}\eta_i \bar{H}_i^* - \eta H\right) \\ &\quad + A(Q + q) - \phi RH\end{aligned} \tag{C-37}$$

Therefore, substituting Equations (C-23) and (C-26) into Equations (C-36) and (C-37) gives the final forms of balance equations:

- the component mass balance equation

$$\phi\rho\frac{dx_i}{dt} = F_{ci} \tag{C-38}$$

$$F_{ci} \equiv F_0(x_{i0} - x_i) + A(\eta_i - x_i\eta) + \phi(R_i - x_i R) \tag{C-39}$$

- the overall mass balance equation

$$\rho\frac{d\phi}{dt}+\phi\frac{\partial\rho}{\partial T}\frac{dT}{dt}+\phi\frac{\partial\rho}{\partial P}\frac{dP}{dt}=F_m \tag{C-40}$$

$$F_m\equiv\rho\sum_{i=1}^{n_c}(x_{i0}F_0-x_iF+A\eta_i+\phi R_i)\bar{V}_i \tag{C-41}$$

- the energy balance equation

$$\phi\rho C_P\frac{dT}{dt}+\phi\left(\rho\frac{\partial H}{\partial P}-\xi\right)\frac{dP}{dt}-\xi P\frac{d\phi}{dt}=Q_e \tag{C-42}$$

$$Q_e\equiv\sum_{i=1}^{n_c}\left\{F_0x_{i0}\left(\bar{H}_{i0}-\bar{H}_i\right)+A\eta_i\left(\bar{H}_i^*-\bar{H}_i\right)-\phi R_i\bar{H}_i\right\}+A(Q+q) \tag{C-43}$$

$$Q_e=F_0\sum_{i=1}^{n_c}x_{i0}\left(\bar{H}_{i0}-\bar{H}_i\right)+A\sum_{i=1}^{n_c}\eta_i\lambda_i-\phi\sum_{j=1}^{n_r}r_jH_{rj}+A(Q+q) \tag{C-44}$$

$$\lambda_i\equiv\bar{H}_i^*-\bar{H}_i \tag{C-45}$$

$$H_{rj}\equiv\sum_{i=1}^{n_c}\alpha_{ij}\bar{H}_i \tag{C-46}$$

These equations are not in standard first-order form. Further there are three derivatives ($dT/dt$, $dP/dt$, and $d\phi/dt$) but only two equations. Either we need another equation, one that defines $d\phi/dt$ for instance, or we must make some simplifying assumptions. For the application of the well-mixed element model to a single-phase system, there are two sets of assumptions, each of which eliminates one of the three derivatives.

- **Liquid Phase System**

In the case of a liquid, we assume that it is almost incompressible, i.e., $\partial\rho/\partial P\approx 0$, $H\approx U$ and $\xi\approx 0$. Further, we assume that the effect of pressure $P$ on $H$ is negligible, i.e., $\partial H/\partial P\approx 0$. These assumptions reduce Equations (C-42) and (C-40) to standard first-order form in $dT/dt$ and $d\phi/dt$, respectively. Therefore, we get the final equations for a well-mixed liquid phase system as follows:

$$\frac{dx_i}{dt} = \frac{1}{\phi\rho} F_{ci} \tag{C-47}$$

$$\frac{dT}{dt} = \frac{1}{\phi\rho C_P} Q_e \tag{C-48}$$

$$\frac{d\phi}{dt} = \frac{1}{\rho} F_m - \frac{\frac{\partial\rho}{\partial T}}{\rho^2 C_P} Q_e \tag{C-49}$$

- **Vapor Phase System**

In the case of a gas, we have $\xi = 1$. Equations (C-40) and (C-42) must then be solved simultaneously for $dT/dt$ and $dP/dt$ to put them in standard first-order form. We need another equation which defines $d\phi/dt$. A vapor expands to fill whatever volume it is in. For example, if there is only one vapor phase in the system of rigid vessel, then its volume will remain constant. This means that $d\phi_V/dt = 0$ (constraint equation). If there are both vapor phase and liquid phase in the system of rigid vessel, then the total volume will remain constant. This means that $d\phi_T/dt = d(\phi_V + \phi_L)/dt = 0$ (constraint equation). Therefore, we get the final equations for a well-mixed vapor phase system as follows:

$$\frac{dy_i}{dt} = \frac{1}{\phi\rho} F_{ci} \tag{C-50}$$

$$\frac{dT}{dt} = \frac{1}{\phi} \frac{\left(1 - \rho\frac{\partial H}{\partial P}\right) F_m + \frac{\partial\rho}{\partial P} Q_e + \left(\rho^2 \frac{\partial H}{\partial P} - \rho + P\frac{\partial\rho}{\partial P}\right)\frac{d\phi}{dt}}{\rho C_P \frac{\partial\rho}{\partial P} + \left(1 - \rho\frac{\partial H}{\partial P}\right)\frac{\partial\rho}{\partial T}} \tag{C-51}$$

$$\frac{dP}{dt} = \frac{1}{\phi} \frac{\rho C_P F_m - \frac{\partial\rho}{\partial T} Q_e - \left(\rho^2 C_P + P\frac{\partial\rho}{\partial T}\right)\frac{d\phi}{dt}}{\rho C_P \frac{\partial\rho}{\partial P} + \left(1 - \rho\frac{\partial H}{\partial P}\right)\frac{\partial\rho}{\partial T}} \tag{C-52}$$

$$\frac{d\phi}{dt} = \frac{d\phi_V}{dt} = \frac{d\phi_T}{dt} - \frac{d\phi_L}{dt} = -\frac{d\phi_L}{dt} \tag{C-53}$$

Note that Equations (C-49), (C-52) and (C-53) usually describe the rate of

change of the *geometric variable*. They are usually referred to as *coordinator* equations. All the other equations, namely, Equations (C-47), (C-48), (C-50), and (C-51) describe the rates of change of the *intensive variables* within a phase. These are usually referred to as *core model* equations. The terms of heat transfer rate, mass transfer rate and chemical reaction rate present in the above all equations describe the transfer rates of the *extensive variables* between phases. These are usually determined by another type of component model known as a *connector*.

Table C-1 Relationship between Model Equations and Modeling Components in Well-Mixed Element

| Modeling Components | Model Equations |
|---|---|
| CoreModels (Conservation Equations) | Equations (C-47), (C-48),(C-50), and (C-51) |
| Connectors (Constitutive Equations) | Equations $q$, $\eta_i$ and $r_j$ |
| Coordinators (Constraint Equations) | Equations (C-49), (C-52) and (C-53) |

## C.2.2 Plug Flow Element (PFE)

Let us start by stating the assumptions that underlie the plug flow and discussing their range of validity. From a physical point of view, we assume that fluid moves through the tabular volume in a continuous series of discs or plugs (Plug Flow Element). Each plug moves through without mixing with the plug either directly behind it or in front of it. Any heat and mass transfer are through the tube wall or phase boundary of the element. Further, we assume that each plug is well-mixed in the direction normal to that of the bulk flow. Since most plug flow element (PFE) volumes are cylindrical in

shape, normal is in the radial direction. This means that the pressure, temperature and composition profiles are uniform in the radial direction. Finally, any resistance to heat or mass transfer occurs entirely at the tube wall or phase boundary. Summarizing these assumptions:

- There is no fluid mixing in the axial direction of flow.
- The fluid is well-mixed in the radial direction; there are no radial gradients of pressure, temperature or composition.
- The only resistance to heat or mass transfer is at the wall of phase boundary.

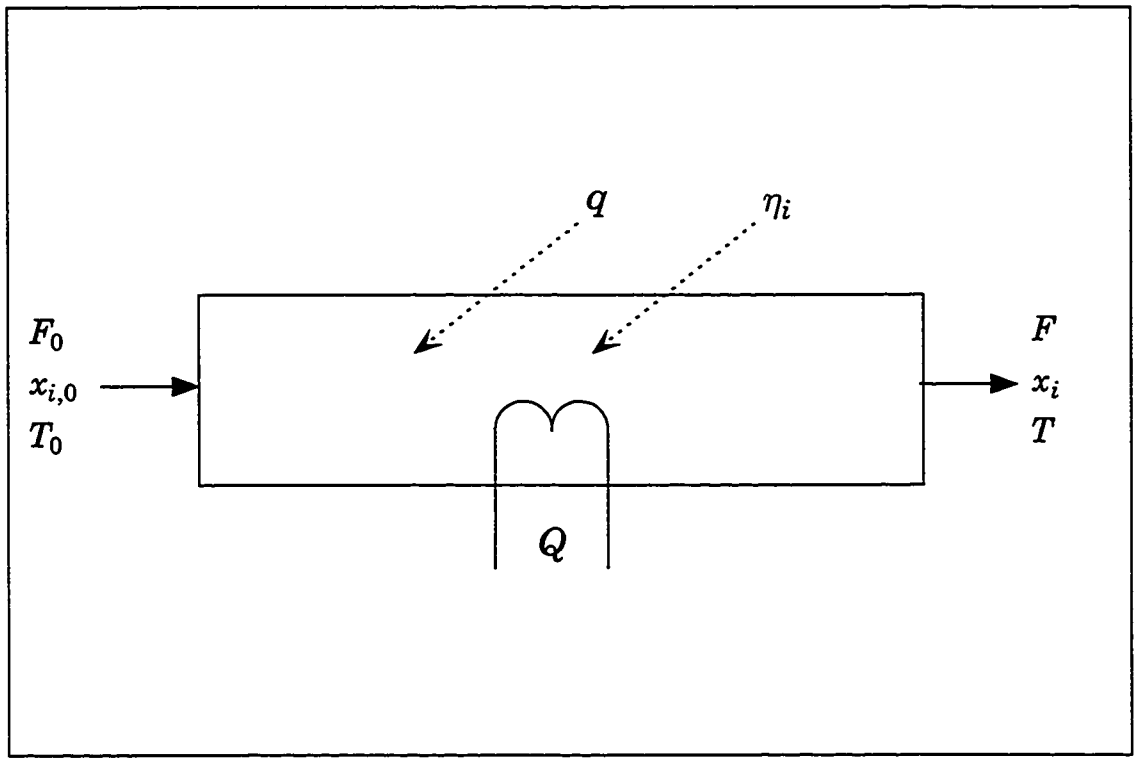



Figure C-2 Plug Flow Element (PFE)

We introduce the following new notation:

$C$ = cross section circumference [m]

$L$ = axial length [m]

$S$ = cross section area [$m^2$]

$z$ = axial distance [m]

Let us now derive the fundamental equations for the plug flow element which allow for heat and mass transfer to or from the element as well as for reaction within the fluid itself [Rinard 1998].

- **Balance Equations**

A component mass balance on the section of differential length $dz$ in the axial direction of flow gives

$$\frac{\partial[S\rho x_i]}{\partial t} + \frac{\partial[Fx_i]}{\partial z} = C\eta_i + SR_i \tag{C-54}$$

$$R_i \equiv \sum_{j=1}^{n_r} \alpha_{ij} r_j \tag{C-55}$$

Summing over $i$ gives an overall material balance

$$\frac{\partial[S\rho]}{\partial t} + \frac{\partial F}{\partial z} = C\eta + SR \tag{C-56}$$

$$\eta \equiv \sum_{i=1}^{n_c} \eta_i \tag{C-57}$$

$$R \equiv \sum_{i=1}^{n_c} R_i \tag{C-58}$$

Expanding the partial derivatives in Equation (C-54) and substituting Equation (C-56) reduces Equation (C-54) to standard form as follows:

$$S\rho\frac{\partial x_i}{\partial t} + F\frac{\partial x_i}{\partial z} = C(\eta_i - x_i\eta) + S(R_i - x_iR) \tag{C-59}$$

An energy balance on the section of differential length $dz$ gives

$$\frac{\partial[S\rho U]}{\partial t} + \frac{\partial[FH]}{\partial z} = C\sum_{i=1}^{n_c} \eta_i \bar{H}_i^* + C(Q+q) \tag{C-60}$$

Expanding the partial derivatives in Equation (C-60) and substituting Equation (C-56) reduces Equation (C-60) to the following form:

$$\begin{aligned} & S\rho\frac{\partial U}{\partial t}+(H-U)\frac{\partial F}{\partial z}+F\frac{\partial H}{\partial z} \\ & =C\left(\sum_{i=1}^{n_c}\eta_i\bar{H}_i^*-\eta U\right)+C(Q+q)-SRU \end{aligned} \tag{C-61}$$

$$H\equiv U+\xi PV \tag{C-62}$$

- **Constitutive Equations**

The final step in the derivation is to expand the time and axial distance partial derivatives of the thermodynamic state functions $H$ and $U$ in terms of $T$, $P$ and $x_i$. Doing so gives:

$$\frac{\partial H}{\partial t}=\frac{\partial H}{\partial T}\frac{\partial T}{\partial t}+\frac{\partial H}{\partial P}\frac{\partial P}{\partial t}+\sum_{i=1}^{n_c}\left(\bar{H}_i\frac{\partial x_i}{\partial t}\right) \tag{C-63}$$

$$\frac{\partial H}{\partial z}=\frac{\partial H}{\partial T}\frac{\partial T}{\partial z}+\frac{\partial H}{\partial P}\frac{\partial P}{\partial z}+\sum_{i=1}^{n_c}\left(\bar{H}_i\frac{\partial x_i}{\partial z}\right) \tag{C-64}$$

$$\frac{\partial V}{\partial t}=\frac{\partial V}{\partial T}\frac{\partial T}{\partial t}+\frac{\partial V}{\partial P}\frac{\partial P}{\partial t}+\sum_{i=1}^{n_c}\left(\bar{V}_i\frac{\partial x_i}{\partial t}\right) \tag{C-65}$$

$$\frac{\partial U}{\partial t}=\frac{\partial H}{\partial t}-\xi\left(P\frac{\partial V}{\partial t}+V\frac{\partial P}{\partial t}\right) \tag{C-66}$$

$$\frac{\partial \rho}{\partial t}=-\rho^2\frac{\partial V}{\partial t} \tag{C-67}$$

$$\frac{\partial \rho}{\partial t}=\frac{\partial \rho}{\partial T}\frac{\partial T}{\partial t}+\frac{\partial \rho}{\partial P}\frac{\partial P}{\partial t}-\rho^2\sum_{i=1}^{n_c}\left(\bar{V}_i\frac{\partial x_i}{\partial t}\right) \tag{C-68}$$

Substituting Equation (C-68) into the overall mass balance Equation (C-56) gives

$$\rho\frac{\partial S}{\partial t}+S\frac{\partial \rho}{\partial T}\frac{\partial T}{\partial t}+S\frac{\partial \rho}{\partial P}\frac{\partial P}{\partial t}+\frac{\partial F}{\partial z}$$
$$=C\eta+SR+S\rho^2\sum_{i=1}^{n_c}\left(\bar{V}_i\frac{\partial x_i}{\partial t}\right) \tag{C-69}$$

Substituting Equations (C-56), (C-66), and (C-67) into the energy balance Equation (C-61) gives

$$S\rho\frac{\partial H}{\partial t}-\xi\left(P\frac{\partial S}{\partial t}+S\frac{\partial P}{\partial t}\right)+F\frac{\partial H}{\partial z}$$
$$=C\left(\sum_{i=1}^{n_c}\eta_i\bar{H}_i^*-\eta H\right)+C(Q+q)-SRH \tag{C-70}$$

Therefore, substituting Equations (C-59) and (C-63) into Equations (C-69) and (C-70) gives the final forms of balance equations:

- the component mass balance equation

$$S\rho\frac{\partial x_i}{\partial t}=f_{ci} \tag{C-71}$$

$$f_{ci}\equiv C(\eta_i-x_i\eta)+S(R_i-x_iR)-F\frac{\partial x_i}{\partial z} \tag{C-72}$$

- the overall mass balance equation

$$\rho\frac{\partial S}{\partial t}+S\frac{\partial \rho}{\partial T}\frac{\partial T}{\partial t}+S\frac{\partial \rho}{\partial P}\frac{\partial P}{\partial t}=f_m \tag{C-73}$$

$$f_m\equiv\rho\sum_{i=1}^{n_c}\left(C\eta_i+SR_i-\frac{\partial Fx_i}{\partial z}\right)\bar{V}_i \tag{C-74}$$

- the energy balance equation

$$S\rho C_P\frac{\partial T}{\partial t}+S\left(\rho\frac{\partial H}{\partial P}-\xi\right)\frac{\partial P}{\partial t}-\xi P\frac{\partial S}{\partial t}=q_e \tag{C-75}$$

$$q_e\equiv-F\sum_{i=1}^{n_c}x_i\frac{\partial \bar{H}_i}{\partial z}+C\sum_{i=1}^{n_c}\eta_i\left(\bar{H}_i^*-\bar{H}_i\right)-S\sum_{i=1}^{n_c}R_i\bar{H}_i+C(Q+q) \tag{C-76}$$

$$q_e=-F\sum_{i=1}^{n_c}x_i\frac{\partial \bar{H}_i}{\partial z}+C\sum_{i=1}^{n_c}\eta_i\lambda_i-S\sum_{j=1}^{n_r}r_jH_{rj}+C(Q+q) \tag{C-77}$$

$$\lambda_i \equiv \bar{H}_i^* - \bar{H}_i \tag{C-78}$$

$$H_{rj} \equiv \sum_{i=1}^{n_c} \alpha_{ij} \bar{H}_i \tag{C-79}$$

- **Approximation by Finite Differences**

The partial differential derivative can be approximated by the finite difference of function evaluations [Schiesser 1991; Silebi & Schiesser 1992]. The finite difference approach is the most popular discretization technique, owing to its simplicity. Finite difference approximations of derivatives are obtained by using truncated Taylor series. Consider the following Taylor expansions

$$u(z+h) = u(z) + h\frac{\partial u}{\partial z} + \frac{h^2}{2!}\frac{\partial^2 u}{\partial z^2} + \cdots + \frac{h^n}{n!}\frac{\partial^n u}{\partial z^n} + O\left(h^{n+1}\right) \tag{C-80}$$

$$u(z-h) = u(z) - h\frac{\partial u}{\partial z} + \frac{h^2}{2!}\frac{\partial^2 u}{\partial z^2} + \cdots + \frac{(-h)^n}{n!}\frac{\partial^n u}{\partial z^n} + O\left(h^{n+1}\right) \tag{C-81}$$

The first order derivative is given by the following three-point finite difference approximations with the same order truncation errors:

- three-point forward difference

$$\frac{\partial u(z)}{\partial z} = \frac{-3u(z) + 4u(z+h) - u(z+2h)}{2h} + O\left(h^2\right) \tag{C-82}$$

- three-point centered difference

$$\frac{\partial u(z)}{\partial z} = \frac{-u(z-h) + u(z+h)}{2h} + O\left(h^2\right) \tag{C-83}$$

- three-point backward difference

$$\frac{\partial u(z)}{\partial z} = \frac{u(z-2h) - 4u(z-h) + 3u(z)}{2h} + O\left(h^2\right) \tag{C-84}$$

The term $O(h^2)$ indicates the remainder which is truncated (*truncation error*) to obtain the approximate derivative. If a better approximation is needed, one could either reduce the size of the mesh or add more information by including higher order neighbors.

In order to simplify the notation, we will label the discretization points with appropriate indices.

$$\frac{\partial u(z_1)}{\partial z} = \frac{-3u(z_1)+4u(z_2)-u(z_3)}{2h} + O(h^2) \tag{C-85}$$

$$\frac{\partial u(z_n)}{\partial z} = \frac{-u(z_{n-1})+u(z_{n+1})}{2h} + O(h^2) \qquad n = 2,3,\cdots,N-1 \tag{C-86}$$

$$\frac{\partial u(z_N)}{\partial z} = \frac{u(z_{N-2})-4u(z_{N-1})+3u(z_N)}{2h} + O(h^2) \tag{C-87}$$

To avoid fictious points, for instance, we use forward difference formula for the first spatial grid point, backward difference for the last point, and centered difference for the all other points, respectively. Since the centered difference involves both neighboring points, there is more balanced information on the local behavior of the function.

We can write Equations (C-85), (C-86), and (C-87) in an alternate and more compact form:

$$\frac{\partial \mathbf{u}}{\partial z} = \frac{1}{2h}\begin{bmatrix} -3 & 4 & -1 \\ -1 & 0 & 1 \\ 1 & -4 & 3 \end{bmatrix}\mathbf{u} + O(h^2) \tag{C-88}$$

Equation (C-88) indicates how the derivative vector $\partial\mathbf{u}/\partial z$ is computed from the dependent variable vector $\mathbf{u}$. The $3\times 3$ matrix is termed a *differentiation matrix*. Note that the elements of the matrix are just the weighting coefficients of Equations (C-85), (C-86), and (C-87).

By including more discretization points, it is also possible to improve the approximation for the first order derivative. The following are five-point

finite difference approximations with higher-order truncation errors:

$$\frac{\partial \mathbf{u}}{\partial z} = \frac{1}{12h}\begin{bmatrix} -25 & 48 & -36 & 16 & -3 \\ -3 & -10 & 18 & -6 & 1 \\ 1 & -8 & 0 & 8 & -1 \\ -1 & 6 & -18 & 10 & 3 \\ 3 & -16 & 36 & -48 & 25 \end{bmatrix}\mathbf{u} + O(h^4) \tag{C-89}$$

Therefore, based on the above three-point or five-point finite difference approximations we can get the following ordinary differential equations for the plug flow element with the following definitions.

$$\frac{\partial u_n}{\partial z} \approx \frac{\Delta u_n}{\Delta z} = \frac{\Delta u_n}{h} \tag{C-90}$$

$$A_n = hC_n \tag{C-91}$$

$$\phi_n = hS_n \tag{C-92}$$

- the component mass balance equations

$$\phi_n \rho_n \frac{dx_{i,n}}{dt} = F_{ci,n} \tag{C-93}$$

$$F_{ci,n} \equiv -F_n(\Delta x_{i,n}) + A_n(\eta_{i,n} - x_{i,n}\eta_n) + \phi_n(R_{i,n} - x_{i,n}R_n) \tag{C-94}$$

- the overall mass balance equations

$$\rho_n \frac{d\phi_n}{dt} + \phi_n\left(\frac{\partial \rho}{\partial T}\right)_n \frac{dT_n}{dt} + \phi_n\left(\frac{\partial \rho}{\partial P}\right)_n \frac{dP_n}{dt} = F_{m,n} \tag{C-95}$$

$$F_{m,n} \equiv \rho_n \sum_{i=1}^{n_c} \{-\Delta(x_{i,n}F_n) + A_n\eta_{i,n} + \phi_n R_{i,n}\}\bar{V}_{i,n} \tag{C-96}$$

- the energy balance equations

$$\phi_n \rho_n C_{P,n} \frac{dT_n}{dt} + \phi_n \rho_n \left(\frac{\partial H}{\partial P}\right)_n \frac{dP_n}{dt} - \xi\left(\phi_n \frac{dP_n}{dt} + P_n \frac{d\phi_n}{dt}\right) = Q_{e,n} \tag{C-97}$$

$$Q_{e,n} = -F_n \sum_{i=1}^{n_c} x_{i,n}\left(\Delta \bar{H}_{i,n}\right) + A_n \sum_{i=1}^{n_c} \eta_{i,n}\lambda_{i,n} - \phi_n \sum_{j=1}^{n_r} r_{j,n} H_{rj,n} + A_n\left(Q_n + q_n\right) \tag{C-98}$$

$$\lambda_{i,n} \equiv \bar{H}^*_{i,n} - \bar{H}_{i,n} \tag{C-99}$$

$$H_{rj,n} \equiv \sum_{i=1}^{n_c} \alpha_{ij} \bar{H}_{i,n} \tag{C-100}$$

- **Liquid Phase System**

In the case of a liquid, we assume that $\partial\rho/\partial P \approx 0$, $\partial H/\partial P \approx 0$ and $\xi \approx 0$. These assumptions reduce Equations (C-97) and (C-95) to standard first-order form in $dT/dt$ and $d\phi/dt$, respectively. Therefore, we get the final equations for the plug flow system of liquid phase as follows:

$$\frac{dx_{i,n}}{dt} = \frac{1}{\phi_n \rho_n} F_{ci,n} \tag{C-101}$$

$$\frac{dT_n}{dt} = \frac{1}{\phi_n \rho_n C_{P,n}} Q_{e,n} \tag{C-102}$$

$$\frac{d\phi_n}{dt} = \frac{1}{\rho_n} F_{m,n} - \frac{\left(\frac{\partial \rho}{\partial T}\right)_n}{\rho_n^2 C_{P,n}} Q_{e,n} \tag{C-103}$$

- **Vapor Phase System**

In the case of a gas, we have $\xi = 1$. Equations (C-95) and (C-97) must then be solved simultaneously for $dT/dt$ and $dP/dt$ to put them in standard first-order form. We need another equation which defines $d\phi/dt$. For example, if there is only one vapor phase in the system of rigid vessel, then $d\phi_{V,n}/dt = 0$ (constraint equation). If there are both vapor phase and liquid phase in the system of rigid vessel, then $d\phi_{T,n}/dt = d\left(\phi_{V,n} + \phi_{L,n}\right)/dt = 0$

(constraint equation). Therefore, we get the final equations for the plug flow system of vapor phase as follows:

$$\frac{dy_{i,n}}{dt} = \frac{1}{\phi_n \rho_n} F_{ci,n} \tag{C-104}$$

$$\frac{dT_n}{dt} = \frac{1}{\phi_n} \frac{\left(1 - \rho \frac{\partial H}{\partial P}\right)_n F_{m,n} + \left(\frac{\partial \rho}{\partial P}\right)_n Q_{e,n} + \xi_{\phi,n} \frac{d\phi_n}{dt}}{\left(\rho C_P \frac{\partial \rho}{\partial P}\right)_n + \left(1 - \rho \frac{\partial H}{\partial P}\right)_n \left(\frac{\partial \rho}{\partial T}\right)_n} \tag{C-105}$$

$$\xi_{\phi,n} \equiv \left(\rho^2 \frac{\partial H}{\partial P} - \rho + P \frac{\partial \rho}{\partial P}\right)_n \tag{C-106}$$

$$\frac{dP_n}{dt} = \frac{1}{\phi_n} \frac{\rho_n C_{P,n} F_{m,n} - \left(\frac{\partial \rho}{\partial T}\right)_n Q_{e,n} - \left(\rho^2 C_P + P \frac{\partial \rho}{\partial T}\right)_n \frac{d\phi_n}{dt}}{\left(\rho C_P \frac{\partial \rho}{\partial P}\right)_n + \left(1 - \rho \frac{\partial H}{\partial P}\right)_n \left(\frac{\partial \rho}{\partial T}\right)_n} \tag{C-107}$$

$$\frac{d\phi_n}{dt} = \frac{d\phi_{V,n}}{dt} = \frac{d\phi_{T,n}}{dt} - \frac{d\phi_{L,n}}{dt} = -\frac{d\phi_{L,n}}{dt} \tag{C-108}$$

Similar to the Well-Mixed Element, the relationship between the model equations and modeling component in Plug Flow Element is shown in Table C-2.

Table C-2 Relationship between Model Equations and Modeling Components in Plug Flow Element

| Modeling Components | Model Equations |
|---|---|
| CoreModels (Conservation Equations) | Equations (C-101), (C-102), (C-104), and (C-105) |
| Connectors (Constitutive Equations) | Equations $q_n$, $\eta_{i,n}$ and $r_{j,n}$ |
| Coordinators (Constraint Equations) | Equations (C-103), (C-107), and (C-108) |

# C.3 Coordinators

## C.3.1 Phase Volume Coordinator

This coordinator is used to determine the volume of the phase. The model equation described by this coordinator is usually $d\phi/dt$.

# C.4 Connectors

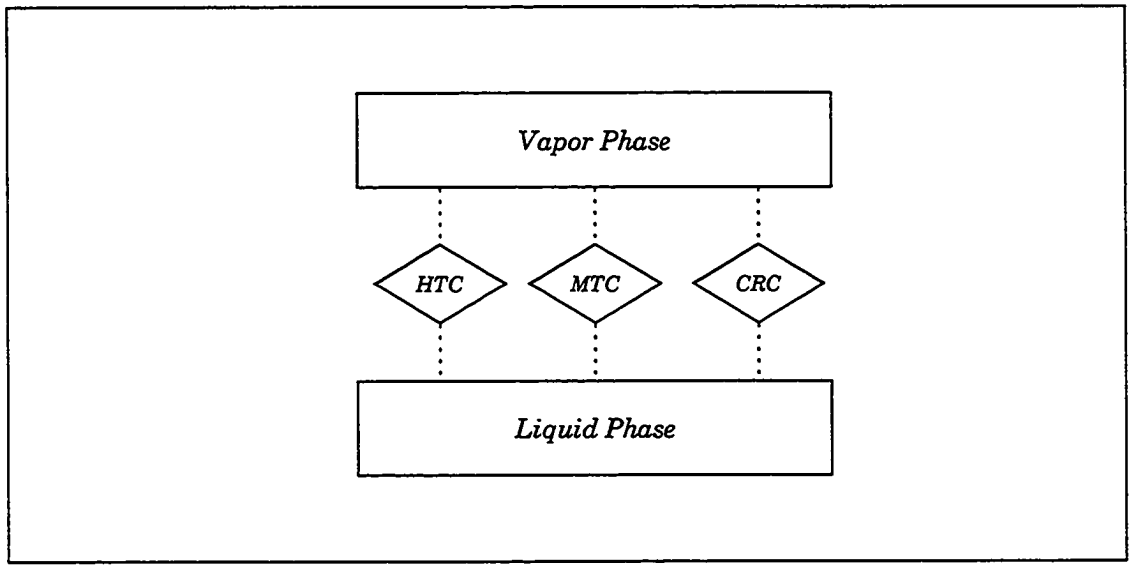



Figure C-3 Connectors between Phases

Let us introduce the following new notation:

$E$ = activated energy of the reaction [kJ/kmol]

$\hat{f}_i^L$ = fugacity of $i^{th}$ component in the liquid mixture [kPa]

$\hat{f}_i^V$ = fugacity of $i^{th}$ component in the vapor mixture [kPa]

$f_i^L$ = fugacity of $i^{th}$ pure component at the liquid temperature

[kPa]

$f_i^V$ = fugacity of $i^{th}$ pure component at the vapor temperature [kPa]

$K_i$ = mass transfer coefficient between phases [$kmol/m^2 \cdot s \cdot kPa$]

$k_0$ = frequency factor of the reaction [$kmol/m^3 \cdot s$]

$q$ = heat transfer rate between phases [$kJ/m^2 \cdot s$]

$R$ = universal gas constant [$kJ/kmol \cdot K$]

$T_L$ = temperature of the liquid phase [K]

$T_V$ = temperature of the vapor phase [K]

$U_h$ = heat transfer coefficient between phases [$kJ/m^2 \cdot s \cdot K$]

$\gamma_i$ = activity coefficient of $i^{th}$ component in the liquid mixture

$\hat{\phi}_i$ = fugacity coefficient of $i^{th}$ component in the vapor mixture

$\eta_i$ = mass transfer rate of $i^{th}$ component [$kmol/m^2 \cdot s$]

### C.4.1 Heat Transfer Connector

This connector is used to determine the heat transfer rate between phases due to the phase temperature difference. The heat transfer rate equations is shown as follows:

$$q = U_h(T_L - T_V) \tag{C-109}$$

### C.4.2 Mass Transfer Connector

This connector is used to determine the mass transfer rate between phases due to the component fugacity difference. The mass transfer rate

equations are shown as follows:

$$\eta_i = K_i\left(\hat{f}_i^L - \hat{f}_i^V\right) \tag{C-110}$$

$$\hat{f}_i^L = \gamma_i x_i f_i^L \tag{C-111}$$

$$\hat{f}_i^V = \hat{\phi}_i y_i P_V \tag{C-112}$$

$$\eta \equiv \sum_{i=1}^{n_c} \eta_i \tag{C-113}$$

## C.4.3 Chemical Reaction Connector

This connector is used to determine the chemical reaction rate within or between phases due to the chemical potential difference of each component. The model equation described by this connector is usually $R_i$.

Consider the $n_r$ independent chemical reactions of $n_c$ chemical species.

$$\sum_{i=1}^{n_{c1}} \alpha_{ij} A_i = \sum_{i=1}^{n_{c2}} \beta_{ij} B_i \qquad j = 1,2,\cdots,n_r \tag{C-114}$$

The macroscopic bulk rate $r_j$ of reaction $j$ may be expressed by

$$r_j = k_0 e^{-E/RT} \prod_{i=1}^{n_{c1}} x_i^{|\alpha_{ij}|} \tag{C-115}$$

The rate of reaction of $i^{th}$ component becomes

$$R_i = \sum_{j=1}^{n_r} \alpha_{ij} r_j \tag{C-116}$$

## C.5 Conclusions

- All model equations being developed here are of index 1 or less, which insure that the system can be reliably integrated using ODEs solvers.
- The set of four types of modeling components is compact and readily comprehensible.

# Notation

## Latin Letters

| | |
|---|---|
| $A$ | mass or heat transfer surface area [$m^2$] |
| $C$ | cross section circumference [m] |
| $C_P$ | heat capacity at constant pressure [kJ/kmol · K] |
| $D$ | diameter of vessel [m] |
| $E$ | activated energy of the reaction [kJ/kmol] |
| $F$ | molar flow rate [kmol/s] |
| $F_0$ | molar flow rate [kmol/s] |
| $\hat{f}_i^L$ | fugacity of $i^{th}$ component in the liquid mixture [kPa] |
| $\hat{f}_i^V$ | fugacity of $i^{th}$ component in the vapor mixture [kPa] |
| $f_i^L$ | fugacity of $i^{th}$ pure component at the liquid temperature [kPa] |
| $f_i^V$ | fugacity of $i^{th}$ pure component at the vapor temperature [kPa] |
| $H$ | molar enthalpy of the material leaving the element [kJ/kmol] |
| $\bar{H}_i$ | partial molar enthalpy of the $i^{th}$ component [kJ/kmol] |
| $\bar{H}_i^*$ | partial molar enthalpy of the $i^{th}$ component crossing the interface [kJ/kmol] |
| $K_i$ | mass transfer coefficient between phases [$kmol/m^2 \cdot s \cdot kPa$] |
| $k_0$ | frequency factor of the reaction [$kmol/m^3 \cdot s$] |
| $k_1$ | forward reaction rate coefficient [$kmol/m^3 \cdot s$] |
| $k_2$ | backward reaction rate coefficient [$kmol/m^3 \cdot s$] |
| $L$ | axial length [m] |
| $n_c$ | number of components |
| $n_r$ | number of reactions |
| $P$ | phase pressure [kPa] |
| $P_L$ | pressure of the liquid phase [kPa] |

| | |
|---|---|
| $P_V$ | real pressure of the vapor phase [kPa] |
| $Q$ | heat input [$kJ/m^2 \cdot s$] |
| $q$ | heat transfer rate between phases [$kJ/m^2 \cdot s$] |
| $R$ | rate of reaction of the system [$kmol/m^3 \cdot s$] |
| $R_i$ | rate of reaction of the $i^{th}$ component [$kmol/m^3 \cdot s$] |
| $r_j$ | reaction rate of the $j^{th}$ chemical reaction [$kmol/m^3 \cdot s$] |
| $S$ | cross section area [$m^2$] |
| $T$ | phase temperature [K] |
| $T_0$ | temperature of inlet flow [K] |
| $T_L$ | temperature of the liquid phase [K] |
| $T_V$ | temperature of the vapor phase [K] |
| $U$ | molar internal energy [kJ/kmol] |
| $U_h$ | overall heat transfer coefficient between phases [$kJ/m^2 \cdot K$] |
| $V$ | phase molar volume [$m^3$/kmol] |
| $\bar{V}_i$ | partial molar volume of the $i^{th}$ component [$m^3$/kmol] |
| $x_i$ | liquid molar fraction. |
| $y_i$ | vapor molar fraction. |
| $z$ | axial distance [m] |

**Greek Letters**

| | |
|---|---|
| $\alpha_{ij}$ | stoichiometric coefficient |
| $\gamma_i$ | activity coefficient of $i^{th}$ component in the liquid mixture |
| $\varepsilon$ | void fraction |
| $\phi$ | phase volume [$m^3$] |
| $\hat{\phi}_i$ | fugacity coefficient of $i^{th}$ component in the vapor mixture |
| $\phi_L$ | liquid phase volume [$m^3$] |
| $\phi_T$ | volume of the vessel [$m^3$] |

| | |
|---|---|
| $\phi_V$ | vapor phase volume [$m^3$] |
| $\eta_i$ | mass transfer rate of $i^{th}$ component [$kmol/m^2 \cdot s$] |
| $\eta$ | rate of mass transfer crossing the interface [$kmol/m^2 \cdot s$] |
| $\rho$ | molar density [$kmol/m^3$] |

**Superscripts**

| | |
|---|---|
| *L* | liquid phase |
| *V* | vapor phase |

**Subscripts**

| | |
|---|---|
| *L* | liquid phase |
| *V* | vapor phase |

**Abbreviations**

| | |
|---|---|
| *4C* | Containments, Core Models, Connectors, and Coordinators |
| *AEs* | Algebraic Equations |
| *CRC* | Chemical Reaction Connector |
| *CSTR* | Continuous Stirred Tank Reactor |
| *DAEs* | Differential Algebraic Equations |
| *DIPPR* | Design Institute for Physical Properties |
| *GUI* | Graphical User Interface |
| *HTC* | Heat Transfer Connector |
| *HTML* | Hypertext Markup Language |
| *IMSL* | International Mathematical and Statistical Libraries |
| *MTC* | Mass Transfer Connector |
| *ODEs* | Ordinary Differential Equations |
| *OOD* | Object-Oriented Design |

| | |
|---|---|
| *OOP* | Object-Oriented Programming |
| *PDEs* | Partial Differential Equations |
| *PFE* | Plug Flow Element |
| *PVC* | Phase Volume Coordinator |
| *SM* | Stream Mixer |
| *SS* | Stream Splitter |
| *SSM* | Substream Mixer |
| *SSS* | Substream Splitter |
| *TFR* | Tubular Flow Reactor |
| *VFC* | Void Fraction Coordinator |
| *W3C* | World Wide Web Consortium |
| *WME* | Well-Mixed Element |
| *XML* | eXtensible Markup Language |
| *ZVM* | Zero Volume Mixer |

# References


[1] Bär, M. & Zeitz, M. (1990), A Knowledge-Based Flowsheet-Oriented User Interface for a Dynamic Process Simulator, *Computers and Chemical Engineering*, **14**, 1275–1280.

[2] Barton, P.I. & Pantelides, C.C. (1994), Modeling of Combined Discrete/Continuous Processes, *AIChE Journal*, **40**, 966–979.

[3] Bird, R.B., Stewart, W.E. & Lightfoot, E.N. (1960), *Transport Phenomena*, John Wiley & Sons, New York. ISBN 0-471-07395-4.

[4] Bogusch, R. & Marquardt, W. (1997), A Formal Representation of Process Model Equations, *Computers and Chemical Engineering*, **21**, 1105–1115.

[5] Bogusch, R., Lohmann, B. & Marquardt, W. (2001), Computer-aided Process Modeling with ModKit, *Computers and Chemical Engineering*, **25**, 963–995.

[6] Boston, J.F., Britt, H.I. & Tayyabkhan, M.T. (1993), Software: Tackling Tougher Tasks, *Chemical Engineering Progress*, **Nov**, 38-49.

[7] Brenan, K.E., Campbell, S.L. & Petzoid, L.R. (1996), *Numerical Solutions of Initial-Value Problems in Differential-Algebraic Systems*, North-Holland, New York. ISBN 0-898-71353-6.

[8] Byrne, G.D. & Ponzi, P.R. (1988), Differential-Algebraic Systems, Their Application and Solutions, *Computers and Chemical Engineering*, **12**, 377–382.

[9] Date, C.J. (1995), *An Introduction to Database Systems*, 6th edition, Addison-Wesley, New York. ISBN 0-201-54329-X.

[10] Deuflhard, P., Nowak, U. & Wulkow, M. (1990), Recent Developments in Chemical Computing, *Computers and Chemical Engineering*, **14**,

1249–1258.

[11] Dieterich, E.E. & Eigenberger, G. (1997), The ModuSim Concept for Modualr Modeling and Simulation in Chemical Engineering, *Computers and Chemical Engineering*, **21**, S805–S809.

[12] Dieterich, E.E., Salden, A., Schäfer, J., Schmidt, J. & Eigenberger, G. (1997), Computer Aided Modeling with BIMAP, *Computers and Chemical Engineering*, **21**, 1191–1201.

[13] Drengstig, T., Wasbo, S.O. & Foss, B.A. (1997), A Formal Graphical Based Process Modeling and Methodology, *Computers and Chemical Engineering*, **21**, S835–S840.

[14] Fowler, M. & Scott, K. (2000), *UML Distilled*, 2nd edition, Addison Wesley Longman Inc. Reading, MA. ISBN 0-201-65783-X.

[15] Franks, R.G.E (1972), *Modeling and Simulation in Chemical Engineering*, John Wiley & Sons, New York. ISBN 0-471-27535-2.

[16] Friedly, J.C. (1972), *Dynamic Behavior of Processes*, Prentice-Hall, Englewood Cliffs, New Jersey. ISBN 0-13-221242-0.

[17] Froment, G.F. & Bischoff, K.B. (1990), *Chemical Reactor Analysis and design*, 2nd edition, John Wiley & Sons, New York. ISBN 0-471-51044-0.

[18] Gani, R., Sørensen, E.L. & Perregaard, J. (1992), Design and Analysis of Chemical Processes through DYNSIM, *Ind. Eng. Chem. Res.*, **31**, 244-254.

[19] Gear, C.W. (1971), *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall, Englewood Cliffs, New Jersey. ISBN 0-136-26606-1.

[20] Gosling, J., Joy, B., Steele, G. & Bracha, G. (2000), *The Java Language Specification*, 2nd edition, Addison-Wesley, New York. ISBN 0-201-

31008-2.

[21] Hangos, K.M. & Cameron, I.T. (1997), The Formal Representation of Process Systems Assumption and Their Implications, *Computers and Chemical Engineering*, **21**, S823–S828.

[22] Hangos, K.M. & Cameron, I.T. (2001), A Formal Representation of Assumption in Process Modeling, *Computers and Chemical Engineering*, **25**, 237–255.

[23] Hillestad, M., Sorlie, C., Anderson, T.F., Olsen, I. & Hertzberg, T. (1989), On Estimating the Error of Local Thermodynamic Models – A General Approach, *Computers and Chemical Engineering*, **13**, 789–796.

[24] Hofmeister, M. (1998), BatchKit – A knowledge Integration Environment for Process Engineering, *Computers and Chemical Engineering*, **22**, 109–123.

[25] Holl, P., Marquardt, W. & Gills, E.D. (1988), DIVA – A Powerful Tool for Dynamic Process Simulation, *Computers and Chemical Engineering*, **12**, 421–426.

[26] Huang, S. (1996), *Model Fidelity Considerations in the Dynamic Simulation of Equilibrium Staged Separation Operations*; Ph.D. Thesis, The City University of New York, New York.

[27] Jarke, M. & Marquardt, W. (1996), Design and Evaluation of Computer-Aided Process Modeling Tools, *First International Conference on Intelligent Systems in Process Engineering*, *AIChE Symposium Series*, **92**(312), 97–109.

[28] Jensen, A.K. & Gani, R. (1999), A Computer-Aided Modeling System, *Computers and Chemical Engineering* **23**, S673–S678.

[29] Jensen, A.K. & Gani, R. (1998), Development of Novel Unit Processes: Review of Computer Aided Modeling Tools, *CAPE-NET* Technical

Working Group.

[30] Kröner, A., Holl, P., Marquardt, W. & Gilles, E.D. (1990), DIVA – An Open System for Dynamic Process Simulation, *Computers and Chemical Engineering*, **14**, 1289–1295.

[31] Lefkopoulos, A. & Stadtherr, M.A. (1993a), Index Analysis of Unsteady-State Chemical Process Systems – I. An Algorithm for Problem Formulation, *Computers and Chemical Engineering*, **17**, 399–413.

[32] Lefkopoulos, A. & Stadtherr, M.A. (1993b), Index Analysis of Unsteady-State Chemical Process Systems – II. Strategies for Determining the Overall Flowsheet Index, *Computers and Chemical Engineering*, **17**, 415–430.

[33] Levenspiel, O. (1972), *Chemical Reaction Engineering*, 2nd edition, John Wiley & Sons, New York. ISBN 0-471-53016-6.

[34] Lohmann, B. & Marquardt, W. (1996), On the Systematization of the Process of Model Development, *Computers and Chemical Engineering*, **20**, S213–S218.

[35] Lund, P.C. (1992), *An Object-Oriented Environment for Process Modeling and Simulation*; Ph.D. Thesis, University of Trondheim, Norway.

[36] Marquardt, W. (1996), Trends in Computer-Aided Process Modeling, *Computers and Chemical Engineering*, **20**, 591–609.

[37] Marquardt, W. (1992), An Object-Oriented Representation of Structured Process Models, *Computers and Chemical Engineering*, **16**, S329–S336.

[38] Marquardt, W. (1991), Dynamic Process Simulation – Recent Progress and Future Challenges, *Chemical Process Control CPC-IV*, *AIChE*,

New York, 131–180.

[39] McCabe, W.L, Smith, J.C. & Harriott, P. (1993), *Unit Operations of Chemical Engineering*, 5th edition, McGraw-Hill, New York. ISBN 0-07-044844-2.

[40] McCarty, B. & McCarthy, B. (1997), *SQL Database Programming with Java*, The Coriolis Group. ISBN 1-57610-1762.

[41] Nilsson, B. (1996), Experiences of Developing Process Model Libraries in OMOLA, *First International Conference on Intelligent Systems in Process Engineering, AIChE Symposium Series*, **92**(312), 388-392.

[42] Oh, M. & Pantelides, C.C. (1996), A Modeling and Simulation Language for Combined Lumped and Distributed Parameter Systems, *Computers and Chemical Engineering*, **20**, 611–633.

[43] Pantelides, C.C. (1988), SPEEDUP – Recent Advances in Process Simulation, *Computers and Chemical Engineering*, **12**, 745–755.

[44] Pantelides, C.C. & Barton, P.I. (1993), Equation-Oriented Dynamic Simulation – Current Status and Future Perspectives, *Computers and Chemical Engineering*, **17**, S263–S285.

[45] Pantelides, C.C. & Britt, H.I. (1994), Multipurpose Process Modeling Environments, *Foundations of Computer-Aided Process Design (FOCAPD), AIChE Symposium Series*, **91**(304), 128–141.

[46] Pantelides, C.C., Gritsis, E., Morison, K.R., & Sargent R.W.H. (1988), The Mathematical Modeling of Transient Systems Using Differential-Algebraic Equations, *Computers and Chemical Engineering*, **12**, 449–454.

[47] Perkins, J.D. (1984), Equation-Oriented Flowsheeting, *Proceedings of the Second International Conference On Foundations of Computer-Aided Process Design (FOCAPD), AIChE Symposium Series*, 309–367.

[48] Perkins, J.D & Sargent, R.W.H. (1982), SPEEDUP – A Computer Program for Steady-State and Dynamic Simulation and Design of Chemical Processes, *AIChE Symposium Series*, **78**(214), 1–11.

[49] Perkins J.D., Sargent, R.W.H., Vázquez-Román, R. & Cho, J.H. (1996), Computer Generation of Process Models, *Computers and Chemical Engineering*, **20**, 635–639.

[50] Perry, R.H., Green, D.W. & Maloney, J.O. (1984), *Perry's Chemical Engineers' Handbook*, 6th edition, McGraw-Hall, New York, ISBN 0-07-049479-7.

[51] Pièla, P.C., Epperly, T.G., Westerberg, K.M., & Westerberg, A.W. (1991), ASCEND – An Object-Oriented Computer Environment for Modeling and Analysis: The Modeling Language, *Computers and Chemical Engineering*, **15**, 53–72.

[52] Ponton J.W., & Gawthrop, P.J. (1991), Systematic Construction of Dynamic Models for Phase Equilibrium Processes, *Computers and Chemical Engineering*, **15**, 803–808.

[53] Preisig, H.A. (1995), MODELLER – An Object-Oriented Computer-Aided Modeling Tool, *Fourth International Conference on Foundations of Computer-Aided Process Design (FOCAPD), AIChE Symposium Series*, **91**(304), 328–331.

[54] Press, W.H., Teukolsky, S.A., Vetterling, W.T. & Flannery, B.P. (1992), *Numerical Recipes in C, The Art of Scientific Computing*, 2nd edition, Cambridge University Press. ISBN 0-521-43108-5.

[55] Preston, A.J., & Berzins, M. (1991), Algorithms for the Location of Discontinuities in Dynamic Simulation Problems, *Computers and Chemical Engineering*, **15**, 701–713.

[56] Ramirez, W.F. (1989), *Computational Methods for Process Simulation*,

Butterworths, Boston. ISBN 0-409-90184-9.

[57] Rinard, I.H. (1998), *Dynamic Modeling for Process Systems Engineering*, Department of Chemical Engineering, The City College of The City University of New York.

[58] Rinard, I.H. (1996), Core Models, Coordinators, and Connectors in the Dynamic Modeling and Simulation of Multiphase Systems, *Computers and Chemical Engineering*, **20**, S969–S974.

[59] Rinard, I.H. (1987), Sensitivity to Modeling Errors in Steady-State Progress Simulation, *Computers and Chemical Engineering*, **11**(6), 707–712.

[60] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. & Lorensen, W. (1991), *Object-Oriented Modeling and Design*, Prentice Hall, Englewood Cliffs, New Jersey. ISBN 0-13-630054-5.

[61] Schiesser, W.E. (1991), *The Numerical Method of Lines: Integration of Partial Differential Equations*, Academic Press, New York. ISBN

[62] Seinfeld, J.H. & Lapidus, L. (1974), *Mathematical Methods in Chemical Engineering, Process Modeling, Estimation and Identification*, Volume 3, Prentice-Hall, Englewood Cliffs, New Jersey. ISBN 3-540-65225-6.

[63] Shacham, M., Macchietto, S., Stutzman, L.F., & Babcock, P. (1982), REVIEW – Equation-Oriented Approach to Process Flowsheeting, *Computers and Chemical Engineering*, **6**, 79–95.

[64] Silebi, C.A. & Schiesser, W.E. (1992), *Dynamic Modeling of Transport Process Systems*, Academic Press, New York. ISBN 0-126-43420-4.

[65] Slattery, J.C. (1999), *Advanced Transport Phenomena*, Cambridge University Press, New York. ISBN 0-521-63565-9.

[66] Sørlie, C.F. (1990), *ModAss – An Intelligent Modeling Assistant*; Ph.D.

Thesis, University of Trondheim, Norway.

[67] Smith, J.M. & Van Ness, H.C. (2000), *Introduction to Chemical Engineering Thermodynamics*; 6th edition, McGraw-Hill, New York. ISBN 0-072-40296-2.

[68] Stephanopoulos, G., Henning, G., & Leone, H. (1990), MODEL.LA: A Modeling Language for Process Engineering, Part I and Part II, *Computers and Chemical Engineering*, **14**, 813–869.

[69] Stephanopoulos, G., Johnston, J., Kriticos, T., Lakshmanan, R., Mavrovouniotis, M., & Siletti, M. (1987), DESIGN-KIT: An Object-Oriented Environment for Process Engineering, *Computers and Chemical Engineering*, **11**, 655–674.

[70] Stephanopoulos, G. (1987), Artificial Intelligence in Process – Current State and Future Trends, *Computers and Chemical Engineering*, **11**, 1259–1270.

[71] Tränkle, F., Zeitz, M., Ginkel, M. & Gilles, E.D. (2000), PROMOT – A Modeling Tool for Chemical Process, *Mathematical and Computer Modeling of Dynamical Systems*, **6**(3), 283–307.

[72] Tyreus, B.D. (1992), Object-Oriented Simulation, *Practical Distillation Control*, W.L. Luyben, Ed., Van Nostrand Reinhold, New York. ISBN 0-442-00601-2.

[73] Unger, J., Kröner, A., & Marquardt, W. (1995), Structural Analysis of Differential-Algebraic Equation Systems – Theory and Applications, *Computers and Chemical Engineering* **19**, 867–882.

[74] Vogel, E.F. (1991), An Industrial Perspective on Dynamic Flowsheet Simulation, *Chemical Process Control CPC-IV, CACHE/AIChE*, New York, 181–208.

[75] Walas, S.M. (1988), *Chemical Process Equipment*, Butterworths,

Boston. ISBN 0-409-90131-8.

[76] Warren, N. & Bishop, P. (1999), *Java in Practice*: *Design Styles and Idioms for Effective Java*, Addison-Wesley, New York, ISBN 0-201-36065-9.

[77] Westerberg, A.W. & Benjamin, D.R. (1985), Thoughts on a Future Equation-Oriented Flowsheeting System, *Computers and Chemical Engineering*, **9**(5), 517–526.

[78] Weiss, M., & Preisig, H.A. (1997), Simplifying Hypotheses in Computer-Aided Modeling: A Singular Perturbation Approach, *Computers and Chemical Engineering*, **21**, S721–S726.