A Computer Tool for the Generation of Process Models from a Physical Description

Patrick James Calvert

A thesis submitted for the degree of
Doctor of Philosophy of the University of London
and for the Diploma of Membership of Imperial College

Centre for Process Systems Engineering
Department of Chemical Engineering and Chemical Technology
Imperial College
London
SW7 2BY

www.ps.ic.ac.uk

April 2004

BSL
LONDON

The pessimist sees difficulty in every opportunity.
The optimist sees opportunity in every difficulty.

Winston Churchill (1874 – 1965)

Abstract

The mathematical modelling of processes has proved indispensable at all stages of process development, from process synthesis and design, to process control and the training of plant operators. The continual increase in computation power has enhanced both the diversity and the potential benefits of process modelling. Despite these improvements, the time-consuming, difficult and error-prone task of formulating consistent and well-posed mathematical models has prevented the engineer from reaping the full benefit of process modelling. Today, the engineer rarely has the capability to build models from scratch. Instead, process models are assembled from existing models, which have previously been written for individual process units. Whilst this has proved satisfactory for many of the traditional chemical processes, the current drives towards the manufacturing of fine chemicals and the use of novel processes have revealed the lack of generality, and the consequent restricted sphere of application of library models.

There is therefore a commercial need for a more intelligent modelling environment that avoids the restrictions associated with a library model yet remains accessible to the engineer. The research described herein is concerned with the development of *COMPASS* (COMPuter Aided Simulation Software), a prototype piece of software that demonstrates the potential of such an environment. The conceptual basis behind *COMPASS* is centred on the generation of process models from a purely physical description. By using a physical-based model-generating tool, the user can construct models for a wide variety of different processes without any detailed modelling knowledge.

Based upon a two-tier foundation of structural representation and phenomenological characterisation, *COMPASS* is able to automatically generate mathematical models for one- and two-phase systems. The structural representation enables a given process to be decomposed into an interconnected network of one- and two- phase regions, whilst the phenomenological characterisation provides a physically-based, systematic approach to which the physical description and the model generating algorithms can be structured around. The validity of *COMPASS* is assessed with several case studies.

Acknowledgements

The work presented in this thesis is an accumulation of the ideas and opinions of many, who unselfishly have given so much of their time to ensure that the research was a success. In particular I am indebted to the support of my 4 x 1 year supervisory relay team, comprising of Professor John Perkins, Professor Roger Sargent, Dr. Steven Asprey and Professor Lawrence. Collectively, their expert advice has shaped the direction of the research for the better and I hope that they can see evidence of their ideas reflected in the work.

I would also like to thank Dr. Mike Colclough, for his help in presenting and marketing the research, Professor Nilay Shah, for acting as co-supervisor and Terrence Crombie, for his most helpful support with all issues relating to the computer programming.

Finally, I would like to acknowledge the financial support from ABB.

Contents

Abstract

Acknowledgements

1. Introduction **15**

1.1 Motivation and Objectives 16

1.2 Computer-Aided Process Modelling 17

1.2.1 Important Aspects of Process Modelling 17

1.2.2 Computer-Aided Process Modelling Software 19

1.2.3 Computer Aided Model Generation Software 23

1.3 Conclusions and Objectives 32

2. An Introduction to the Mathematical Modelling of Process Systems **34**

2.1 An Introduction to Mathematical Modelling 34

2.2 An Introduction to Process Systems Modelling 35

2.2.1 The Physical Mechanisms that Influence the State of the Process and the Properties of Interest 35

2.2.2 A Thermodynamic Basis for Process Modelling 38

2.2.3 A Fluid Dynamic Basis for Process Modelling 40

2.3 A Modelling Strategy Based on Process Decomposition 40

2.31 Modelling a Lumped Single-Phase Region 41

2.32 Modelling a Distributed Single-Phase Region 41

2.33 Modelling a Two-phase System 42

2.4 Summary 48

3. The Modelling Framework **49**

3.1 The Two-Tier Framework 49

3.2 The Structural Representation 50

3.3 The Physicochemical Characterisation of the Zone 54

3.3.1 The Zone Class Structure 54

3.3.2 The Subclass Structure 56

3.4 Summary 59

4. The Modelling Language **61**

4.1 Overview of the Modelling Language 61

4.2 The Phase Section 64

4.3 The Vessel Section 66

4.4 The Zone Section 66

4.4.1 The Zone Class Subsection 67

4.4.2 The Zone Structure Subsection 67

4.4.3 The Physicochemical Behaviour Subsection 70

4.5 The Reservoir Section 81

4.6 The Connection Section 82

4.7 The Flowsheet Section 84

4.8 An Example of the Process Modelling Language 85

4.9 Summary 85

5. The Model Building Procedure **88**

5.1 Model Building Strategy 88

5.2 The Model Building Strategy for the Vessel Entity 89

5.3 The Model Building Strategy for the Zone Entity 91

5.3.1 The Internal Connection Subclass 96

5.4 The Model Building Strategy for the Reservoir Entity 99

5.5 The Model Building Strategy for the Connection Entity 99

5.5.1 The Open Material Connection Model Class 99

5.5.2 The Open Energy Connection Model Class 102

5.5.3 The Closed Material Connection and Closed Energy Connection Model Classes 103

5.5.4 The Pressure Connection Model Class 103

5.6 Summary 103

6. Implementation of the Computer Aided Model Generator **107**

6.1 The Translator 107

6.1.1 The Lexical Analyser 107

6.1.2 The Parser 108
6.2 The Model Building Engine 109
6.2.1 Building the Internal Representation 110
6.2.2 Determining the Validity of the Internal Representation 114
6.2.3 Generating the Process Model 115
6.3 Program Execution 116
6.4 Summary 116

7. Assessing the Validity and Implementation of the Framework 119
7.1 A Buffer Tank 120
7.1.1 *gPROMS* Model Development 122
7.1.2 *COMPASS* Model Development 123
7.1.3 Model Assessment 124
7.2 A Gas-Phase Continuous Stirred Tank Reactor 127
7.2.1 *gPROMS* Model Development 128
7.2.2 *COMPASS* Model Development 128
7.2.3 Model Assessment 129
7.2.4 Model Extension 131
7.3 An Adiabatic Flash Vessel 132
7.3.1 *gPROMS* Model Development 133
7.3.2 *COMPASS* Model Development 133
7.3.3 Model Assessment 135
7.4 A Gas-Phase Plug Flow Reactor 138
7.4.1 *gPROMS* Model Development 139
7.4.2 *COMPASS* Model Development 139
7.4.3 Model Assessment 140
7.4.4 Model Extension 142
7.5 A Drum Boiler 144
7.5.1 Modelling Strategy and Model Assumptions 145
7.5.2 *COMPASS* Model Development 148
7.5.3 Model Assessment 150
7.6 An Adsorptive Reactor 152
7.6.1 Modelling Strategy and Model Assumptions 153

7.6.2 *COMPASS* Model Development 156
7.6.3 Model Assessment 158
7.7 Summary 159

8. Conclusions **161**
8.1 Suggestions for Future Work 163
8.1.1 Strategic Development: A New Generation of Unit-Based Process Modelling Software 163
8.1.2 Broadening the Scope of *COMPASS* – Reducing its Current Limitations 165

9. References **168**

A Nomenclature **177**
A.1 Mathematical Notation 177
A.2 Notation Used in *COMPASS* 179
A.2.1 *COMPASS* Variable and Parameter Symbols 181

B The No Extent Zone Model Class **183**
B.1 The Port Subclass 183
B.2 The Mass Flowrate Subclass 184
B.3 The Composition Subclass 185
B.4 The Pressure Subclass 185
B.5 The Temperature Subclass 186

C The Single-Phase Lumped Zone Model Class **189**
C.1 The Port Subclass 190
C.2 The Extent Subclass 190
C.3 The Composition Subclass 192
C.4 The Temperature Subclass 193
C.5 The Pressure Subclass 196

D The Single-Phase, One-Dimensional, Duct Zone Model Class **198**
D.1 The Port Subclass 198

D.2 The Core Equations Subclass 200
D.3 The Extent Subclass 200
D.4 The Composition Subclass 200
D.5 The Temperature Subclass 202
D.6 The Pressure and Velocity Subclass 205

E The Two-Phase, Lumped Zone Model Class 209
E.1 The Port Subclass 209
E.2 The Core Equations Subclass 211
E.3 The Extent Subclass 212
E.4 The Composition Subclass 213
E.5 The Temperature Subclass 215
E.6 The Pressure Subclass 218
E.7 The Volume Fraction Subclass 218
E.8 The Velocity Subclass 219

F The Two-Phase, One-Dimensional Duct Zone Model Class 224
F.1 The Port Subclass 224
F.2 The Core Equations Subclass 225
F.3 The Extent Subclass 226
F.4 The Composition Subclass 226
F.5 The Temperature Subclass 228
F.6 The Pressure and Velocity Subclass 232
F.7 The Volume Fraction Subclass 238

G Derivation of the Normalising Transformations 240

H Bison Input Specification 242

I Parser Input Specification 245

J The *COMPASS* Input Files 258
J.1 The *COMPASS* Input File for the Buffer Tank Simulation 258
J.2 The *COMPASS* Input File for the Gas Phase Continuous Stirred Tank

Reactor Simulation 261
J.3 The *COMPASS* Input File for the Adiabatic Flash Vessel Simulation 262
J.4 The *COMPASS* Input File for the Plug Flow Reactor Simulation 264
J.5 The *COMPASS* Input File for the Drum Boiler Simulation 266
J.6 The *COMPASS* Input File for the Adsorptive Reactor Simulation 269

K The Generated Models 271
K.1 The Buffer Tank System Model 271
K.2 The Gas Phase Continuous Stirred Tank Reactor System Model 281
K.3 The Adiabatic Flash Vessel Model 286
K.4 The Gas Phase Plug Flow Reactor System Model 294
K.5 The Drum Boiler Model 301
K.6 An Absorptive Reactor 311

List of Figures

2.1 A Hierarchical Modelling Strategy for a Complex Chemical Process 36

2.2 Two-Phase Flow Patterns in Horizontal and Vertical Two-Phase Flow 43

2.3 Averaged Model Formulation 47

3.1 The Structural Representation for a Heated Tank 53

3.2 The Description of a Heated Tank, Liquid CSTR and Gas CSTR written in Terms of the *COMPASS* Language 58

4.1 A Diagram of the Settling Tank and the Corresponding Structural Representation 85

5.1 The Hierarchical Model Building Approach Used within *COMPASS* 90

5.2 Mass Fraction Profile between a Lumped Zone and a 1-D Zone with Diffusion 102

6.1 The Unified Modelling Language Diagram for the Internal Representation 112

6.2 The Unified Modelling Language Diagram for the Formulation of the Mathematical Model 118

7.1 A Diagrammatical Representation of the Buffer Tank System 121

7.2 The Structural Representation of the Buffer Tank System 123

7.3 The Physical Description for Tank T101 Declared in terms of the *COMPASS* Language 124

7.4 Variation in the Liquid Volume within Tank T101 with Time 125

7.5 Variation in the Mass Flowrate through Pipe P104 with Time 126

7.6 Variation in the Mass Flowrate through Pipe P105 with Time 126

7.7 Schematic Diagram of the Reactor Drum Process 127

7.8 The Structural Representation of the Reactor Drum Process 128

7.9 The Physical Description for the Reactor Declared in terms of the *COMPASS* Language 129

7.10 Variation in the Outlet Mass Fraction of Propylene with Time 130

7.11 Variation in the Reactor Temperature with Time 130

7.12 Variation in the Reactor Pressure with Time 131

7.13 Variation in the Outlet Mass Fractions of Cyclopropane and Propylene for Isothermal Operations at 850K 132
7.14 A Diagrammatical Representation of the Flash Vessel 132
7.15 The Structural Representation of the Flash Vessel 134
7.16 The Physical Description for the Flash Vessel Declared in terms of the *COMPASS* Language 135
7.17 Variation of the Temperature in the Flash Vessel with Time 136
7.18 Variation of the Pressure in the Flash Vessel with Time 136
7.19 Mass Fraction of Propane within the Liquid Phase of the Flash Vessel 137
7.20 Mass Fraction of Pentane within the Gas Phase of the Flash Vessel 137
7.21 The Plug Flow Reactor System 138
7.22 The Structural Representation of the Plug Flow Reactor 139
7.23 The Physical Description for the PFR Declared in Terms of the *COMPASS* Language 140
7.24 Axial Variation in the Steady State Mass Fraction of 2-Butyne 141
7.25 Axial Variation in the Steady State Temperature of the Reactor 141
7.26 A Diagrammatical Representation of the Plug Flow Reactor System 142
7.27 Steady State Temperature Profile along the Reactor Length 143
7.28 Difference in the Results for the Temperature for the *gPROMS* Model Library and COMPASS Generated Models 144
7.29 The Drum Boiler 144
7.30 The Structural Representation of the Drum Boiler 148
7.31 The Physical Description of the Riser Tube 149
7.32 Variation of the Drum Pressure with Time 151
7.33 Variation in the Mass Flowrate Through the Downcomer with Time 151
7.34 Variation in the Volume Fraction of Steam along the Length of the Boiling Tube after 0, 200 and 1300 Seconds 152
7.35 The Structural Representation of the Adsorptive Reactor 155
7.36 The Physical Description for the Adsorptive Reactor 157
7.37 Axial Variation in the Gaseous Mass Concentration of Toluene at Various Time Steps 158
7.38 Axial Variation in the Solid Molar Concentration of Toluene 159

8.1 Embedding COMPASS with a flowsheeting package 165

List of Tables

2.1 The State Variables for the Single-Phase Regions and the Two-Phase System 48

3.1 The Class and Subclass Structure 57

4.1 The Attributes for the No Extent Zone Class 72
4.2 The Attributes for the Single-Phase Lumped Zone Class 73
4.3 The Attributes for the Single-Phase 1-D Duct Zone Class 74
4.4 The Attributes for the Two-Phase Lumped Zone Class 76
4.5 The Attributes for the Two-Phase 1-D Duct Zone Class 80
4.6 The Different Connections 83

5.1 The Model Classes and Model Subclasses 95
5.2 Correlations for the Interfacial Area in Vertical Pipe Flow 98
5.3 The Equations for the Open Material Connection Model Class 105
5.4 The Equations for the Closed Material Connection and Closed Energy Connection Model Classes 106

7.1 The Flow Coefficients 121
7.2 The *gPROMS* Model Library Components Used to Model the Buffer Tank 123
7.3 The *gPROMS* Model Library Components Used to Model the CSTR 128
7.4 The Flow Coefficients for Pipes P102 and P103 133
7.5 The *gPROMS* Model Library Components Used to Model the Flash Vessel 133
7.6 The *gPROMS* Model Library Components Used to Model the Reactor System 139
7.7 Design Parameters and Operating Conditions for the Drum Boiler 146
7.8 The Controller Parameters for the Drum Boiler 148
7.9 Design Parameters and Operating Conditions for the Adsorptive Reactor 155
7.10 The CPU Times (s) for the Solution of the *gPROMS* Model Library and *COMPASS* Generated Models 160

A.1 The Pathname Order in which the Identifiers are Expressed 180

E.1 The Different Flow Scenarios 219

E.2 Correlations for the Drift Flux Velocity and the Mass Distribution Parameter for Different Flow Regimes as Proposed by Ishii (1977) 222

Chapter 1

Introduction

A mathematical model is commonly defined as the representation of a real-world process by a set of mathematical relations. Factors that determine the behaviour of the process are identified and related together through a set of mathematical equations. The solution of the mathematical model enables real world problems to be simulated in a virtual environment, allowing investigative work to be carried out that could otherwise be too expensive or too dangerous to perform. Mathematical modelling has proved crucial in a number of scientific disciplines from meteorology to quantitative finance. This thesis is concerned with process modelling, where the behaviour of process plants is described in terms of the physical and chemical phenomena that occur within them and the control and operating procedures that dictate their performance.

Over the last fifty years, mathematical modelling has transformed the chemical and process industries. Since the first electronic computer was constructed at UMIST in 1948, the exponential increase in computational power has enhanced both the scope for application and the potential benefits of mathematical modelling in the chemical and process industries. The solution of the mathematical equations, which underpin the discipline, is no longer a laborious task. Whilst pioneers of chemical engineering took days to solve rigorous engineering calculations, today's engineer is armed with an array of software and hardware that will perform all the necessary calculations at the click of a button. Design information and chemical data that once occupied several manuals are now routinely included in software tools that fits onto a single CD. Not only do these software tools contain all the information, they also help to ensure that the information in used in an appropriate manner.

Within the chemical and process sectors, the insight obtained from mathematical modelling has proved to be indispensable in comparing processes, supplementing human knowledge and enhancing the understanding of processes. Applications of mathematical models include the training of plant operators, troubleshooting of existing plant problems, optimising chemical processes, selection of the control system, scale up and commissioning. With stringent

environmental legislation and tough health and safety regulations, the use of mathematical modelling is essential in finding the best practical solution that does not entail excessive cost.

The rapid development of both computer hardware and modelling software has been met by the demands that industry places on mathematical modelling. The tightening of plant performance specifications, the strengthening of safety and environmental legislation and the constant drive for improved cost effectiveness means that engineers must frequently turn towards modelling software to help with their engineering needs. Modelling software must therefore continue to evolve to match the demands of the engineer by enhancing both the scope of application and the fidelity of the mathematical modelling.

1.1 Motivation and Objective

Since mathematical modelling represents the translation of a real world problem into a mathematical problem, the art of modelling lies in the careful selection of the level of detail that is encapsulated within the model. Frequently the continuum seamless web of life makes the exact abstraction of a precise model impossible and thus any attempt to search for it is futile. There is no single correct answer but several feasible models could be obtained whose adequacy depends on the success of isolating the aspects of the problem that are important and ignoring those that are unimportant. The lack of a "correct" approach makes modelling an artisan subject which relies on the experience and inspiration of the modeler to select the model assumptions and equations that encapsulate the appropriate level of detail and accuracy.

A lack of modelling experience results in two major difficulties: the selection of the correct equations that describe the physical and chemical behaviour of the system and the mathematical solution of the set of equations. Extensive work has been carried out to establish *equation oriented modelling languages* that provide the user with the facility firstly to declare the equations and secondly to solve them using a number of pre-programmed numerical methods. Whilst automating the numerical solution of the model has made modelling more accessible, the formulation of consistent models for these applications is a time-consuming, difficult and error-prone task (Perkins et al. 1996). Attempts to alleviate the difficulties associated with writing the necessary equations have focused on libraries that contain pre-written models for standard items, (e.g. Evans, 1979). Any advantage generated by the pre-written library is restricted by the

limitation that the process must match the physico-chemical conditions assumed in the derivation of the library models.

The need to automatically generate process models without reference to pre-written process-unit models provides the main motivation behind this project. Conceptually, it should be possible to develop a program that can convert a physical description of the phenomena occurring within a process into a mathematical model. Such an approach would have no dependence on prewritten library models, so the only limitations arise from the scope of the predefined language. In practice the difficulty of finding a generic framework and language to describe all chemical processes has limited the application of this concept to the generation of models of the most simplistic type. This inability to generate complex models explains why automatic model generation is still an academic concept and not yet a commercial success.

1.2 Computer-Aided Process Modelling

The important role that mathematical modelling has in process and chemical engineering has generated substantial impetus, from both industry and academia, to develop tomorrow's modelling tools. With the expectations of computer-aided process modelling constantly rising, the specifications for these modelling tools are becoming even more complex. In this section the current state of computer-aided process modeling is described by referring to industrial and academic software that may be widely available or still in its infancy. This section begins with a brief description of some of the important aspects of process modelling (1.2.1). With the key aspects of process modelling established, computer-aided process modelling will be discussed in section 1.2.2. Finally in section 1.2.3, the grounds for computer automated process model generation will be presented and the current status of this field of research will be discussed.

1.2.1 Important Aspects of Process Modelling

There are several aspects of process modelling that must be considered when designing a good piece of computer-aided process modelling software. Some of the more important issues are discussed below.

Hierarchical Process Modelling

As mentioned in section 1.1, the real world, in all its generality, cannot be translated into a mathematical model that represents every aspect of its behaviour. The complexity of the world necessitates that certain aspects are approximated and others are assumed to be negligible. An important facet of the modelling problem is thus the identification of the significant features of the behaviour of the process at hand, and the construction of a model which adequately reproduces those features without irrelevant complexities (Sargent, 1983). For a given process, models of different complexity and accuracy are required at different stages of the process development. Simple models are required early in a process design but more complex ones are needed as the design process proceeds (Sargent, 1990). Similarly, sophisticated models are often used as a basis for developing and validating models that are more robust and less computationally expensive. This leads to a hierarchical modelling approach whereby a family of models may be required to model a given process throughout its entire life cycle.

Model Clarity and Model Structure

Process modelling is a complex task that may involve thousands of equations and numerous model assumptions. To support the modelling of complex processes, it is necessary to provide the user with a structured formalism within which they can build the model. By designing the software architecture so that process models can be developed in a more structured manner, the traceability of the model assumptions is improved and the opportunities for erroneous modelling are reduced. An excellent means towards structured models is to incorporate object-oriented programming techniques within the software. This enables the model to be constructed from a collection of discrete, distinguishable objects that incorporate both data structure and behaviour. As each object is an instance of its class, the model can be structured around sets of model entities that derive from instances of the same base classes. This enables frequently occurring aspects of the model to be declared once for each class. By allowing a model to be constructed from instances of its generic parts, aspects of the model can be reused, reducing the size of the model, the time spent modelling and the likelihood of model-based errors.

Successful Communication Between Modelling Tool and User

The difficulty in generating a physically correct and mathematically well-posed model makes it exceedingly difficult to provide a computer-based environment to guide the user through the modelling process. The paramount importance placed on generating a coherent and solvable mathematical model tends to result in the computer second-guessing the intentions of the user. In such instances, whilst the process-modelling tool may take the modelling "engine" out of the "engineer", one is merely left with "er", a natural response from the user who has insufficient background knowledge to compensate for the deficiencies of the modelling tool. Clearly there is a direct link between the ease-of-use of any computer-aided process-modelling tool and its success; however, the accuracy and understanding of the models generated must not be sacrificed in the quest for simplicity.

When designing a computer-aided modelling tool, it is fair to assume that the user has a good grasp of the rudimentary principles of chemical engineering that form the basis of mathematical modelling, but may be hindered by a limited understanding and experience of the steps that are needed to convert those principles into a mathematical model. The success of any computer-aided mathematical modelling tool can therefore be gauged by the program's ability to represent the modelling process in terms of the rudimentary principles of chemical engineering. Doing so allows the software and user to communicate in a manner that is familiar and well understood by the user. This not only makes the modelling tool easier to use, but by placing the user fully in control of the modelling process, the generated model is more likely to fulfil the role for which it was conceived.

1.2.2 Computer-Aided Modelling Software

In the 1980s each major chemical company had its own in-house process engineering software, developed to support its design operations. Examples of software designed for this purpose include Flowpack and Genesis, developed by ICI and BP respectively. Each company had a team of support engineers, whose task was to develop the software and provide assistance to the engineers who used it. As time progressed, the software became more advanced, the benefits of using the software became more apparent and subsequently the reliance upon the software increased. In the early 1990s the in-house modelling software was replaced by more powerful and cost-effective software maintained and licensed by a handful of global specialist software

companies. Springing from strong academic roots, software companies such as AspenTech (*AspenPlus and Hysys*), SimSci (*Pro II and SIM4ME*) and PSE (*gPROMS*) focused on computer programming techniques that make computer-aided modelling easier to use and more intuitive. These two factors have combined to extend the user-base of simulation tools far beyond its historical trend. The incorporation of such software in the syllabus of many academic chemical engineering courses is evidence of their success.

Many of today's computer-aided process modelling software platforms operate on either an equation-oriented basis (e.g. *gPROMS*) or a block- (modular) oriented basis (e.g. *AspenPlus, SIM4ME*). Equation-oriented modelling tools, such as *SpeedUp* (Perkins and Sargent, 1982), *gPROMS* (Barton and Pantelides, 1994; Oh and Pantelides, 1996) and *ASCEND* (Piela et al, 1991), provide a high-level declarative language, for the declaration of mathematical models in terms of a set of variables and the equations that relate them, and an automated process-modelling environment for the solution of the mathematical model. By separating the mathematical language, used to express the equations, from the mathematical solution procedures, the equation-oriented approach eliminates many of the major difficulties associated with process modelling without sacrificing the flexibility of the approach. In order to make the equation-oriented approach more "user-friendly", *gPROMS* analyses the formulation of the model for inconsistencies and errors, such as redundant equations, under-specification and poor allocation of boundary conditions, before executing the simulation.

To enhance the handling, sharing and checking of complex models, equation-oriented modelling tools provide the facility to model complex unit operations from instances of the basic models, thereby facilitating the reuse of model code. Furthermore, inheritance is used to establish a multi-level class hierarchy, where a child class inherits the attributes and methods of its parent class. This approach is used in both *gPROMS* and *ASCEND*. This enables the process to be expressed in terms of an organised network of connected parts, with each part comprised of a number of predefined parts. Each part may be further refined by specifying additional attributes/methods or by modifying the methods inherited from the parent class. This object-oriented approach improves the ease and reliability with which adjustments can be made to the original model. The absence of prewritten library subroutines means that only the experience of the modeller and the power of the automated solution procedure restrict the capabilities of the equation-oriented approach. However, the level of expertise needed to generate mathematically consistent models makes the model-development a significant part of the overall budget (Perkins and Barton, 1987)

and a potential bottleneck in the widespread use of model-based techniques in industrial practice (Marquart, 1996).

One possible way to reduce the level of expertise needed to construct process models is to provide libraries of basic unit-operation models, which can be combined to model the overall process. This approach has been successfully adopted by the block-oriented software packages (Evans et al., 1979) that use standard flow-sheeting techniques to decompose the process into a number of standardised process units (blocks). Each process unit is modelled by selecting an appropriate pre-programmed subroutine from a library. As each subroutine is pre-programmed, the code is largely fixed and inaccessible to the user. To generate the overall model, the chosen subroutines are interconnected with streams, which denote the flow of material and energy. The relative simplicity and robustness of the block-oriented approach are two of the major factors behind its success. At the same time, it is not clear whether the compilation of a complete library of standard dynamic models is feasible or even useful, given the fact that dynamic process modelling is very often used precisely for studying the behaviour of non-standard processes, or of standard processes under unusual conditions (Pantelides and Barton, 1993).

A powerful extension to the standard block-oriented approach exemplified by *SIM4ME* and *AspenPlus* is implemented in *Modelica*, an object-oriented modelling language designed to allow convenient, component-oriented modelling of complex physical systems. Based around a high-level language, *Modelica* is able to provide an object-oriented modelling paradigm that supports hierarchical model building and reuse of modelling knowledge across several different domains of application (electrical, mechanical, structural, chemical engineering). This paradigm provides an open standard for the format of industrial model libraries and an environment in which they can be used and modified. *Modelica's* model building methodology is centred on instantiating a set of library models with the necessary parameters and appropriate language to fully transform the library model into a complete and fully solvable mathematical model. For cases where model libraries do not exist, models that are based on standardised interfaces can be created either from scratch or from a hierarchy of library-based submodels. For the latter case, in addition to inheriting all the properties of the library-based submodels, additional properties and attributes can be assigned to the model. As all submodels are application-independent, several submodels can be combined to model a given process unit and hence the potential modelling scope of *Modelica* is considerably greater than with conventional block oriented software.

The modelling of chemical processes within the *Modelica* environment is supported through a number of model component libraries. Examples of such libraries include model libraries for the mass flow in process plants (Fabricius and Badreddin, 2002), thermo-hydraulic systems with chemical reaction (Eborn et al, 2000, Tummescheit and Eborn, 2002), thermodynamic applications (Steinmann and Zunft, 2002) and two-phase flow (Jensen and Tummescheit, 2002). Through inheritance, *Modelica* is able to fuse together various submodels, creating complex models from elementary and generic components. Examples of physical aspects covered by the library model components include heat and mass transfer, hydraulic flow and physical properties of the medium in question. The structure of each model can be controlled by the manner in which the model or set of submodels is parameterised. This process of parameterisation may involve assigning appropriate values to the geometric data within the model, selecting an appropriate numerical solution procedure (discretisation method) or declaring the constraints that dictate the appropriate conditions under which each submodel is to be used.

Over the last two decades, the advances in the modelling capabilities of process engineering software have changed the face of chemical engineering. It is inconceivable that today's generation of tightly-integrated continuous plants or flexible batch processes could be designed without such software (Best, 2003). Yet despite the unquestionable advantages that these modelling tools have provided, their limitations are still significant. In the era of reliable desktop applications, our dependence on software tools and our reluctance to question the accuracy of such tools have never been greater. For the engineer, computer-aided process modelling has become just another desktop application with a colourful, attractive, convincing and frequently foolproof graphical-user interface to guide the engineer through the entire modelling process. Although these easy-to-use graphical interfaces may be seen as further evidence of how successful modern computer-aided modelling software has become, the huge difference in the level of difficulty between word-processing a document and modelling a complete chemical process provides an indication, not only of how powerful the software has become, but also of the ease with which the software can be misused. As the user base of today's commercial software has far outstripped the small group of software engineers that developed the in-house company software of the 1980s, it is becoming increasingly difficult to keep the "expert" in the "expert system".

Today's novice engineers often lack experienced mentors to guide and correct them in producing and validating process models and consequently there is often the inclination to view a coherent

and solvable model as sufficient rather than merely necessary (Coon et al., 1998). The difficult task of automating the "art" of process modelling means that it is frequently easier for the software to second-guess the intensions of the user rather to guide them through every stage of the modelling process. This often leads to serious and costly mistakes that remain unnoticed due to the user's exposure to intrinsically reliable desktop applications (such as word processors and spreadsheets) and consequential lack of scepticism. The consequences of computer-generated mistakes may include inadequate capacity, gross over-capacity, operability problems and serious design faults that lead to accidents (Best, 2003).

1.2.3 Computer Automated Model Generation Software

A modelling methodology based on unit operations is frequently inadequate to accurately model modern chemical engineering process systems. In addition to the mistakes associated with the user's lack of understanding of the "hidden" modelling assumptions and the validity of each unit model, there are frequent occasions where the modelling scope is of sufficient complexity that there is no available library model capable of accurately predicting the behaviour of part or all of the process. Although these inadequacies can be overcome with the equation-oriented approach, the level of experience of the non-specialist is frequently insufficient to operate such software. There is therefore a commercial need for software that addresses the inaccurate and restricted application of unit-based modelling software without reducing the software user-base to a handful of global specialist software engineers.

One possible solution lies in providing a passive environment for the user to declare the physico-chemical behaviour of the process and the assumptions that they wish to incorporate within the model. Similar in concept to a high-level command language, the software would record the physical description as a network of interacting physicochemical phenomenon, described through a language of key words that represent particular physical aspects of the process. With the rudimentary chemical engineering concepts such as mass transfer, energy transfer and reaction structured and encapsulated within the software, mathematical models could be generated from this physical description with ease, either for standardised units or for unique and novel processes. By basing the software upon the language of physics, the user is not restricted to a pre-specified and rigid modular language, which in essence puts the words into the user's mouth. At the same time the user does not need to be concerned with translating the language of physics into the language of mathematics, a problem that restricts the use of equation-oriented software to

a limited number of specialists. Most importantly with a declarative front end, the user is aware of the aspects of the process that are being modelled and the unimportant details that have been excluded through the model assumptions. In other words, the user is in control of the complete modelling process.

Evidence of an early attempt to generate a process modelling representation that incorporates the flexibility of the equation-oriented approach and the ease of use of the block-oriented approach can be found in *Model.la*[1] (Stephanopoulos et al., 1990). In *Model.la*, a given process is constructed from a number of process-specific modelling elements, such as Generic Units, Ports and Streams. The behaviour of each modelling element is established in terms of a further set of modelling elements that represent the scope of the model (assumptions and simplifications), the process constraints (a set of relations modelled through a taxonomy of different equations that are encountered in process engineering) and the variables. From a user-provided specification of the modelling scope for each Generic Unit, *Model.la* was able to generate the equations, each in terms of the appropriate set of variables.

More recent research on automated model construction has aimed to decompose the process into components that are more elementary that the process unit, thus allowing the structured description of complex models in terms of a relatively small number of concepts (Pantelides and Britt, 1995). Inspired by the network theory, Giles (1998) developed a four-tier hierarchical approach for the decomposition of a process. The four levels, comprising of the device level, the phase level, the storage level and the molecular level, provide the user with a means to characterise the process at increasing levels of resolution. In Giles's framework, the storage level provides a means to represent the storage of mass, energy and momentum within each phase. To enable the physicochemical behaviour of the phase to be represented at the storage level, coupling elements are used to represent the mechanisms (reaction, diffusion and convective flows) that affect the storage of mass, energy and momentum.

Other prototype software, such as *VEDA* (Marquardt et al., 1993; Bogusch and Marquardt, 1995) focuses on a two-tier characterisation of the process. The first-tier provides the structural representation of the process, whereby the process is decomposed into a number of structural

[1] Whilst the majority of automatic model generating programs are united in terms of the conceptual approach that they adopt, they are less harmonious in the expectations that they place on the user. Early programs such as Model.la required the user to construct the model for each unit by referring to a library of predefined equations. Although such an approach did not require the user to possess any knowledge of the exact equation, a level of expertise was still required to select the correct equations. The reliance that Model.la places on the user to state the required equations means that it is frequency not regarded as an automatic model generator.

elements. The functional characterisation of each element forms the second tier of the approach. In this tier, the necessary phenomenological description (phenomenological components in the case of *VEDA*) that represents the physical behaviour is assigned to each structural element. Since each model element does not act in isolation, it is necessary to link the elements with a set number of connections that carry fluxes of the extensive quantities. Extending the capabilities of the connecting element, *Modeller* (Preisig, 1995) also allows the user to specify the selective permeability of specific components as well as the direction of flow. Marquardt (1996) gives an excellent discussion of the formulation of process models within computer aided modelling environments.

An alternative method for structural decomposition, adopted by Vazquez-Roman (1992), is based on the premise that all material is present in well-defined thermodynamic phases, and hence any process can ultimately be defined as a collection of interacting phases (Perkins et al. 1996). This approach is conceptually simpler as thermodynamic phases can be characterised completely in terms of a small set of standard quantities (e.g. component hold-ups, internal energy) and the corresponding standard conservation equations, while the vast diversity of transport phenomena occurring in processes of practical interest is associated only with the connections between the phases (Pantelides and Britt, 1995). To permit model abstraction, Vazquez-Roman (1992), Cho (1997) and Barber (2000) adopted a reservoir concept to permit the declaration of entities that are sufficiently large that their thermodynamic properties remain constant during the whole of the simulation.

With the conceptual structure in place, the phenomenological characteristics, mathematically represented by the equations that encapsulate the physico-chemical behaviour, can be added to generate the complete model. The phenomenological content needed to customise each element is commonly contained within a reference superstructure. Analogous to a knowledge-based tree, the phenomenological content is formulated by either expanding or neglecting terms within the structure (Dieterich et al. 1997), the control shell (Jensen and Gani, 1996) or the knowledge-based tree (Gerstlauer et al. 1993; Bogusch and Marquardt, 1995). Typically, conservation equations are applied to each element to relate the state variables (pressure, temperature, and composition) to the fluxes between the surroundings and the device itself. Connections are customised through an appropriate transfer law, which relates the rate of transfer to the state variables of the two interconnected elements.

By associating each "branch" of the knowledge-based tree with a command word, the user is able to prescribe a physical description to refine the equations for each element. Alternatively the model equations can be formulated, not from a physical description of the process but from a list of assumptions (Lakner et al., 1999; Hangos and Cameron, 2001). Lakner et al.'s (1999) assumption-driven model editor uses a two-tier assumption-based structure to generate lumped process models. In the first stage, primary assumptions are used to generate the standardised skeletal conservation and constitutive equations needed to represent each device. The conservation equations are then tailored to fit the particular purpose at hand using the secondary assumption. In an unnamed program, Hangos and Cameron (2001) provide a basis to categorise the modelling assumptions, thereby enabling the construction of model hierarchies that share a common assumption-based foundation. By placing the emphasis on the assumptions, and not the physical description, the effect of each assumption is more apparent, making it easier to develop models tailored to a particular purpose.

1.2.3.1 Multi-faceted Computer Generated Process Modelling

By using the principles of inheritance and hierarchical decomposition, the structural representation can be refined to the point where the model is tailored to the particular purpose at hand, a strength that has been previously discussed by Sargent (1990). This allows the generation of model families that provide varying degrees of detail to meet the needs of a diverse range of process engineering activities. Similar to the conceptual architecture used by Vazquez-Roman (1992) and Stephanopoulos et al. (1990), Drengstig et al. (1997) devised a graphical based process-modelling methodology based on the topological abstraction of a process. The topological abstraction achieves both inheritance and hierarchical decomposition through the decomposition of the process into a number of composite and elementary components. A composite component represents a control volume that encompasses several composite/elementary components, whose behaviour can be described in terms of the sum of its parts. Elementary components represent control volumes that can be decomposed no further and require phenomenological abstraction to model their behaviour.

The process of model refinement can also be applied to specific parts of the overall model, allowing the formulation of model units of varying resolution and complexity. This allows the user to focus their attention on the units that have the greatest influence on the overall process. Through an object-oriented approach *Model.la's* (Stephanopoulos et al., 1990) strength lay in its

ability to generate multi-faceted models consisting of multiple levels of abstraction. To maintain continuity between model units, structural information concerning the level of abstraction is conveyed between neighbouring units.

1.2.3.2 The Generation of Accurate and Traceable Process Models

The automation of modelling, apparent in CFD codes such as Fluent and CFX, is frequently a double-edged sword. Whilst the benefits are unquestionable, the automated program will often produce erroneous results for users who incorrectly characterise the process through an inappropriate selection of assumptions. In an attempt to alleviate this problem, *Model.la* (Bieszczad et al., 1999), a development of an earlier program of the same name, attempts to maximise the extent of assistance that is provided. By maximising the interactions between the user and program, *Model.la* ensures that the user declares the model in a gradual and supported manner. The textual input files such as those used in *gPROMS* and *SpeedUp* are replaced with a graphical user interface designed to formulate process models in a sequential manner through a series of drop-down menus and dialogues. The provision of a facility to store the sequential list of model assumptions provides an excellent environment for the exchange and reuse of physical models.

Further program-user interactions are achieved by *Techtool* (Linniger et al., 2000). By utilising object-orientated techniques, physico-chemical expressions can be generated that are sensitive to the context in which they are used. This not only minimises the number of required classes but also ensures that conflicting assumptions are avoided. The generation of process models in *TechTool* is executed through a modelling agent, similar in style to the Microsoft "paperclip". The modelling agent is designed to guide the user sequentially through the model building process, using the model history to evaluate the possible opportunities that exist for further model development without sacrificing the physical correctness of the model.

In a similar manner to *TechTool*, *Modeller* (Preisig, 1995) also ensures that the generated model is mathematically correct. By providing a Windows-based user interface and context-sensitive drop-down menus, *Modeller* is capable of guiding the user to create well-posed models. To ensure this, *Modeller* uses a rigid top-down structure, requiring a declaration of the species and reaction set within each system before the conservation equations are formulated. In an attempt to improve the structure and reusability of the code, *Modeller* (Preisig, 1995) adopts a hierarchical

approach in establishing the systems. Such an approach, which is analogous to the flow-sheeting used in modelling languages, allows the user to group networks together under a reference name.

Extending the capabilities of the computer-aided modelling environment beyond the representation and implementation of process models, *KBMoSS* (Vazquez-Roman et al., 1996) and *ModKit* (Bogusch et al., 2001) provide the user with a process-centred framework that supports the development of the model throughout its entire life cycle. Based on earlier work performed at Imperial College (Vazquez-Roman 1992), *KBMoSS* provides a graphical interface that is designed to ease the evolution of the model. Through six graphical windows, *KBMoSS* separates the flow-sheeting from the model generation thereby enabling sub-unit models to be written independently of the flow sheet. For each process unit, a number of models (schemes) can be written and stored within a specific directory. Each scheme is a self-contained segment of code, allowing the full range of formatting (copying, joining, editing) to be applied. To generate the final model, a scheme is selected for each process unit and inserted into the flowsheet. A similar strategy is also implemented in *ModKit*, which allows families of varying granularity to be stored within a "Model Container" for each process unit. To assist the development of process models within a group of process systems engineers, *ModKit* encourages the communication of modelling ideas by providing a facility to attach textual comments to each assumption. The reuse of process models is achieved through the provision of a directory capable of storing process-unit models, their documentation and a list of modelling strategies.

1.2.3.3 Automating the Modelling Procedure for Processes that Experience Discontinuities

The structural characterisation discussed so far is applicable only for the modelling of processes that run smoothly. Since the process is represented as a fixed network of delimitable elements, the configuration of the elements is unable to change and the process must proceed in a completely continuous way. Real (batch) processes may undergo appreciable discrete changes overlaid on their mainly continuous operation. These discrete events result from, for example, the digital process control, plant equipment maintenance, plant shut-down and start-up (Cho, 1997). These discrete events result in the propagation of phases through the process, with the domains of the various phases constantly changing in shape and position or even number (e.g. liquid accumulation in a tank / the evaporation in a kettle boiler). As the delimitable elements typically refer to the phases, the connections must be flexible to ensure that the configuration of elements can change, reflecting the changes in the process. In response to this need, Cho (1997) attempted

to generate lumped models capable of modelling all eventualities by asking the user to declare all of the possible phases that could occur within a vessel. All connections were made relative to the vessel and not the phase. By determining the position of each declared phase within the vessel, Cho was able to assign each connection to the appropriate phase, thereby generating the network of delimitable elements.

1.2.3.4 Automating the Modelling Procedure for Distributed Systems

A distributed system refers to a region of the process whose physico-chemical behaviour is distributed over one or more of its dimensions. Attempts to automatically generate models for distributed systems are apparent in the program *BIMAP* (Dieterich and Eigenberger, 1995). By simplifying the one-dimensional conservation equations, *BIMAP* was able to generate a mathematically well-posed model in *LaTeX* from. Designed originally to generate models for reactors, *BIMAP's* modelling engine was capable of creating models that consisted of a series of multi-scaled one-dimensional subsystems represented in cartesian, spherical or cylindrical co-ordinates. To enable *BIMAP* to couple multi-faceted, one-dimensional subsystems together, *ModuSim* was written to provide the necessary boundary conditions (Dieterich and Eigenberger, 1997). *ModuSim* replaced the conventional connections and boundaries with subsystems that were capable of dealing with both normal and "orthogonal" boundary conditions[2]. Within *Modusim,* all of the state variables and fluxes for the boundary subsystems were declared as unknowns and determined either through the use of appropriate connecting conservation equations or by equating the potential between neighbouring subsystems. The generation of process models in *BIMAP* excluded the generation of lumped models and models with a dimensionality greater than one. Further *BIMAP* was limited to models that contained only one phase.

Dieterich and Eigenberger (1995) method of neglecting terms in the conservation equations is mirrored in the work carried out by Jensen and Gani (1996). In their unnamed programme, processes are represented by a number of control shells. By using the control shell principle, Jensen and Gani were able to decompose the process into elements which could be modelled from first principles using either lumped or distributed conservation equations. By increasing the

[2] Normal boundaries lie perpendicular to the direction of the co-ordinate system. "Orthogonal" was used for boundaries with no such directionality. An example of an orthogonal dimension is the radial direction of a catalytic pellet submerged within a three dimensional medium.

number of shells used, Jensen and Gani were able to increase the complexity of the model[3]. To close the generated model, property equations were selected from a library set of stored closure equations. Whilst Jensen and Gani (1996) focused on offering a large selection of possible equations to model each shell, the rigid structure obtained through the expansion of a reference model would not allow the user to replace some of the pre-defined equations with specific correlations. Furthermore, little attention has been given to the process by which control shells are interconnected. The lack of a predefined set of fundamental state variables needed to link the control shells together leads one to question the capacity of the methodology to ensure full compatibility between the control shells.

Continuing on the themes established by Vazquez-Roman, Barber's (2000) *gMAGIC* incorporated the principals of fluid dynamics to automatically generate continuous and lumped process models. gMAGIC permitted models of different dimensionality to be constructed, through the use of a number of distinct classes. The concept of connections was also evolved to allow full compatibility between units of different dimensionality. A user interface was established to allow the physiochemical phenomena to be communicated through a number of model assumptions. Based on the work of Gilles (1998), Mangold et al. (2002) devise a network theory capable of structuring the modelling approach for distributed systems. Through the provision of separate fluid-dynamic and thermodynamic subsystems, Mangold et al. (2002) was able to uncouple the fluid dynamics, frequently associated with distributed systems, from the thermodynamic state of the material. This highly flexible approach permits the user to make independent assumptions concerning the thermodynamic state (e.g. constant temperature) and the velocity profile (superficial velocity), a limitation that is apparent in *gMAGIC*.

The principle of using different classes to model distributed systems is also apparent in *Model.La* (Bieszczad, 2000) which allows distributed models to be formed either in terms of a set of partial differential equations (*distributed model-units*) or through a composite of identical lumped models (*staged model-units*). In a staged model-unit, a unit is decomposed into a series of identical subunits allowing the assumptions made for a representative subunit to be automatically propagated to the other subunits. With use of the staged model-entity, a distributed system of two phases in equilibrium can be modelled in terms of a number of transfer units.

---

[3] The overall complexity is constrained by the level of detail encapsulated within the control shell.

1.2.3.5 Automating the Modelling Procedure for Two-Phase Systems

A two-phase system is a region of a process that incorporates material that is in two distinct stable thermodynamic phases. The interactions between these two phases are so complex that it is not feasible to decompose the region into a complex network of single phases. At present the inherent difficulties in automating the art of model building have restricted the scope of prototypes to either lumped systems or systems involving one phase distributed over the region of interest. Both the lumped and distributed modelling of one-phase subsystems are well-established and therefore the structural encapsulation of the relevant modelling techniques has been relatively straightforward. On the other hand, the current level of research activity in two-phase flow makes any attempt to develop a generalised approach for this field extremely ambitious. The evidence of previous two-phase modelling prototypes is sparse, with the only documented successes arising from simplified cases that rely on specific assumptions (Bieszczad, 2000; Cho, 1997; Gilles, 1998; Mangold et al., 2002; Dieterich and Eigenberger, 1995).

An effective approach to automate the modelling of two-phase systems involves grouping the two phases into a single entity called the aggregate phase (Cho, 1997) or device (Gilles, 1998). This allows the user to decompose a given process into a number of two-phase regions and determine the behaviour of each region by considering the properties of the two respective phases. This approach of treating the two-phases as a homogeneous mixture was also used by Bieszczad et al. (1999) to model distributed two-phase systems. Within Bieszczad et al.'s framework, distributed two-phase systems are decomposed into a number of homogeneous lumped models with the two phases in thermal and chemical equilibrium. Since a considerable number of two-phase problems experience neither thermal nor chemical equilibrium, the capabilities of Bieszczard et al.'s (1999) approach are restricted. *BIMA*P (Dieterich and Eigenberger, 1995) provides a more detailed approach to modelling distributed two-phase regions, treating each phase as a separate entity modelled through a set of conservation equations. Such an approach incorporates full consideration of the interfacial exchange of mass, energy and momentum, enabling systems that are not in equilibrium to be modelled. However, the inability to specify the relative motion between the two phases restricts the approach to processes with one phase that is fixed in space (e.g. fixed bed catalytic reactor).

1.3 Conclusions and Objectives

In the opening section of this chapter an introduction to computer-aided modelling software was given. It was noted that the limitations of the current computer-aided modelling software include the high level of modelling experience required to use equation-oriented software and the restricted modelling flexibility of block-oriented software. Motivated by these limitations, the advantages of developing a modelling tool based on the underlying physical and chemical characteristics of chemical engineering processes were presented and a review of the current state of phenomenon-based modelling was undertaken.

Despite the progress that has been made in the field of phenomenon-based modelling, there are still many hurdles that must be overcome before the full conceptual benefits of phenomenon-based modelling can be realised in the commercial sector. One substantial development that must be made is to push back the current frontier of computer-aided process modelling to incorporate process modelling for distributed two-phase regions of the process. The large number of two-phase process systems that cannot be adequately categorised by lumped models is one of the main reasons why computer aided modelling is still in its infancy. Examples of distributed two-phase processes include a settling tank, a condenser, a porous catalytic reactor and a tube boiler. Motivated by the need to extend the scope of computer-aided modelling to incorporate this class of chemical processes, the work presented in this thesis concentrates on the automatic generation of lumped and distributed models for one- and two- phase systems.

Based on the earlier work carried out at Imperial College by Vazquez-Roman (1992), Cho (1997) and Barber (2000), the objective of the research is the development of a computer-aided modelling tool for the automatic generation of one- and two- phase process models. In addition to attempting to automate the modelling process for two-phase processes, the research will also re-examine the methodologies for automating the modelling process for single-phase lumped and distributed processes. Of particular importance is the development of a simple yet powerful object-oriented methodology based upon the rudimentary principles of chemical engineering. Based on a careful formalisation of these principles, it is the objective of this research to demonstrate that it is possible to turn the academic concept of phenomenon-based modelling into a commercial reality.

This thesis is composed of 8 chapters. In Chapter 1 a review of background literature has been given and the motivation for the work has been established. With this background motivation in mind, Chapter 2 will provide a brief overview to process modelling whilst Chapter 3 will describe the structural architecture and the phenomenological characterisation needed to generate one- and two- phase models. In Chapter 4, the formal definition of the language for the representation of one- and two- phase processes will be presented. Chapter 5 will discuss the structuring of the model equations and the logic needed to generate a well-posed and solvable mathematical model, whatever the physical description. The implementation of the framework in the prototype computer program *COMPASS* (*COMP*uter *A*ided *S*imulation *S*oftware) will be discussed in Chapter 6. In Chapter 7, the scope of the framework will be illustrated with six non-trivial computer-generated process models. Finally, in Chapter 8, overall conclusions will be drawn and some suggestions for future work will be made.

Chapter 2

An Introduction to the Mathematical Modelling of Process Systems

In the preceding chapter, the objective of the present research was stated; namely the creation of a computer program capable of automatically generating process models for one and two phase systems from a purely physical description. With this goal in mind, the modelling procedure for one- and two- phase chemical process systems Chapter 2 will be examined and systematised. In the present chapter, this systematic characterisation of process modelling will form the basis behind the model-generating framework discussed in Chapter 3.

2.1 An Introduction to Mathematical Modelling

In Chapter 1, a mathematical model was defined as the translation of an aspect of the real world, which is of particular interest to the modeller, into a solvable mathematical problem. In the context of this work, the aspect of the real world that is of interest to the modeller is referred to as the *system*. To establish a mathematical model, a set of properties that characterise the system with a sufficient degree of accuracy are identified and represented by a set of variables. Taken together, these variables fully describe the representation of the system within the model. This condition is referred to as the *state of the system*. With the state described by this set of variables, previous knowledge can be used to relate the variables through a set of equations. If the knowledge is accurate, the solution of this equation set will yield an accurate prediction of the state of the system. The knowledge may arise, in part, from a set of accepted empirical relations that merely recognise and characterise the correlation between a selected group of the system properties, or from the fundamental laws of science, which aim to represent the causal mechanisms that give rise to the correlations.

If the properties of interest vary with time, the modeler may wish to predict the evolution of the state of the system. Provided that the initial state of the system is specified, an objective forecast of the behaviour of the system can be determined by solving the set of equations over a specific time period. This process is known as *dynamic modelling*.

As the real world is a continuous seamless web of interacting physical, chemical and biological phenomena, the system can never be an isolated entity but must be viewed in the context of the universe of which it is part. As the universe is clearly too large and complex to model, it is necessary to focus on, and approximate particular aspects of the universe that influence the system. This allows the system to be isolated within the model and its behaviour can therefore be modelled. The size of the universe is sufficient to assume that its state is independent of its interactions with the system. It is therefore possible to define a boundary around the system and paramaterise the behaviour of everything outside the boundary that influences the behaviour of system. In the context of this research, the parameterized region outside the boundary is called the *environment* and the boundary may be rigid or movable, physical or imaginary and may allow a transfer of mass, energy, momentum or information across it. For complex systems there may be too many properties of interest that influence the state of the system for it to be possible to obtain an understanding of the system's behaviour. In such cases the system must be decomposed into a number of subsystems and where necessary each subsystem must be further decomposed until eventually the behaviour of the subsystem is sufficiently well understood to enable it to be modelled. Similar to its parent system, each subsystem is fully defined by its boundaries, across which it may be influenced by the neighbouring subsystems. This decomposition is referred to as the *hierarchical decomposition of the system* and is illustrated in figure 2.1.

2.2 An Introduction to Process Systems Modelling

In the context of process systems modelling the system is seen to correspond to a region of the process whose behaviour is of interest to the modeller. The region of the system may be of any scale. The system's environment is merely the region outside the system's boundaries and, depending on the scale of the process being modeled, may be anything from the atmosphere to another part of the process.

2.2.1 The Physical Mechanisms that Influence the State of the Process and the Properties of Interest

Modelling any system requires an expert understanding of the underlying physico-chemical mechanisms that underpin the behaviour of the system. Crudely speaking, the physico-chemical mechanisms that are of most interest to today's chemical engineer can be categorised into three key fields corresponding to:

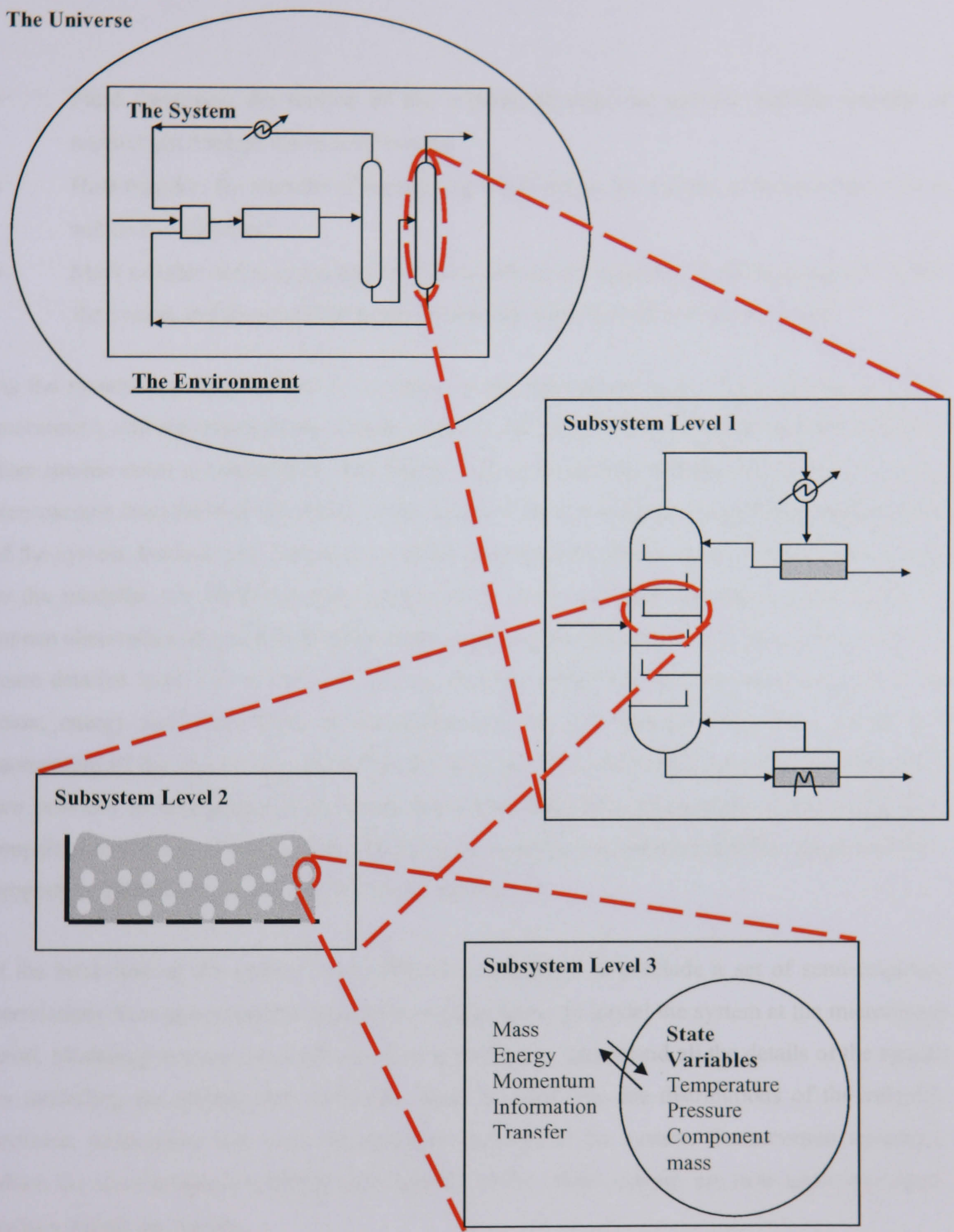


Figure 2.1: A Hierarchical Modelling Strategy for a Complex Chemical Process

- Fluid dynamics; the motion of the material through the process and the transfer of momentum through forces and stresses.
- Heat transfer; the transfer of energy and work within the system or between the system and its environment.
- Mass transfer and reaction kinetics; the transport and reaction of chemical species within the system and the exchange of them between the system and its environment.

As the system is merely defined as a region of the process, the scale of the system may vary enormously and subsequently the length scales of the physico-chemical mechanisms may vary from metres down to nanometres. The largest scale of modelling is frequently referred to as the macroscopic level (Bird et. al, 2001). At this level no attempt is made to understand all the details of the system. Instead, particular aspects of the behaviour of the system that are of most interest to the modeller are identified and represented by semi-empirical correlation based either on human observation or specific simplifications of the physico-chemical mechanisms that occur at a more detailed level. For the three fields, the empirical equations are expressed in terms of the mass, energy and momentum of the system and its environment. The mass, energy and momentum of the system are determined with the use of conservation laws. Each conservation law provides an accounting basis, which is used to relate the accumulation of one of the three properties to the rate of transfer between the system and its environment and the rate at which the property is generated or consumed within the system.

If the behaviour of the system is of sufficient complexity to preclude a set of semi-empirical correlations from being applied accurately, it is necessary to model the system at the microscopic level. Modelling at the microscopic level is an attempt to understand all the details of the system by modelling the system with sufficient rigour to determine the distributions of the velocity, pressure, temperature and mass concentration throughout the system. Conservation equations, which for macroscopic modelling were applied to the whole system, are now applied to small regions within the system.

The final level of modelling is the molecular level, where the fundamental understanding of the mechanisms of mass, momentum and energy transfer is expressed in terms of the molecular structure and the intermolecular forces. At this level the molecular mechanics underlying the

various transport phenomena are very closely related as the same molecular motions and interactions are responsible for the transfer of mass, energy and momentum (Bird et al. 2001).

2.2.2 A Thermodynamic Basis for Process Modelling

The fields of heat and mass transfer incorporate a huge number of empirical and theoretical equations that may be used to characterise the rate of mass and heat exchange between a system and its environment. Expressed in their generic form, the equations relate the rate of exchange of mass and heat to the instantaneous state of the system. From a thermodynamic viewpoint, this instantaneous state is referred to as the thermodynamic state, and a given process can be regarded as a pathway that maps out the thermodynamic state of the material between its initial state (raw materials) and its final state (product). As a system refers to a particular aspect of the process it is therefore intrinsically associated with material undergoing a change in its thermodynamic state and, for a non-reactive system, its behaviour can be modelled by relating the change in its state to the rate of heat and mass transfer between the system and its environment.

The thermodynamic state, defined by Kyle (1999) as: "A state refers to the condition in which the system exists", is fully described by the thermodynamic attributes of interest, such as the temperature, composition and pressure. In process system modelling, it is common to make the assumption that all material undergoing processing is instantaneously in a stable thermodynamic state[4], (Feinberg, 1979). This stable thermodynamic state is referred to as a *phase*. Perhaps the first reference to a phase was given by Gibbs in 1878. He stated:

> "In considering the different homogeneous bodies which can be formed out of any set of component substances, it is convenient to have a term which shall refer solely to the composition and thermodynamic state of any such body without regard to its size or form".

According to the Gibbs statement summarised in the Phase Rule, the thermodynamic state (condition) of any *phase* (material in a stable thermodynamic state) at a given time is completely determined by the knowledge of two independently variable properties in addition to the masses of each chemical species. Typically the two independent variables that are chosen to represent the thermodynamic state are the temperature and the pressure. With knowledge of the two

---

[4] Any thermodynamic state is considered stable if the total entropy amongst all the comparable states is maximised (Vazquez Roman 1992).

independent state variables and the composition of all the components within the phase, all the thermodynamic attributes of interest can be determined.

If the modelling framework is to be established on a thermodynamic basis, it is advantageous to provide a simple way to declare the process in purely thermodynamic terms. Unfortunately, stating the mass of each component is awkward, as the user may have no data concerning the component masses. It is however possible to express the mass of each component in terms of the component density and the volume of the phase. By using the volume (extent) as a means to describe the thermodynamic state of the phase, the user is able to describe the phase in terms of a property that is both easily measurable and well understood. Further, by referring to the volume, it is possible to decompose the process into a number of spatial regions (section 2.3). As the density is an intensive state variable (independent of the size of the phase), it can be easily expressed in terms of another intensive variable. To improve the ease with which the user can describe the thermodynamic state, the component density is replaced by the phase composition, expressed as an array of mass fractions.

If the behaviour of the system is to be modelled by relating the change in the state of the system to the rate of heat and mass transfer between the system and its environment, it is necessary to express the environment in "thermodynamic" terms. This allows the exchange of mass and energy between the system and its environment to be determined through fundamental and empirical equations. The environment is defined as a region of constant state (condition), so it is possible to parameterise the environment in terms of the chosen state variables; namely the pressure, temperature and the composition. The environment is therefore seen as a pressure, temperature or material source or sink (defined by the pressure, temperature and composition of the material) of infinite size.

At this point it should be noted that there are many materials of practical and commercial importance that cannot be described in simple thermodynamic terms, because of micro-structural or compositional complexity at a scale that is small compared to the region of interest. Such materials are often regarded as having additional, internal, state variables that describe the microstructure or properties of interest and whose evolution is described by additional physical models. This class of materials includes many foods and biological materials, household and personal products, structural materials and high value-added fine chemicals and pharmaceutical products. Although the framework is unable to handle this class of material, it should be possible,

in future, to extend it so that complex materials can be described in terms of their additional state variables.

2.2.3 A Fluid Dynamic Basis for Process Modelling

The field of fluid dynamics is concerned with determining the motion of a material as it passes through the system and involves properties such as the velocity, the acceleration and the force within the system. The motion of material through the system dictates the residence time of the material within the system, so the motion of the fluid is coupled with the thermodynamic state of the system. On the microscopic level, the velocity distribution within the system can be determined by simultaneously solving Newton's laws of motion and the principle of mass conservation. If the motion of the material within the system is determined on the macroscopic level, the velocity profile is unknown and therefore the flowrate of material through the system must be determined with the use of semi-empirical correlations that arise partly from analytical solutions of Newton's laws of motion and partly from experimental data.

2.3 A Modelling Strategy Based on Process Decomposition

All materials within the system are described by a set of two state variables and the composition, so a two-tier modelling approach of decomposition and characterisation can be used to model a given process. The first tier of the modelling strategy involves decomposing the system into a number of single-phase regions, each encompassing material that is in a single stable thermodynamic state. The second stage determines the behaviour of each single-phase region by evaluating the geometry and the thermodynamic state of the region. The geometry of the region is determined purely from the extent of the region, which must either be known or be determined as an integral part of the model. The state of each region is represented by the two state variables (pressure and temperature) and the composition of the phase. If there is motion of material through the region, the modelling strategy must also determine the flow (flowrate or velocity) of material through the region. By determining the evolution of the state variables, the physico-chemical behaviour of the single-phase regions can be predicted. With the behaviour of each single-phase region determined, the behaviour of the whole system is merely the cumulative behaviour of its parts. Whilst this procedure is relatively straightforward for processes involving a small number of well-structured single-phase regions, the complex arrangement of the single-phase regions for systems where both phases are flowing poses many more problems.

2.3.1 Modelling a Lumped Single-phase Region

A lumped single-phase region can be defined as a spatial region that encompasses material in one stable thermodynamic state that is uniform throughout the spatial region. At each point within the single-phase region, the thermodynamic state is defined by the same state variables. Hence the behaviour of the single-phase region can be fully described in terms of its extent, one set of state variables and, where necessary, the velocity of the material within the region. For a lumped single-phase region, conservation laws can be used as an accounting basis to determine the variations in the state variables and the extent of the phase throughout the time horizon. Here and throughout the thesis, the phase "time horizon" is used to indicate the period of interest. Hence, if the initial thermodynamic state and extent of the single-phase region are given, the thermodynamic state and extent can be deduced at any point in time. Conservation equations can be used to relate the thermodynamic state and the extent of the single-phase region to:

a) The rate at which mass and energy flow into and out of the single-phase region.
b) The rate of reaction occurring within the single-phase region.

2.3.2 Modelling a Distributed Single-phase Region

A distributed single-phase region can be defined as a region of the process that consists of material in one stable thermodynamic state, which varies continuously over the extent of the single-phase region. As the state is described by a finite set of state variables, the state variables are continuous functions over the region.

To thermodynamically charcterise the single-phase region, the thermodynamic state must be determined throughout its extent. The conservation equations can be applied to a vanishingly small (differential) region fixed in space to determine the state variables at that point. This approach is often referred to as an Eulerian formulation. In a similar manner to a lumped single-phase region, the conservation equations provide an accounting basis to determine the state variables from the input flux, output flux and generation terms. By neglecting radiation heat transfer, each flux can be expressed as the sum of a diffusive component and a convective component. The conservation equations can be integrated (analytically or numerically) over the interior of the single-phase region (domain). The thermodynamic state is thus defined by the state variables expressed as a function over the whole of the extent.

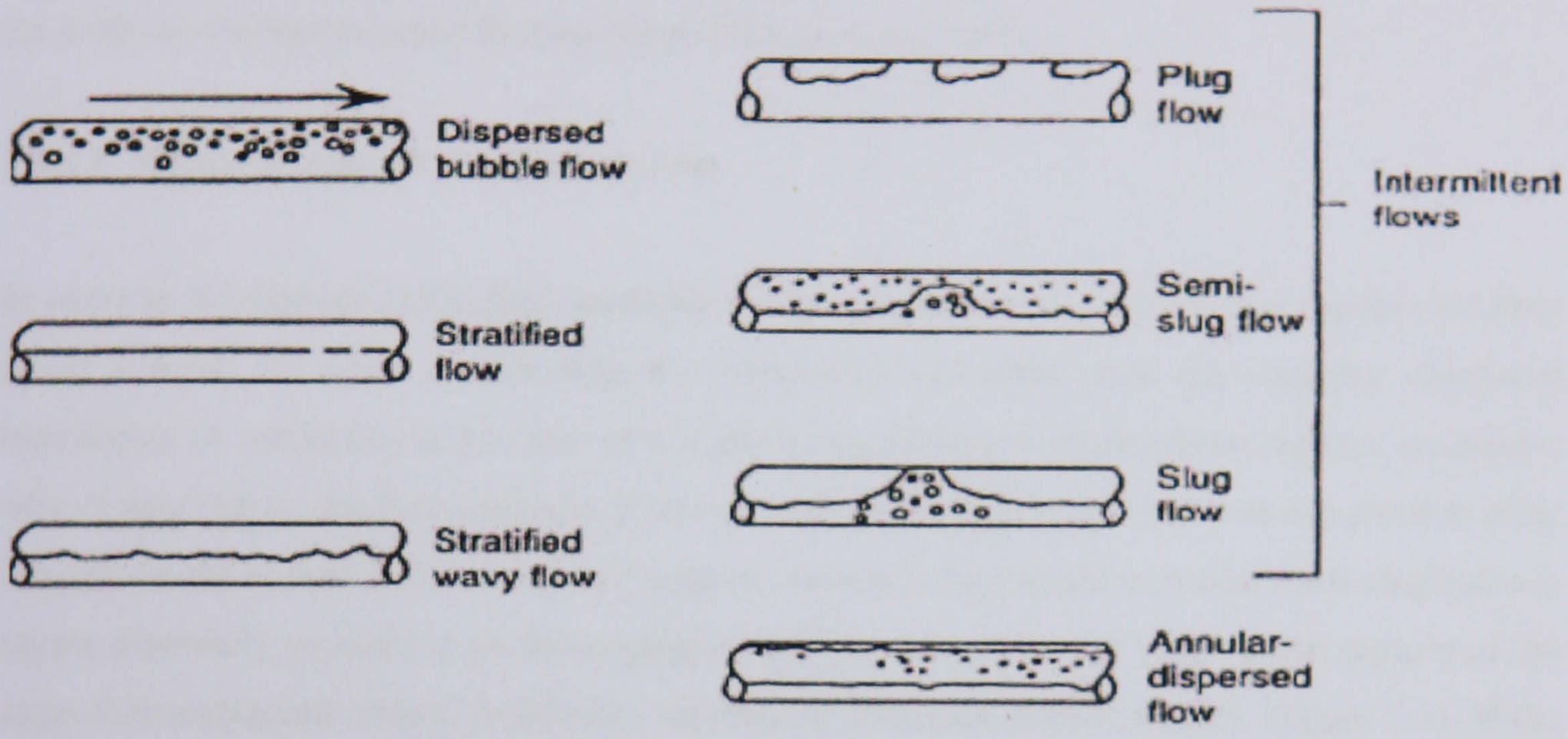
By averaging the conservation equations over a particular dimension, the continuum of state variables in that dimension can be replaced by averaged values. The continuum of state variables can therefore be expressed over three dimensions, two-dimensions or one-dimension. As the single-phase region does not exist in isolation, boundary conditions must be used to specify the conditions at the boundary of the single-phase region, so that the interactions between the region and its environment can be modelled. By integrating the conservation equations over the extent of the *single-phase region*, and using the appropriate boundary conditions, the state variables at any point with respect to the fixed co-ordinate system of the region can be found. The extent of the single-phase region is therefore specified by the range of co-ordinates that the conservation equations are to be integrated over.

2.3.3 Modelling a Two-phase System

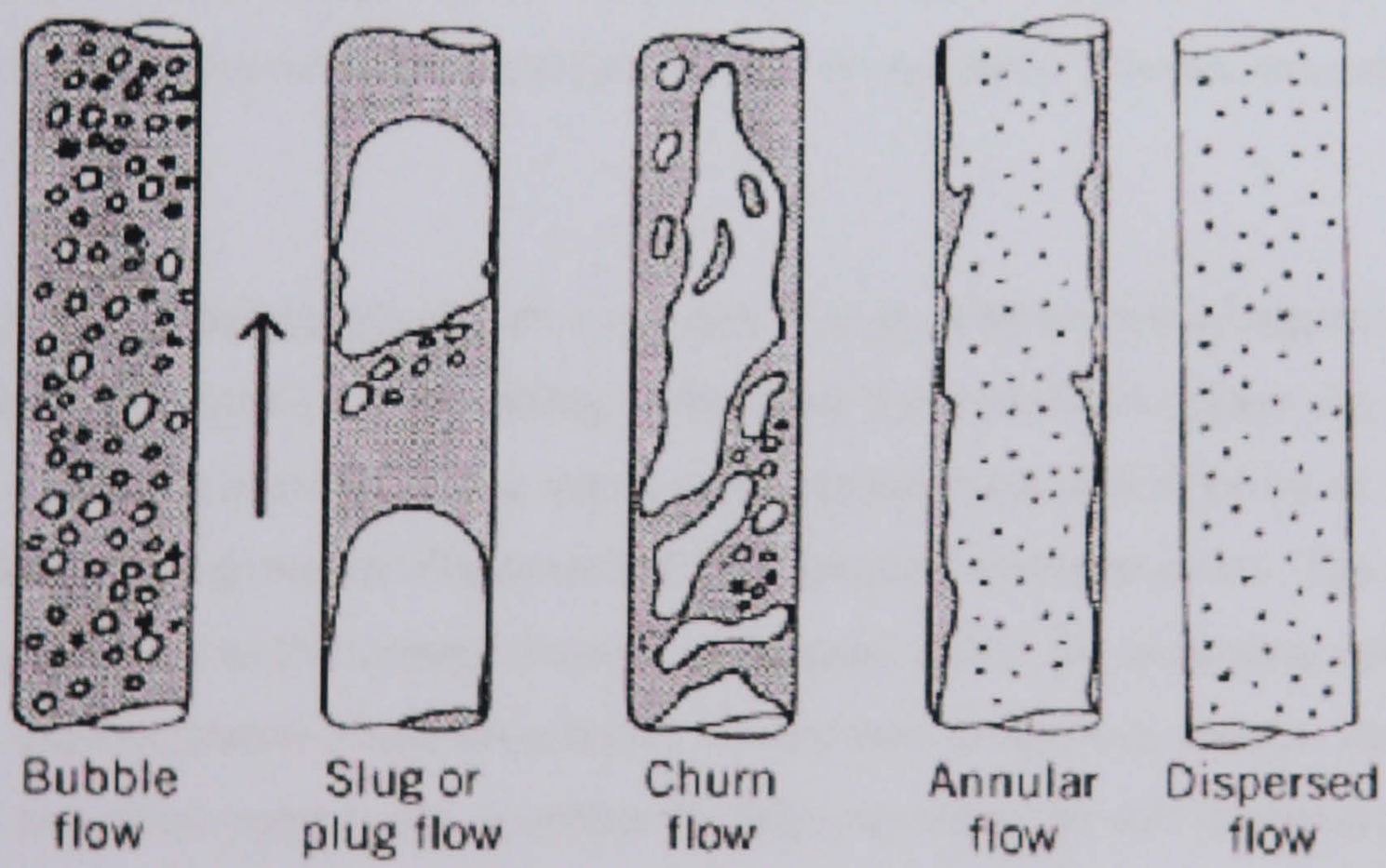
A two-phase system can be defined as a system that involves dynamic interactions between two or more single-phase regions. The complexity of the interfaces separating the single-phase regions is generally sufficient to make it impractical for the system to be broken down into the respective regions. For two-phase systems, the position of the single-phase regions may be static (porous catalyst) or dynamic (two-phase flow). If the position of the single-phase regions is dynamic (two-phase flow), the continuous movement of the phase boundaries and the constant reshaping of the region occupied by a specific phase may make it exceedingly difficult to predict the relative positions of the single-phase regions.

Theoretically at any instance in time, a two-phase system can be decomposed into a number of single-phase regions (Local Instantaneous Formulation) that correspond to one of two thermodynamic phases. Each single-phase region can be fully described by a unique set of state variables (lumped or distributed) and by its extent. A set of conservation equations can be applied to each single-phase region to deduce the evolution of the state variables. By tracking the interfaces that define each single-phase region, the extent can be determined for each single-phase region. The single-phase regions can be coupled together by relating the state variables of the regions either side of the interface (jump conditions). Hence a two-phase system can be thermodynamically defined by coupling all the state variables of all the single-phase regions together. As a two-phase system is a complex collection of single-phase regions it also has extent, corresponding to the cumulative extent of the single-phase regions.

From figure 2.2 it can be seen that, for complex two-phase systems, the process of decomposing a two-phase system into a network of single-phase regions is frequently very problematic. Current computational power is insufficient to enable a two-phase system to be decomposed into single-



Flow Regimes for Horizontal Two-Phase Flow



Flow Regimes for Vertical Two-Phase Flow

Figure 2.2: Two-Phase Flow Patterns in Horizontal and Vertical Two-Phase Flow (from Levy 1999). Modelling each System as a network of single-phase regions is not an easy task.

phase regions for all but highly simplified two-phase systems (stratified flow or a few bubbles). Further, in many cases the interface geometry is poorly understood and the generation and extinction of regions makes it exceedingly difficult to determine the thermodynamic state for single-phase regions that come into existence for a finite time. It is the author's opinion that the presence of interfaces and the difficult task of describing them has been, and will continue to be, the weakest element in codes for two-phase flow, (cf Levy, 1999).

2.3.3.1 Restructuring a two-phase System

In order to develop an automated approach for the generation of mathematical models for two-phase systems we need to overcome the difficulties associated with the complex interfacial interaction. In particular, to be consistent with the modelling of single-phase regions, we need a way of determining the thermodynamic state of each phase within the system at any point in time, throughout the extent of the two-phase system. As it is not practical to model each single-phase region separately we need to use averaging techniques to combine the single-phase regions of the same thermodynamic phase (condition) together to form one common entity (figure 2.3). If the relative positions of the single-phase regions are fixed we can average over a particular spatial region, thereby combining regions of the same thermodynamic phase into one common entity. If the regions are in a constant state of movement we can average over a period of time. If the position of the region is determined by a chaotic or stochastic process, ensemble averaging may be used.

Instantaneous Spatial Averaging: By averaging over an arbitrary spatial region within the system, the single-phase regions corresponding to the same thermodynamic phase can be combined. The result is a spatially averaged entity that is thermodynamically described by its spatially averaged state variables and geometrically described by a spatially averaged extent. The spatially averaged extent corresponds to the volume fraction of material within the averaging volume that is in the relevant thermodynamic phase (condition). As the sum of the two volume fractions must equal one, the two-phase system can therefore be fully described by the thermodynamic state of the material in each phase and by the volume fraction of the material in either phase.

A consequence of the averaging procedure is that the interface between the single-phase regions are lost. Without any knowledge of the interfaces, it is not possible to deduce the interfacial exchange of mass, energy and momentum. To replace this lost information, the total exchange of

mass and energy must be determined either by examining the flux of mass and energy at the surface of a typical single-phase region or with the use of empirical information.

Averaging over the Volume of the system: If the spatial region over which the single-phase regions are being averaged corresponds to the total extent of the system, all the single-phase regions of each thermodynamic phase can be modelled as one averaged entity, thermodynamically described by one set of state variables. The distribution of the two phases is taken to be uniform across the system and hence is described by one volume fraction term.

Spatial Averaging over a Differential Region: In many circumstances, the state variables that define the thermodynamic state of the material in each single-phase region may be regarded as continuous functions over the extent of the system. An example is a porous catalyst pellet where the concentration of the gas phase within the pores is a function of the radial position within the catalyst pellet. Instead of averaging over the total volume of the system, it is possible to average over an infinitely small (differential) region within the system. Through averaging, the material in each thermodynamic phase is thermodynamically characterised by a set of state variables for that differential region. The differential region is therefore fully described by the state variables of each phase and the volume fractions of the two phases within the region. By integrating the differential equation set over the extent of the system, the two phases can be seen as two pseudo-continua simultaneously existing at the each spatial point and time. The system is thus fully described by the state variables for each phase and the volume fractions of the two phases at any point within the extent of the system. As the two-phase system does not exist in isolation, boundary conditions must be used to specify the exterior of the region such that the interactions between the two-phase system and the environment that surrounds it are modelled. As each phase is seen as a pseudo-continuum, boundary conditions must be provided for each phase.

Local Time Averaging: For systems where both phases are flowing, the distribution of the single-phase regions vary with both the time and the spatial position in the system (e.g. gaseous bubbles flowing through a liquid medium). This variation may be in addition to the spatial distribution of the thermodynamic state across the extent of the system. Through time averaging, the single-phase regions that alternately occupy an infinitely small, differential region are sorted into their respective phases and combined to form a single time-averaged identity.

For the infinitely small differential region, conservation equations can be applied to determine both the thermodynamic state and the volume fraction of the material in each thermodynamic phase. By integrating the conservation equations over the extent of the two-phase system, the two phases can be seen as two pseudo-continua simultaneously existing at the same spatial point and time represented by time-averaged state variables and time-averaged extent (volume fraction). With the thermodynamic state and the spatial distribution of the material in each phase defined over the whole extent of the two-phase system, determining the extent of the system completes the system model.

Composite Averaging: The continuum hypothesis associated with integrating the differential equations requires that the length scale of the single-phase regions being averaged must be considerably smaller than the length scale of the characteristic geometric dimension which the averaged identity is to be integrated over. Frequently the scale of the discontinuities associated with two-phase flow prevents the distribution of the two-phase system from being modelled as a continuum in a particular dimension. Examples include slug and annular flow. If time averaging is insufficient, composite (space-time / time-space) averaging may be performed whereby the time-averaged local equations are averaged over a channel cross-sectional area or over a slice. Such composite averaging has proven to be a very useful modelling tool in the analysis of two-phase flow, (Delhaye, 1981).

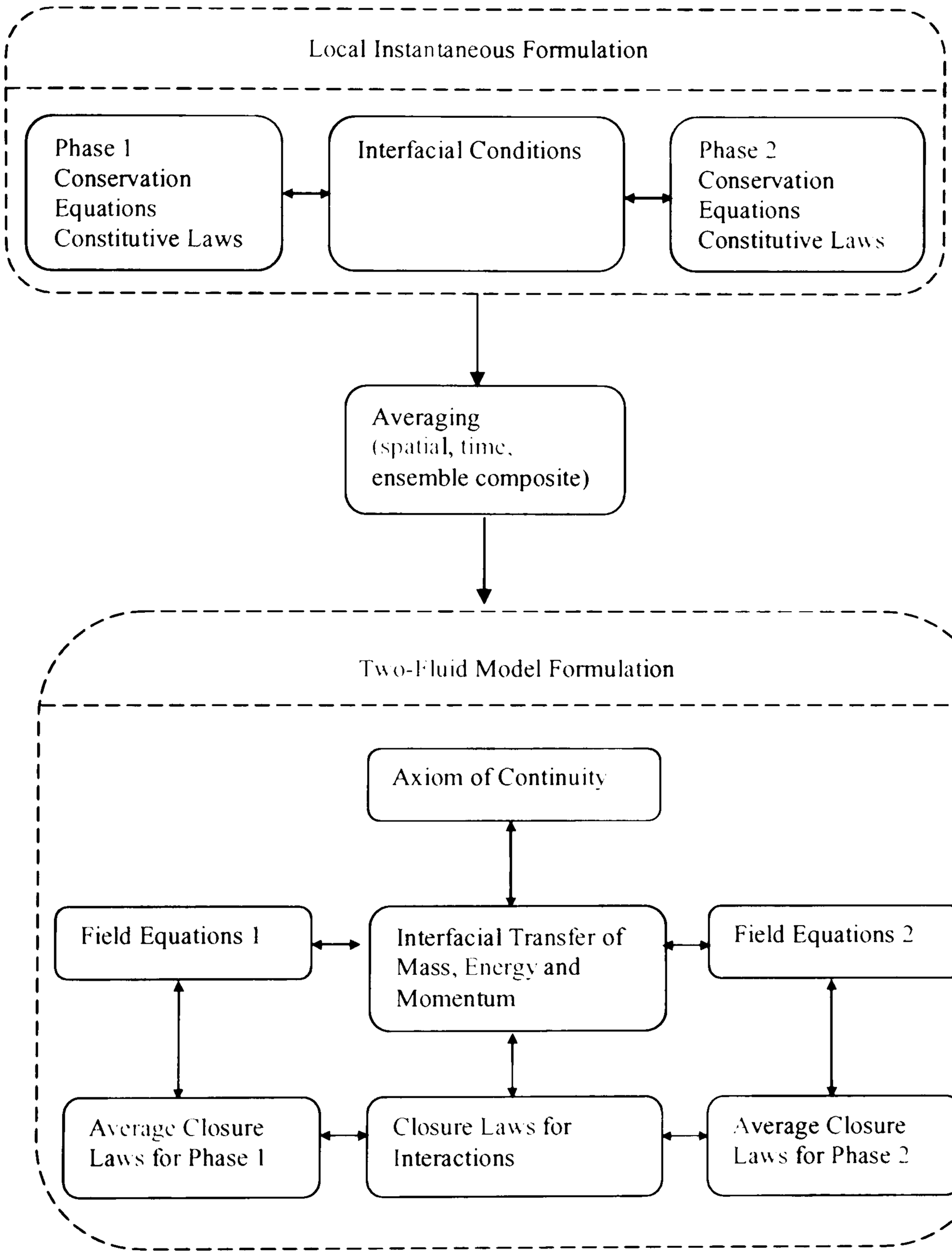


Figure 2.3: Averaged Model Formulation (taken from Barber, 2000).

2.4 Summary

A simplified overview of process modelling was presented in this chapter. Based on this overview and the premise that all material is instantaneously in a stable thermodynamic state a systematic methodology for process modelling was presented. The first step is to decompose a process into a number of single-phase regions and two-phase systems and the second is to determine the state of each single-phase region / two-phase system by evaluating the state variables. The selected state variables for one- and two- phase systems are tabulated in Table 2.1. In Chapter 3, a framework for the automation of process models will be developed based on this systematic characterisation of process modelling.

	Single Phase Region	Two Phase System	Environment
Thermodynamic Properties	Composition Temperature Pressure	Composition of each Phase Temperature of each Phase Pressure	Composition Temperature Pressure
Fluid Dynamic Properties	Velocity	Velocity of each Phase	
Extent	Volume	Volume	
Averaging Properties		Volume Fraction of one Phase	

Table 2.1: The State Variables for the Single-Phase Region and the Two-Phase System.

Chapter 3

The Modelling Framework

In Chapter 2 it was seen that, with the use of a thermodynamic basis and, where necessary, through appropriate averaging techniques, it is possible to decompose a given process into an interconnected network of single-phase regions and two-phase systems. Furthermore for each region or system a modelling approach can be developed that is based on the determination of each state variable. In this Chapter, the well-established themes of computer aided process modelling will be built upon to develop a framework around which the systematic characterisation of process modelling, discussed in Chapter 2, can be structured.

3.1 The Two-Tier Framework

The procedure of decomposing a process into a number of single-phase regions and two-phase systems is sufficiently systematic to allow a substantial part of process model building to be automated. As either the single-phase region or the two-phase system encompasses a region of material whose thermodynamic state can be predicted throughout the time horizon, either entity can be seen as a local model that fully describes the physico-chemical behaviour of a specific spatial region of the process. Hence the process can be seen as a network of local models. In the context of the present work, the local model is called a *Zone*.

The *Zone* is a local model for a two-phase system, which when the volume of one phase falls to zero reduces to a local model for a single-phase region. In either case the *Zone* represents a particular region of the process that encompasses material whose thermodynamic state can be determined from a single set of equations (lumped equations or differential equations that are to be integrated over the whole region) at any point in time throughout the extent of the *Zone*. If the *Zone* is lumped, the region corresponds to the volume of material in the *Zone*. Similarly, if the *Zone* is distributed, the region corresponds to the domain of applicability.

BIBL. LONDON

With the use of the *Zone* entity, a two-tier approach can be formulated whereby a process is firstly broken into a number of elementary units and secondly the appropriate mathematical model for each unit is formulated from the physical description that the user has provided. This approach is well established and has been successfully used by Perkins et al. (1996), Dieterich and Eigenberger (1995)[5] and Bieszczad (2000). The first stage of the approach is to decompose the process into a network of interlinked *Zone*s. The procedure for partitioning a process into a set of *Zone*s is called the **structural representation** and will be discussed in section 3.2. For the second stage, the user must provide a physical description for each *Zone* that fully characterises the extent of the *Zone*, the thermodynamic state of the material within the *Zone* and, where necessary, the flow of material within the *Zone*. For a two-phase *Zone*, the user must also characterise the volume fraction of the two phases. Based on the interpretation of this physical description, the model generator will construct the necessary mathematical equations to fully describe the physico-chemical behaviour of the *Zone*. This process is called the **physicochemical characterisation** of the *Zone* and will be discussed in Section 3.3.

3.2 The Structural Representation:

The purpose of the structural representation is to partition the process into elementary units called *Zone*s. In keeping with previous structural representations, the network of *Zone*s is created by linking a series of *Zones* together with a number of *Connection*s (Perkins et al., 1996, Barber, 2000). The *Connection* entity is used to specify the exchange of mass and energy between two-interconnected *Zone*s or *Reservoirs*. The *Connection* simply matches the state variables between two interlinked *Zone*s, and so it contains no material or energy holdup.

Each *Connection* may convey material, with associated mass and energy, (*Material Connection*) or just energy (*Energy Connection*). Further each *Connection* may be *Open* or *Closed*. An *Open Connection* is used to represent an infinitely thin interface between two *Zones* that poses no resistance to the transfer of mass and energy between the two *Zones*. Consequently, the *Open Connection* matches the state variables of the upstream *Zone* to the corresponding state variables of the downstream *Zone*. For the *Open Material Connection*, the transferred properties, which are equated across the *Connection*, are the mass fraction of each component, the enthalpy, the pressure and the mass flowrate. For the *Open Energy Connection*, the transferred properties are the temperature and the diffusive energy flowrate.

---

[5] Dieterich models phases as Devices.

The *Closed Connection* is used to represent an interface of sufficient substance that it poses a significant degree of resistance to the transfer of mass and energy across it. Consequently, the transfer of mass or energy is deduced from the difference in temperature or mass concentration across the *Connection* and the mass or heat transfer coefficient. The direction of material and energy flow is determined from the respective state variables of the two interlinked *Zones*, so the user does not need to specify the direction of flow. In addition to transferring mass and energy the *Connection* entity may also be used to maintain continuity in the pressure field between two *Zones* at an interface where there is no exchange of mass or energy. For this case, the pressure *Connection* simply matches the pressure at the *Zone's* boundary either to the pressure of a *Reservoir* or to the boundary pressure of the interlinked *Zone*.

To aid the procedure by which a given process is decomposed into an interlinked network of *Zones*, the *Vessel* entity can be used to group a number of *Zones* that form a section of a process unit. This permits a top-down strategy of hierarchical decomposition whereby the *Process* is first broken down into a number of *Vessels* and each *Vessel* is then further decomposed into a number of *Zones*. This provides additional clarity in the model-building framework, since the *Zone* entities, which in essence are merely the building blocks of the framework, can be assigned to objects of actual physical significance. As in previous work, Perkins et al. (1996) and Barber (2000), the *Vessel* entity is defined as a containing surface enclosing one or more *Zone*s. Each *Vessel* may be declared to be closed, in which case the cumulative volume of the *Zones* contained within the *Vessel* is equal to the total volume of the *Vessel*, or open, in which case the volume of the vessel is assumed to be large enough to accommodate all of the *Zones* contained within it.

Each *Zone* has a well-defined *Boundary*, which delimits to the region (extent) of material whose thermodynamic state can be predicted by the *Zone*. If the thermodynamic state of the material (one or two phases) and, where appropriate, the distribution of the two phases is lumped, the *Zone*'s extent is determined from the volume of material within the *Zone* and consequently there is no need to define the *Zone's Boundaries*. However, if either the thermodynamic state or the distribution of the two phases varies over the extent of the *Zone*, the *Zone's* extent must be established in terms of its *Boundaries*. To permit distributed *Zones* of fixed or varying extent to be modelled, the position of each *Boundary* may be fixed, achieved by declaring the co-ordinates of the *Boundary*, or may vary in time (a free *Boundary*). In the latter case, two *Zones* may share a common *Boundary*, so declaring the position of each *Boundary* is regarded as part of the

structural representation of the *Process*, rather than the phenomenological characterisation of the *Zones*. Examples of systems that involve moveable boundaries include:

1. A settling tank, where the extent of the two-phase distributed region is dependent on the build up of the heavy phase at the bottom of the tank.
2. A tube boiler, where the length of pipe that is susceptible to boiling depends on the length of pipe needed to raise the liquid temperature to its boiling point.
3. The process of freezing whereby the "freezing front" propagates through the medium.
4. The accumulation of an organic substrate layer in a extractive membrane bioreactor (Nicotella et al, 2000),

With the *Connection* entity making contact with only a fraction of the *Zone's Boundary*, it is necessary to specify the point on the *Zone* with which the *Connection* is in contact with. This is achieved through the *Port* entity (Cho, 1997; Barber, 2000). For the case where the extent of the *Zone* is defined by its *Boundaries*, the position of the *Port* is assigned to the *Boundary* that it lies on. If the *Zone* has no *Boundaries*, it is clearly not possible to assign a *Port* to a *Zone Boundary* and an additional description must be assigned to the *Port* to describe all characteristics of the *Port* that arise from its position (e.g. the hydrostatic head). As well as enabling the *Connection* to be assigned to a particular point on the *Zone Boundary*, the *Port* entity also ensures that regardless of the physical description of the *Zone* and the mathematical equations that are generated, all interfaces between any *Zone* and any *Connection* are the same. Consequently any combination of *Zones* and *Connections* of different physical behaviour can be achieved without model formulation errors.

In the process representation described so far the process is represented as an isolated entity, a black box, which, when given an input will determine an output. The input to the process can be seen as a connection between the process and the environment. In keeping with previous convention, (Perkins et al., 1996), the *Reservoir* entity is used to model all environments. The *Reservoir* may correspond to a pressure sink, a temperature sink or a material sink or source. An illustration of the structural representation is given in figure. 3.1, where the model for a heated tank is constructed from a network of *Zones*, *Reservoirs*, *Connections* and *Ports*.

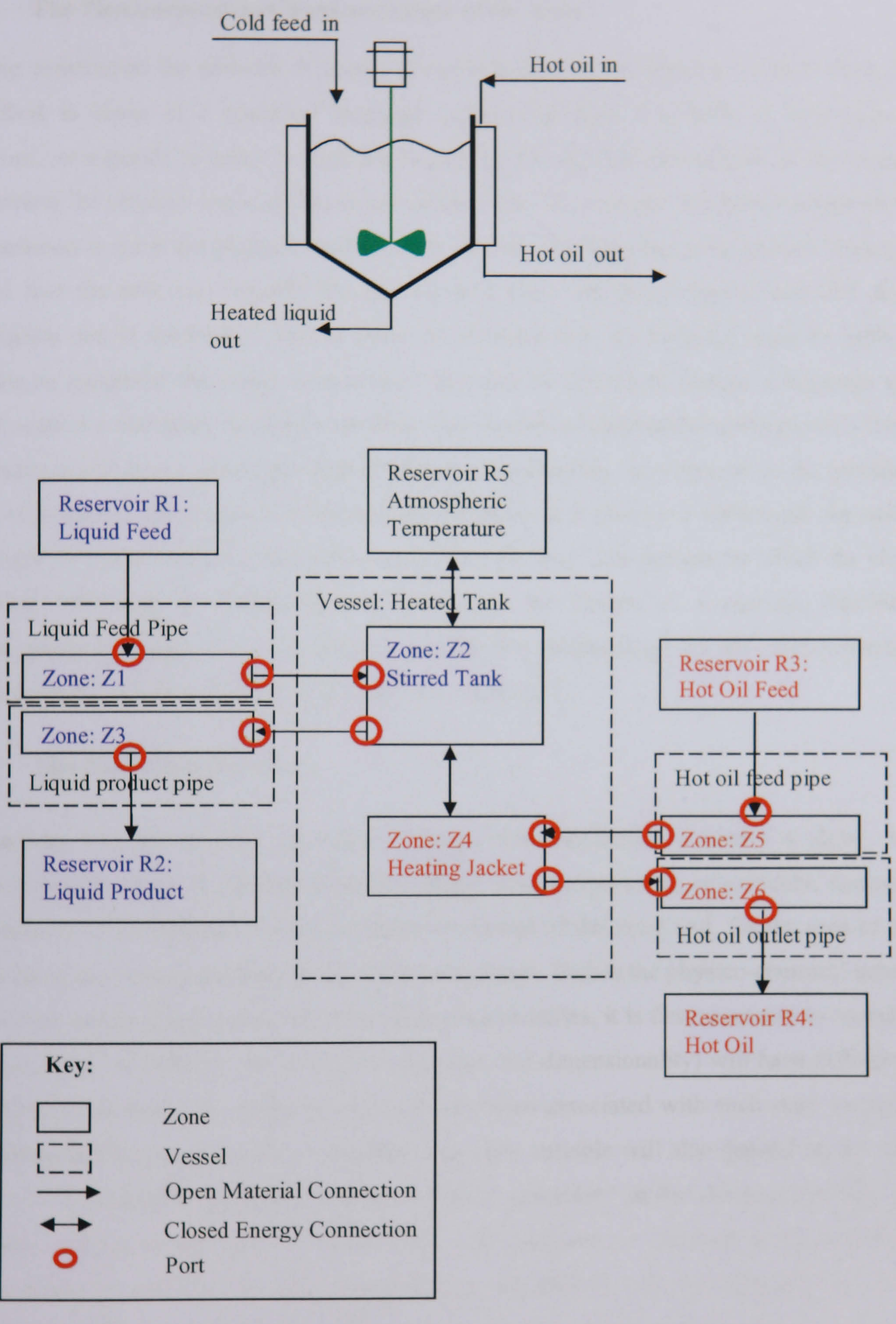


Figure 3.1: The Structural Representation for a Heated Tank

3.3 The Physicochemical Characterisation of the Zone

Having constructed the network of *Zone*s, the physico-chemical behaviour of each *Zone* can be described in terms of a specified language constructed from a number of keywords. Each keyword corresponds to either a particular aspect of the physical description or an assumption concerning the physico-chemical behaviour of the *Zone*. The manner in which the keywords can be combined to form the physical description is critical. The language must be well structured to ensure that the user can describe the process in a clear and unambiguous way and that this description can be interpreted without error. At the same time the language must be sufficiently flexible to maximise the scope of processes that can be described. Before a language can be constructed it is necessary to adopt a methodology capable of characterising the physico-chemical mechanisms that occur within the field of chemical engineering. Analogously to the grammatical rules of language, the physico-chemical characterisation will provide a framework for using the keywords in a non-restrictive and fully comprehensible way. The process by which the physico-chemical behaviour is characterised is critical to the design of a concise, flexible and unambiguous language. A well-structured, yet flexible methodology for the characterisation of the *Zone* entity will be presented in Sections 3.3.1 and 3.3.2.

3.3.1 The Zone Class Structure

In Section 3.1, the process modelling strategy was introduced whereby a *Zone* can be characterised in terms of its state variables. These were the pressure, temperature, composition and velocity of the material within the *Zone*, the extent of the *Zone* and, for the case of a two-phase *Zone*, the volume fraction of one of the two-phases. Before the physico-chemical behaviour of the *Zone* can be characterised in terms of its state variables, it is first necessary to consider the fact that *Zones* of different class (number of phases and dimensionality) will have different state variables. Furthermore, the physico-chemical behaviour associated with each state variable and the subsequent language needed to describe each state variable will also depend on the class of the *Zone*. For example, chemical diffusion, which is dependent on the chemical gradient within the *Zone*, will be present only for those *Zones* that experience a variation in the concentration profile across the extent of the *Zone*. Consequently, the physico-chemical characterisation of the *Zone* must be structured so that the facility to describe the process of diffusion is provided for all *Zones* whose component concentration is distributed, yet excluded for those *Zones* where the component concentration is lumped.

To ensure that the description of a *Zone* can be structured using the actual state variable that define the *Zone's* state, it is necessary, in the first instance, to devise a number of *Zone Classes*, which each *Zone* entity can be instantiated from[6]. Each *Zone Class* provides the appropriate methodology to characterise the physico-chemical behaviour of a particular type of *Zone* (single-phase or two-phase, lumped or distributed), thereby enabling the user to instantiate (classify) the *Zone* in a context-sensitive manner, incorporating all aspects of the *Zone's* behaviour that are relevant and excluding all aspects that are not. For each *Zone Class*, the methodology used to characterise the *Zone's* behaviour reflects the behavioural characteristics that are specific to a particular type of *Zone* whilst ensuring that behavioural characteristics common to a number of different *Zones* are characterised in a generic way. Each *Zone Class* not only contains the appropriate grammatical rules and vocabulary to describe the behaviour of the *Zone*, but also the necessary logic to formulate a process model from the physical description. Currently, five *Zone Classes* have been incorporated within the framework. The basic properties of the five classes are summarised in Table 3.1.

The single-phase *Zone Classes* provide the grammatical rules, vocabulary and modelling logic required to describe and model a *Zone* that encompass material in a single thermodynamic phase. The class system has been chosen to allow a single-phase *Zone Class* to describe an instance where the thermodynamic state or the velocity of the material varies over one of the *Zone's* dimensions. The same dimensional refinement can be found amongst the two-phase *Zone classes*, with the main difference between the two class types arising from the need to characterise the thermodynamic state, velocity and composition of the second phase. The no-extent *Zone* (instantiated from the no-extent *Zone Class)* is used to describe and model a region of the process that is sufficiently small for its volume and holdup to be neglected. This type of *Zone* is treated as a point, allowing algebraic equations, which express the output state variables in terms of the input state variables, to be assigned to the point. Examples of processes that are modelled with such a formulation include a pump, which is modelled with an instantaneous jump in the pressure, the perfect mixing of two streams, the flow of material through a pipe and the change in the flow conditions through a valve.

---

[6] instantiation: producing a particular object from its class template. This involves the allocation of a structure with the types specified by the template, and the initialisation of the instance variables.

3.3.2 The Subclass Structure

The physical description for each class of *Zone* is based around a unique template, which dictates both the content and the structure of the description. The structure of each template is centred on a number of key properties that collectively describe the physical state of the *Zone*. As discussed in the previous chapter, these properties include the composition, the two chosen state variables (pressure and temperature), the extent of the *Zone* and the velocity. The exact set of properties is dependent on the class of *Zone* (No-Extent, Two-Phase Lumped etc) and the reader should refer to Table 3.1, where the set of key properties is given for each *Zone Class*.

With the language template for each *Zone Class* structured around a set of key properties that describe the physical state of the *Zone*, the user can describe the *Zone's* behaviour by referring to each key property independently, listing the factors that influence its value and which consequentially should be considered within the model. By basing the language template for each *Zone Class* around the appropriate set of key properties, the language refers directly to the key properties (composition, temperature, pressure etc) of the *Zone*. This allows the behaviour of the *Zone* to be described in physical terms that are well understood by the user. Furthermore, as the physical description for the *Zone* is constructed from an aggregation of the physical descriptions assigned to each key property, a wide range of different systems can be described and modelled by adjusting the physical description assigned to each key property. An example of this powerful component-oriented approach to which a process can be described by is given in figure 3.2.

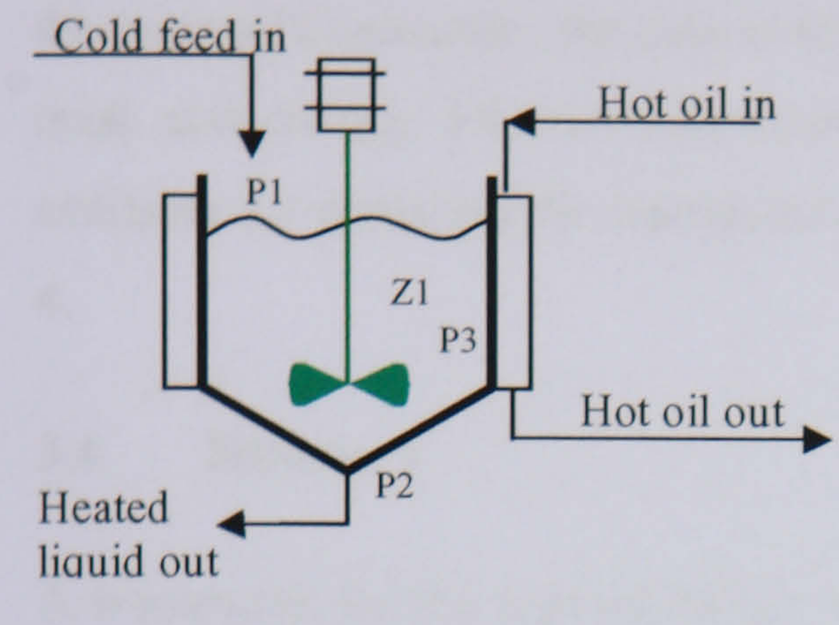
The grammatical rules that dictate how the physical description is established for each key property are contained within a *Zone Subclass*. A *Zone Subclass* exists for each key property within each *Zone Class* and provides both the grammatical rules and the necessary model-building logic required to interpret the physical description and construct a local model that describes the variation of the key property throughout the time horizon. For each key property, the user may provide a specific value or may request that *COMPASS* determines the value of the key property as part of the overall model. For the case where the key property is to be determined by *COMPASS*, the user may declare additional factors that influence the value of the key property and should be included within the model. Examples of this method of characterisation can be seen in figure 3.2.

Class	Number of Phases	Dimensions	Co-ordinates	Key Properties (Subclasses)
Single-Phase, Lumped Class:	1	No-Dimensions	No-Coordinates	Extent, Composition, Temperature, Pressure
Single-Phase, One-Dimensional, Duct Class:	1	One-Dimensional	Axial	Composition, Temperature, Pressure, Velocity
Two-Phase, Lumped Class:	2	No-Dimensions	No-Coordinates	Extent, Phase Composition, Phase Temperature, Pressure, Velocities, Volume Fraction, interfacial exchange of mass and energy
Two-Phase, One-Dimensional, Duct Class:	2	One-Dimensional	Axial	Phase Composition, Phase Temperature, Pressure, Phase velocities, Volume Fraction, Interfacial exchange of mass and energy
No-Extent Class	1	No-Dimensions	No-Coordinates	Flowrate, Heat and Work, Pressure Drop, Assumptions

Table 3.1: The Class and Subclass Structure

Through this process of specifying which key properties are to be determined and which are to be assigned a constant value, the user is able to distinguish between the transport processes (mass, energy and momentum) that are sufficiently significant and need to be included within the model and those that can be neglected. Regardless of which approach is used, a value will exist for each key property and hence all equations of state and empirical correlations can be expressed in terms of these key properties.

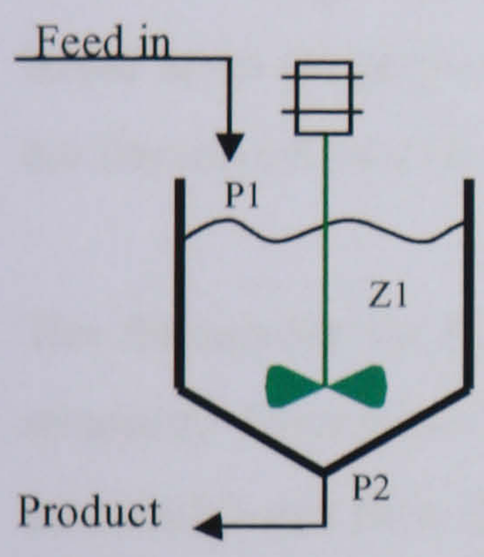
The temperature and composition *Subclasses* for the two-phase classes have been formulated so that the state variables exist for each phase. Consequently the temperatures and compositions of the two phases can be treated independently and it is therefore possible for the user to assign a constant value to the state variable of one phase and request that the program determines the state variable of the second phase. For instances where the temperature of the two phases is homogeneous, the framework allows the state variables for the temperatures of the two phases to be merged into a single variable representing the temperature of the two-phase mixture. For cases where the *Zone* is in chemical equilibrium, the composition of the two-phase mixture is initially determined and equilibrium constants are used to determine the composition of each phase.




```
ZONE : Heated_Tank_Z1 ; #corresponding to the contents of the tank

GEOMETRY : LUMPED ;
PHASE : process_stream ;
PORTS : P1 , P2 : INCLUDE_STATIC_HEAD, P3 ;
DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
        ZONE_DYNAMICS : DYNAMIC ;
        ZONE_EXTENT : VARIABLE ;
        ZONE_COMPOSITION : FIXED ;
                Butanol = 1 ;
        ZONE_TEMPERATURE : VARIABLE
        EXTERNALWORK ;
        ZONE_PRESSURE = 1.0e+5 ;
END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION

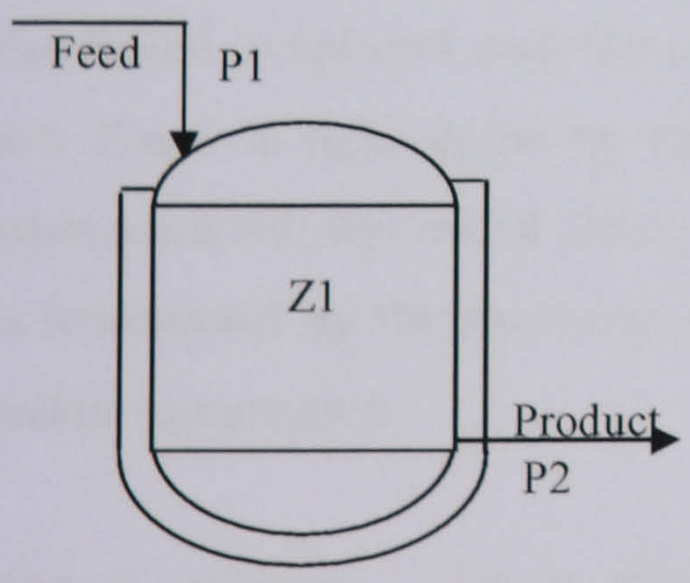
END #zone section
```




```
ZONE : Liquid_CSTR_Z1 ;

GEOMETRY : LUMPED ;
PHASE : process_stream ;
PORTS : P1, P2 ;
DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
        ZONE_DYNAMICS : DYNAMIC ;
        ZONE_EXTENT : VARIABLE ;
        ZONE_COMPOSITION : VARIABLE : REACTION ;
        ZONE_TEMPERATURE : VARIABLE : NEGLECT_KE
        : NEGLECT_PE;
        ZONE_PRESSURE = 1.0e+5;
END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION

END #zone section
```




```
ZONE : Isothermal_Gas_CSTR_Z1 ;

GEOMETRY : LUMPED ;
PHASE : process_stream ;
PORTS : P1, P2 ;
DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
        ZONE_DYNAMICS : DYNAMIC ;
        ZONE_EXTENT = 0.3 ;
        ZONE_COMPOSITION : VARIABLE : REACTION ;
        ZONE_TEMPERATURE = 800 ;
        ZONE_PRESSURE : VARIABLE ;
END# PHYSIOCHEMICAL_BEHAVIOUR_SECTION

END #zone section
```

Figure 3.2: The Description of a Heated tank, Liquid CSTR and Gas CSTR Written in Terms of the *COMPASS* Language. Note the choice of VARIABLE or specified constant values for key variables.

For all *Zones* that encompass two-phases, the exchange of heat and mass between the two phases is characterised by an *Internal Connection*, a *Subclass* within all of the two-phase classes. Within the *Internal Connection*, the user is able to specify the mechanisms for the interfacial exchange of mass and energy. Further information concerning the *Internal Connection* and the language available for declaring the interfacial exchange of mass and energy will be presented in Chapter 4.

3.4 Summary

A framework for the representation for one- and two- phase systems has been presented in this chapter. This representation is of sufficient generality to describe the physico-chemical behaviour of a wide range of chemical processes at varying levels of complexity. As the representation is based upon the physical characterisation of the fundamental principles of chemical engineering, the framework can be used to describe processes in purely physical terms.

The framework for the representation of chemical processes is based on a two-tier approach of structural decomposition and phenomenological characterisation. The structural decomposition uses traditional flow sheeting to decompose the *Process* firstly into a number of *Vessels* and then secondly into a number of *Zones*. To ensure full flexibility in the type of process that can be modelled, the *Zone* entity can be used to represent regions of the process of fixed or variable size, which encompass one or two stable thermodynamic phases and whose properties may be distributed or uniform over the extent of the region. The exchange of mass and energy between two *Zones* is represented by the *Connection* entity, allowing the *Process* to be seen as an interconnected network of *Zones and Connections*. The environment that surrounds the *Process* is represented by the *Reservoir* entity, which can be characterised by its pressure, temperature and/or composition.

With the network of *Zones*, *Reservoirs* and *Connections* in place, it is possible to assign each *Zone* a physical description, communicated through a number of keywords. The structure and vocabulary of this physical description is based on the phenomenological characterisation of the fundamental physico-chemical mechanisms of chemical engineering. In the first instance, the phenomenological characterisation is based on a number of *Zone Classes* that correspond to the dimensions of the *Zone* and the number of phases that the *Zone* encompasses. Each Zone is instantiated (characterised) from a particular *Zone Class* thus allowing the grammatical rules and vocabulary to be used in a manner sensitive to the type of *Zone*. For each *Zone Class*, the

methodology for characterising the physico-chemical behaviour of the *Zone* is based on the key properties of the *Zone Class*. These include the pressure, temperature(s), composition(s) and the velocity(ies) of the phase(s) and the extent of the *Zone*. This component-oriented approach to characterisation provides a powerful way of describing, and eventually modelling, a wide range of physico-chemical phenomena.

In the next chapter the language and grammar required to establish both the structural representation and the phenomenological characterisation of a given process will be discussed. Both the language and the grammar will be based on the *Zone Classes* and *Subclasses* discussed in this chapter.

Chapter 4

The Modelling Language

A framework for the generation of one- and two- phase process models was presented in Chapter 3. Having established the general framework for automation, it is now necessary to devise a declarative approach by which a given process can be described in terms of the framework objects that were presented in Chapter 3.

This chapter describes a specific language by which chemical processes can be described. The scope of the research is quite general, but does not cover all possible processes, as will become apparent in the following chapter. The language is based on the declaration of a set of keywords, which, where possible, are physical terms that require no further explanation. The development of the language is based upon the framework described in Chapter 3, thus ensuring that the language is easy to comprehend, has a clear physical interpretation and can be directly connected to the model-generating program.

4.1 Overview of the Modelling Language

The modelling language is comprised of a vocabulary of keywords and a set of grammatical rules that prescribe the order in which the keywords should be written. The language is case-sensitive and all keywords are expressed in capitals. In addition to describing the process in terms of the keywords, the user may provide additional information to incorporate within the model. Such information may be a name <name>, a number <number> or a reference identifier <id>. Both the names and the reference identifiers may consist of any combination of text, letters, numbers or underscores but must start with a letter. The numbers typically represent physical quantities, but no units are attached. Thus models must be couched in terms of a consistent set of units, such as SI, or cast in a dimensionless form.

The modelling language is composed of a number of sections, with each section comprised of a particular group of objects within the framework (Vessel, Phase, Component) that have been instantiated from the section class. For example the DEFINE_PHASE section is comprised of a number of PHASES. The language for each section can be expressed in its generic form as in definition 4.1

Language Definition 4.1: The Structure for each Section

```
DEFINE_<SECTION NAME>
        Section_Object_1

        Section_Object_2

        Section_Object_3

        ••••
END
```

To facilitate the understanding of the generated model and any error messages that may be printed, each section-object has a name and, where necessary, a reference identifier. The reference identifier is used to identify all equation terms. In addition to its name and reference identifier, each section-object will have a description assigned to it, declared in terms of a number of keywords and the grammatical rules defined for the section. Each section-object may also consist of a further set of sub-sections. Within the description, a colon is used to separate each keyword, user-supplied name or reference identifier. The equals symbol must precede a user-supplied number. A semi-colon indicates the end of a line.

To illustrate the language structure for a section-object, the language declaration for the Phase Object 'Process_stream' is presented in Example 4.1. Representing the components present in the gaseous synthesis of methanol, the language declaration for the 'Process_stream' commences with the keyword 'PHASE', followed by the name of the Phase Object (Process_stream) and a description of the phase. Within the description of the Process_stream, it is stated that the material is in its gaseous phase and that the material can be assumed to be incompressible. In the final part of the description, the chemical species within the phase are declared through the component sub-section. Following the section language structure defined in definition 4.1, the component section begins with the key word 'DEFINE_COMPONENTS' and ends with the keyword 'END'. The component section comprises of three Component Objects, CarbonMonoxide, Hydrogen and Methanol. The language declaration of each Component Object

commences with the keyword 'COMPONENT', followed by the name of the component and a reference identifier, to which the component is referred to within the model equations.

Example 4.1: Example of the Language for a Phase Object

```
PHASE : Process_stream : GAS : INCOMPRESSIBLE ;
      DEFINE_COMPONENTS
            COMPONENT : CarbonMonoxide : a ;
            COMPONENT : Hydrogen : b ;
            COMPONENT : Methanol : c ;
      END #component_section
```

The syntax for each section-object, expressed in terms of the keywords, user supplied identifiers and any further subsections, will be defined in this chapter through a series of language definitions. For each language definition, a concise format has been used whereby the different language combinations for each section-object are presented collectively within one language definition. As it has not been possible to list every language combination explicitly, the following formatting is used for each language definition to ensure that there is no confusion over which aspects of the language are prescriptive and which are optional.

1. Sections of the language in normal text are prescriptive.
2. Sections of the language definition in *italics* are optional and can be excluded from the physical description.
3. A set of different keywords contained within curly bracket indicates that the user should select one keyword from the set.
4. For instances where a set of keywords is followed by an additional declaration that relates to a specific keyword from the set, the keyword in question is underlined. For instances where the succeeding declaration relates to two or more keywords within the curly brackets, then the user should refer to Appendix I, where the complete language is given as part of the parser input specification. More information about the parser can be found in Chapter 6

At the highest level, the overall structure of the program can be seen to comprise of the keyword PROCESS, a user-supplied process name and five main sections corresponding to the Phase Section, Vessel Section, Reservoir Section, Connection Section and the Flowsheet Section, as shown in definition 4.2. Together these sections provide the necessary information to establish, describe and interlink the objects required to characterise the behaviour of a diverse range of

chemical processes. As there may be no interaction between the process and its environment, the use of the *Reservoir* entity may not always be necessary and consequently the Reservoir Section may be omitted from the language declaration.

Language Definition 4.2: The Overall Language Structure

PROCESS : <name> ;

DEFINE_PHASES
    Phase Objects
END

DEFINE_VESSELS
    Vessel Objects
END

DEFINE_RESERVOIRS
    Reservoir Objects
END

DEFINE_CONNECTIONS
    Connection Objects
END

DEFINE_FLOWSHEET
    Flowsheet Objects
END

4.2 The Phase Section

The conceptual approach used to establish the framework developed in Chapter 3 was based on the premise that all material within the process exists in well-defined thermodynamic phases. Consequently each framework entity (*Zone, Reservoir or Connection*) must be characterised in terms of the thermodynamic phase that the object represents or conveys.

All of the phases within the process are established on a global basis within the Phase Section. By defining the phases globally, each framework entity (Zone, Reservoir or Connection) can refer to the predefined list of phases to identify those that exist within the entity. Consequently the same phase may occur within several framework entities yet only needs to be defined once. This not only simplifies the physical description but also aids in the verification of the description as the user can be alerted to an error arising from the inappropriate assignment of a phase to a framework entity.

Each Phase Object is fully described by the set of components within the phase, the thermodynamic state (solid, liquid or gas) and the compressibility (compressible or incompressible) of the phase. If the Phase Object has been defined as incompressible, the density of the phase is assumed to be independent of the phase pressure.

The formal syntax for each Phase Object is given in definition 4.3. The syntax comprises the keyword PHASE followed by three attributes: a user-supplied name, the thermodynamic state of the phase, referred to through the keywords SOLID, LIQUID and GAS, and the compressibility of the phase, declared through the keywords COMPRESSIBLE and INCOMPRESSIBLE. Since each framework entity will refer to the phase through its name, the name must be globally unique, i.e. no two phases can share the same name.

In addition to defining the thermodynamic state and the compressibility of the phase, it is necessary to declare the set of chemical components for each phase. This is achieved within the Component Section, an integral part of each Phase Object. The formal syntax for each Component Object is given in definition 4.4. The declaration comprises the keyword COMPONENT, the name of the component and a reference identifier by which the component will be identified within the generated model. The name and reference identifier must be locally unique, with each component within the phase having a unique name and reference identifier. Components within different phases may share a common name and/or identifier. An example of the syntax for the Phase Object is given in Example 4.1.

Language Definition 4.3: Phase Object Declaration

PHASE : <name> : $\left\{\begin{matrix}\text{SOLID}\\ \text{LIQUID}\\ \text{GAS}\end{matrix}\right\}$: $\left\{\begin{matrix}\text{COMPRESSIBLE}\\ \text{INCOMPRESSIBLE}\end{matrix}\right\}$;

DEFINE_COMPONENTS
    Component_Object_1
    Component_Object_2
    • • •
END

Language Definition 4.4: Component Declaration

COMPONENT : <name> : <id> ;

4.3 The Vessel Section

In Section 3.2, the *Vessel* entity was defined as the region within a containing surface enclosing one or more *Zones*. In addition to grouping the *Zones* together, the *Vessel* entity may provide volumetric constraint on the set of *Zones* that lie within it. If the cumulative volume of the *Zones* within the *Vessel* is equal to the volume of the *Vessel*, the *Vessel* is defined as CLOSED. If the cumulative volume of the *Zones* does not fill the *Vessel*, the *Vessel* is defined as OPEN. For instances where the *Vessel* is defined as CLOSED and contains more than one *Zone*, the cumulative volume of the *Zones* must equal the volume of the *Vessel* and it is therefore necessary to specify the *Vessel* volume.

The formal syntax for the declaration of each *Vessel* Object commences with the keyword VESSEL followed by the vessel name, the vessel reference identifier, a declaration of whether the *Vessel* is CLOSED or OPEN and, where appropriate, a number giving the volume of the *Vessel*. The name of the *Vessel* and the reference identifier must be unique. The remainder of the *Vessel* declaration specifies the *Zones* that occupy the *Vessel*. This is done through the Zone Section. The syntax for each Vessel Object is shown in language definition 4.5.

Language Definition 4.5: Vessel Object Declaration

VESSEL : <name> : <id> : $\left\{ \begin{matrix} \text{OPEN} \\ \underline{\text{CLOSED}} \end{matrix} \right\}$: <*number*> ;

DEFINE_ZONES
    Zone_Object_1
    Zone_Object_2
    • • •
END

4.4 The Zone Section

Defined as a local model for either a two-phase system or a single-phase region, the *Zone* is the key building block of the framework. To enable the *Zone* entity to model a wide range of different processes, five different classes of *Zone* have been created (Section 3.3.1).

The formal declaration of the Zone entity must first identify the class of *Zone* that the user wishes to describe. With the specific class identified the user can then declare the structural elements of the *Zone*, specified through the *Zone's Ports*, which represent the interface between the *Zone* and its environment (other *Zones* or *Reservoirs*), and where appropriate the *Zone's Boundaries*. The

final stage of the declaration process is to describe the physico-chemical behaviour of the *Zone*. This description is sensitive to the class of *Zone* in question. The overall language definition for each Zone Object is shown in definition 4.6.

Language Definition 4.6: Zone Object Declaration

ZONE : <name> ;
Zone Class Subsection
Zone Structure Subsection
Zone Physicochemical Behaviour Subsection

4.4.1 The Zone Class Subsection

With the sole purpose of identifying the class of *Zone* that the user wishes to describe and model, the syntax for the Zone Class Subsection comprises three declarations: the geometry of the Zone (GEOMETRY), the coordinates over which the state of the Zone varies (COORDINATES) and the phases that occupy the *Zone* (PHASE). The GEOMETRY refers to the spatial distribution (NO-EXTENT, LUMPED, 1-D) required to model the *Zone*. For *Zones* with a one-dimensional geometry it is also necessary to specify the domain axis. In the current state, only Cartesian coordinates can be modelled and this is declared by setting the COORDINATES to CARTESIAN. Finally the phases that occur within the *Zone* are specified within the PHASE declaration. The name of each *Phase* must correspond to a *Phase* that has previously been defined within the Phase Section. If two phases are declared, the phase names must be separated with a comma. The formal syntax for the Zone Class Subsection is given below in definition 4.7.

Language Definition 4.7: Zone Class Subsection Declaration

$$\text{GEOMETRY} : \left\{ \begin{matrix} \text{NOEXTENT} \\ \text{LUMPED} \\ \underline{1-D} \end{matrix} \right\} ;$$

COORDINATES : CARTESIAN;

PHASE : <name> , <*name*> ;

4.4.2 The Zone Structure Subsection

The Zone Structure Subsection provides the user with the facility to define the structure of the *Zone*. The *Zone* structure is encapsulated within the framework by the *Zone's Ports* and, where appropriate, the *Zone's Boundaries* and the orientation of the *Zone*. It should be noted that the

Zone's Boundaries are only declared for *Zones* that have a one-dimensional spatial distribution and the *Zone's* orientation (ORIENTATION) is only specified if the COORDINATES have been declared as CARTESIAN. The orientation of the *Zone* is defined as the angle (in degrees) between the Zone's axis and the horizontal plane.

Language Definition 4.8: Zone Structure Subsection Declaration

ORIENTATION : <number> ;
DEFINE_BOUNDARIES
Boundary_Object_1
Boundary_Object_2
• • • •
END

DEFINE_PORTS
Port_Object_1
Port_Object_2
• • • •
END

The Boundary Object: The formal syntax declaration for each Boundary Object is given in definition 4.9. The language may refer to a radial boundary, corresponding to the *Zone's* wetted perimeter, or an axial boundary, corresponding to either the Zone's inlet or outlet. The syntax for the Boundary Object comprises the keyword BOUNDARY, a reference to the type of boundary through the keyword AXIAL or RADIAL and two attributes: a user-specified boundary name and the position of the *Boundary*. If the position of the *Boundary* remains fixed in space, then it should be declared. Alternatively, if the position of the *Boundary* varies with time, the *Boundary* should be declared as FREE and a user-supplied submodel should be added by hand to predict the position of the *Boundary*. Since *COMPASS* deals with scalar terms there is no need to specify the position with reference to a particular origin, however all positions must be positive. The user-specified boundary name must be locally unique, (i.e. no two *Boundaries* within the same *Zone* can share the same name) however *Boundaries* in different *Zones* may use the same name.

Language Definition 4.9: The Boundary Object

$$\text{BOUNDARY : <name> : } \left\{ \begin{matrix} \underline{\text{AXIAL}} \\ \underline{\text{RADIAL}} \end{matrix} \right\} : \left\{ \begin{matrix} \text{number} \\ \text{FREE} \end{matrix} \right\} ;$$

The Port Object: Designed to provide the interface between the *Zone* and its environment, the *Port* entity has been designed to be versatile. Consequently the *Port* declaration is sensitive to the

class of *Zone* to which it is attached. The language definition for the *Port* entity is given in definitions 4.10, 4.10a and 4.10b.

The *Port* object declaration commences with the keyword PORT and a *Zone*-specific, user-supplied *Port* name. If the geometry of the *Zone* is one-dimensional it is necessary to declare the boundary to which the *Port* is attached. The *Boundary* must have previously been declared within the DEFINE_BOUNDARY section and the *Boundary* is referred to through the *Boundary* name. For instances where the *Port* is connected to a two-phase *Zone* it is also necessary to specify the phase to which the *Port* corresponds. Once again the phase must have previously been declared within the Zone Class Subsection.

For the Two-phase Lumped Zone Class, it is possible that there may be relative motion between the two-phases within the *Zone*. This may give rise to a mass flowrate at the *Port*. Unlike the Single-phase Lumped Zone Class, where an additional *Zone* determines the mass flowrate through the *Port*, the mass flowrate at the *Port* must be determined from the relative motion of the two phases. This is achieved through the keyword PORT_MASS_FLOWRATE. When determining the mass flowrate at the *Port* it is necessary to consider both the relative motion between the two phases and the velocity of the *Port*. Currently the velocity at the *Port* may be declared by the user, related to the change in volume of another *Zone* within the *Vessel* (VOLUME_GAIN), or, if the *Port* lies on the interface between a two-phase region and a single-phase region, equated to the velocity at which the interface moves through the *Vessel* (PHASE_DEPLETION).

Finally for all Lumped Zone Classes, it is possible to model the hydrostatic pressure of the *Zone* by considering the hydrostatic head when relating the pressure at each *Port* to the pressure of the *Zone*. This is achieved through the declaration INCLUDE_ HYDROSTATIC_PRESSURE. If this term is omitted, the pressure of the *Port* is simply equated to the pressure of the *Zone*.

Language Definition 4.10: The Port Object

PORT : <name> : *<Boundary_name>* : *<Phase_name>*
Mass Flowrate Section
Hydrostatic Pressure Section
;

Language Definition 4.10a: The Mass Flowrate Section

$$: \text{PORT_MASS_FLOWRATE} \left\{ \begin{array}{c} =< \text{number} > \\ : \text{VOLUME_GAIN} : < \text{zone_name} > \\ : \text{PHASE_DEPLETION} \end{array} \right\}$$

Language Definition 4.10b: The Hydrostatic Pressure Section

: INCLUDE_HYDROSTATIC_PRESSURE

4.4.3 The Physicochemical Behaviour Subsection

The declaration of the physicochemical behaviour for each *Zone* is structured around a number of language subsections, with every *Zone Subclass* represented by one language subsection. As different *Subclasses* exist for different *Zone Classes*, the language is sensitive to the class of *Zone* that the user is describing. For further discussion of the *Zone Classes* and the *Zone Subclasses* please refer to Sections 3.3.1 and 3.3.2 and Table 3.1. In addition to the set of language subsections that represent the *Zone Subclasses*, a language subsection also exists to allow the user to specify the dynamics of the *Zone*. A full list of the language subsections is given in definitions 4.14, 4.15, 4.16, 4.17 and 4.18. For each *Zone*, the Physicochemical Behaviour Subsection begins with the key word DEFINE_PHYSICOCHEMICAL_ BEHAVIOUR and ends with the keyword END. The generic syntax for each Physicochemical Behaviour Subsection is given in definition 4.11.

Language Definition 4.11: The Generic Syntax for the Physicochemical Behaviour Subsection

```
DEFINE_ PHYSICOCHEMICAL_ BEHAVIOUR ;
        Language Subsection_1
        Language Subsection_2
        Language Subsection_3
        • • •
END
```

The syntax for each language subsection begins with the name of the subsection written in capitals (ZONE_EXTENT, ZONE_COMPOSITION, ZONE_TEMPERATURE and ZONE_PRESSURE). If the language subsection refers to a state variable (Temperature, Pressure, Volume Fraction), the subsection name is followed either by the value of the state variable or a

list of attributes that are to be considered when calculating the state variable. A table of possible attributes is listed for each *Zone Class* in Tables 4.1 to 4.4. If the language subsection does not represent a *Zone Subclass* (*Zone* dynamics) the subsection name should be followed by the appropriate set of attributes. Each attribute is preceded by a colon and, if the user specifies the value of the key attribute, the equal sign must precede the value. A semicolon completes the subsection declaration. The generic syntax for each subsection is given in declaration 4.12.

Language Definition 4.12: The Generic Syntax for each Subsection

$$\text{Subsection Name} \left\{ \begin{array}{c} : \text{ATTRIBUTE_1} : \text{ATTRIBUTE_2} \\ = < \text{number} > \end{array} \right\} ;$$

If the language subsection represents a *Zone Subclass* that contains two or more state variables (e.g. the Composition Subclass or the Temperature Subclass for a two-phase Zone) it is necessary to provide the facility to describe both the Subclass and each state variable within the Subclass. This is done through language definition 4.13.

Language Definition 4.13: The Generic Syntax for each Subsection containing multiple State Variables

$$\text{Subclass Name} : \text{ATTRIBUTE_1} : \text{ATTRIBUTE_2} ;$$

$$\left\{ \begin{array}{c} < \text{Component_1_name} > \\ < \text{Phase_1_name} > \end{array} \right\} \left\{ \begin{array}{c} : \text{ATTRIBUTE_1} : \text{ATTRIBUTE_2} \\ = < \text{number} > \end{array} \right\} ;$$

$$\left\{ \begin{array}{c} < \text{Component_2_name} > \\ < \text{Phase_2_name} > \end{array} \right\} \left\{ \begin{array}{c} : \text{ATTRIBUTE_1} : \text{ATTRIBUTE_2} \\ = < \text{number} > \end{array} \right\} ;$$

• • • • •

4.4.3.1 Declaration of the Physicochemical Behaviour for the No Extent Zone Class

In Section 3.3.1, the No Extent Zone Class was established to represent regions of the process that can be regarded as containing no material and whose behaviour can be represented by a jump in the state of the system. For the No Extent Zone Class, the *Subclasses* correspond to the flow rate, the heat and work and the pressure drop. A language subsection exists for each *Subclass*. The formal declaration of the Zone Physicochemical Behaviour Subsection for the No-Extent Zone Class is given in definition 4.14.

Language Definition 4.14: The Physicochemical Behaviour Declaration for the No Extent Zone

```
DEFINE_PHYSICO_CHEMICAL_BEHAVIOUR
        Zone_Flowrate Subsection
        Zone_Heat_and_Work Subsection
        Zone_Pressure_Drop Subsection
        Zone_Assumption Subsection
END
```

The attributes for each language subsection are presented in Table 4.1

Language Subsection	Attributes
Zone_Flowrate	PRESSURE_DRIVEN, CONTINUITY
Zone_Heat_and_Work	HEATFLOW, EXTERNAL_WORK, NONE
Zone_Pressure_Drop	NOT_SPECIFIED, USER_DEFINED_SUBMODEL
Zone_Assumption	ISOTHERMAL, ADIABATIC, POLYTROPIC, NEGLECT_KE, NONE

Table 4.1: The Attributes for the No Extent Zone Class

The purpose of the Zone_Flowrate Subsection is to specify the mass flowrate through the *Zone*. When defining the flowrate it is important to distinguish between a multi input / output *Zone* and a single input / output *Zone*. For a single input / output *Zone* of no extent, the user may either state the mass flowrate explicitly or may declare that the mass flowrate is PRESSURE_DRIVEN. If the user chooses the latter, a generic equation relating the mass flowrate to the pressure drop across the *Zone* is generated. For a multiple input / output *Zone*, the user must declare that CONTINUITY occurs within the *Zone*, such that the sum of the output mass flowrates is equal to the sum of the input mass flowrates.

The Zone_Heat_and_Work Subsection provides the user with the facility to specify whether there is an exchange of heat or the transmission of work between the *Zone* and the *Environment* (HEATFLOW and EXTERNALWORK). Through the Zone_Pressure_Drop Subsection, the pressure drop across the *Zone* may either be declared explicitly, implicitly through a USER_DEFINED_SUBMODEL or not at all (NOT _SPECIFIED). For the latter case, the pressure of each *Port* must be equated to the pressure of another *Zone*, through the appropriate *Connection*.

Finally there may be additional assumptions that need to be considered when calculating the change in the temperature across the *Zone*, the external work and the heat flow that is transmitted between the *Zone* and the *Environment*. These assumptions are declared within the

Zone_Assumption Section. If the flow of material through the Zone is POLYTROPIC, it is necessary to specify the polytropic coefficient.

4.4.3.2 Declaration of the Physicochemical Behaviour for the Single Phase Lumped Zone Class

In Section 3.3.1, the Single Phase Lumped Zone Class was established to model regions of the process that contain material in one stable phase and whose state is uniform across the extent of the *Zone*. For the Single Phase Lumped Zone Class, the *Subclasses* correspond to the extent of the *Zone* and the composition, temperature and pressure of the phase. A language subsection exists for each *Subclass* as well as for the dynamics of the *Zone*. The formal declaration for the Zone Physicochemical Behaviour Subsection for the Single Phase Lumped Zone Class is given in definition 4.15.

Language Definition 4.15: Physicochemical Behaviour Declaration for the Single-Phase Lumped Zone

```
DEFINE_PHSYICO_CHEMICAL_BEHAVIOUR
        Zone Dynamics Subsection
        Zone Extent Subsection
        Zone Composition Subsection
        Zone Temperature Subsection
        Zone Pressure Subsection
END
```

The attributes for each language subsection are presented in Table 4.2.

Subsection	Attributes
Zone_Dynamics	STEADY_STATE, DYNAMICS
Zone_Extent	VARIABLE
Zone_Composition	VARIABLE, FIXED, REACTION
Zone_Temperature	VARIABLE, HEATFLOW, EXTERNAL_WORK, NEGLECT_KE, NEGLECT_PE
Zone_Pressure	VARIABLE

Table 4.2: The Attributes for the Single-Phase Lumped Zone Class

For the Extent, Composition, Temperature and Pressure Subsections, the user may either specify the value of the state variable or declare the state variable to be VARIABLE. If the composition of the *Zone* has been declared as VARIABLE, the user may also specify that a REACTION occurs within the *Zone*. Currently *COMPASS* expresses all reactions in a generic manner and specific reaction kinetic equations must be entered into the generated model by hand. In such cases the user is required to replace the generic rate equation that *COMPASS* has generated, in either the gPROMS input file or the generated text file, with the specific rate equations. For non-

isothermal processes, the Zone_Temperature must be declared as VARIABLE and the user should specify whether there is an internal heat source[7] (HEATFLOW) and a transmission of work (EXTERNAL_WORK) between the *Zone* and its environment. The attributes NEGLECT_KE and NEGLECT_PE refer to the kinetic energy and the potential energy of the material flowing into the *Zone*.

4.4.3.3 Declaration of the Physicochemical Behaviour for the Single Phase 1-D Duct Zone Class

In Section 3.3.1, the Single Phase Duct Zone Class was established to model regions of the process that contained material in one stable phase and whose state was distributed along the axial dimension of the *Zone*. For the Single Phase Duct Zone Class, the *Subclasses* correspond to the composition, temperature, pressure and velocity. A language subsection exists for each *Subclass* as well as for the dynamics of the *Zone*. The formal declaration for the Zone Physicochemical Behaviour Subsection for the Single Phase Lumped Zone Class is given in definition 4.16.

Language Definition 4.16: Physicochemical Behaviour Declaration for the 1-D Duct Zone Class

```
DEFINE_PHSYICO_CHEMICAL_BEHAVIOUR
        Zone_Dynamics Subsection
        Zone_Composition Subsection
        Zone_ Temperature Subsection
        Zone_ Pressure Subsection
        Zone_Velocity Subsection
END
```

For the Composition, Temperature and Pressure Subsections, the user should either specify the value of the state variable or declare the state variable to be VARIABLE. If the state variable has been declared as VARIABLE, additional attributes may be selected from Table 4.3.

Subsection	Attributes
Zone_Dynamics	STEADY_STATE, DYNAMICS
Zone_Composition	VARIABLE, FIXED, REACTION, DIFFUSION
Zone_Temperature	VARIABLE, DIFFUSION, NEGLECT_KE
Zone_Pressure	VARIABLE, NEGLECT_WALL_SHEAR
Zone_Velocity	BULK_FLOW, DIFFUSIVE_FLOW, SUPERFICIAL_VELOCITY

Table 4.3: The Attributes for the Single-Phase 1-D Duct Zone Class

[7] External heat sources are represented by a Closed Energy Connection (Section 4.6).

The formal syntax for the Zone_Velocity Subsection has been written to allow the user to model three different flow scenarios.

1. The Velocity of the Material is known. Whether the material is stationary or flowing, the velocity can be explicitly declared as in definition 4.12. If the material was stationary the following declaration would be made; "ZONE_VELOCITY = 0 ;".
2. There is bulk flow of material through the *Zone* (BULK_FLOW). For this scenario the velocity of material can either be determined throughout the *Zone* or alternatively from the mass flowrate of material at the *Zone's* inlet boundary with the additional attribute SUPERFICAL_ VELOCITY.
3. There is diffusion of material but no overall bulk flow of material through the *Zone* (DIFFUSIVE_FLOW). In such cases the mixture velocity is the sum of the mass averaged component velocities. For instances where there is an equimolar two-component phase or the concentration of the diffusing component is very small, the velocity of the mixture will be negligible. In this case, the velocity can be set to zero, through a declaration as in definition 4.12.

4.4.3.4 Declaration of the Physicochemical Behaviour for the Two Phase Lumped Zone Class

The Two Phase Lumped Zone Class was established to model regions of the process that contain material in two stable phases and whose state is uniform across the extent of the *Zone*. The formal syntax for the Two Phase Lumped Zone Class builds on the language structure established for the Single Phase Lumped Zone Class, with additional consideration given to:

1. The possible relative velocity between the two-phases.
2. The need to describe the temperature either for the two-phase mixture as a whole or for each phase independently.
3. The exchange of heat and mass between the two phases.
4. Possible chemical equilibrium between the two phases.
5. Possible physical equilibrium (eg. Boiling or condensation) between the two phases.
6. The need to describe the volume fraction of one of the two phases.

For the Two Phase Lumped Zone Class, the *Subclasses* correspond to the extent, composition, interfacial exchange of mass, temperature, pressure, volume fraction and velocity. A language subsection exists for each *Subclass* as well as for the dynamics of the *Zone*. The formal

declaration for the Zone Physicochemical Behaviour Subsection for the Two Phase Lumped Zone Class is given in definition 4.17. It should be noted that if there is no interfacial exchange of mass the Internal_Mass_Transfer Subsection should be omitted.

Language Definition 4.17: The Two-Phase Lumped Zone Class Physicochemical Behaviour

```
DEFINE_PHYSICO_CHEMICAL_BEHAVIOUR
        Dynamics Subsection
        Zone_Extent Subsection
        Zone_Composition Subsection
        Internal_Mass_Transfer Subsection
        Zone_ Temperature Subsection
        Zone_Pressure Subsection
        Phase_Volume_Fraction Subsection
        Zone_Velocity Subsection
END
```

The possible attributes that can be used for each language subsection are tabulated in Table 4.4.

Subsection	Attributes
Zone_Dynamics	STEADY_STATE, DYNAMICS
Zone_Extent	VARIABLE
Zone_Composition	Phase Composition: VARIABLE, FIXED, REACTION
Interfacial_Mass_Transfer	EQUILIBRIUM, BOILING, ABSORPTION
Zone_Temperature	Zone Temperature: VARIABLE, BOILING_ISOTHERM, HEATFLUX, EXTERNALWORK, NEGLECT_KE, NEGLECT_PE, INTERNAL_HEAT_TRANSFER Phase Temperature: VARIABLE, BOILING_ISOTHERM, HEATFLUX, EXTERNALWORK, NEGLECT_KE, NEGLECT_PE
Zone_Pressure	VARIABLE
Phase_Volume_Fraction	VARIABLE, FIXED
Zone_Velocity	Definitions 4.17b to 4.17f

Table 4.4: The Attributes for the Two-Phase Lumped Zone Class

The Lumped Two-Phase Composition Subsection allows the user to declare the composition of each phase independently using definition 4.13. For each phase the composition may be assigned by the user or set as VARIABLE. If the composition of either phase is defined as VARIABLE, the user may also declare that a reaction occurs within the phase using the keyword REACTION.

The Internal Mass Transfer Subsection allows the user to declare the transfer of mass between the two phases within the *Zone*. The current modelling scope can handle boiling (BOILING), absorption (ABSORPTION) and chemical equilibrium (EQUILIBRIUM). For both the BOILING and ABSORPTION declarations it is necessary to specify the direction of mass transfer. This is achieved through an arrow (->), with the name of the depleted phase declared to the left of the arrow and the name of the enriched phase declared to the right of the arrow. Both phase names

must refer to phases that have previously been declared and assigned to the *Zone*. For absorption, it is also necessary to declare the components that are absorbed. This is achieved through the Component_Set, a list of components with each entry separated by a comma. Both phases must contain the absorbed components. The interfacial exchange of mass can be declared through the formal language syntax given in definition 4.17a.

Language Definition 4.17a: The Internal Mass Transfer Subsection Declaration

INTERFACIAL_MASS_TRANSFER :

$$\left\{\begin{array}{c} \text{EQUILIBRIUM} \\ \text{BOILING : < phase_name > - > < phase_name >} \\ \text{ABSORPTION : < phase_name > - > < phase_name > : Component_Set} \end{array}\right\};$$

The Lumped Two-Phase Temperature Subsection allows the user to declare the temperature of each phase independently (using definition 4.13) or to declare the temperature of the two-phase mixture as a whole (using definition 4.12). Similar to the Lumped Single-Phase Temperature Subsection, if the temperature of the phase or mixture varies, the user should declare the additional factors that influence the temperature (Table 4.4). If the temperatures of the two phases are different, the user can model the internal transfer of heat through the keyword INTERNAL_HEAT_TRANFER. It should be noted that if there is mass transfer between the two phases, the exchange of energy associated with the transfer of mass is automatically considered. Within the Two-Phase Volume Fraction Section, the user may either declare the volume fraction of each Phase (using definition 4.13) or specify that the volume fraction must be determined by *COMPASS*.

The formal syntax to describe the velocity for the Two-Phase Lumped Zone class has been devised so that a wide variety of different flow scenarios can be handled. By providing the facility to describe the bulk flow of the material and the motion of each phase independently it is possible to describe a two-phase mixture where either both phases are stationary or in a state of movement. Furthermore, the two phases may flow counter-currently or co-currently at different or with identical velocities.

The syntax for the Two-Phase Lumped Velocity Subsection commences with the keyword ZONE_VELOCITY. If both phases are stationary the user should set the velocity of each phase to zero.

Language Definition 4.17b: The Zone Velocity Subsection

$$\text{ZONE_VELOCITY}\left\{\begin{array}{c}:\text{STATIONARY}\ ;\\ \\ ;\end{array}\right\}$$

Mixture_Bulk_Flow Subsection
Relative_Motion Subsection
Flow_Regime Subsection
Phase_Motion Subsection

If the ZONE_VELOCITY has been set to zero (STATIONARY) no further declaration is required. Alternatively, if either phase flows within the *Zone*, the user should describe both the velocity of the mixture and the velocity of each phase through the four subsections given in definition 4.17b. If either phase does flow within the Zone it is first necessary to state whether there is a bulk flow of material within the *Zone*[8]. The formal declaration for the bulk flow of material is given in declaration 4.17c.

Language Definition 4.17c: Mixture Bulk Flow Declaration

$$\text{MIXTURE_BULK_FLOW}:\left\{\begin{array}{c}\text{BULK_FLOW}\\ \text{NO_BULK_FLOW}\end{array}\right\};$$

The second stage in describing the Zone's velocity is to declare whether there is relative motion between the two phases (RELATIVE_PHASE_MOTION) (i.e. do the two phases travel at the same velocity?).

Language Definition 4.17d: Mixture Bulk Flow Declaration

$$\text{INTERNAL_RELATIVE_MOTION}:\left\{\begin{array}{c}\text{NONE}\\ \text{RELATIVE_PHASE_MOTION}\end{array}\right\};$$

Within the current scope of the framework, *COMPASS* is able to handle flow scenarios where the relative motion arises from a difference in the density of the two phases. If the two phases travel at different velocities, the ORIENTATION must have previously been set to VERTICAL and it is necessary to describe the fluid dynamics of the two phases by referring to the occurring flow regime (figure 2.2). By stating the flow regime, appropriate theoretical correlations and experimental equations can be used to predict the relative motion between the two phases. A description of four of the five flow patterns illustrated in figure 2.2 is given by Levy (1999).

---

[8] It should be stressed that the movement of material within the *Zone* does not necessitate that there is an overall bulk flow of material as one phase may merely displace the other.

Bubble Flow: *The liquid is the continuous phase and the gas is dispersed in the liquid in the form of bubbles of varying size and shape.*

Slug Flow: *Slug flow is characterized by large gas bubbles (Taylor bubbles) almost filling the channel and separated by slugs of liquid.*

Annular flow. *Annular flow consists of an annular liquid film and a gas core with or without drops of liquid in it.*

Spray, mist, *The gas occupies most of the cross-sectional area and the liquid is in Dispersed the form of small droplets dispersed in the gas.*

The remaining flow regime is referred to as churn flow and is considered as a transition region between the Slug and Annular regimes. The churn regime is characterized by the merging of two Taylor bubbles, causing a highly agitated liquid "froth" of different sized and shaped bubbles dispersed within the liquid phase. The formal language used to declare the occurring flow regime is given in declaration 4.17e. It should be noted that the dispersed and bubble flow regimes have been merged together.

Language Definition 4.17e: Mixture Bulk Flow Declaration

$$\text{FLOW_REGIME} : \left\{ \begin{array}{c} \text{DISPERSED} \\ \text{SLUG} \\ \text{CHURN} \\ \text{ANNULAR} \end{array} \right\};$$

With the flow regime stated, the next stage of the physical description is to declare which phase is dispersed and which phase is continuous. Although dispersed and continuous phases only exist for dispersed flow, it is useful to adopt this terminology for the other flow regimes. By doing so the appropriate constitutive equations are formulated for the correct phase. The dispersed phase corresponds to the bubbles in bubbly flow, the slugs in slug flow, the gas phase in churn flow and the gas core in annular flow. At present only vertical flow regime are incorporated into the code in order to generate a few useful models. It would be possible, though tedious, to do the same for horizontal, or even inclined, flows.

In addition to declaring the continuous and dispersed phases, for vertically orientated *Zones* it is also necessary to specify whether each phase flows against gravity (UPWARD_FLOW) or with gravity (DOWNWARD_FLOW). If the phase is stationary the keyword STATIONARY should be used. The syntax for the Phase Motion Subsection is given in declaration 4.17f.

Language Definition 4.17f: Mixture Bulk Flow Declaration

$$\text{CONTINUOUS_PHASE : <name> : } \left\{ \begin{matrix} \text{DOWNWARD_FLOW} \\ \text{UPWARD_FLOW} \\ \text{STATIONARY} \end{matrix} \right\} ;$$

$$\text{DISPERSED_PHASE : <name> : } \left\{ \begin{matrix} \text{DOWNWARD_FLOW} \\ \text{UPWARD_FLOW} \\ \text{STATIONARY} \end{matrix} \right\} ;$$

4.4.3.5 Declaration of the Physicochemical Behaviour for theTwo-Phase 1-D Duct Zone Class

In Section 3.3.1, the Two Phase Duct Zone Class was established to model regions of the process that contain material in two stable phases and whose state is distributed along the axial dimension of the *Zone*. For the Two Phase Duct Zone Class, the *Subclasses* correspond to the composition, interfacial exchange of mass, temperature, pressure, volume fraction and velocity. A language subsection exists for each *Subclass* as well as for the dynamics of the *Zone*. The formal declaration for the Zone Physicochemical Behaviour Subsection for the Two Phase 1-D Duct Zone Class is given in definition 4.18.

Language Definition 4.18: Physicochemical Behaviour of The Two-Phase 1-D Duct Zone Class

```
DEFINE_PHYSICO_CHEMICAL_BEHAVIOUR
        Dynamics Subsection
        Zone_Composition Subsection
        Internal_Mass_Transfer Subsection
        Zone_Temperature Subsection
        Zone_Pressure Subsection
        Phase_Volume_Fraction Subsection
        Zone_Velocity Subsection
END
```

The possible attributes that can be used for each Subsection are tabulated in Table 4.5.

Subsection	Attributes
Zone_Dynamics	STEADY_STATE, DYNAMICS
Zone_Composition	Phase Composition: VARIABLE, FIXED, REACTION, DIFFUSION
Interfacial_Mass_Transfer	BOILING, ABSORPTION
Zone_Temperature	Zone Temperature: VARIABLE, BOILING_ISOTHERM, NEGLECT_KE, INTERNAL_HEAT_TRANSFER Phase Temperature: VARIABLE, BOILING_ISOTHERM, NEGLECT_KE
Zone_Pressure	VARIABLE
Phase_Volume_Fraction	VARIABLE, FIXED
Zone_Velocity	See definitions 4.17b to 4.17f and the following text

Table 4.5: The Attributes for the Two-Phase 1-D Duct Zone Class

For a Two-Phase 1-D Duct Zone, the orientation of the Zone has already been defined within the Zone Structure Subsection and the direction in which each phase flows can be determined from the position of the inlet and outlet *Ports*. Therefore, when describing the velocity of the material within the Two-phase 1-D Duct Zone, it is not necessary to describe either the ORIENTATION of the Zone, or the direction in which each phase flows.

The formal syntax used to describe the velocity of the material within the Two-phase 1-D Duct Zone Class is merely an adjustment of that for the Two-phase Lumped Class (4.17b to 4.17f), with declaration 4.17c removed and with declaration 4.17f modified so that the attributes DOWNWARD_FLOW and UPWARD_FLOW are replaced with BULK_FLOW and SUPERFICIAL_VELOCITY.

4.5 The Reservoir Section

In Section 3.2, a *Reservoir* was defined as a sink of effectively infinite size whose state is independent of its interactions with the system. Within the current framework, a *Reservoir* may represent a source or sink of material, temperature or pressure. If the *Reservoir* represents a material sink, the *Reservoir* is fully described by the temperature, pressure and mass fraction of each component within the *Reservoir*. Similarly if the *Reservoir* represents a temperature or pressure sink, the *Reservoir* is fully described by the temperature or pressure of the *Reservoir*.

The formal syntax for the Reservoir Object (definition 4.19) comprises two sections: a declaration of the Reservoir entity and a declaration of the Reservoir's state, achieved by assigning values for the temperature, pressure and component mass fractions. The declaration of the *Reservoir* entity begins with the keyword RESERVOIR followed by two attributes: a user supplied name for the *Reservoir* entity and the name of the phase present within the material sink. The name of the *Reservoir* must be globally unique and the phase must have previously been defined within the Phase Section. If the *Reservoir* represents either a temperature or a pressure sink, the phase which the *Reservoir* represents is of no interest and hence the name of the phase should be omitted from the *Reservoir* declaration.

The generic language declaration for the *Reservoir's* state comprises three language subsections representing the composition, temperature and pressure of the *Reservoir*. Depending on what type of sink the *Reservoir* represents, certain language subsections may be omitted. Each language subsection employs the same grammatical rules as for the Zone Object Physicochemical

Behaviour Subsections with the composition of the *Reservoir* declared according to language definition 4.13 and the pressure and temperature of the *Reservoir* declared according to language definition 4.12.

Language Definition 4.19: The Reservoir Object Declaration

RESERVOIR : <reservoir_name> : <*phase_name*> ;
DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
Reservoir_Composition Subsection
Reservoir_Temperature Subsection
Reservoir_Pressure Subsection
END

An example of the declaration language for the Reservoir Object is given in Example 4.2. In this example a Reservoir is used to represent the feed for a vinyl acetate reactor. The feed consists of three components: ethylene (0.4 wt/wt), acetic acid (0.4 wt/wt) and oxygen (0.2 wt/wt). The temperature and pressure of the feed are 500K and 5×10^5 Pa respectively.

```
RESERVOIR : R101 : feed_phase ;
        DEFINED_PHYSICOCHEMICAL_BEHAVIOUR
                RESERVOIR_COMPOSITION ;
                        ethylene = 0.4 ;
                        acetic_acid = 0.4 ;
                        oxygen = 0.2 ;
                RESERVOIR_TEMPERATURE = 500 ;
                RESERVOIR_PRESSURE = 5E5 ;
END
```

Example 4.2: A Reservoir Declaration

4.6 The Connection Section

In Chapter 3, the *Connection* entity was developed to represent the transfer of mass and energy between two *Zones* or between a *Zone* and a *Reservoir*. The *Connection* entity was also used as a means of maintaining continuity in the pressure field throughout the process. In Section 3.2, five different Connections were established. The five different connections and their purposes are tabulated in Table 4.6.

Connection	Purpose
Open Material Connection	To represent the bulk (convective) transfer of mass and energy resulting from the flow of material. The *Connection* does not influence the rate at which material flows through it.
Closed Material Connection	To represent the diffusive transfer of mass between two *Zones* or between a *Zone* and a *Reservoir* and energy. The rate of mass and energy transfer through the *Connection* is dependent on the properties of the *Connection*.
Open Energy Connection	To represent the diffusive transfer of energy between two *Zones* or between a *Zone* and a *Reservoir* where the rate at which energy is transferred is independent of the transfer mechanism.
Closed Energy Connection	To represent the diffusive transfer of energy between two *Zones* or between a *Zone* and a *Reservoir* where the rate in which energy is transferred is dependent of the transfer mechanism.
Pressure Connection	To equate the pressure either side of an interface between two *Zones* or between a *Zone* and a *Reservoir*.

Table 4.6: The Different *Connections*

The formal syntax for each Connection Object is given in definition 4.20, with the square brackets used to represent that possibility of omitting all the keywords within the brackets. The syntax begins with the keyword CONNECTION and the *Connection* name. The next two attributes are used to identify the type of *Connection* that the user wishes to use. For the Pressure Connection, only one attribute should be specified.

If the user has selected a Material Connection, specified through the keyword MATERIAL, the phase that is transferred by the *Connection* should be declared. For the Closed Material Connection, the diffusing species should be defined as a separate phase within the Phase Section and this phase should be referred to within the *Connection* declaration. If the pressure of the two Zones / Reservoirs are to be equated by the *Connection* the language definition should be terminated with the keyword EQUATE_PRESSURE.

Many physical situations will involve the diffusion of energy across a moving interface that lies between two phases. This is commonly the case when the change of phase also occurs. Examples include the conduction of heat through a reacting hydride bed, where the hydride region shrinks and the metal region expands, and the process of drying, where the saturated liquid region shrinks within the porous solid (falling rate drying). For such cases the flow of energy out of the *Connection* does not equal the flow of energy into the *Connection* as thermal energy is either consumed or generated by the change of phase at the interface. This energy corresponds to the heat of reaction for the hydride bed example and the latent heat for the drying example. Whilst it may not be possible to equate the flow of energy across the interface, it is possible to declare the

temperature of the interface. To allow this type of process to be represented within *COMPASS*, the syntax allows the temperature of the *Connection* to be specified for a *Closed Energy Connection*.

Language Definition 4.20: Connection Object Declaration

$$\text{CONNECTION : <name> : } \left\{ \begin{array}{c} \underline{\text{OPEN}} \\ \underline{\text{CLOSED}} \\ \text{PRESSURE_CONNECTION} \end{array} \right\} : \left\{ \begin{array}{c} \underline{\text{MATERIAL}} \\ \underline{\text{ENERGY}} \end{array} \right\} :$$

$$\left\{ \begin{array}{c} \underline{< \text{phase_name} >} \\ < \text{number} > \end{array} \right\} : [\text{EQUATE_PRESSURE}];$$

4.7 The Flowsheet Section

With each Zone, Vessel, Connection and Reservoir object fully described, it is possible to build the internal representation of the process by interlinking the Zone, Connection and Reservoir framework objects. The internal representation is constructed within the Flowsheet Section by linking each *Connection* to either two *Ports* or to a *Port* and a *Reservoir*. For all *Open Connections* it is also necessary to specify the direction in which material or energy flows.

Within the Flowsheet Section, a Flowsheet Object exists for each *Connection*, specifying the upstream and downstream *Port* or *Reservoir* and where necessary the direction of flow. The formal syntax for each Flowsheet Object commences with the keyword CONNECTION followed by the name of the *Connection* that is being linked, the upstream *Port* name, an arrow $(->)$ representing the direction of flow and the downstream *Port* name. All *Port* names must be globally unique. This is achieved by preceding the *Port* name with the name of the *Zone* to which the *Port* belongs (definition 4.21).

Language Definition 4.21: Global Port Name

Global_Port_Name = Zone_Name . Local_Port_Name

For all *Open Connections* a single arrow $(->)$ is used to represent the direction in which material or energy flows. The upstream Port / Reservoir name precedes the arrow and the downstream Port / Reservoir succeeds the arrow. For all Closed Connections, the direction of flow is arbitrary and a two-way arrow $(<->)$ is used. The formal syntax for the Flowsheet Object is given in definition 4.22.

Language Definition 4.22: Flowsheet Object Declaration

$$\text{CONNECTION : < name> : } \left\{ \begin{array}{c} \text{< Port_name >} \\ \text{< Reservoir_name >} \end{array} \right\} \left\{ \begin{array}{c} \text{->} \\ \text{< ->} \end{array} \right\} \left\{ \begin{array}{c} \text{< Port_name >} \\ \text{< Reservoir_name >} \end{array} \right\};$$

4.8 An Example of the Process Modelling Language

An example of the process modelling language for the description of a settling tank is given in Example 4.3. The settling tank separates an impurity of water from a water / butanol stream. As the density of water is greater than that of butanol, the water droplets settle and coalesce, thereby leaving a clear butanol layer at the top of the tank and a clear water layer at the bottom. In between these two pure layers there is a two-phase region in which the water droplets are dispersed within the butanol phase. It is assumed that the volume fraction of the dispersed phase is uniform throughout the dispersion layer. Further, it is assumed that the water droplets all have a specified radius and their motion can be described by a single kinematic equation. The tank is isolated and consequently there are no input or output streams.

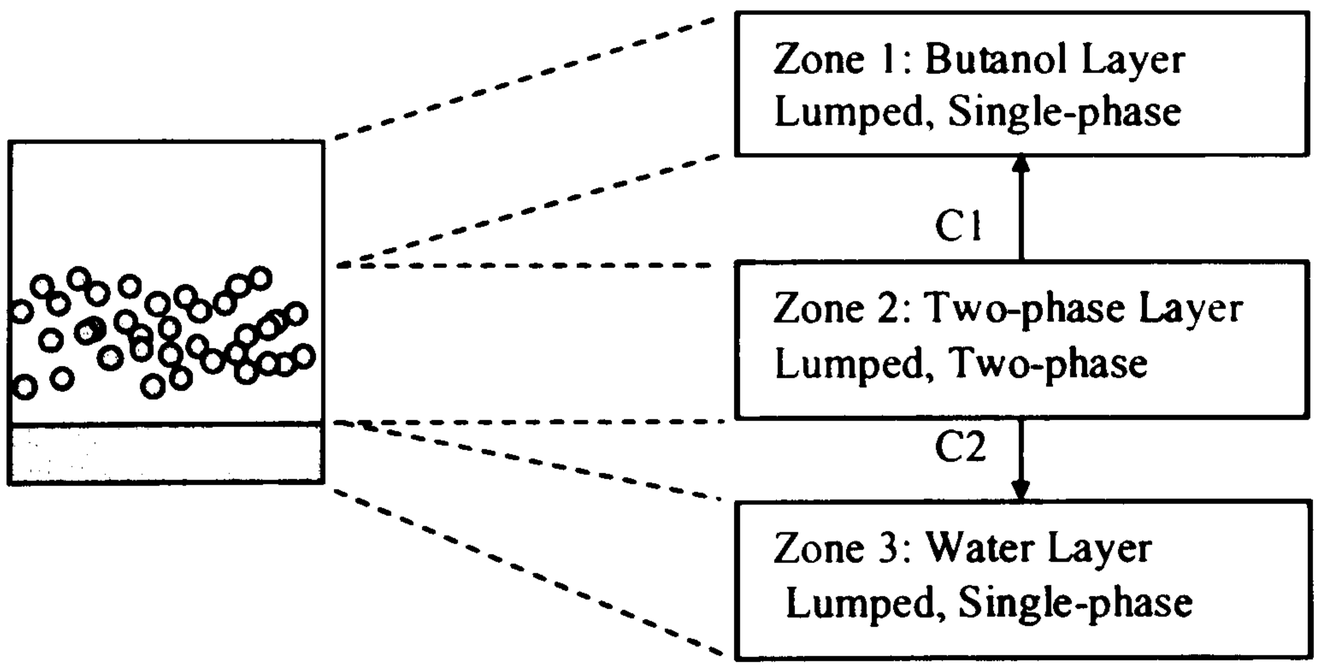


Figure 4.1: A Diagram of the Settling Tank and the Corresponding Structural Representation

4.9 Summary

In this chapter a formal language has been presented in which a given process can be described in purely physical terms. Structured closely around the framework discussed in Chapter 2, the language provides the user with a means to break a given process into a network of framework objects and to describe the behaviour of each object using a set of keywords. The scope of the language allows a wide variety of processes to be described in physical terms that are easily understood. In the forthcoming chapters, a methodology for generating the mathematical model will be developed. The methodology will be based on the physical description described in this chapter.

Example 4.3: Complete Language Example: The Language Description for a Settling Tank

```
PROCESS : Settling_system ;

DEFINE_PHASES
        PHASE : Butanol : LIQUID : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Butanol : a ;
                END #component_section

        PHASE : Water : LIQUID : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Water : a ;
                END #component_section
END #phase section

DEFINE_VESSELS

        VESSEL : Settling_Tank : S101 : OPEN ;

        DEFINE_ZONES
                ZONE : Butanol_layer ;
                        GEOMETRY : LUMPED ;
                        PHASE : Butanol ;
                        DEFINE_PORTS
                                PORT : P1 ;
                        END #Port Section
                        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                                ZONE_DYNAMICS : DYNAMIC ;
                        ZONE_EXTENT : VARIABLE ;
                        ZONE_COMPOSITION : FIXED ;
                                Butanol = 1 ;
                        ZONE_TEMPERATURE = 293 ;
                        ZONE_PRESSURE = 1.013E5 ;
                END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION

ZONE : Two_phase_layer ;
        GEOMETRY : LUMPED ;
        PHASE : Butanol, Water ;
        DEFINE_PORTS
        PORT : P1: PORT_MASS_FLOWRATE : VOLUME_GAIN : Water_layer ;
        PORT : P2: PORT_MASS_FLOWRATE : PHASE_DEPLETION ;
        END #Port Section
        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                ZONE_DYNAMICS : DYNAMIC ;
                ZONE_EXTENT : VARIABLE ;
                ZONE_COMPOSITION ;
          PHASE_COMPOSITION : Water : FIXED :
                Water = 1 ;
          PHASE_COMPOSITION : Butanol : FIXED ;
                Butanol = 1 ;
ZONE_TEMPERATURE = 293 ;
ZONE_PRESSURE = 1.013E5 ;
PHASE_VOLUME_FRACTION : VARIABLE :
ZONE_VELOCITY ;
MIXTURE_BULK_FLOW : NO_BULK_FLOW ;
```

```
                INTERNAL_RELATIVE_MOTION : RELATIVE_PHASE_MOTION
                FLOW_REGIME : DISPERSED ;
                CONTINUOUS_PHASE : Butanol : UPWARD_FLOW  ;
                DISPERSED_PHASE : Water : DOWNWARD_FLOW ;
        END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION

ZONE : Water_layer ;
        GEOMETRY : LUMPED ;
        PHASE : Water ;
        DEFINE_PORTS
                PORT : P1 ;
                END #Port Section
                DEFINE_PHYSIOCHEMICAL_BEHAVIOUR
                        ZONE_DYNAMICS : DYNAMIC ;
                        ZONE_EXTENT : VARIABLE ;
                        ZONE_COMPOSITION : FIXED ;
                                Water = 1 ;
                        ZONE_TEMPERATURE = 293 ;
                        ZONE_PRESSURE = 1.013E5 ;
                END# PHYSIOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section
END #vessel section

DEFINE_CONNECTIONS
        CONNECTION : C1 : OPEN : MATERIAL : Butanol ;
        CONNECTION : C2 : OPEN : MATERIAL : Water ;
END #connection section

DEFINE_FLOWSHEET
        CONNECTION : C1 : Two_phase_layer.P2 -> Butanol_layer.P1  ;
        CONNECTION : C2 : Two_phase_layer.P1 -> Water_layer.P1 ;
END #flow sheet section
```

Chapter 5

The Model Building Procedure

In the previous chapter a formal language syntax and a vocabulary of key words were presented that allow a given process to be described in purely physical terms. In this chapter, the model building procedure for the construction of single- and two- phase models will be discussed. The chapter begins with a discussion of the overall model building strategy and then continues with a discussion of the model building logic that will be used to construct a suitable mathematical model for each framework object (Vessel, Zone, Reservoir or Connection).

5.1 Model Building Strategy

The object-oriented nature of the framework supports the development of a model building strategy that is based upon the systematic generation of a model for each element within the framework. For each framework object, an appropriate mathematical model is formulated from the physical representation, which is declared in terms of the process modelling language. The modelling strategy begins with each *Vessel* entity and formulates the appropriate mathematical model for the *Vessel* in addition to the model for each *Zone* that is contained within the *Vessel*. In the second stage of the model building strategy, the appropriate model for each *Reservoir* is constructed by assigning a user-supplied value to the appropriate state variables. The final stage of the model building strategy links the *Reservoirs* and *Zones* together, by formulating the appropriate model for each *Connection* entity.

The mathematical model for each *Vessel*, *Zone*, *Reservoir* or *Connection* entity is constructed by using a number of model building algorithms that analyse the physical description and select the appropriate equations that constitute the local model. These algorithms are collectively referred to as the "the model building logic". The model building logic is structured into a number of *model classes*, with each *model class* assigned to a particular framework entity. For all *zone model classes*, the model building logic is further decomposed into a number of *model subclasses* with each *model subclass* assigned to a particular key property (composition, pressure, temperature, extent, velocity etc.).

Each equation is formulated by instantiating the appropriate equation class. An equation class exists for each type of equation (lumped component mass balance, distributed 1-D energy balance, etc.) and contains both the equation terms and the model building logic required to construct the appropriate form of the equation. The model building logic interprets the physical description provided by the user and determines the equation terms and the order in which they are to be included into the equation. Due to the large number of equation classes, it is not appropriate to discuss the model building logic for each equation. Instead the generic form of the equation is presented and it is expected that the reader will obtain a sufficient understanding of the equation building logic from the terms within each generic equation. For example, time derivatives are omitted for all steady state models, the convective flux is omitted if the material within the Zone is stationary and the diffusive flux is omitted if there is no diffusion. In the final stage of the model building strategy, each equation term is constructed from a specific symbol, the name of the framework object and, for the case of a two-phase *Zone*, the name of the phase. A list of symbols used is given in Appendix A. An illustration of the model building strategy is shown in figure 5.1.

5.2 The Model Building Strategy for the Vessel Entity

The *Vessel Model Class* contains the necessary model building logic to model the *Vessel* entity. An instance of the *Vessel Model Class* is made for each *Vessel* entity declared in the process description. The model-building algorithm for the *Vessel Model Class* is split into two sections. The first section generates Eq. 5.1 for instances where there are two or more *Zones* within the *Vessel* and the *Vessel* has been defined as being closed. Eq. 5.1 equates the accumulative volume of each Zone (V^Z) within the vessel to the volume of the Vessel V^{VSL}.

$$V^{VSL} = \sum_{Z=1}^{NZ} V^Z \qquad [5.1]$$

The second part of the algorithm formulates the mathematical model for each *Zone* contained within the *Vessel*.

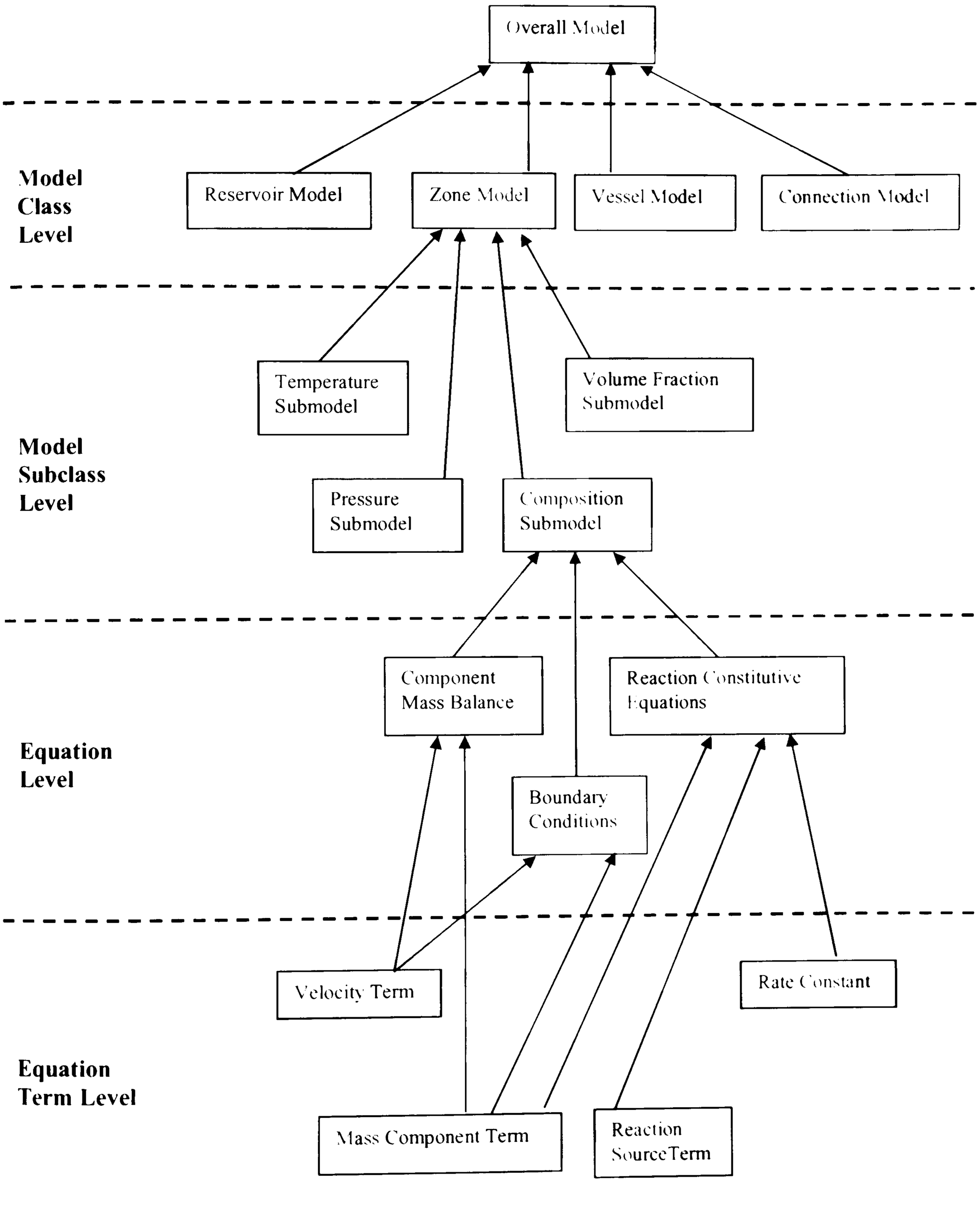


Figure 5.1: The Hierarchical Model Building Approach used within COMPASS

5.3 The Model Building Strategy for the Zone Entity

The model building logic for each *Zone* is structured around a number of *zone model classes* and *model subclasses*. A *zone model class* corresponds to each *zone class* and contains all the necessary model building logic required to construct a model for a particular class of *Zone*. The model building strategy for each *Zone* therefore commences with instantiation of the appropriate *zone model class*. Currently five *zone model classes* have been constructed. These are the *No Extent Zone Model Class*, the *Single-Phase Lumped Zone Model Class*, the *Single-Phase 1-D Duct Zone Model Class*, the *Two-Phase Lumped Zone Model Class* and the *Two-Phase 1-D Duct Cone Model Class*. The model building procedure for each of these *model classes* can be found in Appendices B, C, D, E and F respectively.

The model building logic for each *zone model class* is structured around a number of *model subclasses* (figure 5.1). Each *zone model subclass* interprets the physical description assigned to a particular key property of the *Zone* (composition, temperature, pressure, extent ect.) and, based on this information, formulates a set of mathematical equations that describe the property in question. The overall model for each *Zone* is then formed from the collection of local models created by each *zone model subclass*. A list of the *zone model classes* and *zone model subclasses* is given in Table 5.1.

By constructing the overall model from a collection of local models that have been tailored to fit a particular aspect of the *Zone's* physical description, the modelling strategy is able to benefit from the advantages of an object-oriented approach. Since different instances of each local model can be constructed, a substantial number of different local model combinations can be created. Unsurprisingly, the scope within which a model can be custom-built to fit a specific physical description is very wide. At the same time, structuring the large number of model building algorithms around a discrete number of *model subclasses* makes it easier to verify the model building logic and hence the reliability of the model building is enhanced.

Due to the number of model building algorithms that have been written for each model subclass, it is sadly not possible to describe each algorithm in detail. However, by referring to Appendices B, C, D, E and F it is possible to obtain an excellent overview of the model building logic that is embedded within each *model subclass*. To provide an illustration of the model building logic, the formulation of a local model by the algorithms embedded within the *Composition Subclass* of the

Single-Phase, One-dimensional, Duct Zone Model Class will be described in detail. The equations presented below are obtained by working through the model building logic documented in Appendix D, Section D.4, using the physical description for a plug flow reactor provided in Example 5.1. It should be noted that the *Composition Subclass* interprets the description assigned to the composition key property, which has been highlighted in bold text below.

Example 5.1: The input language for a Plug Flow Reactor

```
VESSEL : Reactor1 : R101  ;
DEFINE_ZONES
      ZONE : Z1 ;
            GEOMETRY : 1-D ;
            COORDINATES : CARTESIAN ;
            PHASE : Product_stream ;
            ORIENTATION : 0 ;
            DEFINE_BOUNDARIES
                  BOUNDARY : Inlet   : AXIAL  : 0 ;
                  BOUNDARY : Outlet  : AXIAL  : 5 ;
                  BOUNDARY : Wall    : RADIAL : 3 ;
            END #DEFINE_BOUNDARIES
            DEFINE_PORTS
                  PORT : P1 : Inlet ;
                  PORT : P2 : Outlet ;
                  PORT : P3 : Wall ;
            END #DEFINE_PORTS
            DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                  ZONE_DYNAMICS : DYNAMIC ;
                  ZONE_COMPOSITION : VARIABLE : REACTION ;
                  ZONE_TEMPERATURE : VARIABLE :  NEGLECT_KE ;
                  ZONE_PRESSURE = 1.0E5 ;
                  ZONE_VELOCITY : BULK_FLOW ;
      END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
END #zone section
```

The model building logic commences by determining whether the user has declared the composition of the phase. For example 5.1, the composition of the phase has not been declared and must therefore be determined by solving a normalised mass balance for each component. As each of the *Zone's Boundaries* has been assigned a fixed coordinate and with there being no chemical diffusion through the *Zone* or mass transfer through the 'Wall' *Boundary*, the following form of the normalised component mass balance is generated for each component within the phase 'Product_stream':

$$\frac{\partial \rho_i^z}{\partial t} + \frac{1}{(Z_2 - Z_1)} \frac{\partial}{\partial \hat{z}} v^z \rho_i^{\prime} - R_i^z = 0 \qquad \forall i = NCM \qquad \forall z \in (0,1] \qquad [5.2]$$

The user should refer to Appendix A for the definitions of the nomenclature. To determine the domain of applicability to which Eq. 5.2 will be integrated over, an algorithm has been written to assess whether there is any chemical diffusion within the *Zone* (required to determine whether one or two boundary conditions are needed) and to determines the direction of flow through the *Zone* (deduced from the appropriate *Connections*). In this case, material flows into the *Zone* through the Inlet *Boundary* and the domain of applicability is formulated to reflect this.

By working through the model building logic documented in section D.4, it can be seen that, with no diffusion, the next step of the model-building logic generates a set of equations that equate the source term for chemical reaction to the product of the stoichiometric coefficient and the rate of reaction. It should be noted that Eq. 5.3 is generated for each component within the phase.

$$R_i^z = \sum_{j=1}^{j=NR} \upsilon_{ij} r_j \qquad \forall \mathrm{i} = \mathrm{NCM} \qquad [5.3]$$

The next stage of the model-building algorithm generates a set of equations that relate the mass fraction of each component to the component density and the total density of the phase. It should be noted that the *Core Subclass* (discussed later in this Chapter) generates an equation that equates the phase density to the sums of the component densities

$$x_i^z = \frac{\rho_i^z}{\rho^z} \qquad \forall \mathrm{i} = \mathrm{NCM} \qquad [5.4]$$

Finally, with the appropriate equations formulated for the interior of the domain of applicability, the model building algorithms focus on generating the appropriate boundary conditions. By using the same algorithm that was used to determine the domain of applicability for the normalised component mass balance, it is deduced that only one set of boundary conditions are required at $\hat{z} = 0$. As there is no diffusion within the *Zone*, the diffusive terms are omitted from Eqs. D.16 and D.17 and the following set of two equations is generated for each component.

At $\hat{z} = 0$

$$F_i^P = A^Z v^Z \rho_i^Z \qquad \forall \mathrm{i} = \mathrm{NCM} \qquad [5.5]$$

$$F_i^P = F^P x_i^P \qquad \forall \mathrm{i} = \mathrm{NCM} \qquad [5.6]$$

The model building logic contained within each *zone model subclass* works in a similar manner to that described above for the *Composition Subclass*, with the logic analysing the physical description assigned to the appropriate key property and formulating an appropriate set of equations that describe the key property. As the overall model for the *Zone* is formed from the collection of local models for each *Subclass*, the model building logic for each *Zone* merely runs through the set of Subclasses in a sequential manner, activating the model building logic for each *Subclass*. The model building logic for each *Zone* terminates once the local model for the last key property has been generated.

To support the construction of an interface for each *Zone*, the *model subclass* structure (for each *model class*) is extended to include the *Port* entity, with the *Port Model Subclass* containing the appropriate model building logic to formulate equations for each *Port* framework entity within the *Zone*. The model building logic and the generic equations that are generated by each *zone model subclass* can be found in Appendices B, C, D, E and F.

For different physical descriptions, a given equation may be generated by different *subclasses*. To ensure that, for a specific physical description, the equation is generated only once, a substantial amount of logic is embedded in the program. In certain instances it is more efficient to extract the equation from each *subclass* and declare it once within the *Core Model Subclass*. Each equation within the *Core Model Subclass* is generated automatically without using any model logic to determine whether the physical description merits its generation.

It should be stressed that the coupled nature of the equations within a model means that it is frequently unwise to assign a particular purpose to an individual equation. However, it is permissible to identify a set of equations with particular functions (calculating the pressure / temperature / composition) for the purpose of accounting. The allocation of the equation set to particular purposes must be such that when all the purposes are combined, the overall model is physically correct and mathematically well-posed. To ensure that the overall model is correct, for certain instances where there is coupling, each *zone model subclass* must not only consider the physical description assigned to the specific state variable but also the physical description assigned to the other state variables.

Model Class	Model Subclasses								
	Port Model Subclass	Core Model Subclass	Flowrate Model Subclass	Extent Model Subclass	Composition Model Subclass	Temperature Model Subclass	Pressure Model Subclass	Velocity Model Subclass	V. Fraction Model Subclass
No Extent	✓	✗	✓	✗	✓	✓	✓	✗	✗
Single-Phase Lumped Zone Model Class	✓	✗	✗	✓	✓	✓	✓	✗	✗
Single-Phase 1-D Duct Zone Model Class	✓	✓	✗	✓	✓	✓	✓	✓	✗
Two-Phase Lumped Zone Model Class	✓	✓	✗	✓	✓	✓	✓	✓	✓
Two-Phase 1-D Duct Zone Model Class	✓	✓	✗	✓	✓	✓	✓	✓	✓

Table 5.1: The Model Classes and Model Subclasses

5.3.1 The Internal Connection Subclass

The *Internal Connection Subclass* exists for the *Two-Phase, Lumped Zone Model Class* and the *Two-Phase, 1-D Duct Zone Model Class*. The *Internal Connection Subclass* generates the necessary equations to determine the interfacial exchange of mass and energy between the two phases.

The Interfacial Exchange of Component Mass: The current scope of the framework permits the interfacial exchange of mass to arise either from absorption or from phase change. If there is absorption between the two phases, the interfacial exchange of mass is expressed in terms of the interfacial area and the mass flux.

$$\Gamma_{IC,i} = a j_{IC,i} \qquad [5.7]$$

In Eq. 5.7, a is the interfacial area per unit volume and j_i is the interfacial mass flux of component i. The current scope of the framework requires the user to provide a suitable equation for the interfacial mass flux. Suitable equations include the two-film theory, the penetration model or the surface renewal model. For systems that experience a change of phase within the *Zone*, the interfacial exchange of mass associated with this phase change is determined from the heat flux between the phase undergoing the phase transition and its environment. If the two phases within the *Zone* are at different temperatures, the environment in question is the second phase. The following energy jump equation can be used to relate the rate of mass transfer to the interfacial rate of heat transfer between the two phases.

$$\Gamma_i = \frac{\lambda \left| q_{if,1} + q_{if,2} \right|}{\left| h_2 - h_1 \right|} \qquad [5.8]$$

In Eq. 5.8 h_2 and h_1 are the enthalpies of the second and first phase respectively and λ is a flag that is set to +1 if phase 2 is undergoing the phase transition and -1 if phase 1 is undergoing the phase transition. The terms $q_{if,1}$ and $q_{if,2}$ represent the interfacial heat transfer for the two phases defined by:

$$q_{if,1} = h_{if,1} a (T_{PT} - T_1) \qquad [5.9]$$

$$q_{if,2} = h_{if,2} a (T_{PT} - T_2) \qquad [5.10]$$

where T_{PT} is the phase transition temperature (saturation temperature for boiling and condensation, freezing temperature for freezing and melting). Clearly, for systems involving a phase transition, one of the two phases will be at the phase transition temperature and consequently either $q_{if,1}$ or $q_{if,2}$ is eliminated from Eq. 5.8. For certain systems, it is possible that both phases are at the phase transition temperature and hence any change in the overall energy of the system must result in the interfacial exchange of mass. It can be shown that by assuming that the thermal diffusion of energy is negligible the interfacial exchange of mass is equal to:

$$\Gamma_{IC,i} = \frac{1}{|h_2 - h_1|} \left(\frac{|Q_W| P_W}{A} + \rho_m g + \frac{\partial}{\partial t} P \right) \qquad [5.11]$$

In Eq. [5.11], Q_W is the heat flux through the wall (specified through a Connection), P_W is the wetted perimeter, A is the cross sectional area, ρ_m is the density of the two-phase mixture, g is the gravitational acceleration and P is the pressure. In addition to determining the rate of component mass transfer between the two phases it is also necessary to determine the total exchange of mass between the two phases. This is achieved by summing the component interfacial mass transfer rates.

$$\Gamma_{IC} = \sum_{i=1}^{NCM} \Gamma_{IC,i} \qquad [5.12]$$

where NCM represents the number of components.

<u>The Interfacial Exchange of Heat:</u> The interfacial exchange of heat can be expressed in terms of the interfacial area per unit volume (a) and the interfacial flux of heat (q_{if}).

$$\Lambda = a q_{if} \qquad [5.13]$$

The interfacial flux of heat can be further expanded in terms of the interfacial heat transfer coefficient (h_{if}) and the temperature difference between the two phases.

$$q = h_{if}\left(T_2 - T_1\right) \qquad [5.14]$$

At present, the interfacial heat transfer coefficient must be supplied by the user. It is proposed that a library could be developed containing a set of suitable correlations for the heat transfer coefficient.

The Interfacial Area: The interfacial area is used to determine the interfacial exchanges of both mass and energy. A few correlations are embedded within *COMPASS* for illustrational purposes and are for vertical pipe flow only. The correlations for the interfacial area for vertical pipe flow are given in Table 5.2.

Flow Regime	Interfacial Area
Droplet	$a = \frac{3\varepsilon_d}{r_{sm}}$
Slug and Churn	$a = \frac{4.5}{D}\frac{\varepsilon_d - \varepsilon_{gs}}{1-\varepsilon_{gs}} + \frac{3\varepsilon_{gs}}{r_{sm}}\frac{1-\varepsilon_d}{1-\varepsilon_{gs}}$
Annular	$a = \frac{4C_{an}}{D}\sqrt{\varepsilon_d}$

Table 5.2: Correlations for the Interfacial Area in Vertical Pipe Flow (Ishii, 1984)

In Table 5.2, r_{sm} is the Sauter mean droplet radius, D is the hydraulic diameter, ε_d is the volume fraction of the dispersed phase, ε_{gs} is the average volume fraction of the gas in the liquid slug section and C_{an} is the roughness parameter due to the film waves, $C_{an} \geq 1$. A suitable constitutive equation is required for the Sauter mean radius. The average volume fraction of gas in the liquid slug can be determined from Eq 5.15 (Levy, 1999).

$$\varepsilon_{gs} = 0.3e^{-8\left(\frac{\varepsilon_d - 0.3}{0.45}\right)} \qquad [5.15]$$

5.4 The Model Building Strategy for the Reservoir Entity

The model building logic contained within the Reservoir Model Class determines which state variables (Composition, Temperature, Pressure) the user has declared a value for and generates Eqs. 5.16, 5.17 and 5.18 accordingly.

$$x_i^R = \text{user supplied value} \qquad [5.16]$$

$$T^R = \text{user supplied value} \qquad [5.17]$$

$$P^R = \text{user supplied value} \qquad [5.18]$$

where x is the mass fraction, T is the temperature and P is the pressure, superscript R refers to the *Reservoir* and subscript i refers to the component. If the *Reservoir* is linked to *a Closed Material Connection*, it is necessary to determine the density of the material within the *Reservoir*, and a generic equation of state is generated.

$$\rho^R = f\left(T^R, P^R, x_i^R\right) \qquad [5.19]$$

This generic form of the equation of state is widely used within *COMPASS*. It is envisaged that a thermodynamic library would be available to the user to provide specific PVT data.

5.5 The Model Building Strategy for the Connection Entity

The model building logic for the *Connection* entity is structured around five model classes, with each model class corresponding to a particular type of Connection. The five different model classes are, the *Open Material Connection Model Class*, the *Open Energy Connection Model Class*, the *Closed Material Connection Model Class*, the *Closed Energy Connection Model Class* and the *Pressure Connection Model Class*.

5.5.1 The Open Material Connection Model Class

The *Open Material Connection* is used to represent the transfer of material between two *Zones* that results from the flow of material. Consequently, the *Open Material Connection* is inherently associated with the exchange of both mass and energy. As the *Connection* represents bulk

movement, the thermodynamic state of the material transferred by the *Connection* is determined from the upstream *Zone*. Consequently the equations generated by the *Open Material Connection Model Class* simply equate the thermodynamic state variables and the mass flowrate of the downstream *Port* to the appropriate state variables of the upstream *Port* or *Reservoir*. The current scope of the program permits the *Open Material Connection* to transfer one phase only. The *Open Material Connection Model Class* deduces the type of the upstream and the downstream model entities and, based on this information, derives the appropriate equations from a library.

The equations generated by the *Open Material Connection* Model Class are summarised in Table 5.3. The upstream model entities are shown in the rows and the downstream model entities in the columns. For the *Open Material Connection Class*, the model entities are classified into four key groups: *Reservoirs*, no extent and lumped *Zones*, 1-D duct *Zones* with no axial diffusion and 1-D duct *Zones* with axial diffusion. For each possible interconnection, the appropriate equations can be deduced from Table 5.3. It can be seen that the equations are expressed in terms of the state variables of the appropriate *Port* rather than the *Zone* itself. With each *Port* assigned to a specific phase within the *Zone*, the same equations are generated for instances where the *Connection* is attached to a two-phase Zone as for instances where the *Connection* is linked to a single-phase *Zone*. As the *Port* entity has been specifically formulated to provide a consistent interface between the *Zone* and the *Connection*, the state variables of the two interlinked *Ports* can be easily related without fear of generating a physically incorrect or mathematically ill-posed model. The exception to this rule is the *Reservoir* entity. The state variables of each *Reservoir* can be referred to directly and hence there is no need to use the *Port* entity to create a consistent interface.

For the case where the *Connection* is linked to either a *Zone* of no-extent or a lumped *Zone*, the formulated *Connection* equations simply equate the state variables (enthalpy, mass fraction, pressure) and the mass flowrate of the appropriate *Ports* across the *Connection*. A similar approach is adopted for a *Connection* linked to a *Reservoir*, except that the equations are formulated to ensure that the constant thermodynamic state of the *Reservoir* is preserved throughout the simulation. Consequently, for the case where the downstream model entity is a *Reservoir*, only the pressure of the *Reservoir*, which has previously been assigned a fixed value by the user, is equated to the pressure of the upstream model entity. As the *Reservoir* entity requires no information concerning the flow of mass through the *Connection*, this term is not used within the *Connection* submodel.

For the case where one or more of the linked *Zones* is one-dimensional, it is necessary to consider the diffusive flow of mass and energy across the *Connection* in addition to equating the state variables either side of the *Connection*. The diffusive flow of mass and energy introduce some additional issues. In many instances there may be a discontinuity in the model across the *Connection* with diffusion of mass or energy on one side of the *Connection* but not the other. In all instances it is necessary to enforce the continuity in the mass and energy fluxes across the *Connection*. Attention must therefore be given to the formulation of the *Connection* equations so that discontinuities can be avoided and so that the formulated equations maintain the conservation of mass and energy across the *Connection*.

For instances where there is a single-phase, one-dimensional duct *Zone* without diffusion linked to the *Connection*, the base transfer properties of the *Connection* are the same as for the case where the *Connection* is linked to a no-extent or lumped *Zone*; namely the state variables (enthalpy, mass fraction, pressure) and the mass flowrate. This represents the bulk flow of material through the *Connection*. For cases where there is diffusion in one or more of the linked *Zones*, it is necessary for the *Connection* to maintain the continuity in the total heat and species fluxes. If the upstream *Zone* experiences diffusion, the mass and energy fluxes at the upstream *Port* in contact with the *Connection* are set to zero for all cases except when there is also diffusion within the downstream *Zone*. For the latter case, the total fluxes of heat and species at the two interlinked *Ports* are equated. Similarly, as the *Connection* transfers the properties from the upstream *Port* to the downstream *Port*, the diffusive flux at the downstream *Port* is equated to zero for the case where there is no diffusion within the upstream *Zone*.

In effect, this formulation treats each linked *Port* as though it lies on the discontinuity between the two *Zones* and provides the downstream *Zone* entity with information concerning the physical behaviour of the process upstream of the interface. This information is used to formulate the appropriate boundary conditions for the *Zone*. Consequently, the state variables at the *Port* may be different to those of the *Zone* at the *Zone Boundary*. The difference between the state variable at the *Port* and those at the boundary to which the *Port* is linked to is shown in figure 5.2.

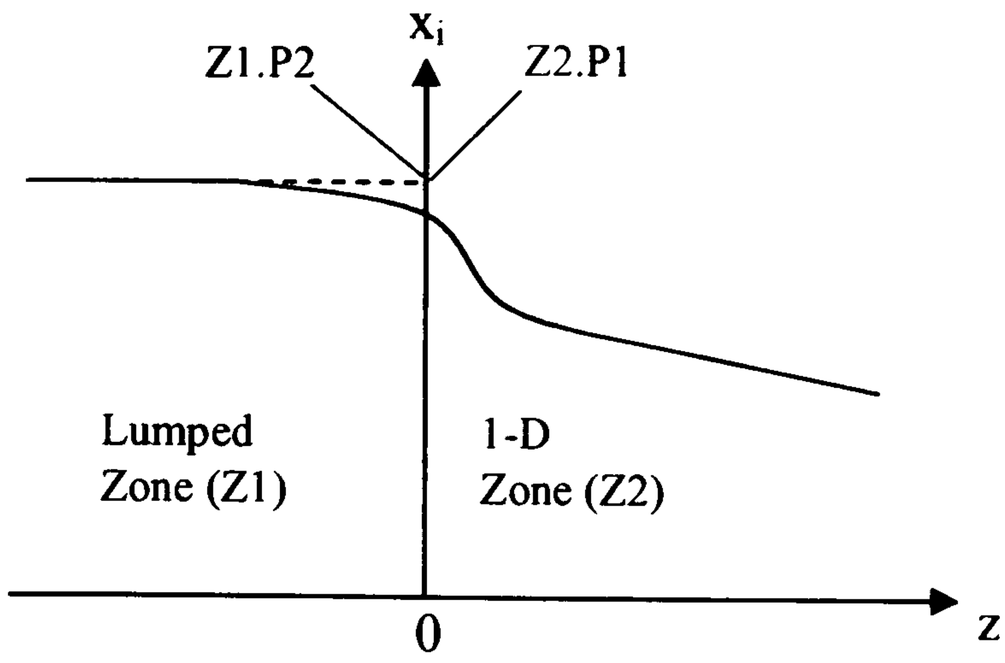


Figure 5.2. Mass Fraction Profile between a Lumped Zone and a 1-D Zone with Diffusion. Note how the mass fraction at the *Port* may be different to the mass fraction at the boundary to which the *Port* is attached to.

This logic, combined with the logic used to generate the boundary conditions within the single-phase, one-dimensional duct class, is capable of generating Danckwerts inlet and exit boundary conditions for all combinations of diffusive and non-diffusive *Zones* (Barber, 2000).

5.5.2 The Open Energy Connection Model Class

The *Open Energy Connection* is used to represent the transfer of energy between two *Zones* resulting from the temperature gradients in the two *Zones* on either side of the interface. As there must be a temperature gradient on either side of the interface, the *Open Energy Connection* can only link two *Ports* attached to two axial boundaries. For instances where there is no phase change occurring at the boundary, the *Open Energy Connection Model Class* generates the following two equations:

$$T^{P1} = T^{P2} \qquad [5.20]$$

$$Q_D^{P1} = Q_D^{P2} \qquad [5.21]$$

For cases where there is a phase change (evaporation, freezing) occurring at the interface that the *Connection* represents, the input diffusive flux is no longer equal to the outlet diffusive flux. In such cases the following equations are generated.

$$T^{P1} = T^{P2} \qquad [5.22]$$

$$T^{P1} = T^{C} \qquad [5.23]$$

where T^C is the user-specified *Connection* temperature.

5.5.3 The Closed Material Connection and Closed Energy Connection Model Classes

The *Closed Material* and *Closed Energy Connections* are used to model the diffusive flow of mass and energy between two *Zones* or between a *Zone* and a *Reservoir* for cases where there is no flow and where there is a discontinuity in the state variables across the *Connection*. As the transfer is diffusive, the rate of transfer is dependent on the state variables of both *Zones* at the appropriate *Port* position and the flow direction is arbitrary. For the case where material is transferred, it is assumed that the energy associated with the diffusive flow of material is negligible. The appropriate equations for the *Closed Material Connection and Closed Energy Connection Model Classes* are shown in Table 5.4.

5.5.4 The Pressure Connection Model Class

The *Pressure Connection* allows the user to equate the pressure of one *Port* to the pressure of another *Port* or *Reservoir* for the specific case where there is no flow of material between the two model entities.

$$P^{P1} = P^{P2} \qquad [5.24]$$

5.6 Summary

In this Chapter, a model building procedure has been presented for the development of process models for one- and two- phase systems. The model building strategy is based upon the systematic generation of a model for each Vessel, Zone, Reservoir and Connection entity within the framework.

The model building procedure for each framework entity commences with the selection of an appropriate *Model Class*. The *Model Class* contains the necessary model logic to select the appropriate set of equations for a specific type of framework entity. The selection of the equation

set that constitutes the local model for each framework entity may be done directly or, as for all of the *Zone Model Classes*, indirectly through a set number of Model Subclasses. The model building logic for each Model Class and each Model Subclass is fully documented in this chapter and in appendices B, C, D, E and F.

In the next tier of the model building strategy the appropriate form of each equation is constructed. Although the equation building logic has not been described explicitly, the user should be able to obtain a sufficient understanding of the model building logic from the generic form of each equation that has been presented. In the final tier of the model building strategy, each equation term is formulated from a specific symbol, the name of the framework object and, where appropriate, the name of the phase.

In the next chapter the implementation of the process representation and the model building procedure discussed in this chapter to form a prototype piece of software called *COMPASS* (COMPuter Aided Simulation Software) will be discussed.

	Zone 2: Downstream Zone (P2/R2)				
Zone 1: Upstream Zone (P1/R1)		Reservoir	No-Extent /Lumped	1-D Duct No-Diffusion Axial Port	1-D Duct Diffusion Axial Port
	Reservoir	-	$h^{P2} = h^{R1}$ $x_i^{P2} = x_i^{R1}$ $P^{P2} = P^{R1}$	$h^{P2} = h^{R1}$ $x_i^{P2} = x_i^{R1}$ $P^{P2} = P^{R1}$	$h^{P2} = h^{R1}$ $x_i^{P2} = x_i^{R1}$ $P^{P2} = P^{R1}$ $F_{D,i}^{P2} = 0$ $Q_D^{P2} = 0$
	No-Extent or Lumped	$P^{R2} = P^{P1}$	$h^{P2} = h^{P1}$ $x_i^{P2} = x_i^{P1}$ $P^{P2} = P^{P1}$ $F^{P2} = F^{P1}$	$h^{P2} = h^{P1}$ $x_i^{P2} = x_i^{P1}$ $P^{P2} = P^{P1}$ $F^{P2} = F^{P1}$	$h^{P2} = h^{P1}$ $x_i^{P2} = x_i^{P1}$ $P^{P2} = P^{P1}$ $F^{P2} = F^{P1}$ $F_{D,i}^{P2} = 0$ $Q_D^{P2} = 0$
	1-D, Duct No-Diffusion Axial Port	$P^{R2} = P^{P1}$	$h^{P2} = h^{P1}$ $x_i^{P2} = x_i^{P1}$ $P^{P2} = P^{P1}$ $F^{P2} = F^{P1}$	$h^{P2} = h^{P1}$ $x_i^{P2} = x_i^{P1}$ $P^{P2} = P^{P1}$ $F^{P2} = F^{P1}$	$h^{P2} = h^{P1}$ $x_i^{P2} = x_i^{P1}$ $P^{P2} = P^{P1}$ $F^{P2} = F^{P1}$ $F_{D,i}^{P2} = 0$ $Q_D^{P2} = 0$
	1-D, Duct Diffusion Axial Port	$P^{R2} = P^{P1}$	$h^{P2} = h^{P1}$ $x_i^{P2} = x_i^{P1}$ $P^{P2} = P^{P1}$ $F^{P2} = F^{P1}$ $F_{D,i}^{P1} = 0$ $Q_D^{P1} = 0$	$h^{P2} = h^{P1}$ $x_i^{P2} = x_i^{P1}$ $P^{P2} = P^{P1}$ $F^{P2} = F^{P1}$ $F_{D,i}^{P1} = 0$ $Q_D^{P1} = 0$	$h^{P2} = h^{P1}$ $x_i^{P2} = x_i^{P1}$ $P^{P2} = P^{P1}$ $F^{P2} = F^{P1}$ $F_{D,i}^{P2} = F_{D,i}^{P1}$ $Q_D^{P2} = Q_D^{P1}$

Table 5.3: The Equations for the Open Material Connection Model Class

It should be noted that the pressure is only equated across the *Connection* when specified by the user through the command EQUATE_PRESSURE. The equations given in Table 5.3 are those for an Open Material Connections linked to a single-phase Zone. For instances where the *Connection* is linked to a two-phase *Zone*, the equations reference the state variables of the appropriate phase.

	Zone / Reservoir 2 (Z2, P2, R2)				
Zone / Reservoir 1 (Z1, P1, R1)	-	Reservoir	No-Extent / Lumped	1-D, Duct Axial Port	1-D, Duct Radial Port
	Reservoir	-	$F_i^{P2} = kA\left(\rho^{R1}x_i^{R1} - \rho_i^{P2}\right)$ $Q^{P2} = \alpha A\left(T^{R1} - T^{P2}\right)$	$F_{D,i}^{P2} = kA\left(\rho^{R1}x_i^{R1} - \rho_i^{P2}\right)$ $Q_D^{P2} = \alpha A\left(T^{R1} - T^{P2}\right)$	$\Gamma_i^{P2}(z) = k\left(\rho^{R1}x_i^{R1} - \rho_i^{P2}(z)\right)$ $q^{P2}(z) = \alpha\left(T^{R1} - T^{P2}(z)\right)$
	No-Extent/ Lumped	$F_i^{P1} = -kA\left(\rho_i^{P1} - \rho^{R2}x_i^{R2}\right)$ $Q^{P1} = -\alpha A\left(T^{P1} - T^{R2}\right)$	$F_i^{P2} = kA\left(\rho_i^{P1} - \rho_i^{P2}\right)$ $Q^{P2} = \alpha A\left(T^{P1} - T^{P2}\right)$ $-F_i^{P2} = F_i^{P1}$ $-Q^{P2} = Q^{P1}$	$F_{D,i}^{P2} = kA\left(\rho_i^{P1} - \rho_i^{P2}\right)$ $Q_D{}^{P2} = \alpha A\left(T^{P1} - T^{P2}\right)$ $-F_{D,i}^{P2} = F_i^{P1}$ $-Q^{P2} = Q^{P1}$	$\Gamma_i^{P2}(z) = k\left(\rho_i^{P1} - \rho_i^{P2}(z)\right)$ $q^{P2}(z) = \alpha\left(T^{P1} - T^{P2}(z)\right)$ $F_i^{P1} = -\int_L P_W \Gamma^{P2}(z)dz$ $Q^{P1} = -\int_L P_W q^{P2}(z)dz$
	1-D, Duct Axial Port	$F_{D,i}^{P1} = -kA\left(\rho_i^{P1} - \rho^{R2}x_i^{R2}\right)$ $Q_D^{P1} = -\alpha A\left(T^{P1} - T^{R2}\right)$	$F_i^{P2} = kA\left(\rho_i^{P1} - \rho_i^{P2}\right)$ $Q^{P2} = \alpha A\left(T^{P1} - T^{P2}\right)$ $-F_i^{P2} = F_{D,i}^{P1}$ $-Q^{P2} = Q_D^{P1}$	$F_{D,i}^{P2} = kA\left(\rho_i^{P1} - \rho_i^{P2}\right)$ $Q_D^{P2} = \alpha A\left(T^{P1} - T^{P2}\right)$ $-F_{D,i}^{P2} = F_{D,i}^{P1}$ $-Q_D{}^{P2} = Q_D{}^{P1}$	-
	1-D, Duct Radial Port	$\Gamma_i^{P1}(z) = -k\left(\rho_i^{P1}(z) - \rho^{R2}x_i^{R2}\right)$ $q^{P1}(z) = -\alpha\left(T^{P1}(z) - T^{R2}\right)$	$\Gamma_i^{P1}(z) = -k\left(\rho_i^{P1}(z) - \rho_i^{P2}\right)$ $q^{P1}(z) = -\alpha\left(T^{P1}(z) - T^{P2}\right)$ $F_i^{P2} = -\int_L P_W \Gamma^{P1}(z)dz$ $Q^{P2} = -\int_L P_W q^{P1}(z)dz$	-	$\Gamma_i^{P2}(z) = k\left(\rho_i^{P1}(z) - \rho_i^{P2}(z)\right)$ $q^{P2}(z) = \alpha\left(T^{P1}(z) - T^{P2}(z)\right)$ $-\Gamma_i^{P2}(z) = \Gamma_i^{P1}(z)$ $-q^{P2}(z) = q^{P1}(z)$

Table 5.4: The Equations for the Closed Material Connection and Closed Energy Connection Model Classes

Chapter 6

Implementation of the Computer Aided Model Generator

In this Chapter, the implementation of the framework, language and the model generating strategy described in Chapters 3, 4 and 5 will be discussed. The concepts introduced in the earlier chapters have been implemented in a prototype piece of prototype software called *COMPASS* (COMPuter Aided Simulation Software). The logic and context of the framework have been written in C++. The current prototype software requires the user to provide the physical description as a text input and generates the appropriate mathematical model in both a text format and in a form that is compatible with *gPROMS*. The software consists of two key components, the translator and the model-formulating engine.

6.1 The Translator

The purpose of the translator is threefold. Firstly an input file, expressed in terms of a predefined modelling language, is processed to identify the language terms. Secondly the order in which the language terms are identified is analysed to evaluate whether any pre-declared grammatical structures occur within the file (Levine et al., 1990). Finally, on discovery of a predefined grammatical structure, an internal representation of the physical process, expressed in terms of the conceptual elements of the framework, is generated from a set of functions within the model-formulating engine. If the translator is unable to identify a specific grammatical structure, an error message is generated, thereby enabling the translator to validate the syntactical correctness of the input file. The translator consists of two components, the lexical analyser and the parser.

6.1.1 The Lexical Analyser

The lexical analyser searches the input file, attempting to identify certain predefined textual tokens or units. The predefined textual tokens include character strings, numbers, punctuation symbols, predefined symbols, and specific text symbols such as the end of line symbol. The

output from the lexical analyser is a stream of lexical tokens that is fed to the parser to be processed further.

As in Barber (2000), FLEX (Fast Lexical Analyser Generator) has been used to construct the scanner. Distributed by the GNU Project[8], (*G*nu's *N*ot *U*nix), FLEX is freely available and there are no restrictions on its redistribution. The C code generated by FLEX converts the tokens into a form that the lexical analyser can use to scan the input text file extremely quickly and at a rate that is independent of the number of expressions that it is able to match. The FLEX specification consists of three major sections, the definition section, the rules section, and the user subroutine section. The definition section allows initial C++ program code to be copied verbatim into the lexical analyser. Data definitions and function protocols, used within the rules section, are declared so that the generated code can reference them before their use. The rules section is used firstly to declare the text patterns that are used to identify the tokens and secondly to specify the resulting action that is executed when a particular token is matched (sending the token to the parser). Within the third section, the user subroutine section is used to specify any C++ code that will be copied into the lexical analyser. Typically this section defines additional subroutines to enable errors to be reported and handled. The FLEX specification file is given in Appendix H.

6.1.2 The Parser

The parser receives a set of tokens from the lexical analyser and identifies the structure and relationships amongst the tokens in the set. To enable the parser to identify the token structure, a list of rules is written to define the relationships between the tokens. This is frequently referred to as the grammar and serves to define the syntax and the structure of the modelling language. The parser automatically detects whenever a set of input tokens matches a specific rule in the grammar and, when such a rule is matched, the parser will execute a function that generates an internal representation of the physical description. If no rule is identified from a set of input tokens, the parser prints an error to the screen alerting and directing the user to the syntax error.

Again, as in Barber (2000), GNU project's Bison has been used to implement the parser. The parser is fully integrated with the scanner, calling the scanner whenever it requires a token from the input. It should be noted that tokens that are of no interest to the parser, such as comments,

---

[8] The GNU project is sponsored by the Free Software Foundation. More information can be obtained from www.gnu.org.

white space and end of line symbols, are discarded by the scanner, allowing it to continue to the next token without needing to address the parser. Just as the lexical scanner, the parser consists of three sections, the definition section, the rule section and the user subroutine section. The definition section is used to define the entire set of tokens that are to be received from the lexical scanner. It is this set of tokens that constitutes the modelling language. The actual grammar is defined within the rules section as a set of production rules. Each production rule consists of a unique identity symbol, a structure of tokens or identity symbols that the production rule attempts to identify (in the appropriate order), and a function that is executed once the order of the tokens has been matched. Each production rule can recognise an identity symbol used in a previous production rule, so it is possible to build a complex grammar from a large number of production rules. The third and final section within the parser is the user subroutine section, which enables extra subroutines to be written that allow the parser to perform additional tasks. No extra subroutines have been used in *COMPASS*. A copy of the parser input file is given in Appendix I.

6.2 The Model Building Engine

The model-building engine encapsulates the full physical description in an internal representation, checks the validity of this representation and finally generates the appropriate mathematical model. The model-building engine incorporates object-oriented programming techniques to enable an internal computational representation of the process to be constructed from a collection of discrete, distinguishable entities called *objects* that incorporate both data structure and behaviour (executed through a number of functions). Each *object* has its own inherent identity and represents a particular aspect of the framework. Characterized by a set number of *attributes* and *operations*, the *object* is able to model the particular aspect of the framework that it represents, independently of all other aspects of the framework.

The physical characteristics of each aspect of the framework are fully encapsulated by the *object's attributes*, a set of data values held by the *object*. In addition to the *object's attributes* each *object* also possesses a fixed number of *operations*, implemented through a number of functions or transformations, which describe the behaviour of the *object*. It is these functions that are used to generate the mathematical model for each *object*. It is the dual functionality of each *object* to store both data and a fixed number of *operations* that enables a mathematical model to be generated in one stage rather than the several stages that have been used by previous authors (Barber, 2000; Cho, 1997).

6.2.1 Building the Internal Representation

The external aspects of each *object*, which are accessible to other *objects*, can be separated from the internal implementation details of the *object*, which are hidden from other *objects*. This encapsulation enables a network of *objects* to be constructed through the relationships that are established as a result of the interdependency between two or more *objects*. Each *object* incorporates a data structure, so it is possible to obtain an internal representation of the process by first linking the *objects* in an appropriate manner and then storing the appropriate data within the attributes of each *object*. This approach is similar to the physical symbol table used by Barber (2000) and Cho (1997).

Each *object* is an instance of its *class,* coexistent with the other instantiations[9] of the class through the common *attribute* names and *operations*; yet independent of the other instantiations through its own values for each *attribute*. The concept of the *class* is well suited to the modelling framework, since the model entities, the building blocks of the structural representation, evolve from instances of a few base classes. The internal representation can therefore be created with relative ease by instantiating objects from a specific number of *classes* (*Vessel, Reservoir and Zone)* and interlinking these instances.

Each *class* represents a collection of *objects* with identical operations and attribute names, so *classes* are not isolated entities but instead are interrelated in a similar manner to the *objects* that they group. The relationship between two *classes* is represented by an *association*, which fully describes the link between the two *classes*. Each *association* links each *object* in one *class* to the *objects* in the other, and is exemplified by a verb in the physical description (models, transfers, links, etc.). Consequently each *association* is inherently bi-directional. By establishing the *association* between *classes* rather than *objects*, freedom is given to the manner in which *objects* within each *class* can be interrelated. Consequently, each *association* can be singular, whereby an *object* in one *class* is related to just one *object* in the linked *class*, multiple, whereby an *object* in one *class* is related to more than one *object* in the linked *class,* and optional, whereby an *object* in one *class* may or may not be related to one or more *objects* that it is linked to.

[9] instantiation: producing a particular object from its class template. This involves the allocation of a structure with the type specified by the template, and the initialisation of instance variables with default values or values provided by the constructor function of the class. For further definitions please refer to: http://wombat.doc.ic.ac.uk/foldoc/

In addition to one *class* being associated with another, it is possible to have two other forms of inter-class relationship: *aggregation* and *inheritance*. An *aggregation* relationship is defined as "a part of" relationship in which one *class*, the "assembly" *class*, can be seen as being comprised of a number of *objects* formed from instances of the second linked *class*. An example of an *aggregation* within the model-generating framework is the assembly of *Vessels* that make up a *Process*. *Inheritance* is used to describe the relationship whereby one *class* (the *child class*) is a more refined version of the interlinked *class* (the *parent class*). Through *inheritance*, the *child class* inherits the *attributes* and *operations* of its *parent class;* additional *attributes* and *operations* are added to distinguish it from its *parent class*. Further, the *child class* is also able to modify the *operations* inherited from the *parent class* (polymorphism or overloading). An excellent discussion of object-oriented modelling is presented by Rumbaugh (1991).

The internal representation is generated through a set of functions that have been written within the parser. Each function firstly generates an *object* by instantiating the appropriate *class* and secondly interconnects the generated *object* according to the relationships illustrated in figure 6.1. In figure 6.1, the nodes on the Unified Modelling Language diagram represent *classes,* whilst the arcs represent the *associations* between the different classes. On certain arcs, a filled circle represents a *multiple association* and an open circle represents an *optional association*. *Inheritance* is illustrated through the triangle symbol and *aggregation* is illustrated by the diamond symbol.

The computational representation is centred on the *Process* object and is initially constructed from a set of *Connections*, *Model Entities* and *Phase* objects. Each *Connection* object inherits a set of common attributes and functions from the *Connection* base or *parent class,* enabling common behaviour to be abstracted from the specific *Connection classes* and declared within the *parent class*. This inheritance permits all *Connection* objects to be treated collectively when required (such as creating a link with a *Zone*) whilst maintaining the unique behaviour of the different *Connection* types. The *Model Entity* class provides an opportunity to group the objects formed from the *Vessel* and *Reservoir* classes together, facilitating the storage of the internal representation and the model generation. The *Reservoir class* contains information concerning the *Connection* to which it is linked, whilst information concerning the geometry of the *Vessel* is stored within the *Vessel class*. The final core component of the *Process* is a set of *Phase objects* that represent the *Phases* that occur within the *Process*. In addition to storing

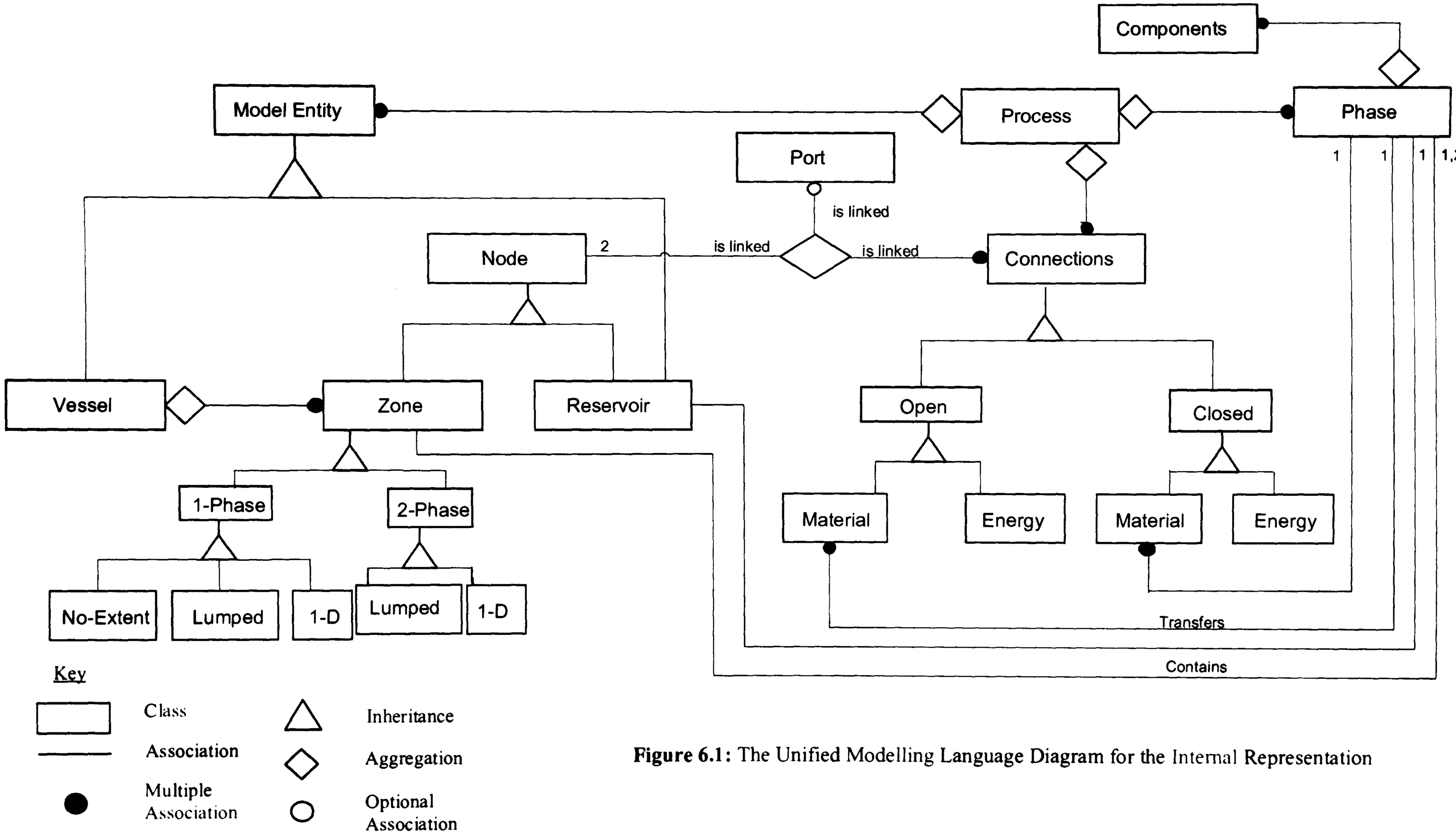


Figure 6.1: The Unified Modelling Language Diagram for the Internal Representation

information concerning the phase type and the compressibility of the *Phase*, each *Phase* object contains a set of *Component objects*, instantiated from the *Component class*. With the *Process* established in terms of the *Connections, Reservoirs, Vessels* and *Phases*, the physical behaviour of the *Process* can be fully described by instantiating a number of *Zone objects* and assigning the appropriate physical information that has been provided by the user. As for the *Connection classes*, each *Zone* type is modelled by a *class* that possesses a specific set of *attributes* and functions required to fully encapsulate the specific physical behaviour of the *Zone* type which it represents. The different *Zone classes* inherit the same common *attributes* and *operations* from the *Zone base class* enabling them to be treated collectively when required. Common *Zone attributes* stored within the *Zone* base class include the *Zone* name, a set of reference pointers to the *Ports* and *Connections* that the *Zone* is linked to and a reference pointer to the *Phase(s)* that the *Zone* incorporates. The *inheritance* of the *Zone* classes enables the functions required to establish the internal physical description to be overloaded (polymorphism). Through overloading, the functions required to establish the physical description for the *Zone* entity can be specifically tailored for each *Zone* type. This ensures that each *Zone* is assigned the appropriate physical description.

The final part of the internal representation is the network of links between the *Zones* and *Connections*. A *Connection* may either be linked to a *Reservoir* or a *Zone*, so the internal representation treats the *Zones* and *Reservoirs* as a common object, the *Node*, and the interlinked network is generated as a network of *Nodes* and *Connections*. With both the *Reservoir* and the *Zone* objects inheriting the *attributes* and *functions* of the *Node* class, each *Zone* or Re*servoir* inherits the information about the *Connections* to which the object is linked. The interlinked network of *Zones*, *Reservoirs*, *Connections* and *Ports* is represented through a tertiary association (indicated by the lozenge in Fig. 6.1). The tertiary association firstly permits each *Zone* to be linked to both a set of *Ports* and a set of *Connections* and secondly permits each *Connection* to be linked to two *Zones* as well as two *Ports*. Finally each *Port* is linked to one *Zone* and one *Connection*. This versatile formulation enables the appropriate mathematical model for each *Zone* object to be expressed in terms of either its *Ports* or its *Connections* and the mathematical model for each *Connection* to be expressed in terms of either the *Zones* or *Ports* to which it has been linked. As a *Reservoir* does not possess any *Ports*, the association linking the *Port* to both the *Reservoir* (*Node*) and the *Connection* is optional.

6.2.2 Determining the Validity of the Internal Representation

An important aspect of constructing the internal representation of the *Process* is to evaluate the consistency of the physical description that has been provided by the user. In addition to checking the syntactical structure of the input file, the software also examines the validity of the physical description stored with the internal representation. By alerting the user to an erroneous physical description, the program is able firstly to guide the user towards an unambiguous and complete physical description and secondly to avert any unsuccessful attempt to process a non-feasible physical description. Whilst it is impossible to ascertain whether the description fully represents the process that the user is trying to model, it is possible to examine the internal representation to ensure that, within the modelling framework, the description is meaningful, unambiguous and contains all the necessary information required to generate a mathematical model.

For each class, a number of C++ functions have been assigned to determine the validity of the data assigned to the attributes of the class. Each function ensures that the physical description is valid for the class and may be executed either when the internal representation is being constructed, or when the process model is being generated. The functions used to report the inconsistencies in the internal representation can be broadly categorised into three groups, corresponding to structural errors in the representation, flowsheeting errors and inconsistencies with the physical description assigned to each *Zone*. Structural inconsistencies include the inappropriate assignment of one model object (*Vessel*, *Connection*, *Zone*, *Reservoir*, *Phase*) to another (e.g. assigning three *Phases* to a *Zone*) and the referencing of a model entity that has not been previously defined. Flowsheeting errors result from an erroneous declaration of the interlinked network of *Ports*, *Connections*, *Zones* and *Reservoirs* and may include a redundant or over-specified *Zone*, *Port* or *Connection* or an inconsistency between the *Phase* of a *Connection* and the *Phase* of an interlinked *Zone*. Inconsistencies in establishing the physical description for each *Zone* and each *Connection* include assigning the wrong description to a particular *Zone* type and either overstating or understating the necessary physical description. It should be noted that by overloading the functions used to store the physical description for each *Zone*, the number of functions required to determine the validity of the physical description of each *Zone* is minimized.

6.2.3 Generating the Process Model

Once the internal representation of the *Process* has been validated, it is possible to use this representation to build the mathematical model. Embedded within each *Zone*, *Connection*, *Vessel* or *Reservoir* object is a function that generates the appropriate mathematical model for the *object*. To generate the appropriate model, the function firstly instantiates the appropriate *model class* and secondly calls the appropriate operations within the new *model class object* to generate and write the model to the output file. The operations within each *model class* are based on the physical information stored within the *Zone*, *Connection*, *Vessel* or *Reservoir*, accessed by a link that is created when the *object* is instantiated. The UML diagram for the model generating aspect of the code is given in Figure 6.2.

The software structure used to generate and write the mathematical model has been developed according to the framework discussed in Chapter 5. Each *model class* consists of a parameter list, a variable list and either a set of *Equation objects* (if the model consists of one equation) or a set of *model subclasses*. An *Equation object* has been written for each equation that occurs within the framework and encapsulates the data and behaviour required to generate that equation. To ensure that the appropriate form of the generic equation is generated, each *Equation object* accesses the appropriate *Zone object* for the required information needed to formulate the equation. Based on this information, the *Equation object* combines the fragments of the equation in the appropriate order. The equation fragments are created by instantiating the appropriate *Equation Term classes*.

Each *Equation Term class* represents a particular symbol and contains the functionality to construct the appropriate form of the *Equation Term* from the symbol, the name of the *Zone object* and, where necessary, the name of the *Phase object*. With the *Equation objects* instantiated, the local *Model Subclass* model is generated by calling the appropriate functions within each *Equation object*, formulating the equation by instantiating the appropriate *Equation Terms*, and copying the *Equation object's* parameter and variable list into the relevant *Model Subclass* list. The *model subclass* provides a means of grouping a set of equations together, structuring both the generated equations and the functions required to generate them. These interpret the physical description of the *Zone object* and instantiate the appropriate set of *Equation classes*. In addition to structuring the equations, each *model subclass object* may also add additional text to the equation to improve the clarity of the generated model. To enable the equations to be structured and formulated in a versatile manner, each *model subclass object* may

contain any number of *subclasses*. The model building procedure is essentially the same as before, with the *subclass* instantiating the lowest tier of *subclasses* and each *model subclass* object on the lowest tier instantiating the appropriate *Equation classes*. The final part of the model-building algorithm for each *subclass* is to copy the parameter and variable list for the *subclass* into the appropriate *model class object* list.

6.3 Program Execution

The program takes an input file and generates an output file. The input file should be saved as the <file_name> followed by the suffix ".gwizz". The generated text based output file will be saved as the <file_name> followed by the suffix ".mod" and the generated gPROMS input file will be saved as the <file_name> followed by the suffix ".gPROMS". The program is executed through the following command:

generate <file_name>

The first argument is the name of the executable. The second argument is the name of the text file containing the input description. The suffix ".gwizz", should be omitted when supplying the file name as the argument.

6.4 Summary

In this Chapter the development of a prototype piece of software called *COMPASS*, that implements the model building concepts described in Chapters 3, 4 and 5 has been presented. Written in C++ and making full use of the associated benefits of object oriented programming, *COMPASS* is capable of automatically generating process models from a physical description.

COMPASS is comprised of two key sections: the translator and the model-building engine. The translator analyses the input file, identifying keywords and grammatical structures. On discovery of a predefined grammatical structure, an internal representation of the process is generated from a specific function within the model-building engine. The model-building engine encapsulates the full physical description in an internal representation, checks the validity of this representation and finally generates the appropriate mathematical model. Object oriented techniques are used to structure the model-building engine around a number of discrete objects. The methodology for structuring the model-building engine is shown in the two UML diagrams (Figs. 6.1 and 6.2). In

the next chapter the application of *COMPASS* with be demonstrated with several examples that include standard and novel process units. This will also serve to show that the software itself is finished and, as far as can be assessed, free of errors.

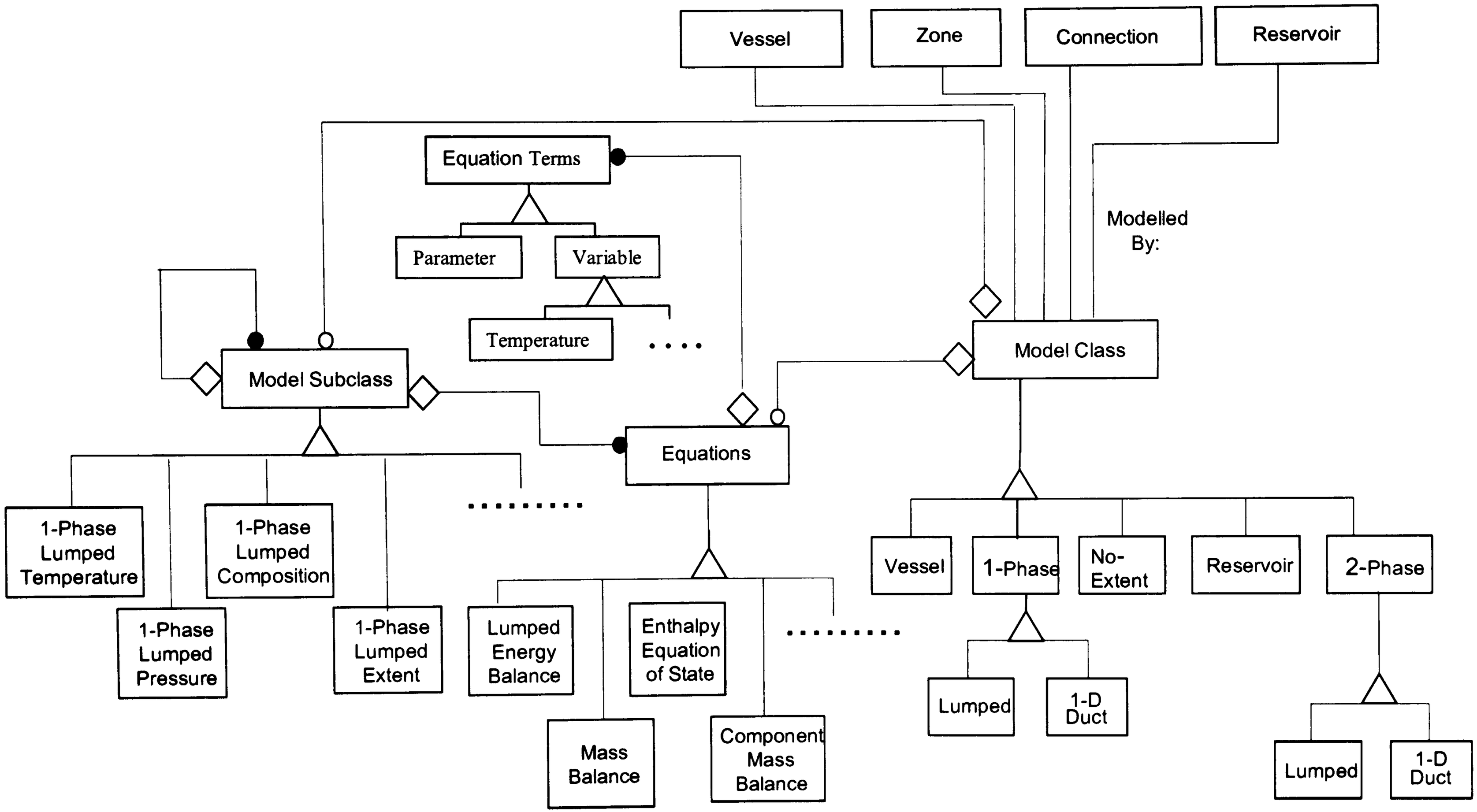


Figure 6.2: The Unified Modelling Language Diagram for the Formulation of the Mathematical Model

Chapter 7

Assessing the Validity and Implementation of the Framework

The previous four chapters have described the development of a framework for the automatic generation of process models and the subsequent implementation of the framework in a prototypical piece of software called *COMPASS* (COMPuter Aided Simulation Software). In this chapter the results from six case studies will be presented to assess the validity of the framework and its implementation. Selected from over twenty examples that have been tested, the six case studies have been chosen to illustrate the meticulousness of the model-building framework, demonstrate the current scope of application and provide an indication of *COMPASS's* potential as a next-generation modelling tool.

The only truly effective means of assessing the performance of the framework and its implementation is painstaking line-by-line examination of the output, and comparison with models created by experts. This has been carried out for twenty different examples, and in all cases the *COMPASS* software has been found to work perfectly. This process is difficult to illustrate in the format of a PhD thesis, so here the focus is on a comparison of results obtained by the different modelling approaches.

The six examples that have been selected to illustrate the success of the model-generating framework are: a buffer-tank system, a gas-phase CSTR, a flash vessel, a gaseous phase plug flow reactor, a drum boiler and a two-phase adsorptive reactor. For the first four case studies, the equivalent *gPROMS* library model components (supplied by Process Systems Enterprise) were used to validate the *COMPASS* generated models. For the remaining two case studies (drum boiler and two-phase adsorptive reactor) the *COMPASS* generated models were validated against documented models sourced from the literature (Adam and Marcheti, 1999; Yongsunthon and Alpay, 1999).

For all six case studies, a uniform presentation structure has been used that comprises of three sections: an introduction to the case study, a discussion of how the model is developed using

COMPASS and an assessment of the accuracy of the generated model. The introduction provides the background to the case study, describing the process that is to be modelled, the design parameters and the operating conditions. Additional information that is required to model the process, such as empirical correlations and control settings, is also provided at this stage. In the model development section, the construction of the model in the *COMPASS*-modelling environment is described. In this section, the representation of the process as a network of *Zones, Reservoirs and Connections* is presented along with a discussion of the model class that is used to model each *Zone*. For those case studies that involve a comparison between the *COMPASS* generated model and the *gPROMS* model library, the model library components used within the reference *gPROMS* model are also listed. The full physical description for each case study, established in terms of the *COMPASS* language, can be found in Appendix J. For each case study, the physical description has been specified in a manner that replicates the model assumptions of the reference model. To illustrate the clarity and versatility of the physical language, an annotated section of the physical description for each case study is presented within this chapter.

Within the final section of each case study, the accuracy of the *COMPASS* generated model is evaluated against the reference model. Both the reference models and the *COMPASS* generated models were solved using *gPROMS* 2.3[10] on a Windows platform, thereby allowing a direct comparison of the model results. To enable all thermodynamic properties to be determined, the necessary *gPROMS* code was added to allow *gPROMS* to interface with Multiflash, a thermodynamic package supplied by Infochem Computer Services. Where necessary, additional kinetic rate equations and constitutive equations were also added by hand. All of the constitutive equations added to each *COMPASS* generated model are discussed, as they arise, in the appropriate section. Finally, for each case study it is shown that the solution results of the *COMPASS* generated models are consistent with those generated from the reference model, thereby proving the validity of both the model generation approach and its software implementation.

7.1 A Buffer Tank

This example considers the flow of fluid through a system comprised of a buffer tank, a pump and a splitter. The buffer tank is fed with a two-component liquid phase, comprised of equal

[10] gPROMS is an equation-oriented modelling environment designed for process systems modelling. The user should refer to Chapter 1 for more information.

quantities of water and methanol, at a mass flowrate of 50 kg/s. The tank is open to the atmosphere and the temperature of the whole system remains constant at 293K. The outflow from the tank is partly driven by the hydrostatic head of the liquid within the tank. To increase the flowrate, pump X101 is attached to outlet pipe P102. The outlet of pump X101 is attached to pipe P103, which itself is attached to splitter S101. The pressure drop across the splitter is assumed to be negligible. Pipes P104 and P105 are attached to the splitter. The outlet pressure for both pipes is assumed to remain at 1 atm throughout the simulation. A diagrammatical representation of the process is given in figure 7.1

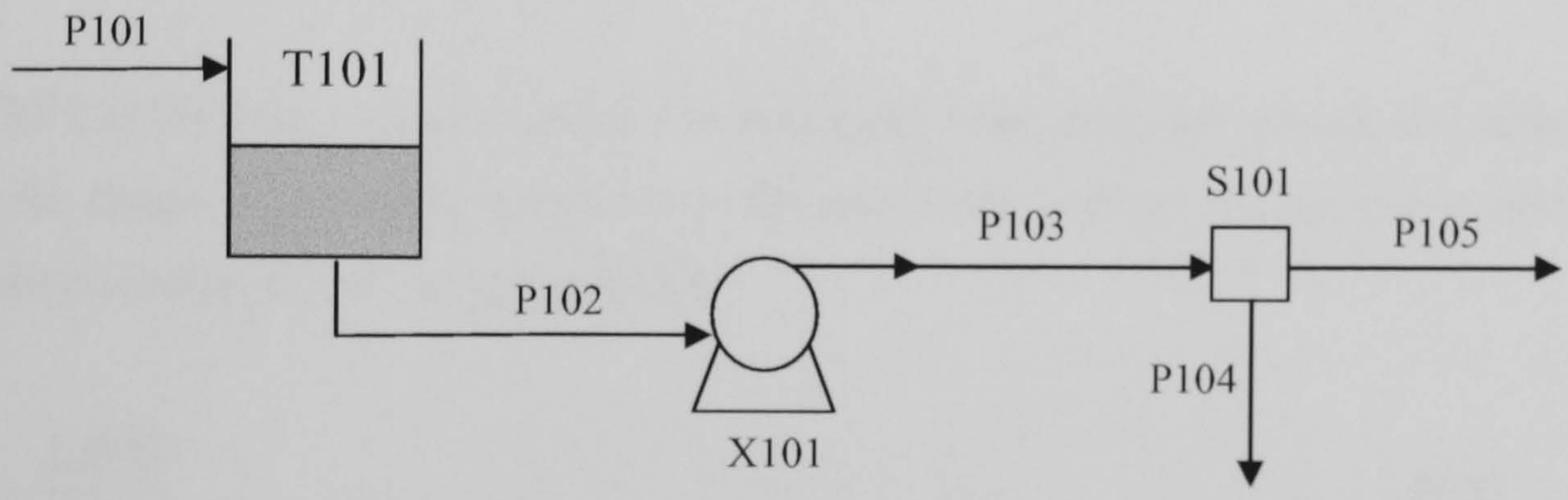


Figure 7.1: A Diagrammatical Representation of the Buffer Tank System

The Pipes: It is assumed that the mass flowrates (F) of material through P102, P103, P104 and P105 can be accurately predicted through the following equation.

$$F = \gamma \Delta P \qquad [7.1]$$

where γ is the flow coefficient (Table 7.1) and ΔP is the pressure drop across the pipe.

Pipe	Flow Coefficient (m.s)
P101	1×10^{-4}
P102	1×10^{-4}
P103	1×10^{-4}
P104	8×10^{-5}
P105	6×10^{-5}

Table 7.1: The Flow Coefficients

The Tank: The tank has a height of 10m and a diameter of 1.5m. Pipe P101 is attached to the top of the tank and pipe P102 is attached to the bottom of the tank. The initial volume of liquid within the tank is $3m^3$, with the same composition as the feed.

The Pump: The volumetric flowrate (Q) through the pump can be related to the pressure rise across the pump through the following equation:

$$1 = \left(\frac{1}{FRPS}\right)^4 \frac{\Delta P^2}{\Delta P_{DS}^2} + \left(\frac{1}{FRPS}\right)^2 \frac{Q^2}{Q_{DS}^2} \qquad [7.2]$$

where FRPS is the pump speed expressed as fractional revolutions per second and subscript DS refers to the design value. FRPS is defined as the ratio of the load revolutions per second (LRPS) to the design revolutions per second (DRPS).

$$FRPS = \frac{LRPS}{DRPS} \qquad [7.3]$$

The LRPS can be calculated through the following equation.

$$LRPS = \frac{Q\Delta P}{2\pi \mathrm{T}} \times \frac{100}{\text{efficiency}} \qquad [7.4]$$

In Eq. 7.4. T is the load torque of the pump. For this simulation it is assumed that the pump efficiency is 90%, the design revolutions per second speed is 1×10^3 s^{-1}, the design volume flowrate is 0.08 m^3s^{-1} and the design differential pressure is 1×10^6 Pa. The load torque is such that the pressure rise across the pump remains at 8.5×10^5 Pa.

7.1.1 gPROMS Model Development

A mathematical model was developed with the use of the *gPROMS* Model Library. The following model library components were used to model the buffer tank system.

Process Component	gPROMS model library component
Feed Material	Source
T101	Liquid_Tank
X101	Centrifugal_Pump
S101	Splitter
P101	Flow_Device
P102, P103, P104 and P105	Simple_pipe
Outlet to P105 and P106	Sink

Table 7.2: The gPROMS Model Library Components Used to Model the Buffer Tank

7.1.2 *COMPASS* Model Development

To encapsulate the behaviour of the process with *COMPASS*, the process was represented by eight *Zones,* as shown in figure 7.2, corresponding to the tank (T101), the pump (X101), the splitter (S101) and the five pipes (P101 – P105). In addition, three *Reservoirs* were used to model the inlet to P101 and the outlets of the two pipes P104 and P105. The network of *Zones* and *Reservoirs* was created with the use of ten *Connections*. The process representation for the buffer tank system is shown in figure 7.2. The process unit that each *Zone* represents, and the model class used to characterise each *Zone* are shown under the *Zone* name.

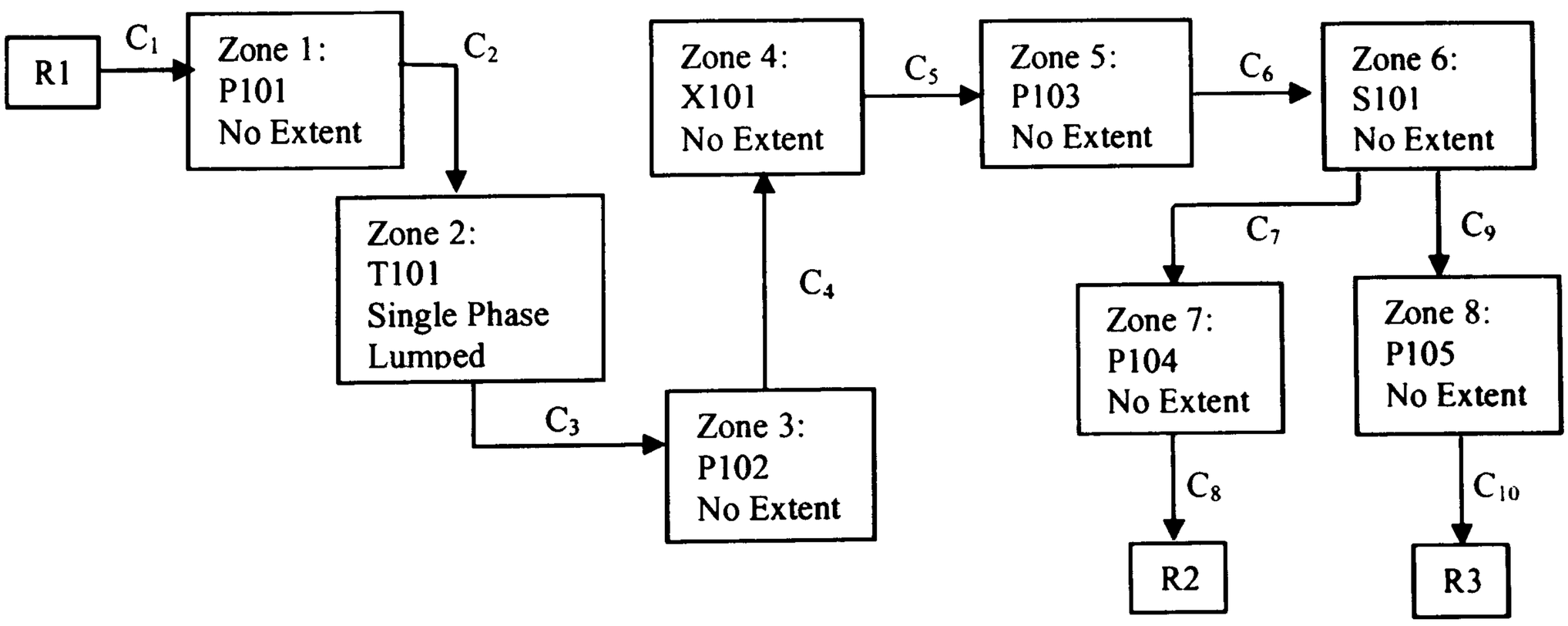


Figure 7.2. The Structural Representation of the Buffer Tank System

The full physical description for each *Zone* is given in Appendix J. For illustrational purposes, the section of the physical description for tank T101 is shown in figure 7.3. Tank T101 is modelled

by one *Zone*, a single phase lumped *Zone* referred to as Z2. The *Zone's* interface is established through *Ports* P1 and P2, with P1 corresponding to the tank inlet and P2 corresponding to the tank outlet. Consideration of the hydrostatic pressure at *Port* P2 is included within the model through the command 'INCLUDE_HYDROSTATIC_PRESSURE'. By modelling the pressure variation through the *Zone* as a discontinuity at *Port* P2, the pressure within the *Zone's* interior can be set to atmospheric pressure (Line 17). The dynamic variation in the volume of liquid within the tank is modelled firstly by declaring the *Zone's* extent to be a variable (Line 12) and secondly by requesting a dynamic model (Line 11). With the composition and temperature of the material within tank T101 remaining constant throughout the simulation, appropriate values are assigned to these attributes (lines 13 – 17).

```
Line 1          VESSEL : Tank : T101  ;
     2                  DEFINE_ZONES
     3                          ZONE : Z2 ;
     4                          GEOMETRY : LUMPED ;
     5                          PHASE : product_stream ;
     6                          DEFINE_PORTS
     7                                  PORT : P1 ;
     8                                  PORT : P2 : INCLUDE_HYDROSTATIC_PRESSURE ;
     9                          END #DEFINE_PORTS
     10                         DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
     11                         ZONE_DYNAMICS : DYNAMIC ;
     12                         ZONE_EXTENT : VARIABLE ;
     13                         ZONE_COMPOSITION : FIXED ;
     14                                 Methanol = 0.5 ;
     15                                 Water = 0.5 ;
     16                                 ZONE_TEMPERATURE = 293 ;
     17                                 ZONE_PRESSURE = 1.013e+5 ;
     18                         END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION
     19         END #zone section
```

Figure 7.3: The Physical Description for Tank T101 Declared in Terms of the *COMPASS* Language

7.1.3 Model Assessment

Assessment of the *COMPASS* generated model was carried out by comparing the simulated results for the volume of liquid within the tank and the mass flowrate through pipes P104 and P105. Equations 7.2, 7.3 and 7.4 were added by hand to the model for Zone 3 (the pump). The results of the comparison can be found in figures 7.4, 7.5 and 7.6, where the simulated results for the liquid volume within the tank (figure 7.4) and the mass flowrate through Pipes P104 and P105 are plotted (figures 7.5 and 7.6). In all figures a solid red line represents the results obtained from the gPROMS model library simulation and the black dots represent the results obtained from the

COMPASS model simulation. The results from the *COMPASS* generated model show excellent agreement with those from the *gPROMS* Model Library model.

Both the generated text and gPROMS files can be found on the CD attached to the back of this thesis. Sadly the size of generated model files prevented them from being printed in this thesis.

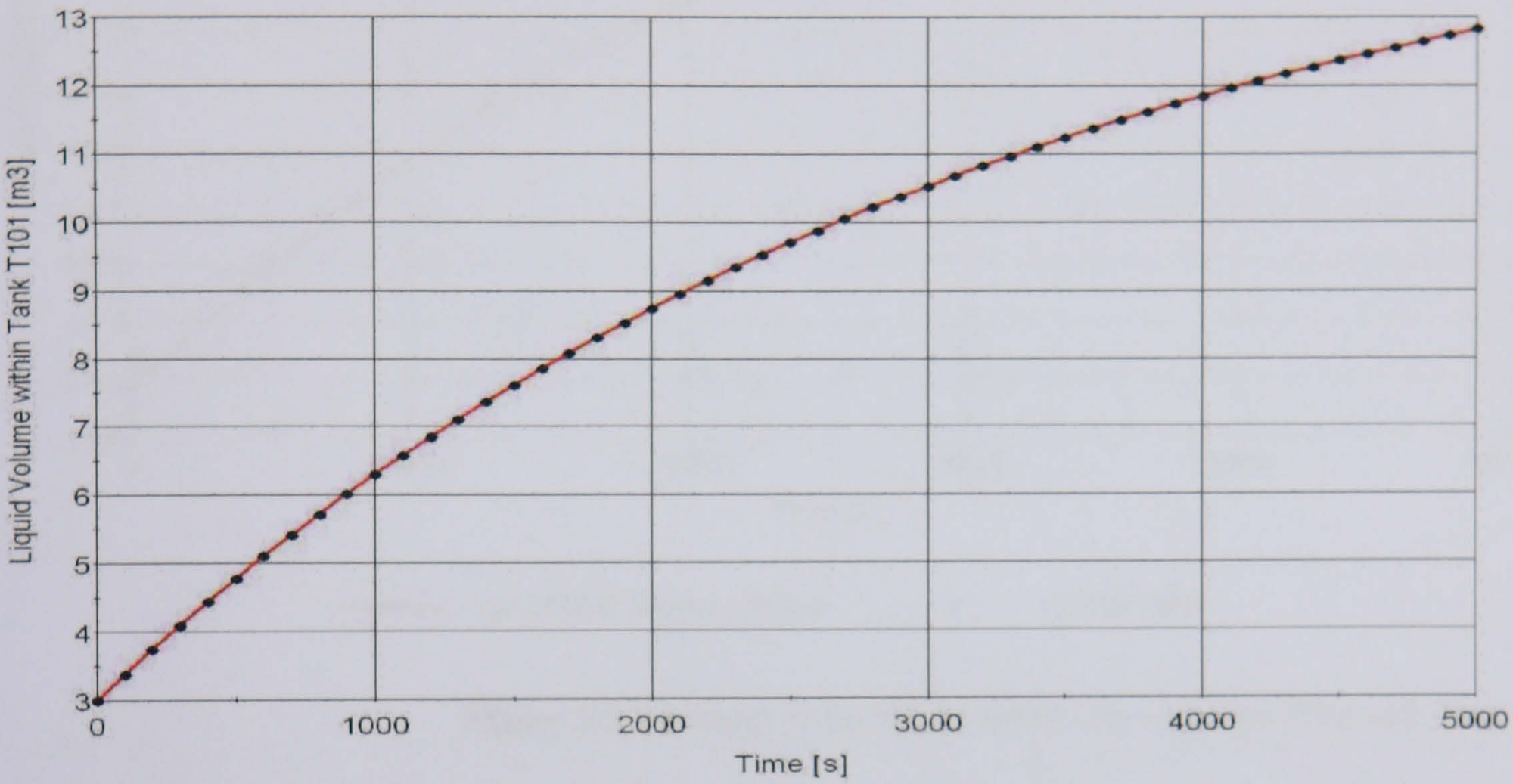


Figure 7.4: Variation in the Liquid Volume within Tank T101 with Time

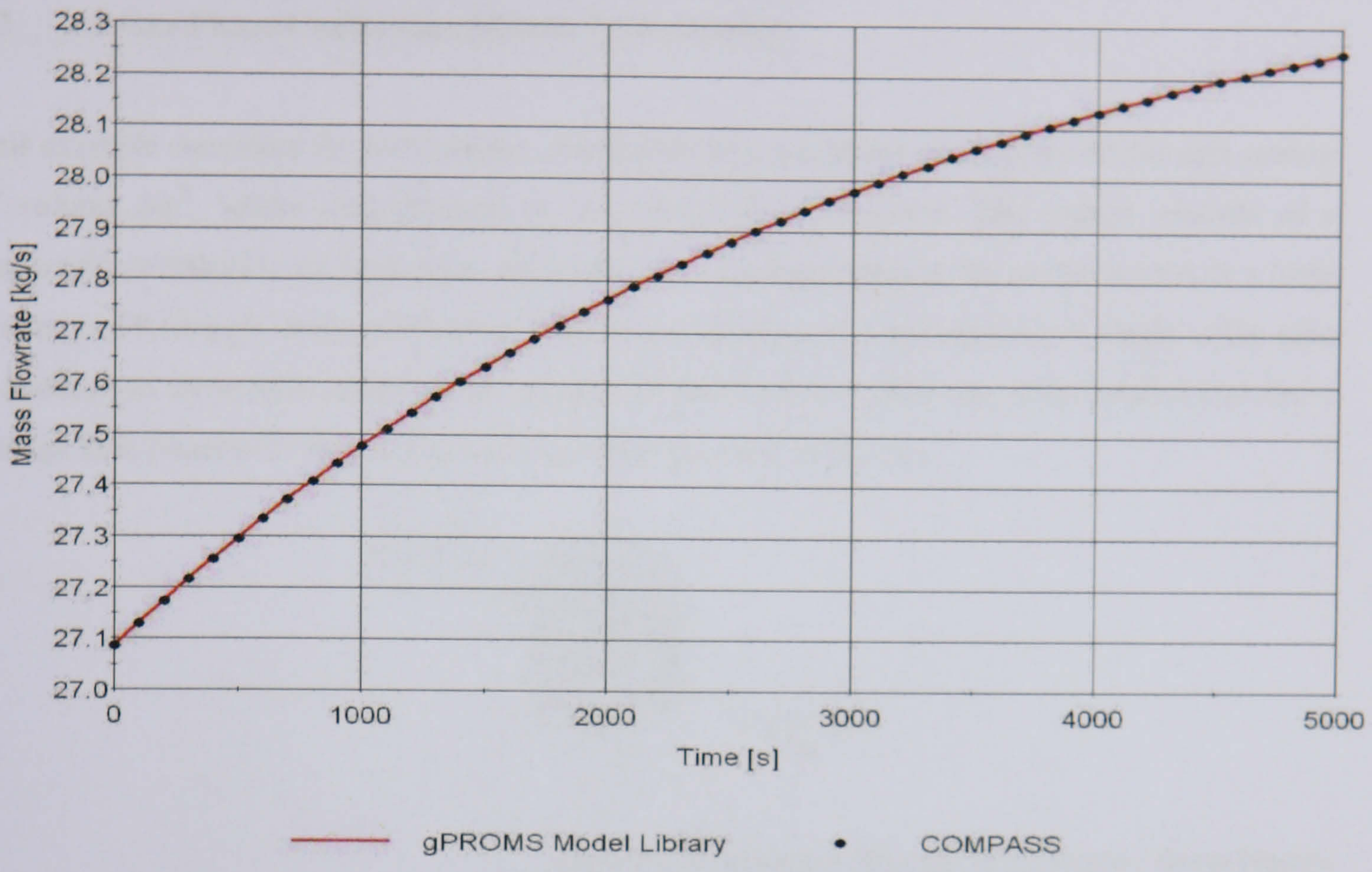


Figure 7.5: Variation in the Mass Flowrate through Pipe P104 with Time

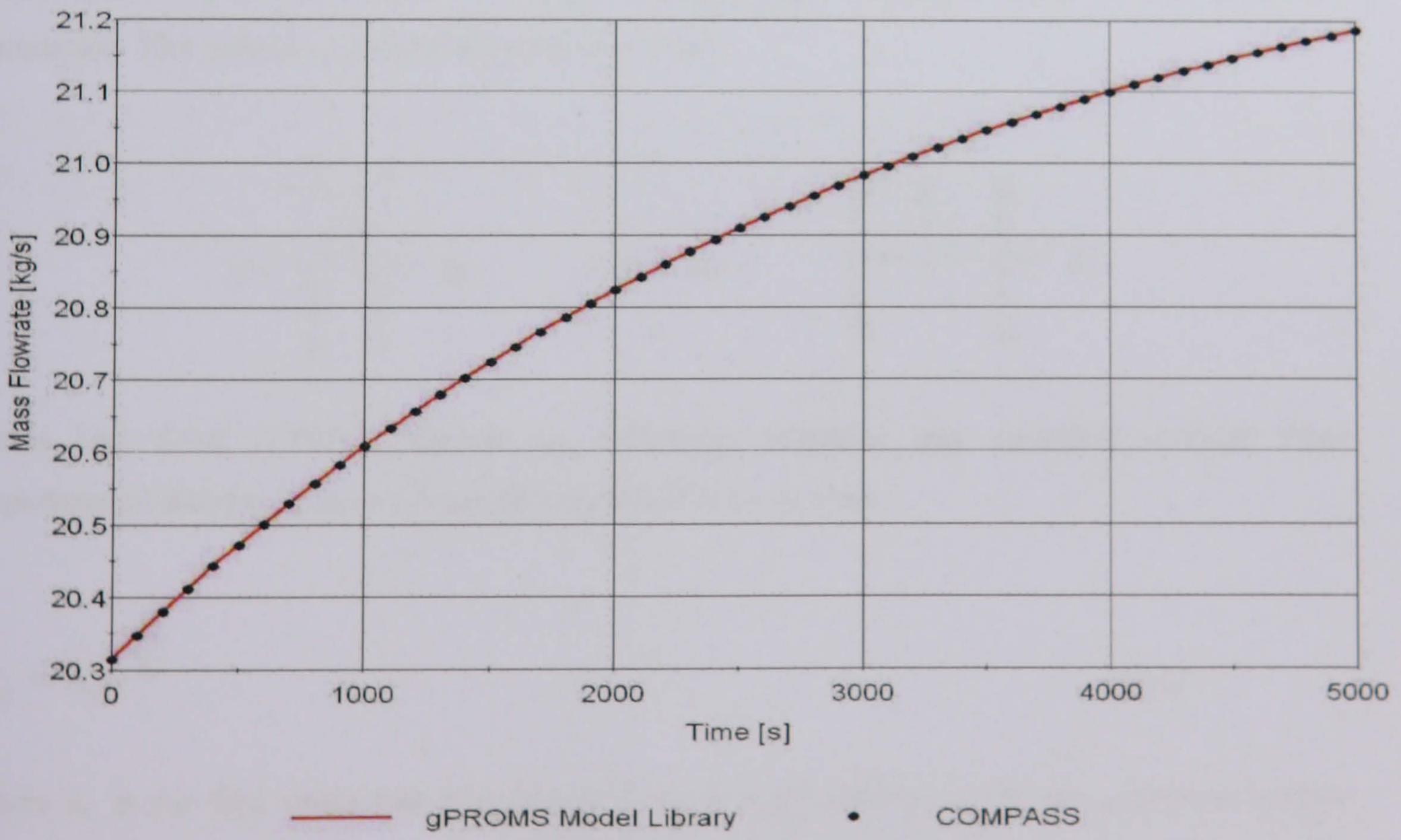


Figure 7.6: Variation in the Mass Flowrate through Pipe P105 with Time

7.2 A Gas-Phase Continuous Stirred Tank Reactor

This example considers the formulation of a model for a gas-phase continuous stirred tank reactor of volume $2m^3$, where cyclopropane is isomerized into propylene. The system consists of a reactor drum linked to an inlet pipe and outlet pipe. Cyclopropane is fed to the reactor at a mass flowrate of 0.01 kg/s, a temperature of 780K and a pressure of 1 atmosphere. A single outlet pipe is attached to the reactor, allowing the product of the reaction to flow out of the reactor and into a storage tank (reservoir) that remains at a constant pressure of 0.9 bar.

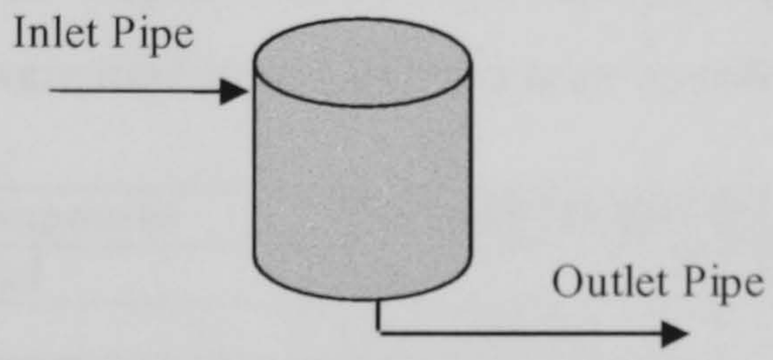
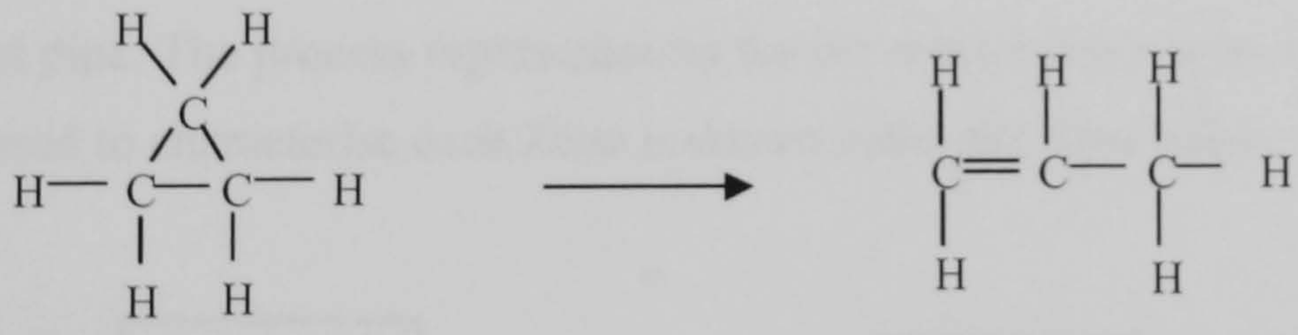


Figure 7.7: Schematic Diagram of the Reactor Drum Process

The reactor is assumed to be adiabatic and is initially at a temperature of 780K and a pressure of 1 atmosphere. It is assumed that there is pure cyclopropane within the reactor at the start of the simulation. The following first order reaction occurs:



Davis and Scott (1964) proposed the following arrhenius rate equation, devised from experimental data taken from a borosilicate glass reactor at 1bar:

$$k_c = k_0 e^{\frac{E_a}{RT}} \qquad [7.5]$$

where k_c is the first order rate constant (s^{-1}), k_0 is 5.225×10^5 s^{-1}, E_a is the activation energy (2.801×10^5 $kJ.kmol^{-1}$), R is the universal gas constant (8.314 $kJ.kmol^{-1}.K^{-1}$) and T is the temperature of the reactor (K). It is assumed that the exit pressure of the outlet pipe remains at 0.9

bar. The flow of material through the outlet pipe is pressure driven and it is assumed that the mass flowrate (F) through the pipe can be accurately predicted through the following equation.

$$F = \gamma \Delta P \qquad [7.6]$$

where ΔP is the pressure drop across the pipe and γ is the flow coefficient (1×10^{-6} m.s).

7.2.1 gPROMS Model Development

A mathematical model was developed with the use of the *gPROMS* Model Library. The following model library components were used to model the reactor system.

Process Component	gPROMS Model Library Component
Feed Material	Source
Inlet Pipe	Flow_Device
Reactor Vessel	Reactor_drum_kinetic
Outlet Pipe	Simple_pipe
Storage Tank	Sink

Table 7.3: The gPROMS Model Library Component Used to Model the CSTR

7.2.2 *COMPASS* Model Development

The process was represented by three *Zones* corresponding to the inlet pipe, the gas phase in the tank and the outlet pipe. The process representation for the reactor drum process is shown below. The model class used to characterise each *Zone* is shown under the *Zone* name.

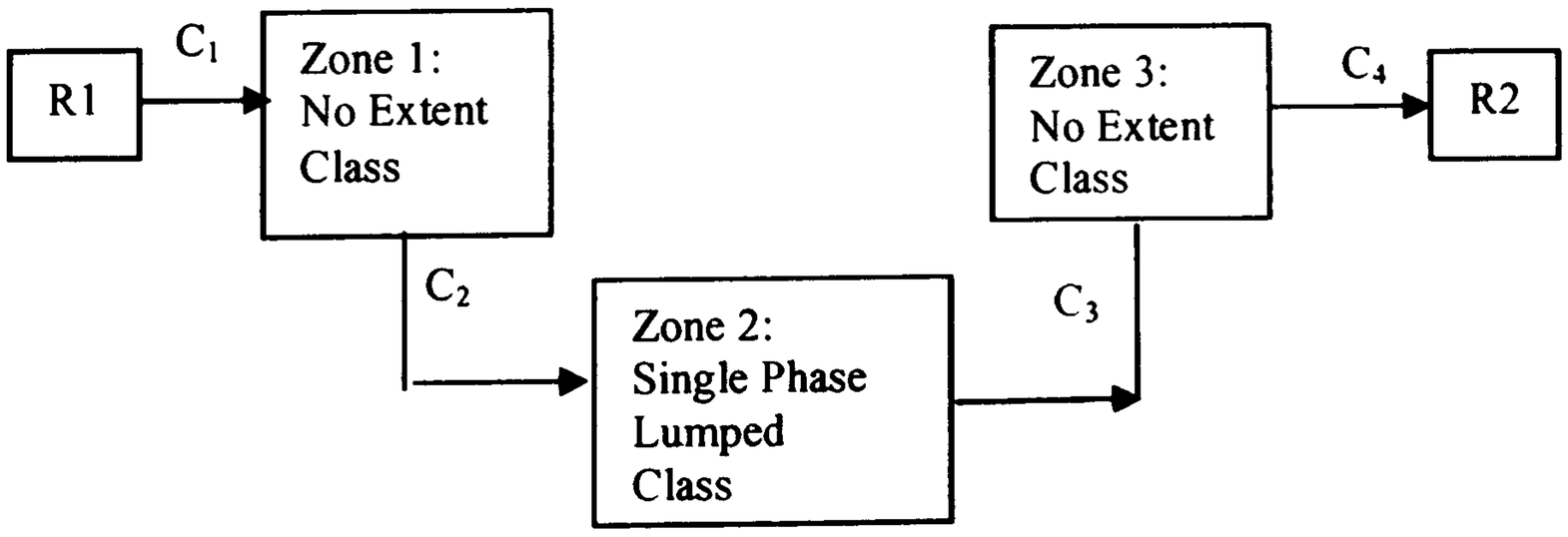


Figure 7.8: The Structural Representation of the Reactor Drum Process

The formal description of the process, declared in terms of the process modelling language *COMPASS*, is given in Appendix J. For illustrational purposes, the section of the physical

description for reactor is shown in figure 7.9. The reactor is modelled by one *Zone*, a single phase lumped *Zone* called Z2. The material within the tank is in one stable phase referred to as the product_stream, defined previously as a gaseous two-component phase. The *Zone's* interface is established through *Ports* P1 and P2, corresponding to the tank inlet and tank outlet respectively. The aspects of the reactor's behaviour that are of interest to the modeller include the variation in the phase composition caused by the reaction (line 13) and the subsequent change in the temperature (line 14). From line 14 it can be seen that the thermal energy of the material within the phase is assumed to be considerably greater that the kinetic and potential components. As the reactor is a closed pressure vessel, the extent of the *Zone* is assigned a constant value (line 12) and the pressure of the reactor is specified as an unknown (line 15).

```
Line 1          VESSEL : Reactor : R101 : CLOSED ;
     2          DEFINE_ZONES
     3                  ZONE : Z2 ;
     4                  GEOMETRY : LUMPED ;
     5                  PHASE : product_stream ;
     6                  DEFINE_PORTS
     7                          PORT : P1 ;
     8                          PORT : P2 ;
     9                  END #DEFINE_PORTS
     10                 DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
     11                 ZONE_DYNAMICS : DYNAMIC ;
     12                 ZONE_EXTENT = 2 ;
     13                 ZONE_COMPOSITION : VARIABLE : REACTION ;
     14                 ZONE_TEMPERATURE : VARIABLE : NEGLECT_KE : NEGLECT_PE ;
     15                 ZONE_PRESSURE : VARIABLE ;
     16                 END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION
     17         END #zone section
```

Figure 7.9: The Physical Description for the Reactor Declared in Terms of the COMPASS Language

7.2.3 Model Assessment

Assessment of the *COMPASS* generated model was carried out by comparing the results for the temperature, pressure and mass fraction of the two components within the reactor. The results of the comparison can be found in figures 7.10, 7.11, and 7.12, where the variation in the mass fraction of propylene (figure 7.10), temperature (figure 7.11) and pressure (figure 7.12) are plotted. In all figures a solid red line represents the results from the *gPROMS* model library simulation and the black dots represent the results from the simulation of the *COMPASS* model. The results from the *COMPASS* generated model again show excellent agreement with the *gPROMS* Model Library model.

The text language output file generated by *COMPASS* can be found in Appendix K. The generated *gPROMS* file is stored in electronic form on the CD attached to the back of this thesis.

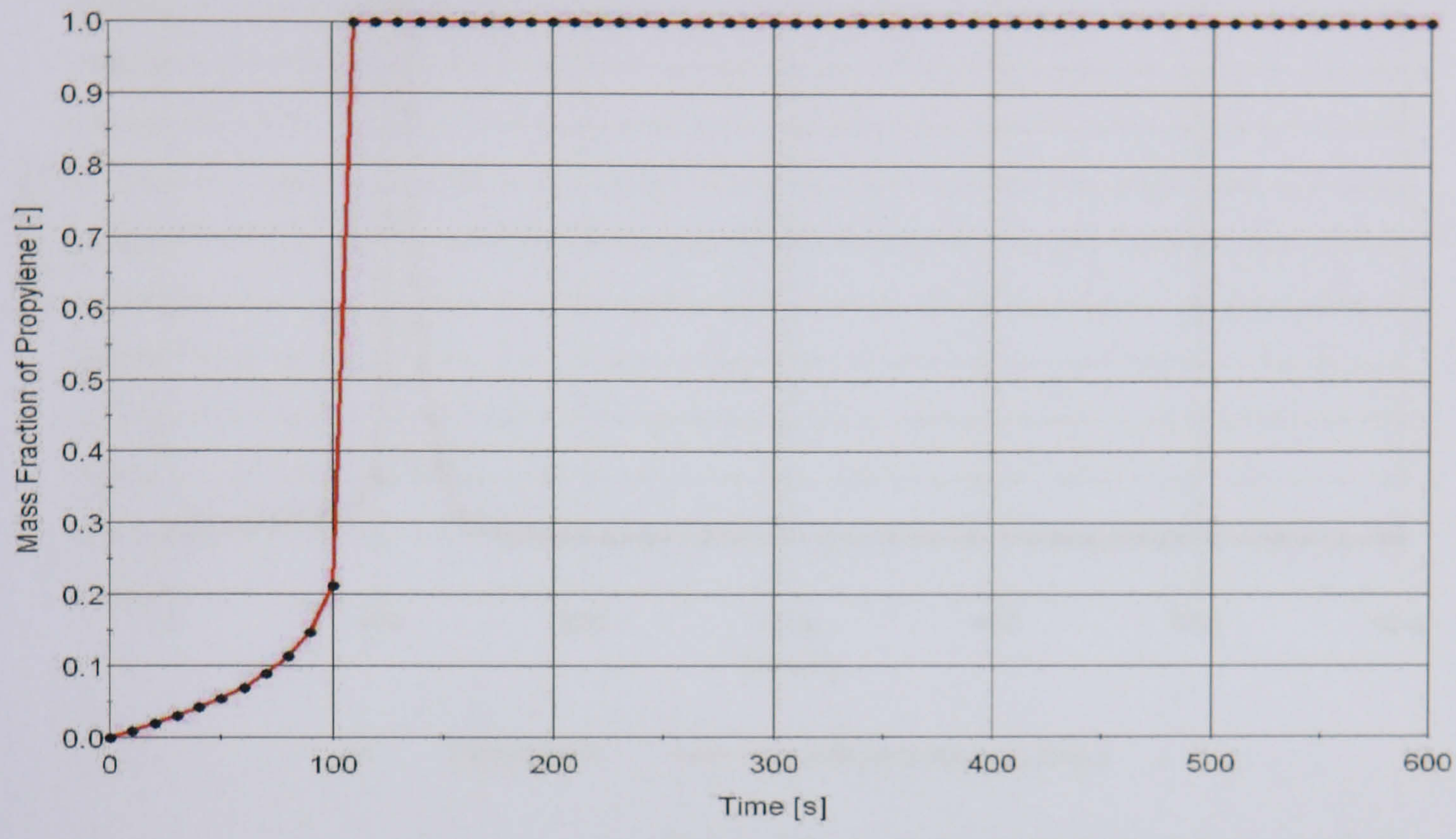


Figure 7.10: Variation in the Outlet Mass Fraction of Propylene with Time

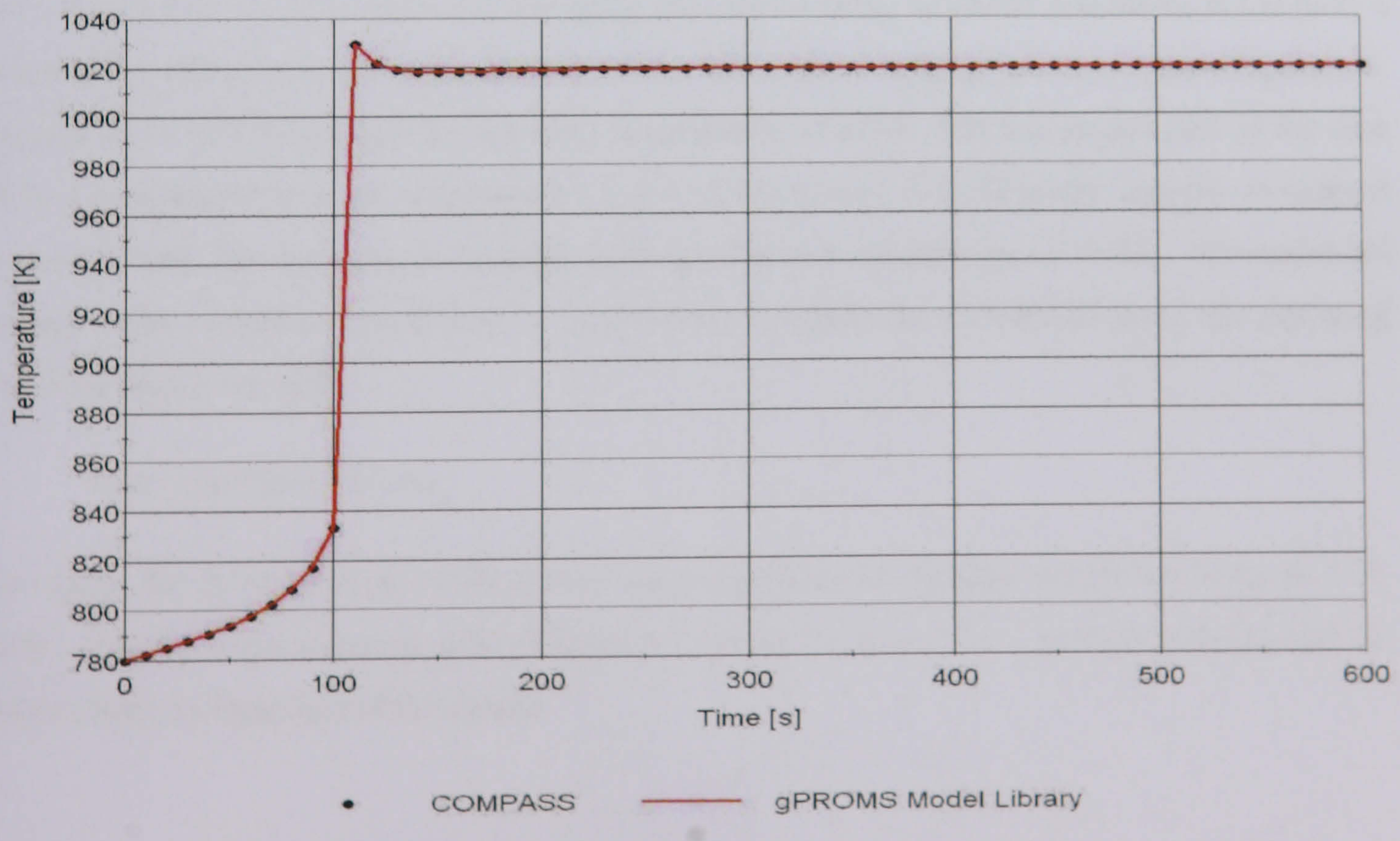


Figure 7.11: Variation in the Reactor Temperature with Time

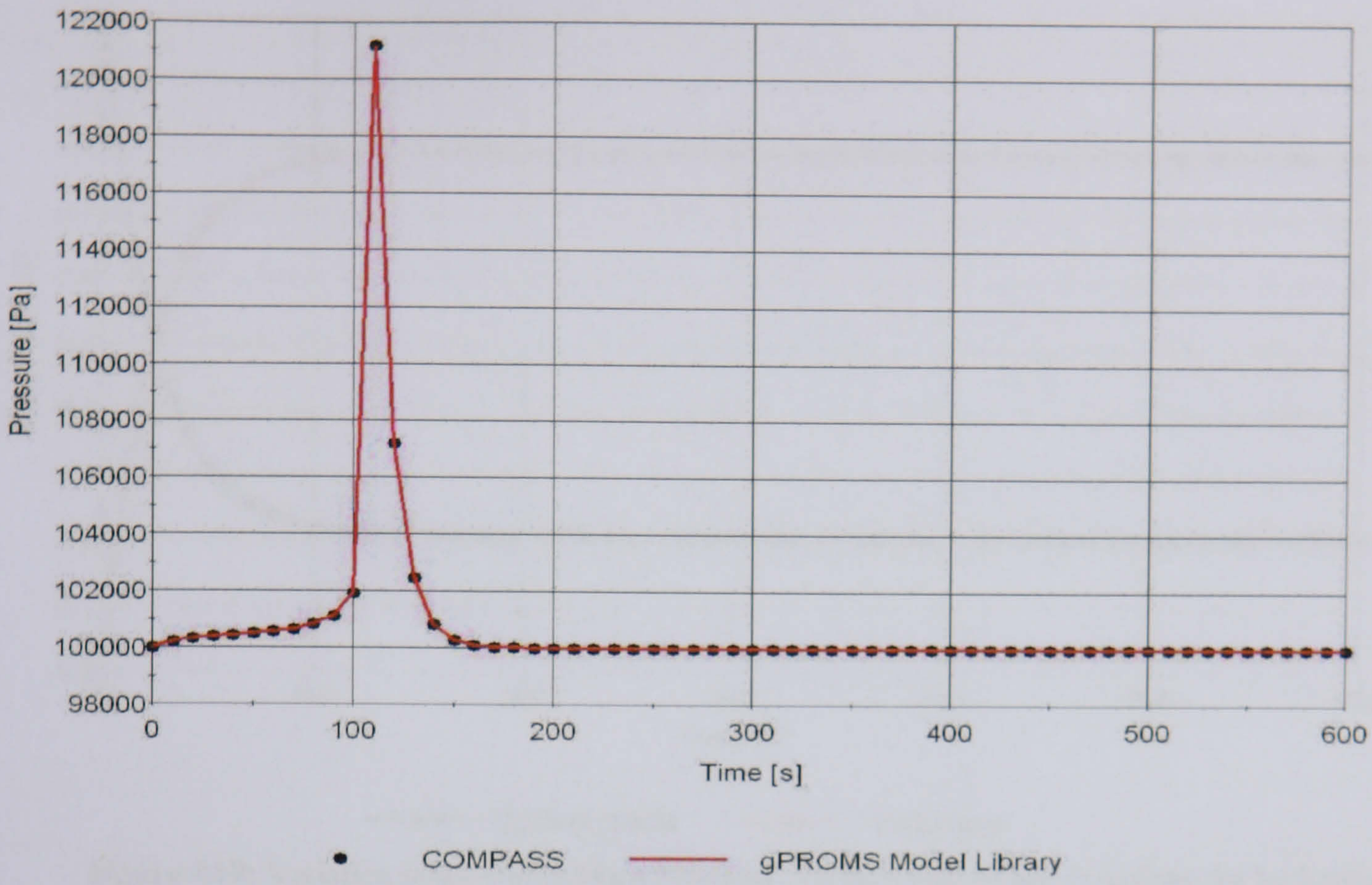


Figure 7.12: Variation in the Reactor Pressure with Time

7.2.4 Model Extension

From figure 7.11 it can be seen that operating the reactor under adiabatic conditions could have a catastrophic effect on the process. It is therefore advisable to attempt to control the temperature. Holland et al. (1979) suggest an operating temperature of 850K. For the second part of the case study it is assumed that the temperature control of the reactor is sufficiently accurate to support the assumption that the reactor operates isothermally at a temperature of 850K. To model the process under isothermal conditions, it is necessary to adjust the *COMPASS* input file replacing line 14 in figure 7.9 with:

```
ZONE_TEMPERATURE = 850 ;
```

The results for the simulation of the reactor under isothermal conditions are shown in figure 7.13. Sadly, as there is no available gPROMS library model for this case, it has not been possible to make a direct comparison of the results.

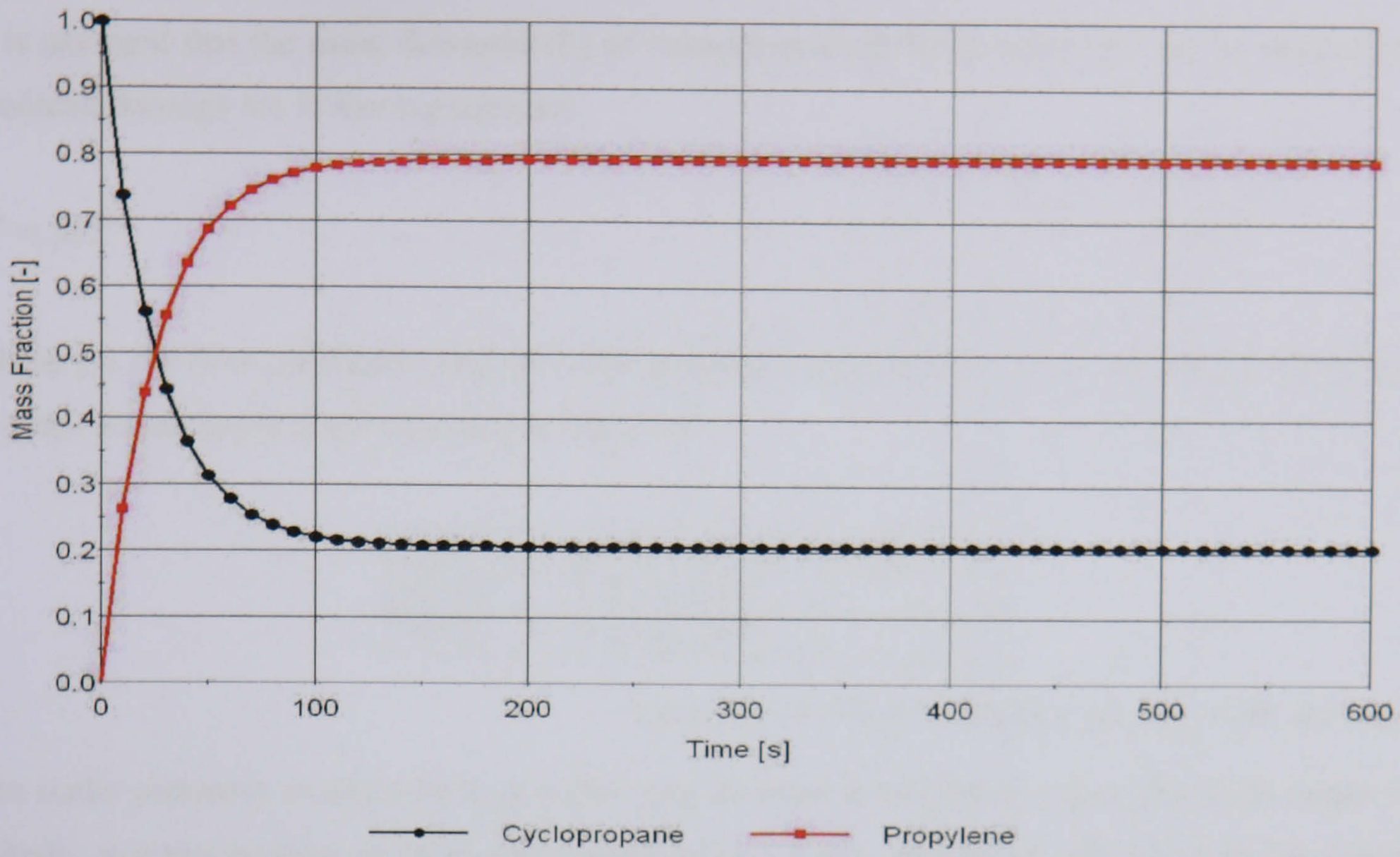


Figure 7.13: Variation in the Outlet Mass Fractions of Cyclopropane and Propylene for Isothermal Operation at 850K.

7.3 An Adiabatic Flash Vessel

This case study considers the formulation of a model for an adiabatic flash vessel (V101) that contains two phases in chemical and thermal equilibrium. A liquid stream comprised of equal mass fractions of propane and pentane is pumped into the vessel at a mass flowrate of 10 kg/s. The pressure of the inlet stream is 10 atmospheres and the temperature of the inlet stream is 293K. The cylindrical flash vessel has a volume of $5m^3$ and a diameter of 1.5m. The liquid and gas phases flow out of the tank through pipes P102 and P103 respectively. The flow of material through each pipe is pressure driven. A diagrammatical representation of the process is shown in figure 7.14.

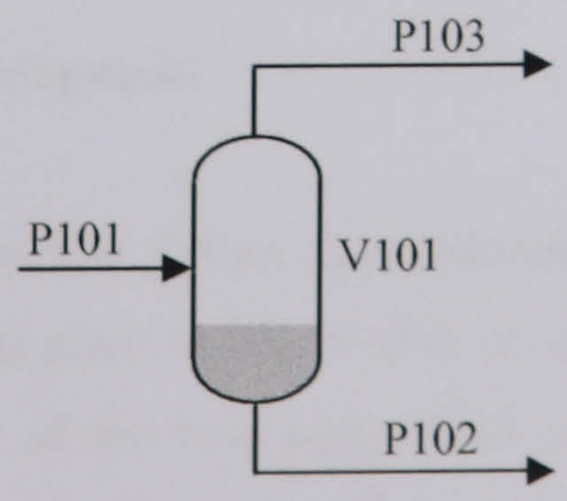


Figure 7.14: A Diagrammatical Representation of the Flash Vessel

It is assumed that the mass flowrates (F) of material through P102 and P103 can be adequately predicted through the following equation.

$$F = \gamma \Delta P \qquad [7.7]$$

where γ is the flow coefficient and ΔP is the pressure drop across the pipe. The flow coefficients for the two outlet pipes are tabulated in Table 7.4.

Pipe	Flow Coefficient (m.s)
P102	4.576×10^{-4}
P103	2.576×10^{-4}

Table 7.4: The Flow Coefficients for Pipes P102 and P103

The outlet pressures of pipes P102 and P103 are assumed to remain at 1 atm. The flash vessel is initially at a temperature of 267K and a pressure of 1.2 atm. The initial volume of liquid within the flash vessel is $1.9m^3$, at equilibrium with the gas.

7.3.1 gPROMS Model Development

A mathematical model was developed for the adiabatic flash vessel with the use of the *gPROMS* Model Library. The following model library components were used to model the flash system.

Process Component	**gPROMS model library component**
Feed Material	Source
V101	Flash_Drum
P101	Flow_Device
P102 and P103	simple_pipe
Outlet to P102 and P103	Sink

Table 7.5: The gPROMS Model Library Components Used to Model the Flash Vessel

7.3.2 COMPASS Model Development

The process was represented by four *Zones,* corresponding to the inlet pipe (P101), the flash vessel (V101) and the two outlet pipes (P102, P103). In addition three *Reservoirs* were used to model the feed and the outlets of the two pipes P102 and P103. The network of *Zones* and *Reservoirs* was created with the use of six *Connections*. The process representation for the flash

vessel system is shown in figure 7.15. The component that each *Zone* represents, and the model class used to characterise each *Zone* are shown under the *Zone* name.

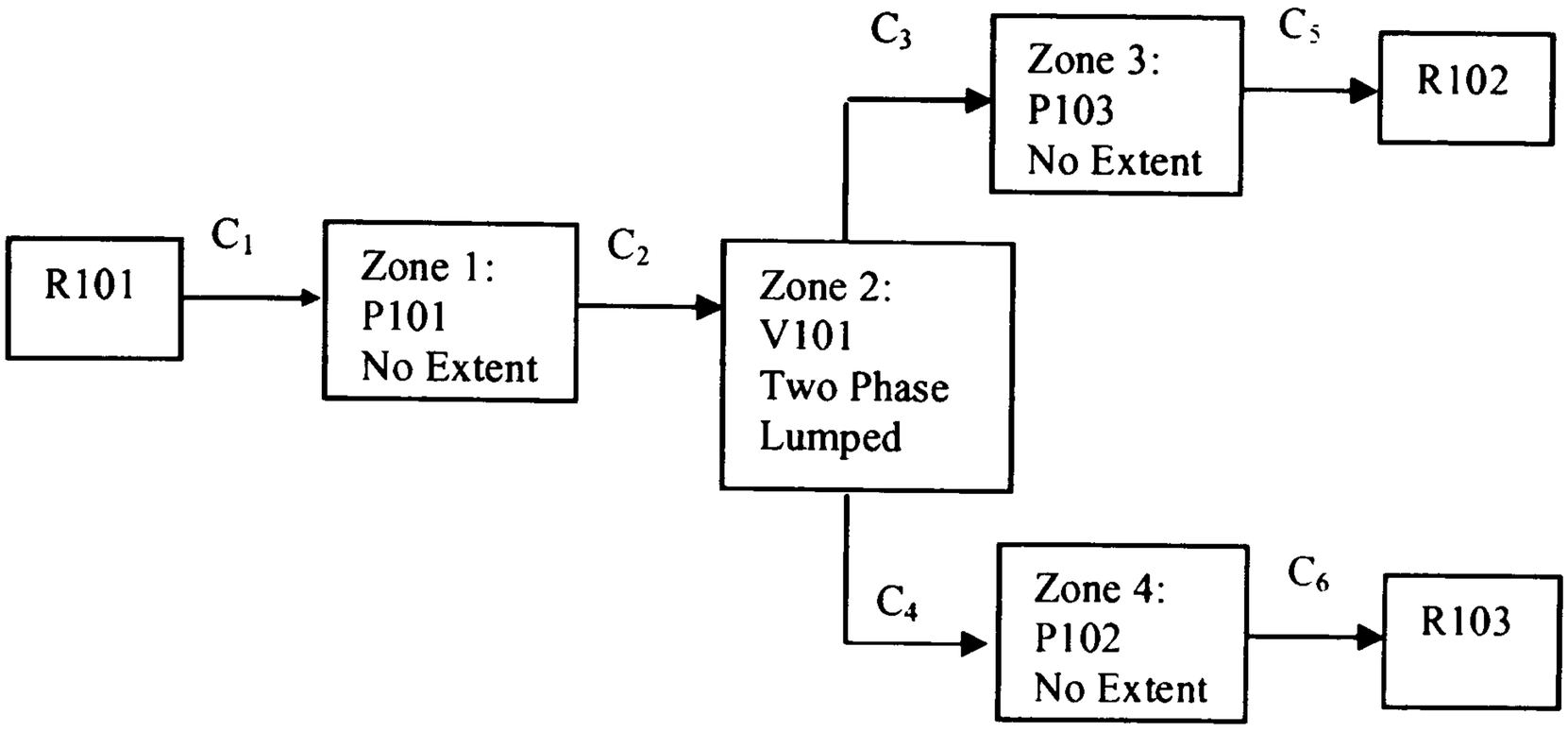


Figure 7.15: The Structural Representation of the Flash Vessel

The complete *COMPASS* language declaration for the flash vessel case study is given in Appendix J. To illustrate the physical language for a two-phase lumped *Zone*, the physical description for the flash vessel is shown in figure 7.16. The flash vessel is modelled by a two phase lumped *Zone*. The two phases within the vessel are referred to as GP, the gas phase, and LP, the liquid phase. The name, phase type, compressibility and component set of each phase has been declared previously within the input file. The *Zone* has three *Ports*. P1 corresponds to the inlet to the tank whilst P2 and P3 correspond to the tank outlets. From the *Connections* linked to the two outlet *Ports* it can be deduced that the gaseous phase flows through *Port* P2 and the liquid phase flows through *Port* P3. The hydrostatic pressure of the liquid at *Port P3* is also considered (Line 9). With the vessel being closed, the *Zone's* extent is equated to the volume of the vessel (line 13). Due to the chemical equilibrium between the two phases, the composition of both phases is unknown and varies over time (lines 15 and 16). The chemical equilibrium between the two phases is specified through the keyword 'EQUILIBRIUM'. The temperature, pressure and volume fraction of the two phases are unknown and must be determined as part of the overall model (lines 18 – 20). Finally, as the phases form two stratified layers there is no relative motion between them and the velocity of the mixture is declared as being stationary (line 21).

```
Line 1          VESSEL : Flash_Vessel : FV101 : CLOSED : 5 ;
     2          DEFINE_ZONES
     3                  ZONE : Z2 ;
     4                  GEOMETRY : LUMPED ;
     5                  PHASE : GP, LP ;
     6                  DEFINE_PORTS
     7                          PORT : P1 ;
     8                          PORT : P2 ;
     9                          PORT : P3 : INCLUDE_HYDROSTATIC_PRESSURE ;
     10                 END #DEFINE_PORTS
     11                 DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
     12                 ZONE_DYNAMICS : DYNAMIC ;
     13                 ZONE_EXTENT = 5 ;
     14                 ZONE_COMPOSITION  ;
     15                         PHASE_COMPOSITION : LP : VARIABLE ;
     16                         PHASE_COMPOSITION : GP : VARIABLE ;
     17                         INTERFACIAL_MASS_TRANSFER : EQUILIBRIUM ;
     18                 ZONE_TEMPERATURE : VARIABLE : NEGLECT_KE : NEGLECT_PE  ;
     19                 ZONE_PRESSURE : VARIABLE ;
     20                 PHASE_VOLUME_FRACTION : VARIABLE ;
     21                 ZONE_VELOCITY : STATIONARY ;
     22                 END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION
     23         END #zone section
```

Figure 7.16: The Physical Description for the Flash Vessel Declared in Terms of the *COMPASS* Language

7.3.3 Model Assessment

Assessment of the *COMPASS* generated model was carried out by comparing the simulated results for the pressure (figure 7.17) and temperature (figure 7.18) of the flash vessel and the composition of each phase within the flash vessel (figures 7.19 and 7.20). In figures 7.17 – 7.20, a solid red line represents the results from the gPROMS model library simulation and the black dots represent the data obtained from the *COMPASS* model. As in the two previous examples, the results from the *COMPASS* generated model shows excellent agreement with the *gPROMS* Model Library model. Both the output text file and the output gPROMS file are stored electronically on the CD attached to the back of this thesis.

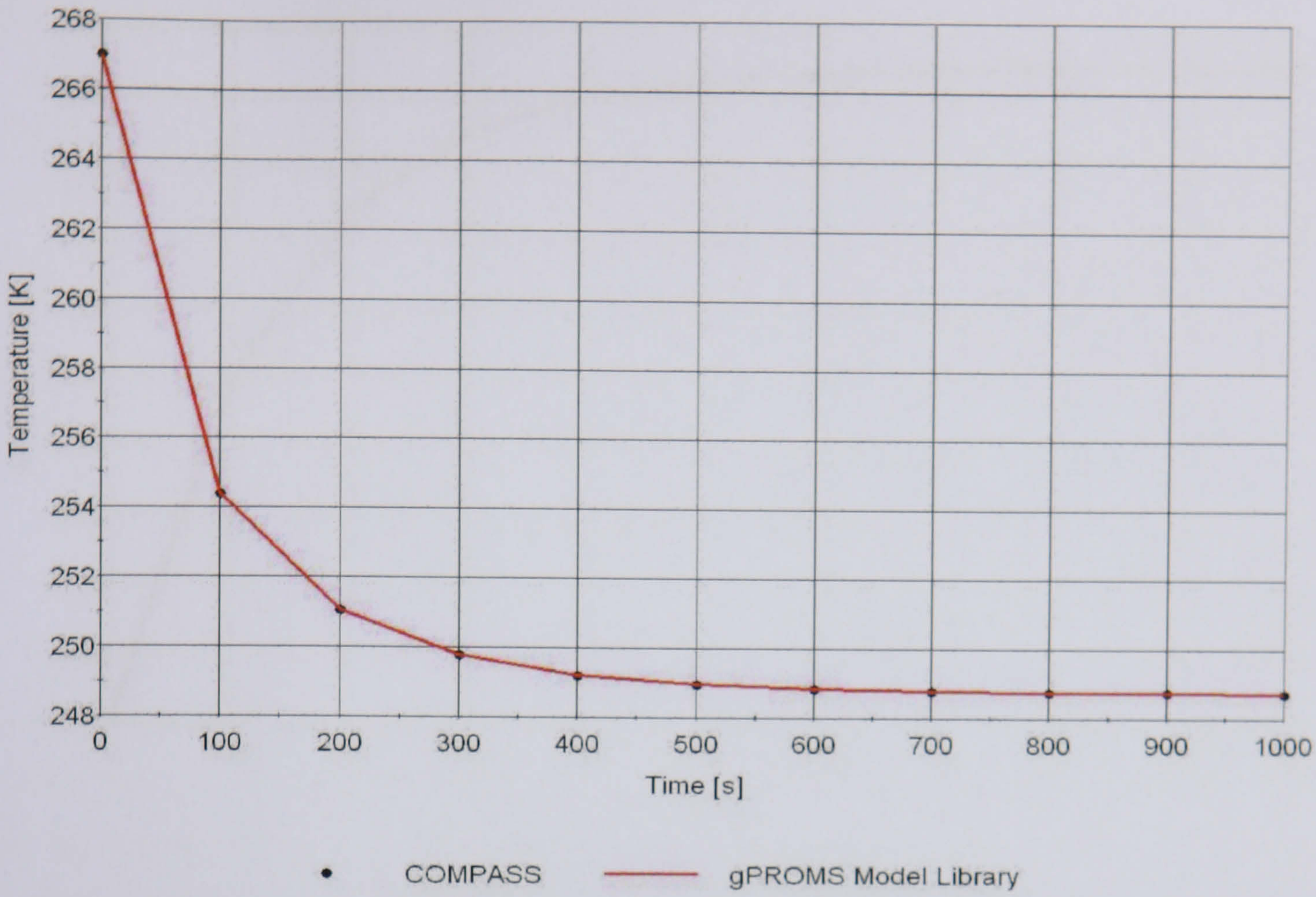


Figure 7.17: Variation of the Temperature in the Flash Vessel with Time



Figure 7.18: Variation of the Pressure in the Flash Vessel with Time

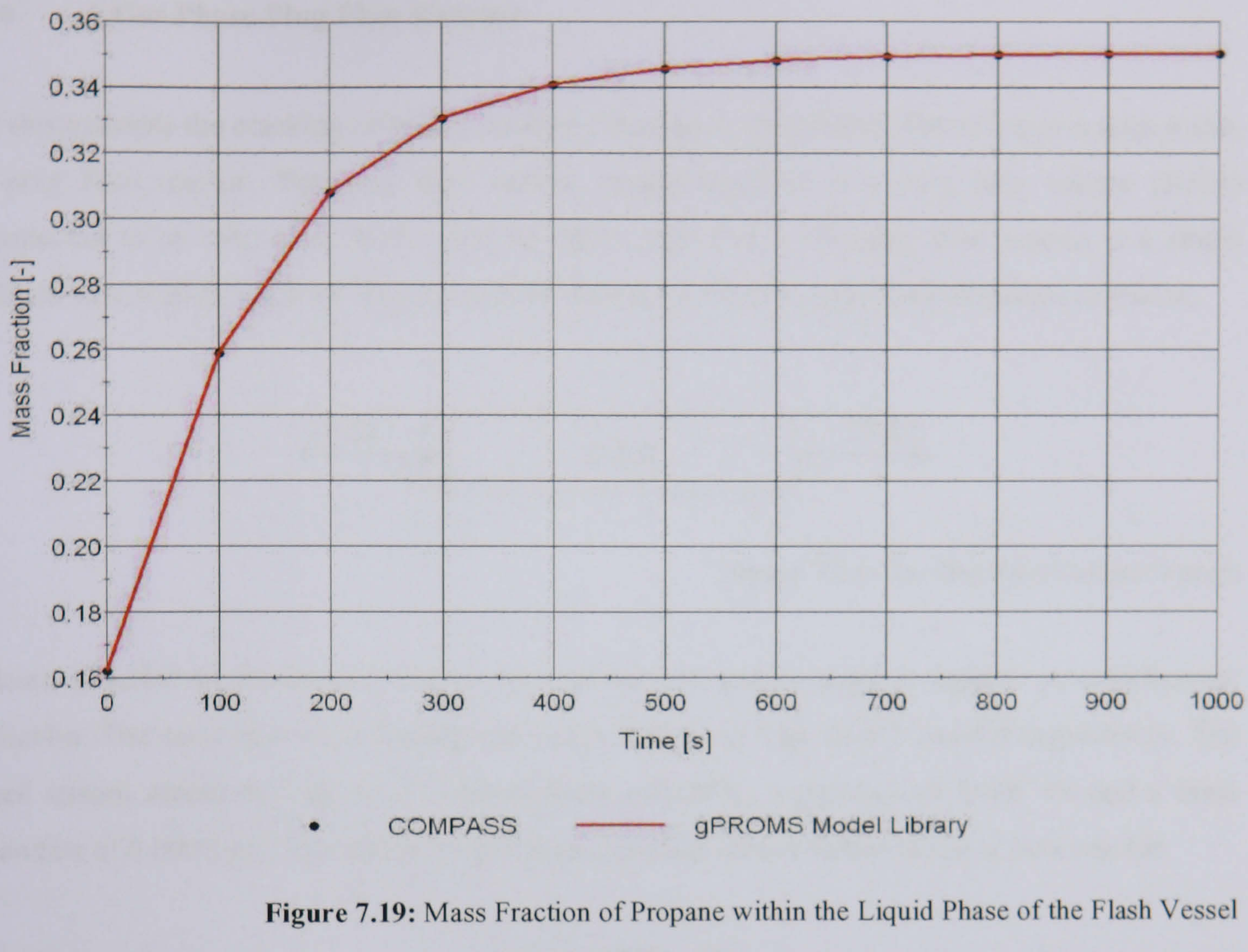


Figure 7.19: Mass Fraction of Propane within the Liquid Phase of the Flash Vessel

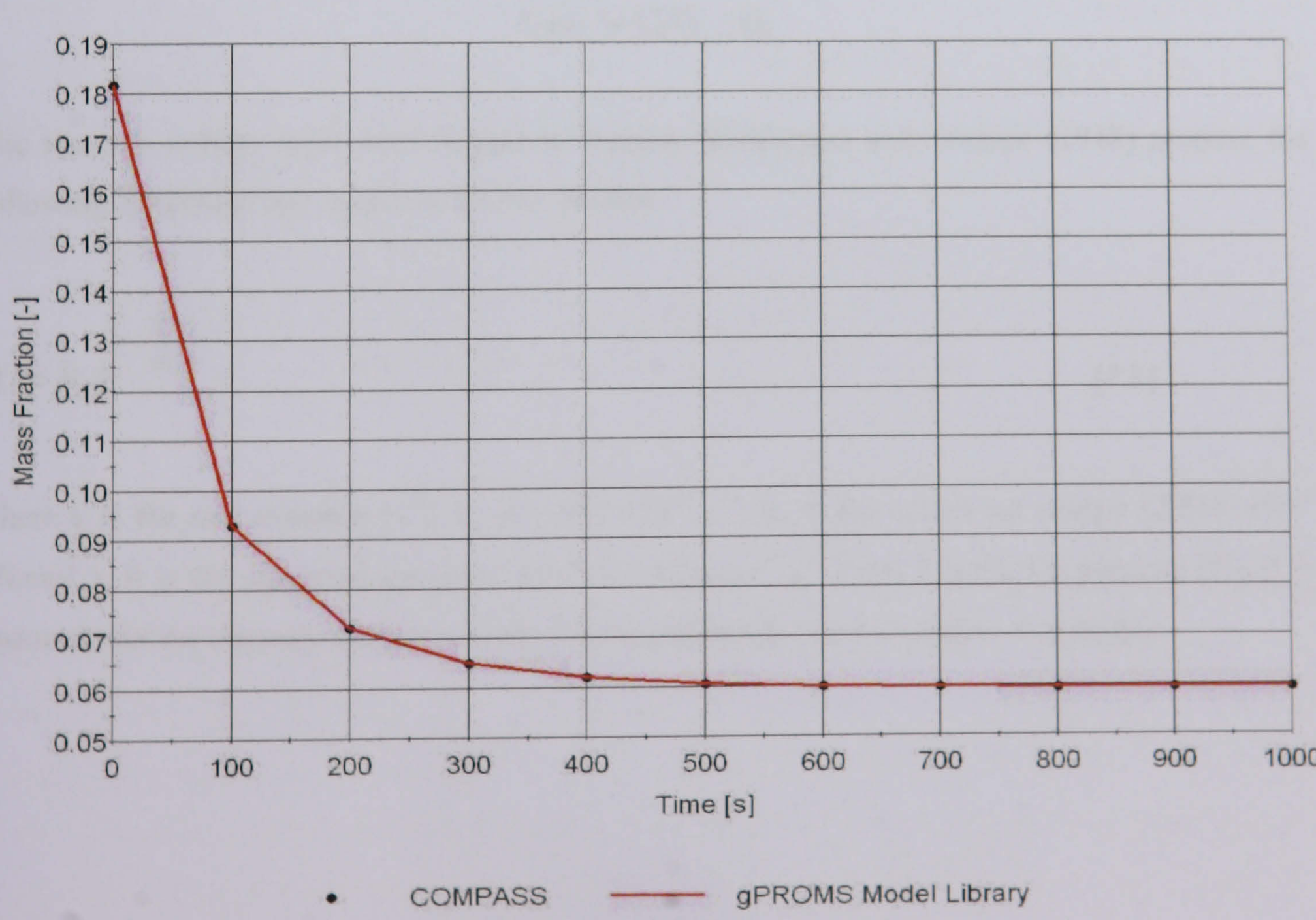


Figure 7.20: Mass Fraction of Pentane within the Gas Phase of the Flash Vessel

7.4 A Gas Phase Plug Flow Reactor

In this example the cracking of butene to form 2-butyne is considered. The reaction occurs within a plug flow reactor. The plug flow reactor system consists of a plug flow reactor (R101) connected to an inlet pipe (P101) and an outlet pipe P102. The plug flow reactor is a single 12inch tube with a length of 10m, an outside diameter of 0.32m and a wall thickness of 0.01m.

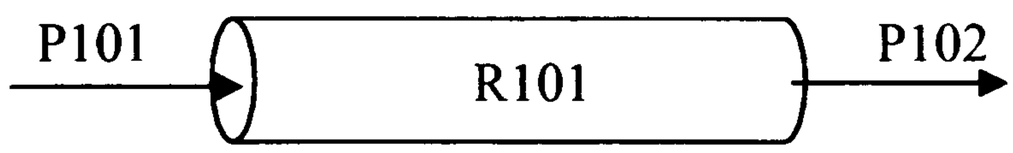


Figure 7.21: The Plug Flow Reactor System

Steam is added to the reactor feed to provide the necessary energy to support an endothermic reaction. The mass fraction of butene and steam within the feed are 0.1 and 0.9 respectively. The feed stream enters the reactor at a temperature of 820°K, a pressure of 1×10^5 Pa and a mass flowrate of 0.002 kg/s. The following gas phase reaction occurs within the plug flow reactor.

$$C_4H_8 \rightarrow C_4H_6 + H_2$$

The reaction is first order with respect to Butene. Beckberger and Watson (1948) propose the following Arrhenius rate equation for the reaction:

$$k = k_0 e^{\left(-\frac{E_a}{RT}\right)} \qquad [7.8]$$

where k is the rate constant (s^{-1}), k_o is 1.937×10^{15} s^{-1}, E_a is the activation energy (2.511×10^{15} $kJkmol^{-1}$), R is the universal gas constant (8.314 $kJ.kmol^{-1}.K^{-1}$) and T is the temperature (K). It is assumed that the pressure drop across the tube is negligible and the reactor is adiabatic.

7.4.1 gPROMS Model Development

A mathematical model was developed with the use of the *gPROMS* Model Library. The following model library components were used to model the reactor system.

Process Component	gPROMS model library component
Feed Material	Source
Inlet Pipe	Flow_Device
Plug Flow Reactor	Tubular_reactor
Outlet to Reactor	Sink

Table 7.6: The gPROMS Model Library Components Used to Model the Reactor System

7.4.2 *COMPASS* Model Development

To encapsulate the behaviour of the process with *COMPASS*, the process was represented by two *Zones* corresponding to the inlet pipe and the plug flow reactor. The outlet is connected directly to a reservoir. The process representation for the plug flow reactor system is shown in figure 7.22. The model class used to characterise each *Zone* is shown under the *Zone* name. To illustrate the versatility of *COMPASS*, two models have been generated. For the first simulation, the development of the velocity along the reactor axis is modelled. For the second simulation, it is assumed that the velocity is constant throughout the reactor.

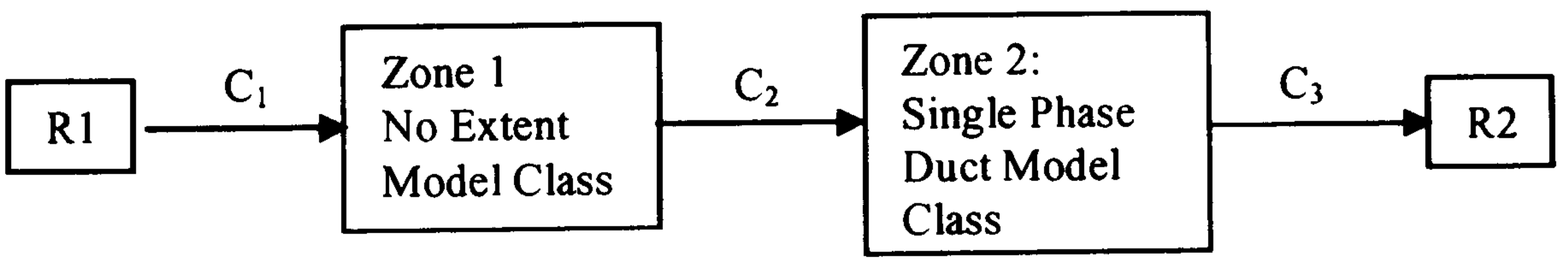


Figure 7.22: The Structural Representation of the Plug Flow Reactor

The formal description of the process, declared in terms of the process modelling language *COMPASS*, is shown in Appendix J. To illustrate the physical language for a single-phase duct *Zone*, the language for the reactor R101 is provided in figure 7.23. Similarities between the physical description for the gas CSTR and the physical description for the gas PFR can be observed with the small number of differences between the two descriptions arising from the need to establish the *Zone's* extent in terms of the coordinates of the *Zone* (lines 9 – 12), the need to specify the orientation of the *Zone* as being horizontal (line 7), the allocation of each *Port* to a

boundary (lines 14 – 15) and the addition description for the velocity of material within the Zone (line 22).

```
Line 1          VESSEL : Reactor1 : R101  ;
     2          DEFINE_ZONES
     3                  ZONE : Z2 ;
     4                  GEOMETRY : 1-D ;
     5                  COORDINATES : CARTESIAN ;
     6                  PHASE : Product_stream ;
     7                  ORIENTATION : 0 ;
     8                  DEFINE_BOUNDARIES
     9          BOUNDARY : Inlet  : AXIAL  : 0 ;
     10                 BOUNDARY : Outlet : AXIAL  : 10 ;
     11                 BOUNDARY : Wall   : RADIAL : 0.3 ;
     12         END #DEFINE_BOUNDARIES
     13         DEFINE_PORTS
     14                 PORT : P1 : Inlet ;
     15                 PORT : P2 : Outlet ;
     16         END #DEFINE_PORTS
     17         DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
     18                 ZONE_DYNAMICS : DYNAMIC ;
     19                 ZONE_COMPOSITION : VARIABLE : REACTION ;
     20                 ZONE_TEMPERATURE : VARIABLE :  NEGLECT_KE ;
     21                 ZONE_PRESSURE = 1.0E5 ;
     22                 ZONE_VELOCITY : BULK_FLOW ;
     23         END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
```

Figure 7.23: The Physical Description for the PFR Declared in Terms of the *COMPASS* Language

7.4.3 Model Assessment

Assessment of the *COMPASS* generated model was carried out by comparing the simulated results for the steady state temperature and mass fraction profiles along the reactor axis. The results of the comparison can be found in figures 7.24 and 7.25, where the axial development of for the mass fraction of 2-butyne and the temperature are plotted respectively. The results from the *COMPASS* generated model show excellent agreement with those from the *gPROMS* Model Library model, when the variation in velocity along the reactor is taken into account (red line and black dots respectively). The blue line shows the results of the *COMPASS* model with a uniform velocity. The results are less accurate than the others, but show the flexibility of the *COMPASS* language.

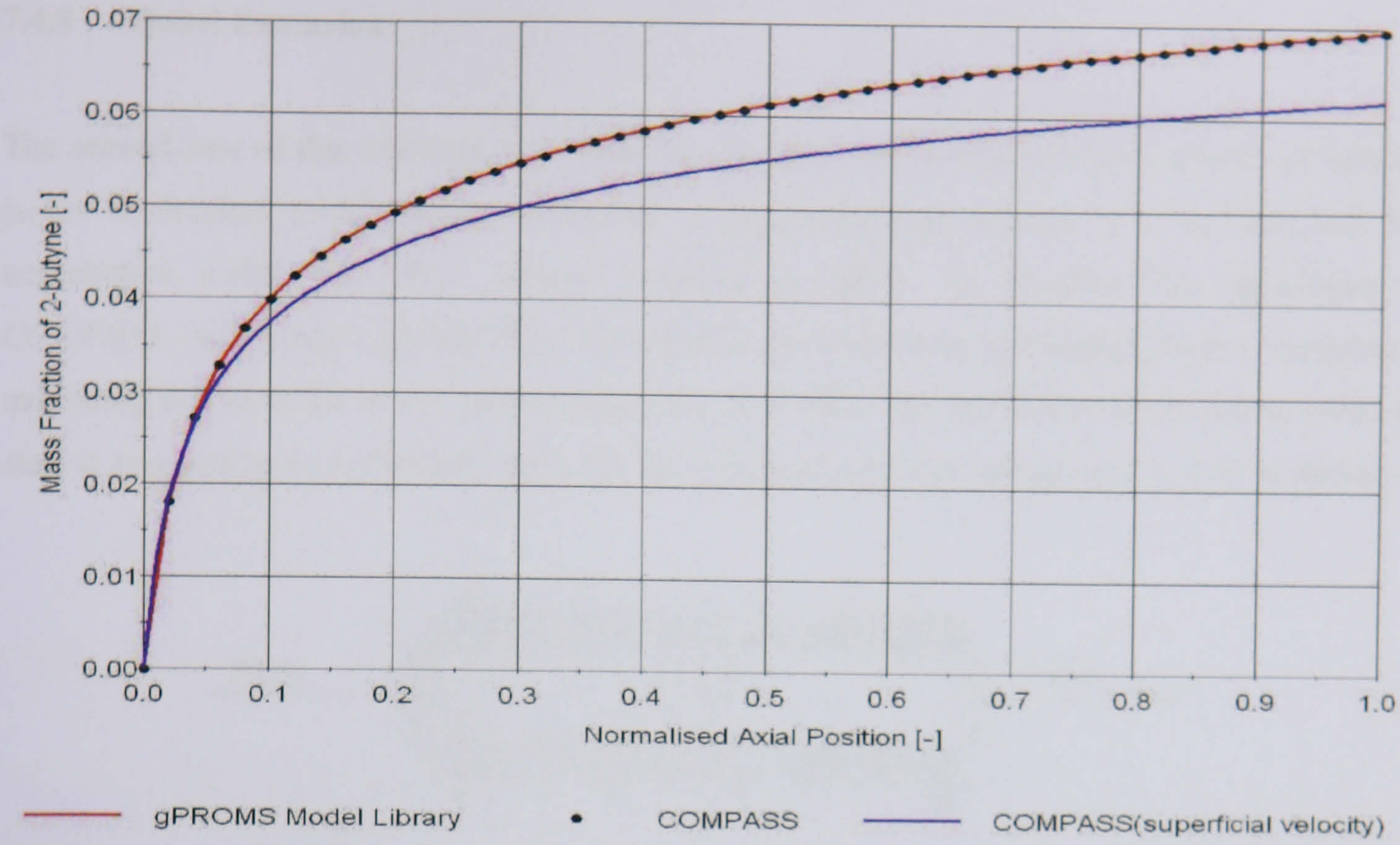


Figure 7.24: Axial Variation in the Steady State Mass Fraction of 2-Butyne

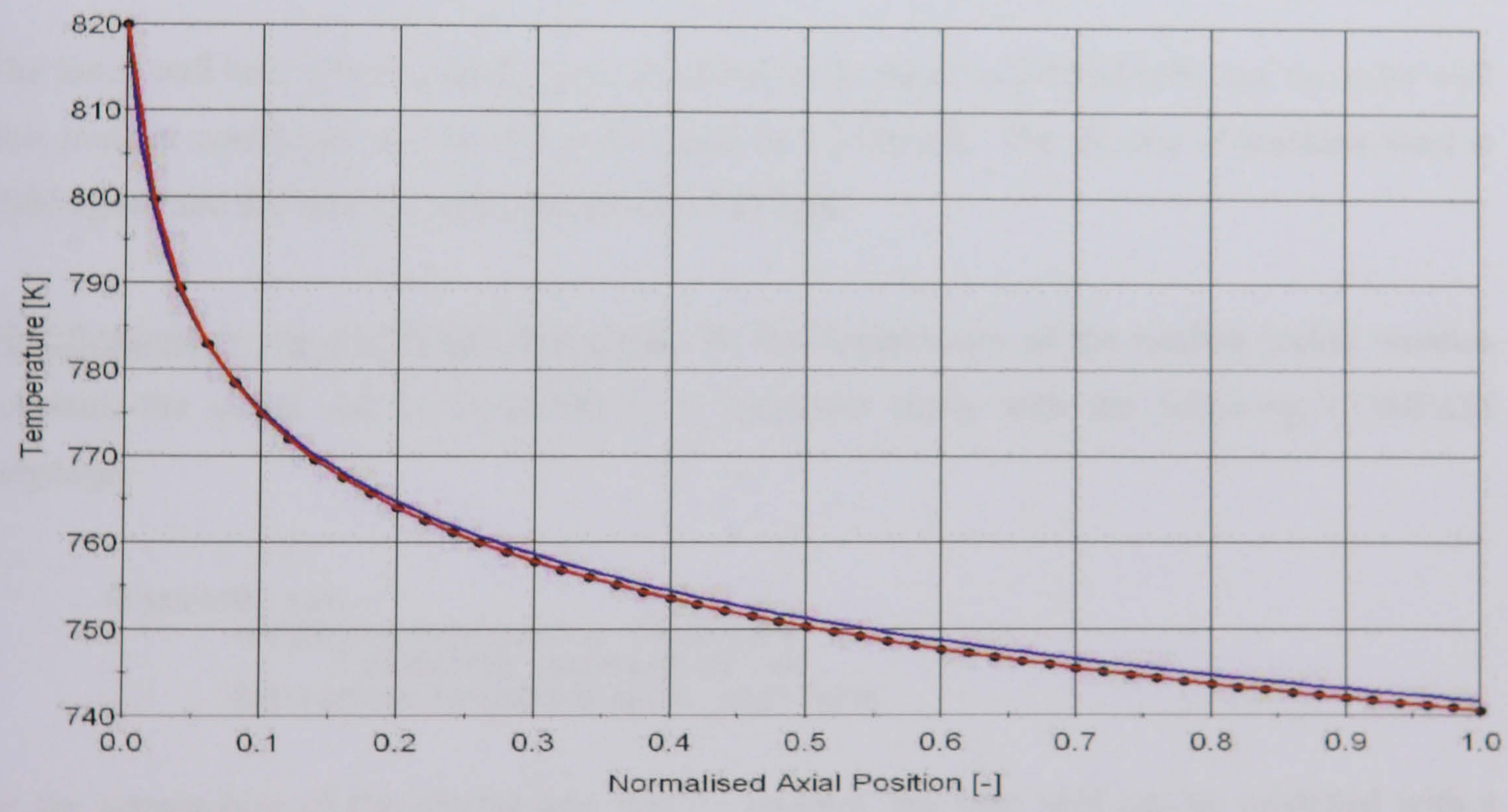


Figure 7.25: Axial Variation in the Steady State Temperature of the Reactor

7.4.4 Model Extension

The second part of this example considers the non-adiabatic operation of the reactor. A heating jacket is attached to the reactor providing an additional heat source. It is assumed that the temperature within the jacket remains constant at 800°K. To illustrate the capabilities of *COMPASS*, the thermal capacity of the tube wall is assumed to be sufficiently large to necessitate modelling the variation of the wall temperature with time. The thermal conductivity of stainless steel is assumed to be sufficiently great for the temperature of the reactor tube wall to be uniform.

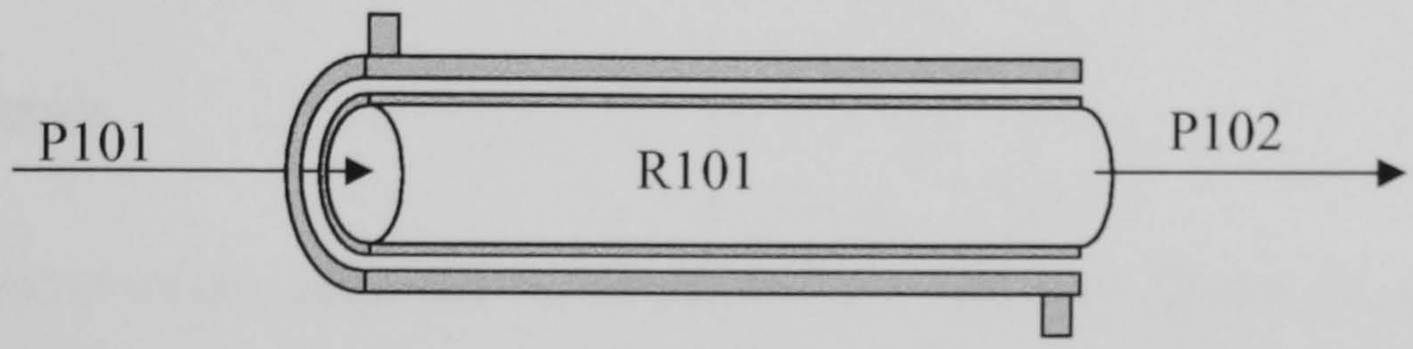


Figure 7.26: A Diagrammatical Representation of the Plug Flow Reactor System

The inner wall heat transfer coefficient is assumed to be equal to 0.02 kJ/m^{2}K and the outer wall heat transfer coefficient is assumed to be equal to 1.5 kJ/m^{2}K. The density of stainless steel is 7800 kg/m^3 and the specific heat capacity is 0.5 kJ/kgK.

Modifications to the *COMPASS* input file: As the temperature of the heating jacket remains constant, the jacket can be modelled by a Reservoir entity with the following *COMPASS* language.

```
RESERVOIR : R103  ;
        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                RESERVOIR_TEMPERATURE = 800 ;
        END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
```

As the temperature of the reactor tube wall is uniform, the tube wall can be modelled with a Lumped Zone. The physical description for the tube wall is as follows:

```
ZONE : Z3 ;
        GEOMETRY : LUMPED ;
        PHASE : StainlessSteel ;
        DEFINE_PORTS
                PORT : P1 ;
                PORT :  P2 ;
        END #DEFINE_PORT
        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
```

```
                ZONE_DYNAMICS : DYNAMIC ;
                ZONE_EXTENT = 0.097 ;
                ZONE_COMPOSITION : FIXED ;
                        StainlessSteel = 1 ;
                ZONE_TEMPERATURE : VARIABLE : NEGLECT_KINETIC_ENERGY :
                NEGLECT_POTENTIAL_ENERGY ;
                ZONE_PRESSURE = 1.013e+5 ;
END# PHYSIOCHEMICAL_BEHAVIOUR_SECTION
```

Two *Closed Energy Connections* are used to represent the transfer of energy between the inner tube and the wall and between the wall and the jacket respectively.

```
CONNECTION : C104 : CLOSED : ENERGY :  Z2.P3  ->  Z3.P1  ;
CONNECTION : C105 : CLOSED : ENERGY :  Z3.P2  ->  R103  ;
```

Model Assessment

The mathematical model generated by *COMPASS* was assessed against the *gPROMS* Model Library. The comparison of the axial temperature profiles generated by the *COMPASS* and *gPROMS* Model Library models is shown in figure 7.27. The difference between the two temperatures is shown in figure 7.28. It can be seen that the two models show excellent agreement with the relative temperature difference being less than 0.01%.

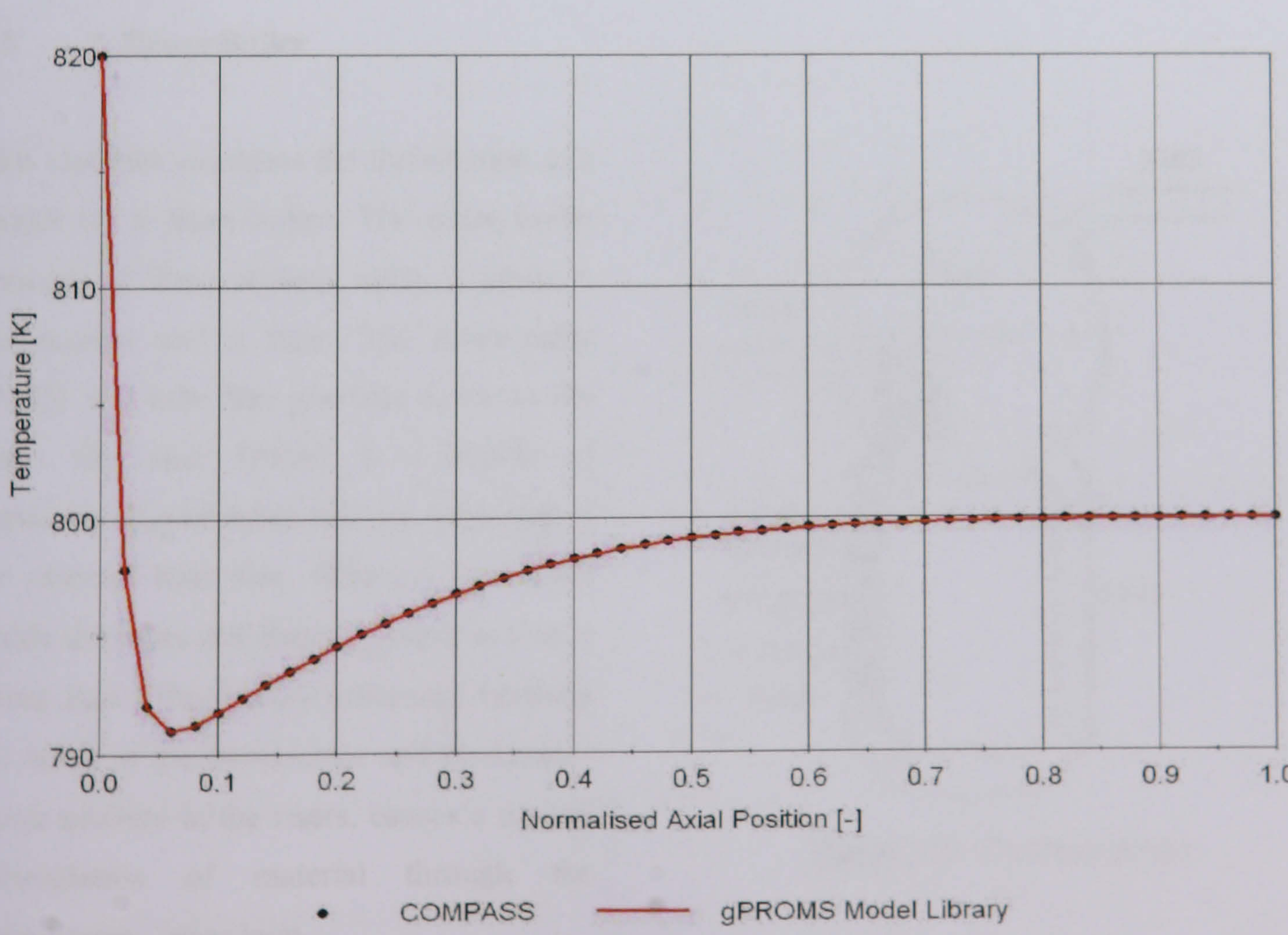


Figure 7.27: Steady State Temperature Profile along the Reactor Length

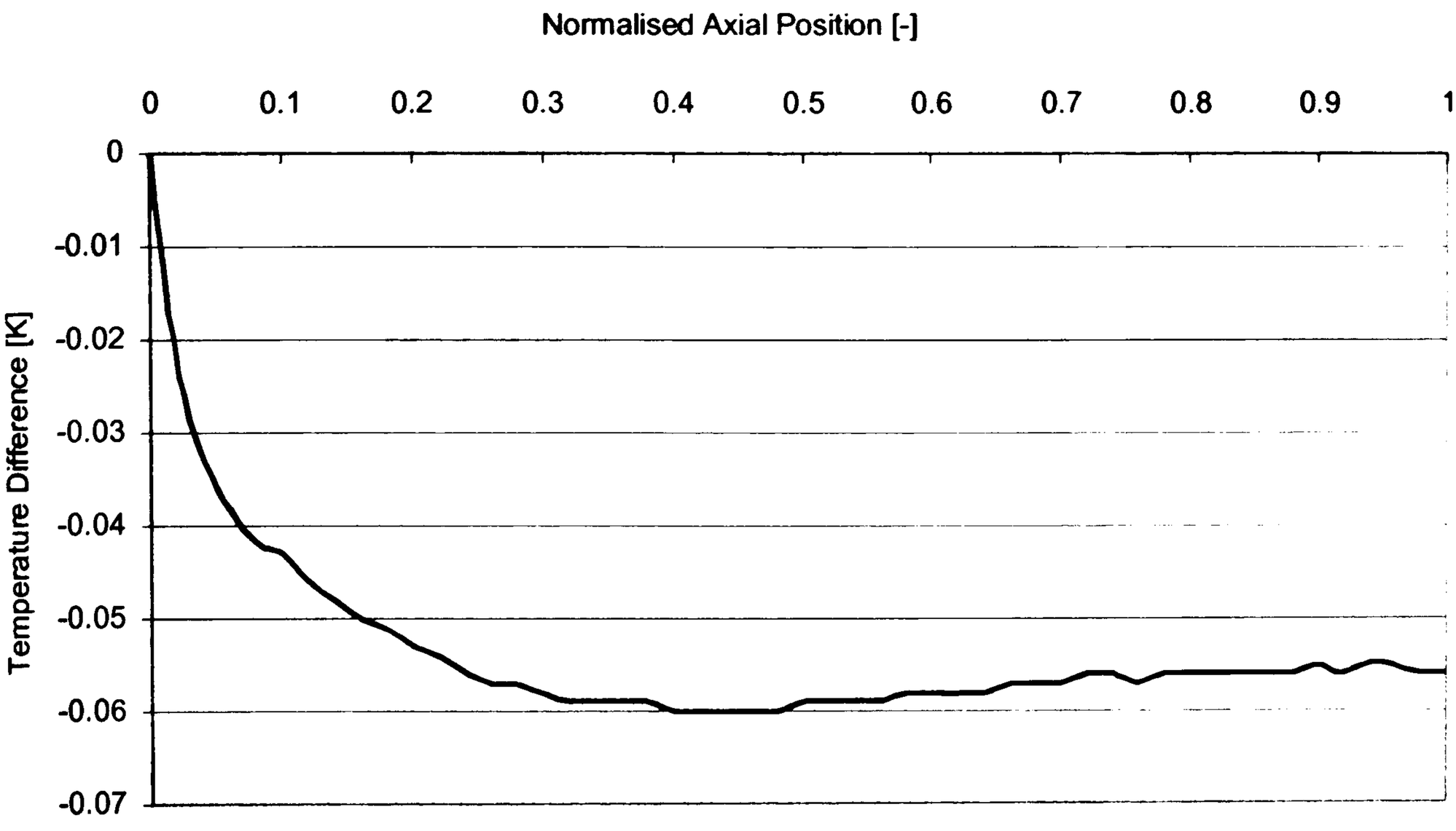


Figure 7.28: Difference in the Results for the Temperature for the *gPROMS* Model Library and *COMPASS* Generated Models

7.5 A Drum Boiler

This example considers the formulation of a model for a drum boiler. The drum boiler consists of three process units, a drum, a downcomer and a riser. The downcomer (P103) is a tube that provides water to the riser. The riser (P104) is a bundle of vertically aligned tubes that are subjected to an external heat flux. Steam is generated inside the tubes and flows upwards in a two-phase flow. The density difference between the water in the downcomer and the steam / water mixture in the risers, causes a natural recirculation of material through the downcomer – riser loop.

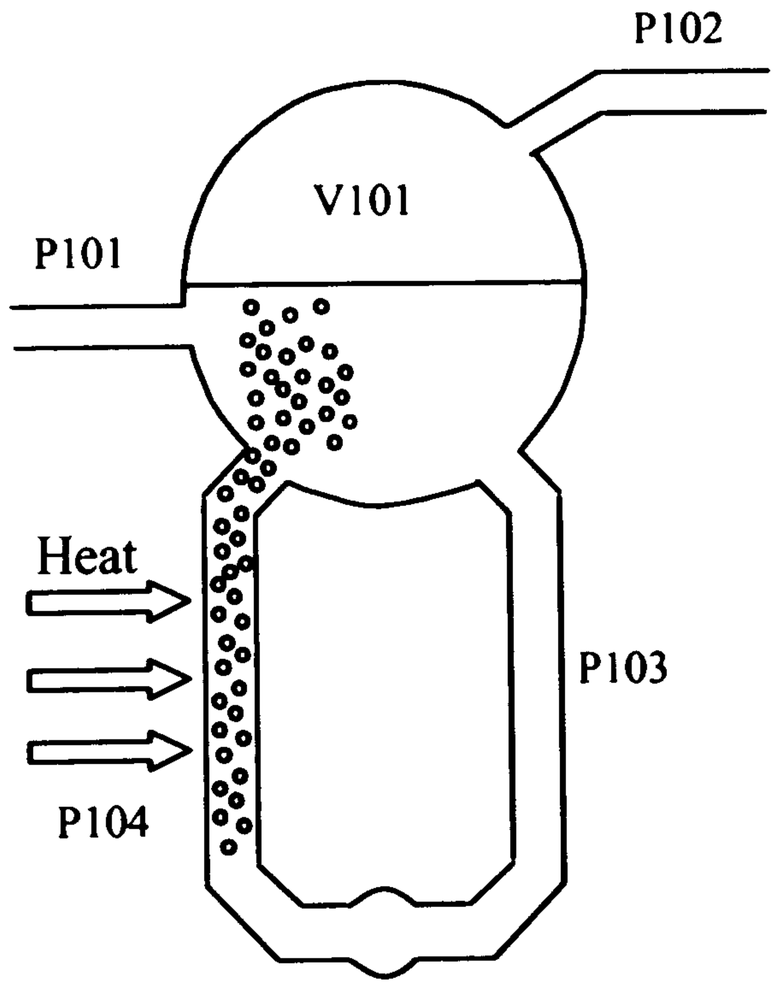


Figure 7.29: The Drum Boiler

The main process unit is the drum (V101), which is an accumulation tank located at the top of the boiler. The drum receives the vapour-liquid mixture from the riser tubes, allows the gas and liquid to separate and attenuates steam demand disturbances. Liquid water is fed to the drum through P101 and steam leaves the drum through P102. The temperature inside the drum remains at the boiling point of water.

The complexity of the drum boiler means that a wide variety of models can be generated of varying levels of complexity. The model that *COMPASS* generates for this example is based, as closely as possible, on the same set of assumptions that were used by Adam and Marcheti (1999). According to Adam and Marcheti (1999) the model is a "halfway-complexity model developed to provide an adequate description of the operation and to give information about many of the important physical aspects". By using similar assumptions to those used by Adam et al, it is possible to directly compare the two models and hence assess the model generating algorithms within *COMPASS*. The design parameters and operating conditions for the drum boiler are shown in Table 7.7.

7.5.1 Modelling Strategy and Model Assumptions

This section describes the modelling strategy and assumptions for each component of the drum boiler. In addition to declaring the physical behaviour of the drum boiler, additional equations must be added to the generated model to cover aspects of the operation that are beyond the current scope of *COMPASS*. The additional equations are listed for each component.

<u>The Drum:</u> The model of the drum is based on the following assumptions:

1. The drum remains at the boiling temperature of water throughout the simulation.
2. The phases separate as soon as they enter the drum. The volume of steam below the water level is therefore assumed to be negligible.
3. The drum is adiabatic.
4. The kinetic energy of material entering the drum can be neglected.
5. Both phases are at the same temperature and pressure.
6. The hydrostatic pressure within the drum is negligible.

Typical operating procedures for the drum boiler will be based on the control of the liquid level within the drum. Adam and Marcheti (1999) propose the following equation to relate the liquid

level to the volume of liquid (V), the minimum liquid level (l_{min}) and the corresponding liquid volume (V_{min}).

$$l = l_{min} + \frac{V - V_{min}}{\tilde{A}_{sep}} \qquad [7.9]$$

Where $\tilde{A}$ is the mean total area, which is related to the length of the drum (L), the drum diameter (D) and the minimum mixture height through Eq. 7.10.

$$\tilde{A} = L\sqrt{D^2 - (d - l_{min})^2} \qquad [7.10]$$

The Downcomer: As the cross sectional area of the downcomer tubes is considerably greater that the cross sectional area of the riser tubes, the wall friction of the downcomer is assumed to be negligible. The dynamics of the downcomer are assumed to be at steady state.

Parameter	Value
Riser Tube Bundle	
Length,	25.0m
Diameter,	0.0508m
Number of tubes,	440
Wall heat flux	2.61×10^4 Wm^{-2}
Separation drum	
Length	8.0m
Diameter	1.8m
Initial height of mixed phase	0.7m
Initial Pressure	8.36×10^6 Pa
Initial Temperature	571.3K
Feed water stream	
Flow	31.41 kg s^{-1}
Pressure	8.4×10^6 Pa
Temperature	571.3 K
Steam stream	
Flow	31.41 kg s^{-1}

Table 7.7: Design Parameters and Operating Conditions for the Drum Boiler

Key:

The Riser Tubes: In keeping with the model assumptions used by Adam and Marcheti (1999), the dynamics of the riser are assumed to be sufficiently fast to allow the steady-state assumption to be adopted. The water entering the riser tubes is assumed to be at the boiling temperature so that the water begins to boil as soon as it enters the riser tube. The temperature at any point within the boiling tube is predicted by a boiling isotherm. The flow of steam and water through the riser is modelled by assuming that both phases travel at the same velocity.

The wall shear stress per unit volume ($Pa.m^{-3}$), used in the momentum balance for the riser, is calculated in an identical way to that of Adam and Marcheti (1999). Eq. 7.11 is used to relate the shear stress to a friction factor (f), which is calculated with the use of the homogeneous model through Eq. 7.12.

$$\tau_w = \left(\frac{2f}{D_t}\right)\left(\rho_m v_m{}^2\right) \qquad [7.11]$$

$$f = 0.0014 + 0.125\,\mathrm{Re}^{-0.32} \qquad [7.12]$$

where Re is the Reynolds number $= \dfrac{\rho_m v_m D_t}{\mu_m}$, v_m is the mixture velocity in the tubes, D_t is the tube diameter and ρ_m and μ_m are the density and viscosity of the two-phase mixture respectively.

Control Loop: A control loop is used to control the liquid level within the drum. The control loop attempts to control the liquid level at a set point of 0.7m by varying the mass flowrate of water through pipe P101. The controller is Proportional, Integral. Equation 7.13 is used to express the overall closed loop control dynamics.

$$O_C = B + K_c\varepsilon + \frac{K_c}{\tau}\int_0^t \varepsilon dt \qquad [7.13]$$

where O_c is the manipulated variable, B is the controller bias, K_c is the proportional gain, τ is the overall integral time constant and ε is the difference between the set point value and the actual value of the controlled variable. The values used for the overall proportional gain and the overall time constant of the control loop are given in table 7.8.

Parameter	Level Control
Controller Bias (B)	34.1
Proportional Gain (K_c)	10
Integral time constant (τ)	200

Table 7.8: The Controller Parameters for the Drum Boiler

7.5.2 *COMPASS* Model Development

The structural representation for the drum boiler is shown in figure 7.30. To model the behaviour of the process with *COMPASS*, the process was represented by five *Zones* corresponding to the feed water pipe (P101), the steam outlet pipe (P102), the downcomer (P103), the risers (P104) and the drum (V101). Reservoirs were used to represent the inlet feed conditions (R1) and the downstream (R2). The network of *Zones* and *Reservoirs* was created with the use of eight *Connections*. The component that each Zone represents, and the model class used to characterise each *Zone* are shown under the *Zone* name.

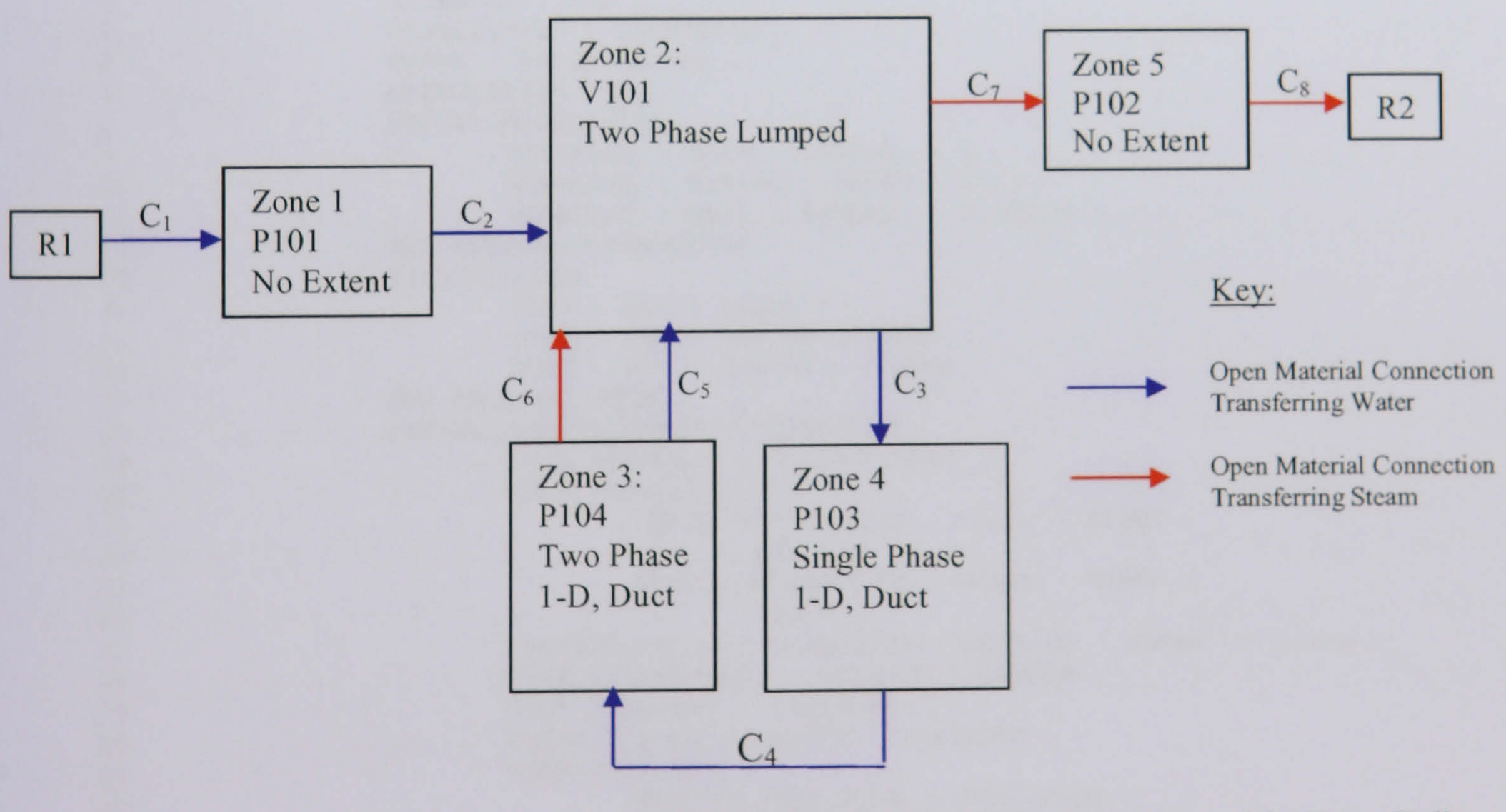


Figure 7.30: The Structural Representation of the Drum Boiler

The *COMPASS* language declaration for the drum boiler is given in Appendix J. To illustrate the physical language for a two-phase duct *Zone*, the language for a riser tube (P104) is provided in figure 7.31. The physical description commences by establishing the coordinates of the *Zone* (lines 4 and 5), the number of phases within the *Zone* (line 6) and the vertical orientation of the *Zone* (line 7). As with the single-phase duct *Zone*, the extent of the *Zone* is established by stating the dimensions of the riser tube (lines 9 – 11). The *Ports* are then defined and assigned to the appropriate boundary to which they lie on. As both the water and steam phases comprise of a single component, the composition of each phase has been declared. The boiling of the water within the tube is modelled through an internal connection between the two phases with the 'BOILING' library model used to determine the rate of mass transfer (line 25). As the water remains at its boiling point, a boiling isotherm is used to calculate the temperature profile along the tube (line 26). So as to be consistent with the assumptions of Adam et al. (1999), both phases are assumed to travel at the same velocity (line 30). The physical description for the velocity of the two phases can be found on lines 29 to 33.

```
Line 1          VESSEL : Boiling_Tube : P104 ;
     2          DEFINE_ZONES
     3                  ZONE : Z3 ;
     4                  GEOMETRY : 1-D ;
     5                  COORDINATES : CARTESIAN ;
     6                  PHASE : Water, Steam ;
     7                  ORIENTATION : 90 ;
     8                  DEFINE_BOUNDARIES
     9                          BOUNDARY : Inlet : AXIAL  : 0  ;
    10                          BOUNDARY : Outlet : AXIAL : 25 ;
    11                          BOUNDARY : Wall : RADIAL  : 0.02504 ;
    12                  END #DEFINE_BOUNDARIES
    13                  DEFINE_PORTS
    14                          PORT : PD1 : Inlet ;
    15                          PORT : PD2 : Outlet : Water ;
    16                          PORT : PD3 : Outlet : Steam ;
    17                  END #DEFINE_PORTS
    18                  DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
    19                          ZONE_DYNAMICS : STEADY_STATE ;
    20                          ZONE_COMPOSITION ;
    21                                  PHASE_COMPOSITION : Water : FIXED ;
    22                                          H2O = 1 ;
    23                                  PHASE_COMPOSITION : Steam : FIXED ;
    24                                          H2O = 1 ;
    25                          INTERFACIAL_MASS_TRANSFER : BOILING : Water -> Steam ;
    26                          ZONE_TEMPERATURE : BOILING_ISOTHERM ;
    27                          ZONE_PRESSURE : VARIABLE ;
    28                          PHASE_VOLUME_FRACTION : VARIABLE ;
    29                          ZONE_VELOCITY ;
    30                                  MIXTURE_BULK_FLOW : BULK_FLOW ;
    31                                  INTERNAL_RELATIVE_MOTION : NONE ;
    32                                  CONTINUOUS_PHASE : Water  ;
    33                                  DISPERSED_PHASE : Steam ;
    34                  END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
    35          END #zone section
```

Figure 7.31: The Physical Description for the Riser Tube

Before the *COMPASS* generated model could be solved, the following modifications were made to ensure compatibility with the model published by Adam et al. (1999):

1. The equations of state were rewritten in a format compatible with Multiflash.
2. Equations [7.9] and [7.10] were written to enable the liquid level to be determined.
3. Equations [7.11] and [7.12] were used to determine the wall shear stress.
4. The drum level control loop was modelled with Eq. [7.13].
5. The mass flowrates through *Connections* C5 and C6 were multiplied by the number of riser tubes.

7.5.3 Model Assessment

The *COMPASS* generated mathematical model was assessed against the model published by Adam and Marcheti (1999). The boiler was simulated for 100s with the operating parameters specified in table 7.8. After 100s the steam flowrate through pipe P102 was increased by 5% and the simulation was run for a further 1200s. The results are shown in figures 7.32, 7.33 and 7.34. In figure 7.32 the variation in the drum pressure is plotted for the whole simulation time. It can be seen that there is a good agreement between the *COMPASS* generated model and the Adam and Marchetti model. The swelling effect, caused by the steam bubbles below the drum liquid level, was considered within the Adam and Marchetti model, but not the *COMPASS* generated model; this may account for this discrepancy. In figure 7.33, the mass flowrate through the downcomer is compared for the two models. The mass flowrate through the downcomer is approximated within the Adam and Marchetti model, so it is likely that the simulation results produced from the solution of the *COMPASS* generated model are more accurate than those from the Adam and Marchetti model. The final plot shows the volume fraction distribution through the riser tube at three different times in the simulation. It can be seen that the *COMPASS* generated model and the Adam and Marchetti model show excellent agreement. The generated output text file is given in Appendix K. Both the text and the gPROMS output files are can be found in electronic format on the CD attached to the back of this thesis.

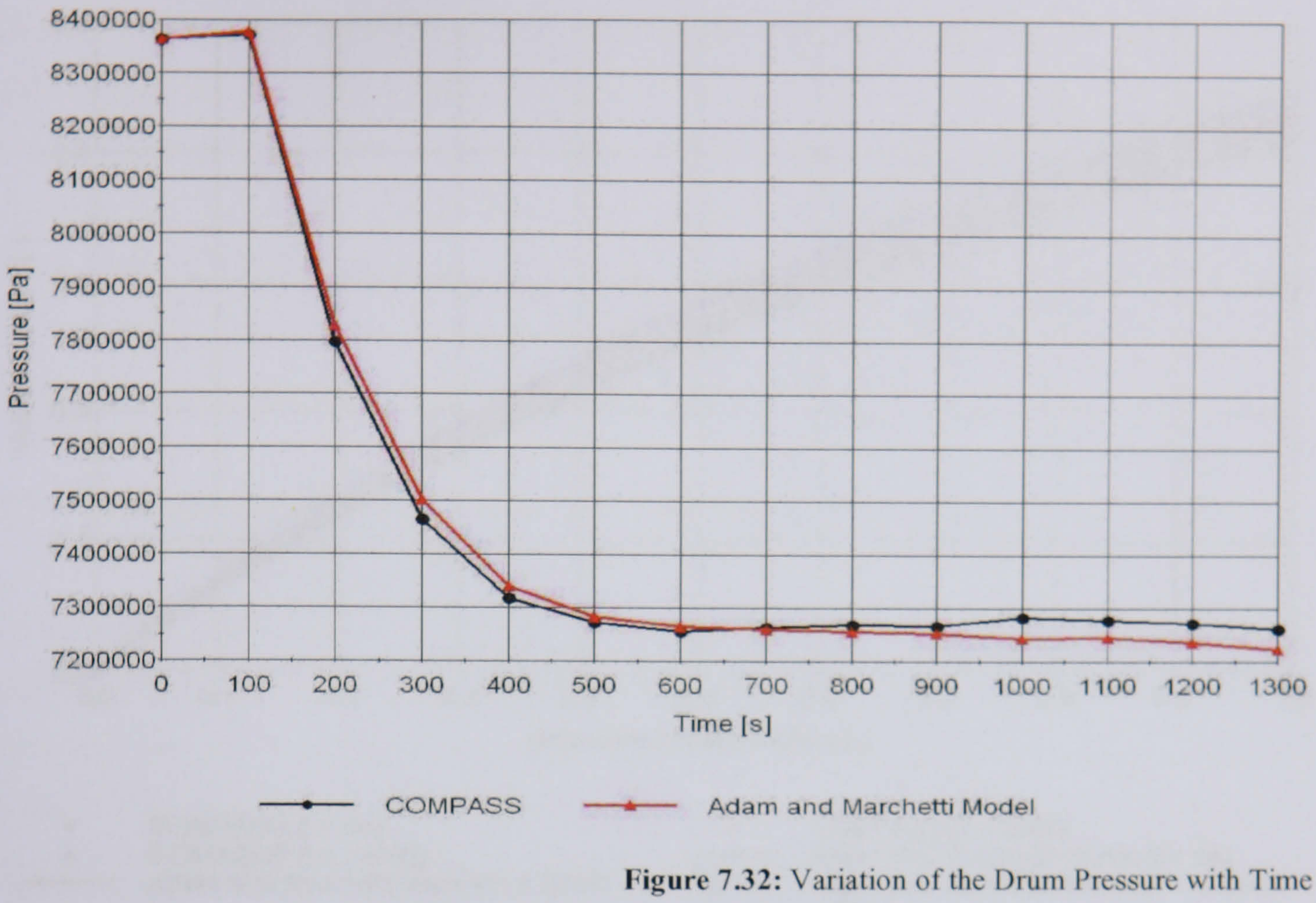


Figure 7.32: Variation of the Drum Pressure with Time

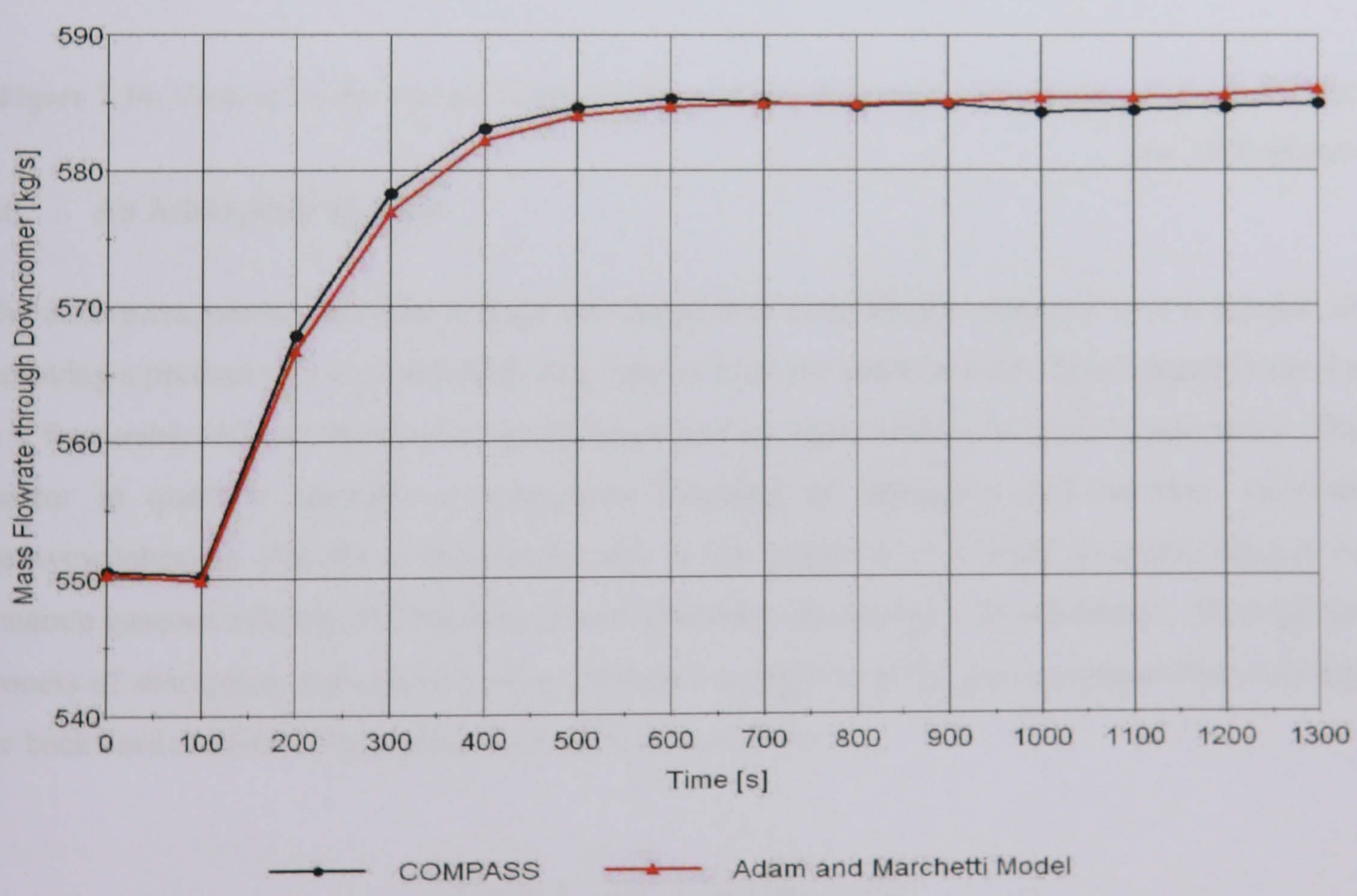


Figure 7.33: Variation in the Mass Flowrate through the Downcomer with Time

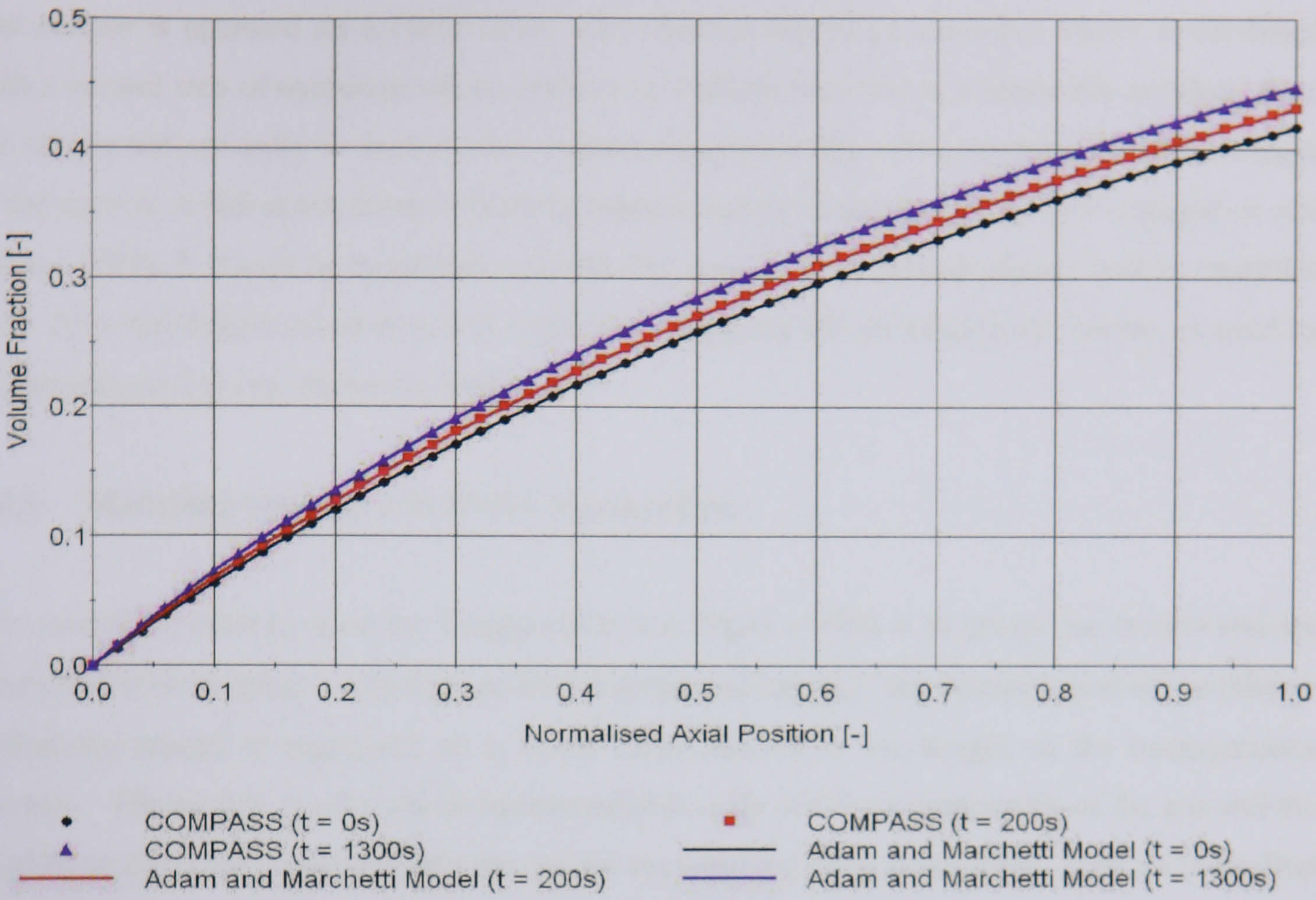


Figure 7.34: Variation in the Volume Fraction of Steam along the Length of the Boiling Tube after 0, 200 and 1300 seconds

7.6 An Adsorptive Reactor

The adsorptive reactor provides a huge advantage over conventional reactors as it is capable of removing a product or a reaction-inhibiting species from the reaction zone. This frequently results in a favourable shift in the reaction equilibrium and an improvement in product selectivity. The reactor in question provides a synergistic coupling of separation and reaction. Gaseous methycyclohexane (MCH) is dehydrogenated in the presence of a solid Pt-Al_2O_3 catalyst to generate gaseous toluene. The toluene is then adsorbed into a zeolite 5A adsorbent. Through the process of adsorption, the concentration of toluene is reduced in the gaseous phase thus reducing the backward reaction. The overall reaction is shown below:

$$C_6H_{11}(CH_3) \xleftrightarrow{Pt-Al_2O_3} C_6H_5(CH_3) + 3H_2$$

$$C_6H_5(CH_3) \xrightarrow{\text{Zeolite 5A}} C_6H_5(CH_3)ads$$

The reactor is operated on a cyclic basis. The reaction procedure described above is combined with a second step of operation whose purpose is to firstly remove the toluene (the product) from the zeolite and secondly to increase the temperature prior to the next reactant feed being charged to the reactor. A full explanation of both the reactor and the kinetics is given by Yongsunthon and Alpay (1999). It should be noted that only the first step of the operating process will be modelled here. Here the design parameters and operating conditions for the adsorptive reactor, as used by Yongsunthon et al., are shown in Table 7.9.

7.6.1 Modelling Strategy and Model Assumptions

The modelling strategy used by Yongsunthon and Alpay (1999) is to group the Zeolite and the catalyst as a homogeneous solid phase with volume fraction (ε_s). The concentration of the toluene within the zeolite is expressed on a molar basis relative to the weight of the homogeneous mixture. The reactor is assumed to operate adiabatically and the temperatures of the gas and the catalyst at any axial position within the reactor are assumed to be same. There is no pressure drop across the reactor and the kinetic energy of the gas phase is assumed to be negligible.

The rate of reaction is modelled as a source term within the gas phase. An effectiveness factor (η) is used to account for the difference between the overall rate of reaction and the surface rate of reaction. The overall rate of reaction (r_r) is related to the surface rate of reaction (r_s) by:

$$r_r = \eta r_s \qquad [7.14]$$

The surface reaction rate is represented by power-law reaction kinetics.

$$r_s = k_f\left(c_A - \frac{c_B c_C^{\,3}}{K_e}\right) \qquad [7.15]$$

Where r_s is the surface rate of reaction, k_f is the pseudo-first-order rate constant, c is the molar gas concentration and K_e is the reaction equilibrium constant. Subscripts A, B and C refer to methycylohexane, toluene and hydrogen respectively. The pseudo-first-order rate constant is described by the following Arrhenius-type equation.

$$k_f = a_1 e^{\left(\frac{-a_2}{RT}\right)} \qquad [7.16]$$

where a_1 and a_2 are constants. The reaction equilibrium constant is given by:

$$K_e = \frac{b_1}{T^3} e^{\left(\frac{-b_2}{RT}\right)} \qquad [7.17]$$

where b_1 and b_2 are constants and T is the reactor temperature. The actual reaction rate is related to the surface rate through the effectiveness factor (Eq. 7.14).

The rate of adsorption of toluene is described by the linear driving force between the equilibrium loading of the adsorbent and the solid-phase concentration of the adsorbent.

$$\Gamma = k\varepsilon_S \rho_S R_{m,B}\left(c_B^* - c_B\right) \qquad [7.18]$$

Where Γ is the interfacial rate of mass transfer, c_B is the molar concentration, $R_{m,B}$ is the relative molecular mass of toluene, superscript * refers to the molar concentration of the solid phase in equilibrium with the gas phase. Assuming a linear isotherm, the equilibrium loading can be related to the partial pressure (p_B) through Eq. 7.19.

$$c_B^* = m_B p_B \qquad [7.19]$$

where m_B is the adsorbent isotherm gradient. The temperature dependency of the adsorbent isotherm gradient is described by the van't Hoff equation.

$$\frac{m_B}{m_{o,B}} = e^{\left\{-\frac{\Delta H_a}{R}\left(\frac{1}{T}-\frac{1}{T_o}\right)\right\}} \qquad [7.20]$$

where $m_{o,B}$ is the adsorbent isotherm gradient at T_o (573K) and ΔH_a is the heat of adsorption. Eq. 7.21 relates the molar concentration of toluene to the mass fraction of toluene within the solid phase.

Parameter	Value
Feed Conditions	
Velocity	0.2 ms^{-1}
Temperature	623K
Pressure	2×10^5 Pa
Reactor	
Length	1 m
Diameter	0.3m
Pressure	2×10^5 Pa
Fluid volume fraction	0.7
Effective dispersion coefficient	0.001 m^2/s
Bed thermal conductivity	0.1 W/mK
Reactor Bed Properties	
Density	800 kg/m^3
Specific heat capacity	920 J/kgK
Catalyst: Pt/Al_2O_3	
Heat of reaction	2.12×10^5 J/mol
Overall effective factor	1
a_1	1.93×10^9 s^{-1}
a_2	1.08×10^5 kJ/kmol
b_1	5.94×10^{32} $(mol.K/m^3)^3$
b_2	2.12×10^5 (kJ/kmol)
Absorbent: zeolite 5A	
ΔH_a	80×10^3 J/mol
$m_{o,B}$	1.65×10^{-5} $molPa^{-1}kg^{-1}$
k_B	1 s^{-1}

Table 7.9: Design Parameters and Operating Conditions for the Adsorptive Reactor

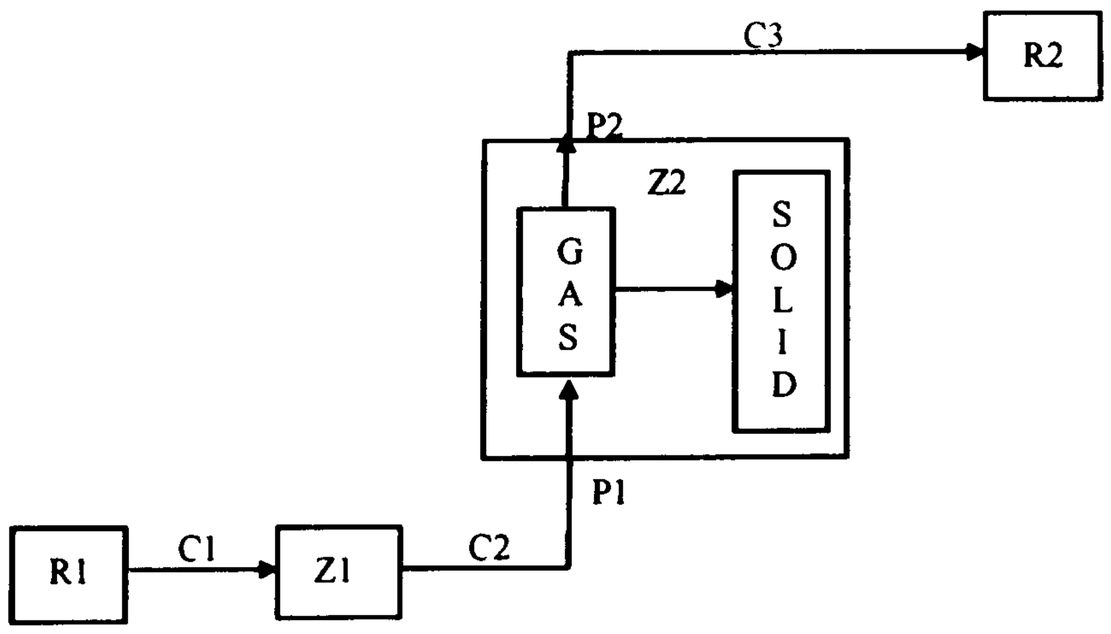


Figure 7.35: The Structural Representation of the Adsorptive Reactor

$$c_{S,B} = \frac{\rho_s x_B}{R_m} \qquad [7.21]$$

7.6.2 *COMPASS* Model Development

The process representation for the adsorptive reactor is given in figure 7.35. To model the behaviour of the process with *COMPASS*, the process was represented by two *Zones* corresponding to the feed pipe and the reactor. Reservoirs were used to represent the inlet feed conditions (R1) and the downstream conditions (R2). The network of *Zones* and *Reservoirs* was created with the use of three *Connections*.

The complete *COMPASS* language declaration for the adsorptive reactor is given in Appendix J. An excerpt of the physical description for Zone Z2 is given in figure 7.36. Based on an identical template to that which was used to model the boiling tube in section 7.5.2, the physical description for the reactor (figure 7.36) demonstrates that the same generic template can be used to model multiphase systems of varying physical behaviour.

The main differences between the physical description for the boiling tube and the physical description for the adsorptive reactor can be found in the 'DEFINE_PHYSICOCHEMICAL_BEHAVIOUR' section. For the adsorptive reactor the composition of both phases varies both with time and over the axial domain. The reaction is declared to occur within the 'Product' (gas) phase (line 20) whilst the adsorption of toluene form the 'Product' phase to the 'Solid' phase is described by reference to the 'ABSORPTION' library model on lines 22 – 23. With the temperature of the two-phases being homogeneous at any axial position within the reactor, the physical description for the temperature of the reactor is established for the two-phase mixture as a whole rather than for each phase separately (line 24). The assumption that there is no pressure drop across the reactor allows the pressure to be declared as a constant value (line 25). Similarly, the volume fraction of each phase also remains constant and can be declared by the user (lines 26 – 28). The final part of the physical description specifies that the velocity of the 'Product' phase is uniform along the reactor length and that the 'Solid' phase is stationary.

```
Line 1          VESSEL : Reactor1 : P101  ;
     2          DEFINE_ZONES
     3          ZONE : Z2 ;
     4                  GEOMETRY : 1-D ;
     5                  COORDINATES : CARTESIAN ;
     6                  PHASE : Product, Solid ;
     7                  ORIENTATION : 0 ;
     8                  DEFINE_BOUNDARIES
     9                  BOUNDARY : Inlet : AXIAL : 0  ;
     10                         BOUNDARY : Outlet : AXIAL : 1 ;
     11                         BOUNDARY : Wall : RADIAL : 0.15 ;
     12                 END #DEFINE BOUNDARY
     13                 DEFINE_PORTS
     14                         PORT : P1 : Inlet : Product ;
     15                         PORT : P2 : Outlet ;
     16                 END #DEFINE_BOUNDARY
     17                 DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
     18                         ZONE_DYNAMICS : DYNAMIC ;
     19                         ZONE_COMPOSITION  ;
     20                                 PHASE_COMPOSITION : Product : VARIABLE : REACTION ;
     21                                 PHASE_COMPOSITION : Solid : VARIABLE  ;
     22                                 INTERFACIAL_MASS_TRANSFER : ABSORPTION : Product ->
     23                                 Solid : Toluene  ;
     24                         ZONE_TEMPERATURE : VARIABLE : DIFFUSION : NEGLECT_KE ;
     25                         ZONE_PRESSURE = 2.0E+5 ;
     26                         PHASE_VOLUME_FRACTION : FIXED ;
     27                                 Product = 0.3;
     28                                 Solid = 0.7 ;
     29                         ZONE_VELOCITY ;
     30                                 MIXTURE_BULK_FLOW : BULK_FLOW ;
     31                                 INTERNAL_RELATIVE_MOTION : RELATIVE_PHASE_MOTION ;
     32                                 CONTINUOUS_PHASE : Product : SUPERFICIAL_VELOCITY ;
     33                                 DISPERSED_PHASE : Solid : STATIONARY ;
     34                 END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
     35         END #zone section
```

Figure 7.36: The Physical Description for the Adsorptive Reactor

Before the *COMPASS* generated model could be solved, the following modifications were made to ensure compatibility with the model published by Yongsunthon and Alpay (1999):

1. Equations 7.14 - 7.17 were added to allow the reaction to be modelled.
2. Equations 7.18 – 7.21 were added to enable the adsorption of toluene to be modelled.
3. No equations of state could be found for the zeolite and catalyst, so it was necessary to use an overall specific heat capacity for the solid phase (zeolite and catalyst). An overall specific heat capacity was used for the reactor bed, so specific terms for the heat of reaction and the heat of adsorption had to be added by hand to the homogeneous energy balance[11].

[11] It is normally not necessary to incorporate these terms as the overall enthalpy incorporates the energy of formation.

7.6.3 Model Assessment

Assessment of the *COMPASS* generated model was carried out by comparing the results for the mass concentration of toluene within the gas phase and the molar concentration of toluene within the solid phase. The results of the comparison are shown in figures 7.37 and 7.38, where the gaseous mass concentration of toluene (figure 7.37) and the solid molar concentration of toluene are plotted (figures 7.38) at several points in time. In all figures a solid line represents the Yongsunthon and Alpay (1999) model and a dotted line represents the results obtained from the simulation of the *COMPASS* model. The results from the *COMPASS* generated model show very good agreement with those generated from the solution of Yongsunthon et al.'s model.

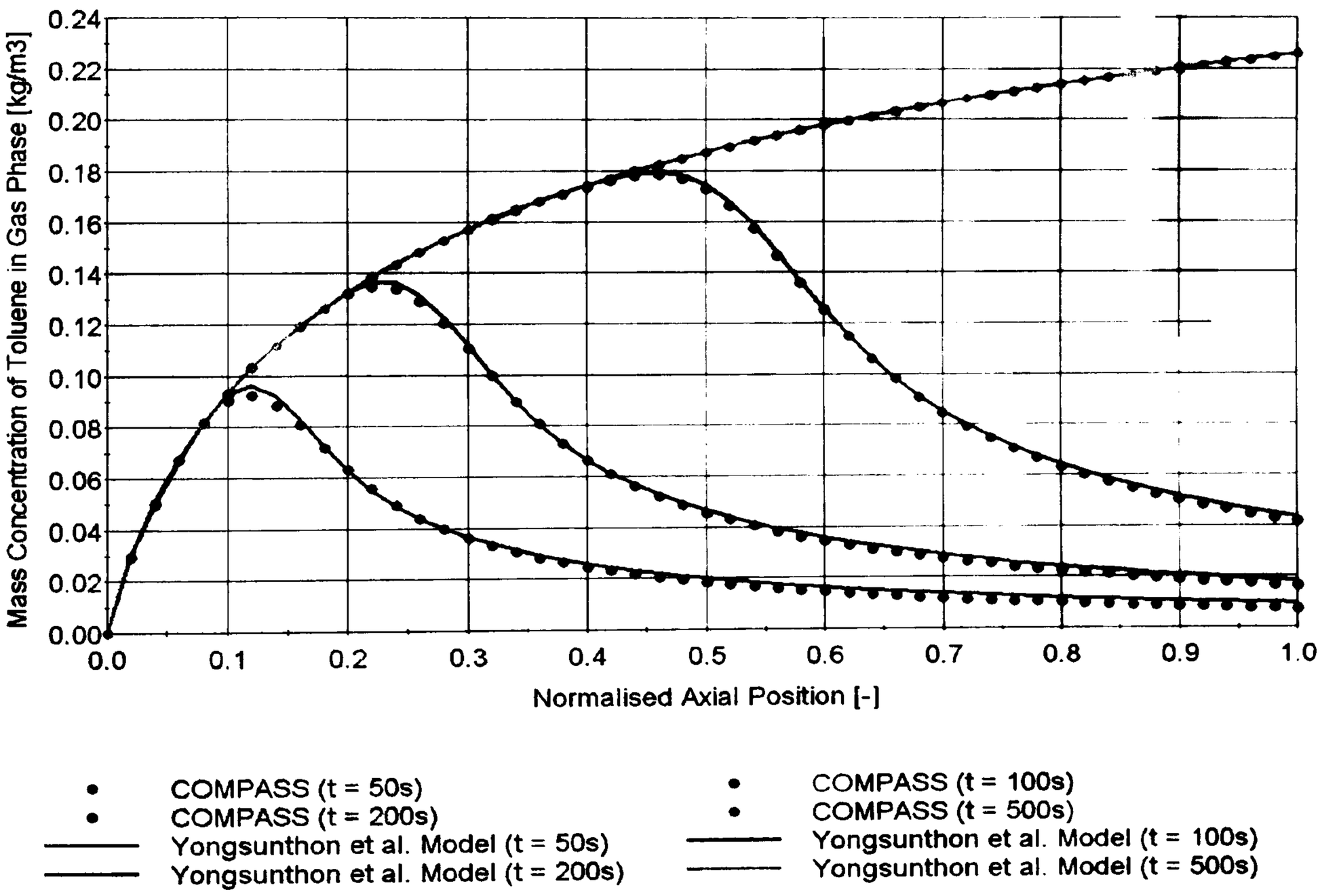


Figure 7.37: Axial Variation in the Gaseous Mass Concentration of Toluene at Various Time Steps

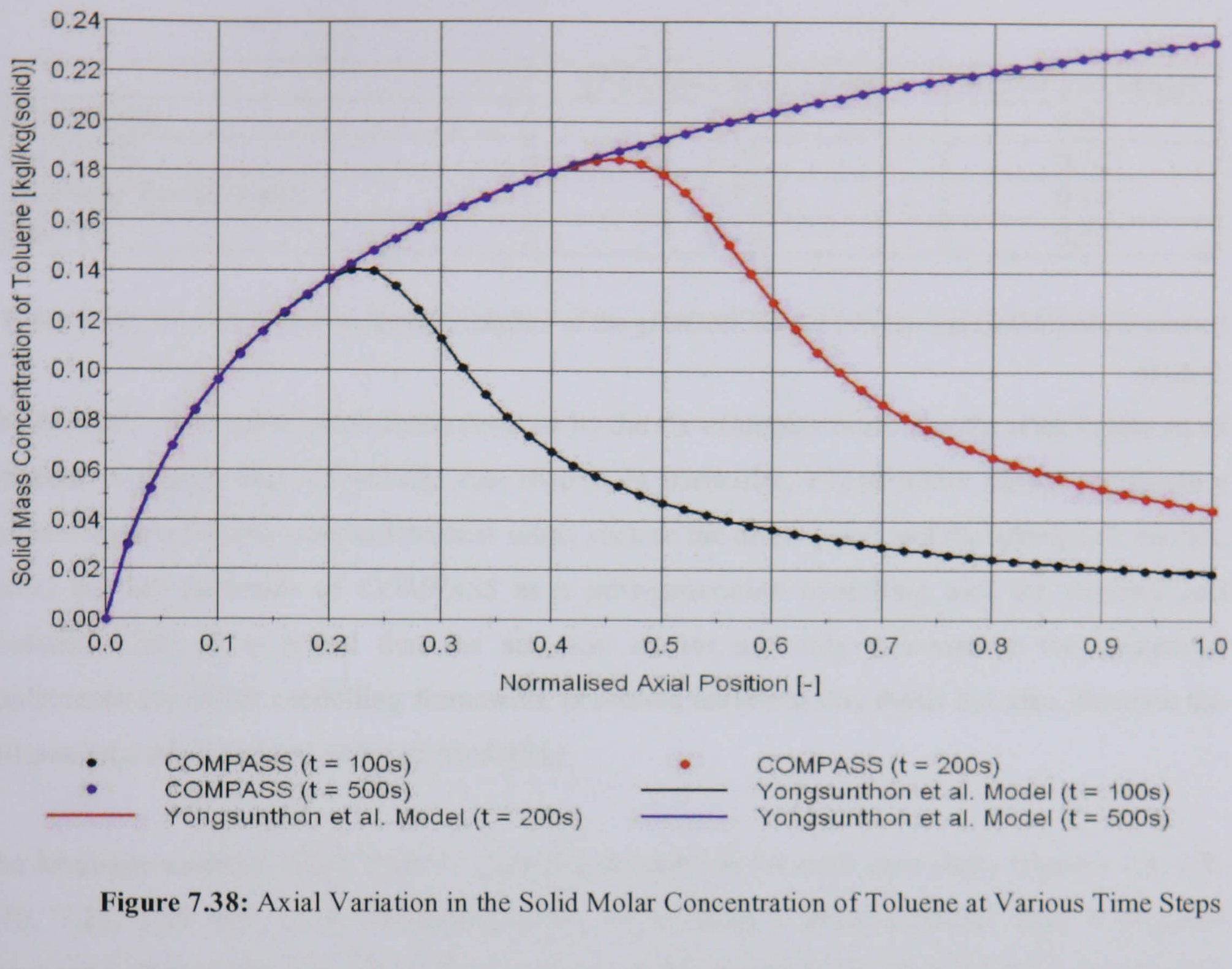


Figure 7.38: Axial Variation in the Solid Molar Concentration of Toluene at Various Time Steps

7.7 Summary

In this chapter, six examples have been used to demonstrate and validate the framework for automatic model generation described earlier in this thesis. For all examples, there was a very good agreement between the simulation results of the *COMPASS* generated model and those of the reference model.

An interesting aspect of the validation simulations was that for lumped systems[12], the *COMPASS* generated model took less time to solve than the corresponding gPROMS library model, as shown in Table 7.10. The *COMPASS* model was formulated to fit the specific purpose at hand, so the generated model involved fewer equations and consequently the simulation of the *COMPASS* generated model took less time than the *gPROMS* library model.

[12] This observation was sadly reversed for the plug flow reactor as the careful design of the gPROMS library model meant that the model was more robust than the distributed model generated by *COMPASS*.

	gPROMS Model Library	*COMPASS* Model
Buffer Tank	0.19	0.05
Gas CSTR	0.50	0.12
Adiabatic Flash Vessel	0.53	0.14
Gas PFR	0.45	0.53

Table 7.10: The CPU times (s) for the Solution of the *gPROMS* Model Library and *COMPASS* Generated Models.

The diversity of physical phenomena covered by the six examples illustrates the wide spectrum of different processes that *COMPASS* can model. In particular, *COMPASS's* ability to generate process models for non-standard process units, such as the drum boiler and the adsorptive reactor, reveal the full potential of *COMPASS* as a next-generation modelling tool for industry and academia alike. It is hoped that the six case studies not only demonstrate the successful implementation of the modelling framework presented earlier in this thesis but also illustrate the full potential of phenomenon based modelling.

The language excerpts taken from the physical description for each case study (figures 7.3, 7.9, 7.16, 7.23, 7.31 and 7.35) demonstrate that, by building a process model from a physical description, the emphasis is placed firmly on the model assumptions. Such emphasis results in a high level of interaction between the modelling software and the user. With the user aware of the model assumptions, they should be able to formulate a model that is both accurate and concise, incorporating all behavioural aspects of importance and discarding those of little significance. By placing the user firmly in control of the modelling process, the likelihood of erroneous modelling relating to a misunderstanding of the modelling assumptions is small.

In the final chapter, the conclusions drawn from the research are presented and some suggestions for the next stage of research are given.

Chapter 8

Conclusions

In this thesis, a prototype piece of software called *COMPASS* has been developed that is capable of generating mathematical models for one- and two- phase chemical processes. Based on the paradigm of physicochemical modelling, *COMPASS* is able to interpret a user-supplied physical description of a given process and generate the appropriate mathematical model.

Building on the concepts of Vazquez-Roman (1992), Cho (1997) and Barber (2000), *COMPASS* is based on the well-founded premise that a process can be regarded as a network of interacting thermodynamic phases, with each phase fully described in terms of a finite set of state variables. By accepting this premise, a systematic two-tier framework for process representation has been established. The first tier of the framework allows the process to be represented as a network of elementary building blocks interlinked through a number of *Connections*, with each *Connection* representing the transfer of mass and energy. To allow complex two-phase processes to be modelled, a new entity called the *Zone* is formed to replace the phase as the elementary building block. A *Zone* entity is a local model for a two-phase system, which, when the volume of one phase falls to zero, becomes a local model for a single-phase region, so it is able to represent either a single-phase region or a two-phase region. By partitioning the process first into *Zones* and then into phases, the framework is able to represent the complex interfacial interactions of two-phase systems.

With the network of *Zones* in place, the second tier of the framework provides a physical-based, object-oriented mechanism by which the user can describe the phenomenon occurring within each *Zone* in the network. For each *Zone*, the physical description is established in terms of the state variables. To ensure that the process can be described in physical terms that are easily understood by the user, the chosen state variables are the composition (described by the component mass fractions), the temperature and the pressure. In addition to describing the thermodynamic state of

each phase, it is also necessary to describe the volume of the *Zone*, the mass flowrate of the material through the *Zone* and, for a two-phase *Zone*, the volume fraction of the two phases. For a two-phase *Zone*, the state variables for the two phases may be described collectively or independently. For each state variable, the user should specify a value or, for instances where the value is not known, the assumptions that *COMPASS* may use to determine that variable.

Behind the user interface is a sophisticated model-building procedure that interprets the physical description, analyses the description for errors and applies object-oriented techniques to build the appropriate mathematical model. The model-building procedure consists of a sequence of algorithms that encapsulate the modelling logic needed to generate the appropriate equations. Dependent on the physical description provided by the user, equations are combined to form mathematical submodels for each state variable; these mathematical submodels are then combined to form mathematical models for each modelling element (*Zone* or *Reservoir*). All mathematical equations such as conservation laws, equations of state and empirical correlations are stored within *COMPASS* as a collection of equation fragments, which are combined to create the appropriate form of the equation. This ability to construct process models from the basic terms of the equations allows the generation of a well-posed and solvable mathematical model time after time, provided only that the basic physical description supplied to *COMPASS* is correct.

The capability of the modelling framework and the success of the model generating algorithms within *COMPASS* have been assessed with over twenty case studies. Examples of standard and more complex process units make it possible to directly compare the modelling capabilities of *COMPASS* against the capabilities of commercial modelling software. It can be seen that, when compared against flowsheet software (e.g. AspenTech), *COMPASS* provides a greater scope of application, can generate model families of varying degrees of complexity and felicity and places a greater emphasis on the model assumptions. By eliminating the need to write the mathematical model, *COMPASS* provides a potential market advantage over equation-based modelling software, as it is able to increase the modelling user base from a small group of specialists to the whole community of chemical engineers with a rudimentary understanding of chemical engineering.

The excellent agreement between the results of the *COMPASS* generated models and the reference models clearly show that *COMPASS* has met the objectives of the research. What is less

clear is whether the ideas behind *COMPASS* can be incorporated into a state-of-the-art commercial modelling tool. Within the final section of this thesis some key criteria for the development of *COMPASS* to the point of commercialization are identified and a set of targets are suggested that would need to be achieved if *COMPASS* is to pave the way for the next generation of computer-aided process modelling tools.

8.1 Suggestions For Future Work

The general suggestions for future work are concerned with advancing the design of *COMPASS* to the stage where it can reach its full market potential as a physical based modelling tool. To achieve this aim, the proposals for future work are inspired by both commercial and academic requirements, with consideration given to the potential market for a physically-based modelling tool and the current limitations of *COMPASS*. The suggestions for future work are split into two sections. The first section is concerned with the strategic development of *COMPASS*. In this section some of the possible barriers that may restrict the commercialisation of physically-based modelling are identified and requirements for future work are proposed. In the second section, the motivation for the proposals is based on the current limitations of *COMPASS* with the suggestions for future work aiming to enhance the modelling power and scope of application of the software.

8.1.1 Strategic Development: A New Generation of Unit-Based Process Modelling Software

The industrial requisite that all models are extensively tested and error-free results in few chemical engineering companies developing process models in-house. Whilst *COMPASS* reduces the potential for error involved in model construction it cannot prevent the user from providing an incorrect physical description. It is anticipated that the risk of erroneous modelling will restrict its widespread industrial use as a stand-alone product. The real potential of *COMPASS* lies in its ability to construct library-type models for unique and novel processes in a quick and resource-efficient manner. The library models would be constructed and validated by a provider of advanced model-based technology for the client to use as part of a flowsheeting package. To ensure that *COMPASS* can be integrated seamlessly into a flowsheeting package, both the physical language and the generated model should be expressed in a generic and fully application-independent manner. Through a strategic alliance with a flowsheet software provider,

COMPASS could permit each library model to be expressed in purely physical terms (figure 8.1). This would support the rapid development of library models, improve the clarity and understanding of each library model and provide the user with the opportunity to adjust the library model to fit the purpose at hand. Before *COMPASS* can be integrated within a flowsheeting package several modifications would have to be made:

a) An interface would need to be developed to ensure that the physical description and the generated model are fully compatible with the flowsheet environment.

b) The physical declaration should be split into two components: a context-independent section, which the expert would specify, and a context-dependent section that can be specified by the user when instantiating the process unit model. The user-supplied description need not necessarily be restricted to a declaration of the model parameters but could also include certain well-understood model assumptions.

c) The *COMPASS's* software would need to be embedded within the flowsheeting software. The *COMPASS* description should lie behind the icon for each library model and formulate the model equations for that process unit.

d) A facility to document the assumptions used for the development of each library model should be provided.

e) Each library model should have the facility to store several physical declarations, thereby allowing model families to be developed for similar process units.

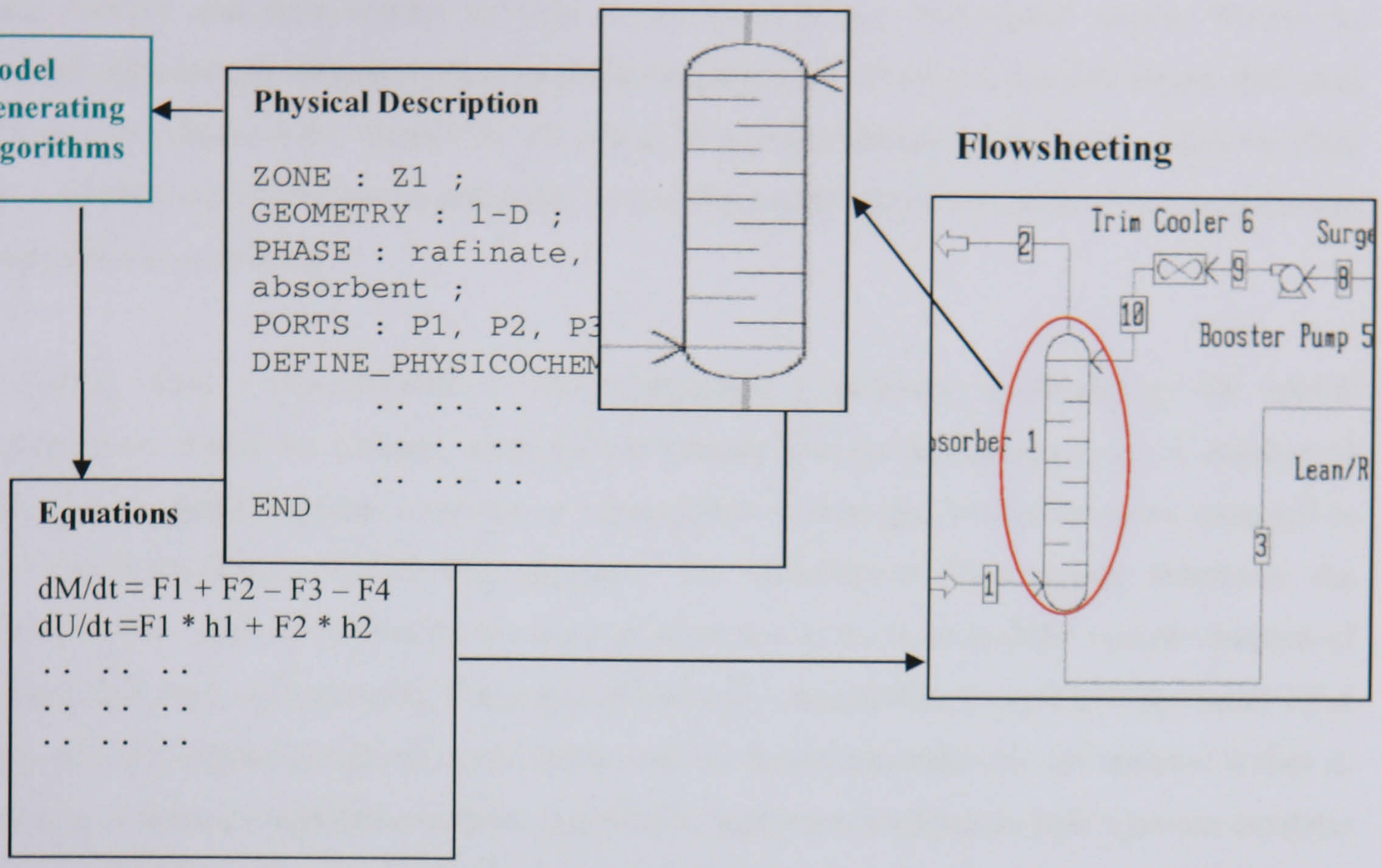


Figure 8.1: Embedding *COMPASS* within a flowsheeting package. The physical description provides an additional layer between the library model and the model equations.

8.1.2 Broadening the Scope of *COMPASS* – Reducing its Current Limitations

The purpose of this section is to present some suggestions for future work that enhance the modelling capabilities of *COMPASS*.

A Framework for the Multi-Hierarchical Decomposition of a Process: Enhancing the power with which a process can be broken down into the set of elementary units (*Zones*) that *COMPASS* uses to characterise the process, would improve the overall scope of application. One possible strategy to improve this process of model fragmentation is to adopt a method of hierarchical decomposition, where each stage of the process model is more refined than the previous one.

For a two-phase *Zone*, the region of space that is represented by the *Zone* is fully characterised by the extent of the *Zone*, the volume fraction of the two phases and their respective thermodynamic state. In its simplest form, the two-phase *Zone* can be regarded as a container for the two entities (currently phases), which can generate models to predict the overall spatial region that the two

entities occupy and their volume fraction at any point within that spatial region. Whilst the structural approach developed within this thesis equates each entity to a specific phase. this need not necessarily be the case. Instead, by allowing the user to represent each entity within the *Zone* by an assembly or collection of additional *Zones*, the modelling power of the framework would be substantially increased.

By making such a modification to the framework, a powerful methodology for spatial decomposition would be formed, whereby the process can be decomposed into a number of single- or two- phase regions, with one or more regions submerged within the phase assigned to the higher-level *Zone*. Within this approach, the higher-level *Zone* would determine the thermodynamic state of the first phase, the overall extent of the *Zone* and the volume fraction of the phase and the *Zone* assembly. The assembly of *Zones* would then be split into the lower level *Zones,* with each *Zone* determining the extent and the thermodynamic state of material within it. Examples of process units that could be modelled by this representation include a porous catalytic reactor, where the diffusion of chemical species within each particle could be modelled by an additional *Zone*, a shrinking core fuel pellet, comprised of a gas phase, an outer ash layer and an inner fuel core, and process units that incorporate submerged tubes or catalytic rod bundles (e.g. a shell and tube heat exchanger). Such an extension of the framework would allow any given two-phase *Zone* to be further decomposed into an assembly of *Zones*, of which one or more may be two-phase *Zone(s)*, so the modified framework could support an unlimited number of hierarchical levels.

Incorporating the Two-fluid Model within the Framework: The current application of the framework to two-phase systems where the velocity of both phases is non-zero is restricted by the capabilities of the drift flux model. The limitation of the drift flux approach therefore confines the application of the framework to the bubble, droplet and slug regimes. To avoid this limitation there is a need to incorporate the two-fluid model into the framework thus, enabling the stratified and annular flow regimes to be modelled.

Extending the Number of Model Classes to Handle Radial Distribution Profiles:
At the present time, *COMPASS* is only able to handle systems whose behaviour is either uniform or distributed over the axial domain. Future work could extend the number of classes to enable regions of the process that experience a radial distribution in either the cylindrical or spherical coordinates to be modelled.

Empirical Correlations and Library Models: Due to the time constraints of this research, the current state of COMPASS assumes that there is an available library of empirical correlations and transfer equations. The future development of *COMPASS* would require the construction of a library of suitable correlations. Possible correlations that might be included within the framework include the Ergun equation for the pressure drop through a packed bed, the Francis formula for the flow over a weir, various heat transfer coefficient equations and reaction rate equations. The library should not only store the correlations but also provide information concerning the range of applicability of each correlation.

Modelling Discontinuities: The rigid modular nature of the framework restricts the application of *COMPASS* to processes that do not experience discontinuities. To overcome this limitation, the framework could be modified so that the network of *Zones* is no longer rigid but can vary depending on the occurrence of the discontinuities. One possible solution is to assign a logical condition to each *Connection*. If the logical condition is met, then the mass and energy flowrate through the *Connection* is calculated according to the equations presented in this thesis. If the logical condition is not matched, then the mass and energy flowrate through the *Connection* would be equated to zero.

Whilst the outcome of the research presented in this thesis falls short of producing a piece of software that can be commercialized, it has been demonstrated that, with reasonable success, the work does bridge the gap between the academic dream of using the computer to generate models from a physical description and the commercial reality of developing a piece of software that will actually achieve this.

It is hoped that the success of this research will provide others with the motivation to follow in the same lines, using a similar basis to characterise chemical processes and building on the model construction methods discussed in this thesis. If others can find the solutions to the points raised in the suggestions for future work section, then it is quite possible that their work will create a next-generation process modelling tool that place little dependency on the expert. If this is so then the chemical industry may no longer be dependant on the expertise of modelling consultancy companies and the history of the process modelling industry may move full circle, with process models once again developed in house.

References

Adam, E.J. AND J.L. Marcheti (1999). Dynamic simulation of large boilers with natural recirculation. *Computers and Chemical Engineering,* **23**, pp. 1031–1040.

Astrom, K.J. AND R.D. Bell (2000). Drum-boiler dynamics. *Automatica*, **36**, pp. 363-378.

Azbel, D. (1981). Two-Phase flows in Chemical Engineering. Cambridge University Press.

Barber, J. (2000). Computer Generation of Process Models. Ph.D. Thesis, University of London.

Barton, P.I. AND C.C. Pantelides (1994). Modelling of combined discrete/continuous processes. *AIChE.J.*, **40**, pp. 966-979.

Basta, N., (1997). Process simulation – having it your way. *Chemical Engineering*, **104(4)**, pp. 155–160.

Batistella, C. AND M. Maciel (1996). Modeling, simulation and analysis of molecular distillators, centrifugal and falling film. *Computers and Chemical Engineering,* **20**, pp S19-S24.

Beckberger, L.H. AND K.M. Watson (1948). Catalytic dehydrogenation of normal butane to butadiene in the presence of steam. *Chemical Engineering Progress*, **44**, pp. 229.

Best, R. (2003). Computers: are they always right? *The Chemical Engineer,* **743**, pp. 40–41.

Bieszczad, J., A. Koulouris AND G. Stephanopoulos (1999). Model.La: A phenomena-based modeling environment for computer-aided process design. *In Fifth International Conference of Computer Aided Process Design FOCAPD '99*, Colorado, USA. Paper C03.

Bieszczard, J. (2000). A framework for the language and logic of computer-aided phenomena-based process modeling. Ph.D. Thesis, Massachusetts Institute of Technology.

Bird, R. B., W. Stewart AND E.N. Lightfoot (2001). Transport Phenomena. John Wiley & Sons.

Bogusch, R. AND W. Marquardt (1995). A formal representation of process model equations. *Computers and Chemical Engineering*, **19**, pp. S211-S216.

Bogusch, R., B. Lohmann AND W. Marquardt (2001). Computer-aided process modelling with MODKIT. *Computers and Chemical Engineering*. **25**, pp. 963-995.

Borkowski, J.A. (1992). TRAC-BF1/MOD1. Models and correlations. NUREG/CR-4391. EGG-2680, Rev. 4.

Chexal, B AND G. Lellouch (1986). A full-range drift-flux correlation for vertical flows. EPRI NP-3989-SR.

Cho, J.H. (1997). Computer Aids for Mathematical Model Building. Ph.D. Thesis, University of London.

Coon, J., M. Kusch, M. Rowland AND J. Cunningham (1998). Simulation software and engineering expertise: A marriage of necessity. AICHE Spring Meeting, New Orleans.

Crank, J. (1984). Free and Moving Boundary Problems. Oxford Science Publication.

Crowe, C., M. Sommerfield AND T. Yutaka (1998). Multiphase Flows with Droplets and Particles. CRC Press.

Davis, B.R. AND D.S. Scott, (1964). Rate of isomerization of cyclopropane in a flow reactor. *Industrial and Engineering Chemistry Fundamentals,* **3**, pp 20-23.

Deckwer, W.D. (1985). Reaktionstechnik in blasensaulen. Otto Salle Verlag, Frankfurt, Main.

Dekkers, R., H.A. Preisig AND M. Weiss (1999). SyMoM – a symbolic manipualtion package for dynamic models of physical systems. *Computers and Chemical Engineering*, **23**, pp. S657-S660.

Delhaye, J.M., M. Giot AND M.L. Riethmuller (1981). Thermohydraulics of Two-phase Systems for Industrial Design and Nuclear Engineering. Hemisphere publishing corporation.

Denn, M. (1987). Process Modeling, Longman Scientific Technical.

Dieterich, E. E. AND G. Eigenberger (1995). Bimap – A tool for computer aided modeling in

chemical reaction engineering. *Computers and Chemical Engineering*, **19**, pp. S773-S778.

Dieterich, E.E. AND G. Eigenberger (1997). The ModuSim concept for modular modeling and simulation in Chemical Engineering. *Computers and Chemical Engineering*, **21**, pp. S805-S809.

Dieterich, E.E., A. Salden, J. Schmidt AND G. Eigenberger (1997). Computer aided modeling with Bimap. *Computers and Chemical Engineering*, **21**(10), pp. 1191-1201.

Drengstig, T., S. Wasbo AND B.A. Foss (1997). A formal graphical based process modeling methodology. *Computers and Chemical Engineering* **21**, pp. 472-476.

Eborn, J., H. Tummescheit AND K.J. Astrom (1999). Physical system modeling with Modelica, http://www.control.lth.se/~hubertus/ThermoFluid/publ/ifac99.pdf

Eborn, J., H. Tummescheit AND F.J. Wagner (2000). Development of a Modelica base library for modeling of thermo-hydraulic systems. In *Proceedings of the 41st SIMS Simulation Conference, SIMS 2000*, pp. 253-266. SIMS, Copenhagen, Denmark.

Evans, L.B., J.F. Boston, H.I. Britt, P.W. Gallier, P.K. Gupta AND B. Joseph (1979). Aspen: An advanced system for process engineering. *Computers and Chemical Engineering*, **3**, pp. 319-327.

Fabricius, S. AND E. Badreddin (2002). Modelica library for hybrid simulation of mass flow in process plants. Proceedings from the 2nd International Modelica Conference, pp. 225-234.

Feinberg, M. (1979). On Gibbs' phase rule, Archive for Rational Mechanics and Analysis, **70**, pp. 219-234.

Fell, N. (1997). Simulation almost as good as the real thing. *The Chemical Engineer*, **637**, pp. 15-18.

Gani, R. AND J. O'Connell (2001). Properties and CAPE: from present uses to future challenges. *Computers and Chemical Engineering*, **25**, pp. 3-14.

Gerstlauer, A., M. Hierlemann AND W. Marquardt (1993). On the representation of balance equations in a knowledge-based process modelling tool. Chisa 1993.

Gibbs, J. W. (1878). On the equilibrium of heterogeneous substances. *Trans. Conn. Acad.* **3**. pp. 343-524. In The Scientific Papers of J.W. Gibbs, Vol 1, New York, Dover Publications. 1961.

Gilles, E. (1998). Network theory for chemical processes. *Chemical Engineering Technology,* **21**, pp. 121-132.

Hangos, K.M. AND I.T. Cameron (2001). A formal representation of assumptions in process modelling. *Computers and Chemical Engineering*, **25**, pp. 237-255.

Hangos, K.M. AND I.T. Cameron (2001). Process Modelling and Model Analysis. Academic press.

Hewitt, G.F. AND D.N. Roberts (1969). Studies of two-phase flow patterns by simultaneous x-ray and flash photography. Technical report, UKAERA Report AERE-M 2159.

Hirt, C.W. AND B.D. Nichols (1979). Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of Computational Physics*, **39**, pp. 201-225.

Holland, C. D. AND G.A. Rayford (1979). Fundamentals of Chemical Reaction Engineering. Prentice-Hall, Inc.

Hsu, Y. AND R. Graham (1976). Transport Processes in Boiling and Two-Phase Systems. Hemisphere Publishing Cooperation.

Ishii, M. (1975). Thermo-Fluid Dynamic Theory of Two-Phase Flow. Eyrolles, Paris

Ishii, M. (1977). One-Dimensional drift-flux and constitutive equations for relative motion between phases in various two-phase flow regimes, ANL-77-47

Ishii, M. AND K. Mishima (1984). Two-fluid model and hydrodynamic and constitutive relations. *Nuclear Engineering and Design,* **82**, pp. 107-126.

Ishii, M. AND N. Zuber (1979). Drag coefficient and relative velocity in bubbly, droplet or particulate flows. AIChE Journal, **25**(5).

Jaunzemis, W., (1967). Continuum Mechanics. The Macmillan Company

Jensen, A.K AND R. Gani (1995). Development and application of problem specific "local"

process simulators in CAPE. *Computers and Chemical Engineering*, **19**, pp. S311-S316.

Jensen, A. AND R. Gani (1996). A computer aided system for generation of problem specific process models. *Computers and Chemical Engineering*, **20**, pp. S145-S150.

Jensen, J. AND H. Tummescheit (2002). Moving boundary models for dynamic simulations of two-phase flows. 2nd International Modelica Conference,
http://www.modelica.org/Conference2002/papers.shtml

Jiang, S.Y, X.X. Wu, Y.J. Zhang AND H.J. Jia (2001). Thermal hydraulic modelling of a natural circulation loop. *Heat and Mass Transfer*, **37**, pp. 387-395.

Kyle, B. G. (1999). Chemical and process thermodynamics. Prentice Hall.

Lakner, R., I.T. Cameron AND K.M. Hangos (1999). An assumption-driven case-specific model editor. *Computers and Chemical Engineering*, **23**, pp. S695-S698.

Lapin, A. AND A. Lubert (1994). Numerical simulation of the dynamics of two-phase gas-liquid flows in bubble columns. *Chemical Engineering Science*, **49**, pp. 3361-3674.

Levine, R., T. Mason AND D. Brown (1990). Lex and Yacc. O'Reilly and Associates, Inc.

Levy, S. (1999). Two-Phase Flow in Complex Systems. John Wiley and Sons.

Linniger, A.A., S. Chowdhry, V. Bahl, H. Kredl AND H. Pinger (2000). A systems approach to mathematical modeling of industrial processes. *Computers and Chemical Engineering*, **24**, pp. 591-598.

Lohmann, B. AND W. Marquardt (1996). On the systemization of the process of model development. *Computers and Chemical Engineering*, **20**, pp. S213-S218.

Mangold, W., S. Motz AND E.D. Gilles (2002). A network theory for the structured modelling of chemical processes. *Chemical Engineering Science*, **57**, pp. 4099-4116.

Marquardt, W., A. Gerstlauer AND E. Gilles (1993). Modeling and representation of complex objects: chemical engineering perspective. In Proc. 6th Int. Conf. On IEA/AIE, 219-228, Edinburgh.

Marquardt, W. (1996). Trends in computer-aided process modeling. *Computers and Chemical Engineering*, **20**, No. 6/7, pp. 591-606.

Mi, Y., M. Ishii AND L.H. Tsoukalas (2001). Flow regime identification methodology with neural networks and two-phase flow models. *Nuclear Engineering and Design*, **204**, pp. 87-100.

Mishima, K. AND M. Ishii (1984). Flow regime transition criteria for upward two-phase flow in vertical tubes. *Int. J. Heat Mass transfer*, **27**(5), pp. 723-737.

Moe, H. (1995). Dynamic Process Simulation. Studies on Modeling and Index Reduction. Ph.D. Thesis, University of Trondheim.

Nicollela, C., P. Pavasant AND A.G. Livingston (2000). Substrate counter diffusion and reaction in membrane attached biofilms: mathematical analysis of rate limiting mechanics. *Chemical Engineering Science*, **55**, pp. 1385-1398.

Oh, M. (1995). Modelling and Simulation of Combined Lumped and Distributed Processes. Ph.D. Thesis, University of London.

Oh, M. AND C.C. Pantelides (1996). A modelling and simulation language for combined lumped and distributed parameter systems. *Computers and Chemical Engineering*, **20**, 6/7, pp. 611-633.

Pantelides, C.C. AND Barton, P. I. (1993). Equation-oriented dynamic simulation: current status and future perspectives. *Computers and Chemical Engineering*, **17**, pp. S263 – S285.

Pantelides, C.C. AND H.I. Britt (1995). Multipurpose process modelling environments. AICHE Annual Meeting.

Perkins, J.D. AND G.W. Barton (1987). Modelling and simulation in process operation. In G.V. Reklaitis and H.D. Spriggs (Eds.), Computer Aided Process Operations, pp. 287-316.

Perkins, J. D. AND R.W.H Sargent (1982). Speedup: A computer program for steady-state and dynamic simulation and design of chemical processes. *AIChE Symp. Ser.*, **78**(214), pp 1-11.

Perkins, J. D., R.W.H. Sargent, R. Vazquez-Roman AND J.H. Cho (1996). Computer Generation of Process Models. *Computers and Chemical Engineering*, **20**, No. 6/7, pp. 635-639.

Pickering, P. (1994). Instabilities in Low Pressure Boiling Systems. Thesis, University of London.

Piela, P. C., T.G. Epperly, K.M. Westerberg AND A.W. Westerberg (1991). Ascend: An object-oriented computer environment for modeling and analysis: The modeling language. *Computers and Chemical Engineering*, **15**(1), pp. 53-72 .

Preisig, H.A. (1995). MODELLER - An object oriented computer-aided modelling tool. *AIChE Symp. Ser.*, **91**(304), pp. 328-331.

Preisig, H. (1996). Computer aided modelling – two paradigms on control. *Computers and Chemical Engineering,* **20**, pp. S981-S986.

Richardson, S.M. (1989). Fluid Mechanics. Hemisphere Publishing Corporation.

Rohsenow, W.M., J.P. Hartnett AND I.C. Young (1998). Handbook of Heat Transfer. McGraw-Hill.

Rumbaugh, J. (1991). Object Oriented Programming and Design. Prentice-Hall International.

Sargent, R.W.H. (1983). Advances in modelling and analysis of chemical process systems. *Computers and Chemical Engineering* 7(4), pp. 219–237.

Sargent, R.W.H. (1990). Process design – what next? In J.J. Siirola, I. E. Grossmann AND G. Stephanopoulos, (Eds.). Foundations of Computer-Aided Process Design. Elsevier.

Sowa, C. (1997). Make the move to dynamic simulation. *Chemical Engineering Progress,* **93**(8), pp. 44 - 47.

Steinmann, W. AND S. Zunft (2002). TechThermo – A library for Modelica applications in technical thermodynamics. 2nd International Modelica Conference, pp. 217-224.

Stephanopoulos, G., G. Henning AND H. Leone (1990). Model.La. A modelling language for Process Engineering-II. Multifaceted Modeling of Process Systems. *Computers and Chemical Engineering*, **14**(8), pp. 847-869.

Stephanopoulos, G., J. Johnston, T. Kriticos, R. Lakshmanan, M. Marvrovouniotis AND C. Siletti (1987). Design-Kit: An object-oriented environment for process engineering.

Computers and Chemical Engineering, **11**(6), 655-674.

Taitel, Y., D. Bornea AND A. Dukler (1980). Modelling flow pattern transitions for steady upward gas-liquid flow in vertical tubes. *AIChE* **26**(3), pp. 345-354.

Trankle, R., A. Gerslauer, M. Zeitz AND E. Gilles (1997). Application of the modeling and simulation environment ProMoT/DIVA to the modeling of distillation processes. *Computers and Chemical Engineering*, **21**, pp. S841-S846.

Tummescheit, M., J. Eborn AND F. Wagner (2000). Development of a modelica based library for modelling of thermo-hydraulic systems. Modelica Workshop 2000, pp. 41-51.

Tummescheit, H. AND J. Eborn (2002). Chemical reaction modeling with Thermofluid/MF and Multiflash. Proceedings for the 2nd International Modelica Conference, pp. 31-39.

Vazques, G.E., I. Ponzoni, M.C. Sanchez AND N.B. Brignole (2001). ModGen: a model generator for instrumentation analysis. *Advances in Engineering Software*, **32**, pp. 37-48.

Vazquez-Roman, R. (1992). Computer Aids for Process Model Building. Ph.D. Thesis, University of London.

Vazquez-Roman, R., J.M.P. King AND R. Banares-Alcantara (1996). KBMoSS: A process engineering modelling support system. *Computers and Chemical Engineering*, **20**, S309-S314.

Walas (1995). Chemical Reaction Engineering Handbook of Solved Problems. Gordon and Breach Publishers.

Wallis, G. (1969). One-Dimensional Two-Phase Flow. McGraw-Hill Book Company.

Wasbo, S. (1996). Furnace Modelling Using Object-Oriented Principles. Ph.D. Thesis. University of Trondheim.

Wasbo, S.O. AND B.A. Foss (1996). Modelling unit processes using formal language description and object-orientation. *Mathematical Modelling of Systems*, **1**(1), pp. 0-30.

Weiss, M. AND H.A. Preisig (1997). Simplifying hypotheses in computer-aided modelling: a singular perturbation approach. *Computers and Chemical Engineering*, **21**, pp. S721-S726.

Westerweele, M.R., H.A. Preisig AND M. Weiss (1999). Concept and design of modeller, a

computer-aided modelling tool. *Computers and Chemical Engineering*, **23**, pp. S751-S754

Westerweele, M.R. (2003). Five Steps for Building Consistent Dynamic Process Models and their Implementation in the Computer Tool Modeller. Technische Universiteit Eindhoven.

Whitaker, S. (1968). Introduction to Fluid Mechanics. Prentice-Hall

Yongsunthon, I. AND E. Alpay (1999). Design of periodic adsorptive reactors for the optimal integration of reaction, separation and heat exchange. *Chemical Engineering Science*, **54**, pp. 2647 – 2657.

Zuber, N. AND J.A. Findlay (1965). Average volumetric concentration in two-phase flow Systems. *Journal of Heat Transfer*, **87**, pp. 453-468.

Appendix A

Nomenclature

A.1 Mathematical Notation

The following notation has been used in this thesis. To aid in the avoidance of ambiguity, appropriate SI units are identified for each quantity. However it is emphasized that any consistent set of units could be used, or the equation could be represented in dimensionless form.

a	(m^{-1})	Interfacial area per unit volume
A	(m^2)	Cross sectional area
b	(-)	Scalar term
B	(-)	Controller bias
c	(mol/m^3)	Molar concentration
C_{an}	(-)	Roughness parameter
C_o	(-)	Mass distribution parameter
C_v	(-)	Momentum distribution parameter
d	(m)	Particle diameter
D	(m)	Vessel diameter or hydraulic diameter
D_i	(m^2/s)	Diffusion coefficient
E	(J/kg)	Total specific energy
E_a	(J/mol)	Activation energy
f	(-)	Friction factor
F	(kg/s)	Mass flow rate
g	(m/s^2)	Gravitational acceleration
h	(J/kg)	Enthalpy
H	(J/kg)	Total enthalpy and specific kinetic energy of the phase
H_a	(J/mol)	Heat of absorption
j	(Kg/m^2s)	Diffusive mass flux
J	(m/s)	Mass flux or velocity of the volume centre of the mixture
k	(s^{-1})	Rate constant
k	(J/m.K.s)	Thermal conductivity
K	(-)	Equilibrium coefficient
K_c	(-)	Proportional gain
K_e	(mol^3/m^9)	Reaction equilibrium constant
l	(m)	Liquid level
L	(m)	Domain length
m_B	(mol/Pa.kg)	Absorbent isotherm gradient
M	(kg)	Mass of material
n	(kg/m^2s)	Combined mass flux
n	(-)	Polytropic coefficient
NCM	(-)	Number of Components
NPB	(-)	Number of Ports that experience bulk flow
NPD	(-)	Number of Ports that experience diffusive flow
NZ	(-)	Number of Zones
O_c	(-)	Manipulated variable
p	(Pa)	Partial pressure
P	(Pa)	Pressure
P_w	(m)	Wetted perimeter

q	(J/m^2s)	Conductive heat flux.
Q	(J/s)	Heat flow rate
r	(Kg/m^3s)	Reaction rate
r_g	(m)	Bubble radius
r_{sm}	(m)	Sauter mean radius
R	(Kg/m^3s)	Reaction source term
R	(J/molK)	Universal gas constant
R_e	(-)	Reynolds number
R_m	(kg/kmol)	Relative molecular mass
s	(J/kg.K)	Entropy
t	(s)	Time
T	(K)	Temperature
T	(-)	Variable of interest
u	(J/kg)	Internal energy
U	(J/kg)	Total energy (internal energy + kinetic energy of the phase)
v	(m/s)	Velocity
V	(m^3)	Volume
W	(J/s)	Work done
x	(-)	Component mass fraction
X	(-)	Phase mass fraction
z	(m)	Position in the axial domain
$\hat{z}$	(-)	Position in the normalised axial domain
Z	(m)	Coordinate position

Greek Symbols

α	($J/m^2s.K$)	Heat transfer coefficient
ε	(-)	Set point variance
ε	(-)	Volume fraction
γ	(-)	Flow coefficient
η	(-)	Effective factor
λ	(-)	Flow direction
μ	(Pa.s)	Viscosity
ν	(m^3/kg)	Specific volume
θ	(^{o})	Angle of inclination
ρ	(kg/m^3)	Density
σ	(N/m)	Surface tension
τ	(s^{-1})	Integral time constant
τ	(Pa)	Shear stress
υ	(-)	Stoichiometric coefficient
Γ	(Kg/m^3s)	Interfacial mass flow per unit volume
Λ	(J/m^3s)	Interfacial heat flow per unit volume
T	(N.m)	Load torque

Subscripts

1	1^{st} axial boundary or 1^{st} phase
2	2^{nd} axial boundary or 2^{nd} phase
B	Bulk flow
c	Continuous phase

D	Diffusive
d	Dispersed phase
dj	Drift flux
DS	Design value
f	Pseudo first order
gs	Gas within the slug
i	Component i
IC	Internal *Connection*
IF	Interfacial
in	Inlet *Port* condition (No Extent Zone)
j	Reaction j
k	Phase k
m	Mixture
out	Outlet *Port* condition (No Extent Zone)
PT	Phase transition
r	Overall
s	Solid
s	Surface
w	Wall
z	Axial dimension

Superscripts

B	Boundary
M	Mechanical component
P	Port
R	Reservoir
VSL	Vessel
Z	Zone
*	Equilibrium value

A.2 Notation Used in the *COMPASS* Generated Models

The notation for all model variables and parameters consists of the variable or parameter symbol followed by a set of identifiers that provide each variable or parameter with a unique identity. Each symbol represents a generic type of variable or parameter and is selected from the list given below (A.2.1). The set of identifiers refer to the model name (*Zone* or *Reservoir*), the *Port* name, the name of the phase and the name of the component. The syntax that dictates the order and the form in which the identifiers are written is dependent on whether the model is generated in Text form or as a gPROMS input file.

For all cases where the model is generated in Text, the identifiers are contained within a set of brackets that directly follow the symbol. A comma separates each identifier. The general

formatting for each variable and parameter is given is as follows, with the sequence of identifiers given in table A.1.

Symbol(Identifier, Identifier, Identifier, Identifier)

The list of identifiers must start with the local model name, which refers to the name of the *Zone* or *Reservoir*. The following set of identifiers is dependent on the exact physical meaning of the variable or parameter, but may include the name of the *Port*, phase or component that the variable represents. All identifiers that are incorporated into the variable or parameter name must follow the order specified in table A.1.

To ensure compatibility with gPROMS, the *COMPASS* notation for models formulated as gPROMS input files adheres to the gPROMS rules and conventions. For all variables defined locally (within a particular *Zone* or *Reservoir* model), the notation commences with the variable or parameter symbol followed by the list of identifiers, separated by underscores.

Symbol_Identifier_Identifier_Identifier

The list of identifiers no longer commences with the local model name. However, except for the model name, the order of identifiers is the same as for text generated models (table A.1).

Text Formatted Models	gPROMS Formatted Models
Local Model Name	Port Name
Port Name	Phase Name
Phase Name	Component Name
Component Name	

Table A.1: The Pathname Order in which the Identifiers are Expressed

For the Connection submodels, the variables within each local model are referenced globally using the following notation.

Local Model Name . *Symbol_Identifier_Identifier_Identifier*

A.2.1 *COMPASS* Variable and Parameter Symbols

The following symbols are used within the *COMPASS* generated models. The appropriate SI units are given to aid in the avoidance of ambiguity. Again it is emphasised that any consistent set of units could be used within *COMPASS*, as could dimensionless variables.

a	(m^{-1})	Interfacial area per unit volume
A	(m^2)	Cross sectional area
alpha	(J/m^3s)	Interfacial heat flow per unit volume
b	(-)	Scalar term
C	(kg/m^3)	Mass concentration
Co	(-)	Mass distribution parameter
D	(m)	Diameter
den	(kg/m^3)	Density
F	(kg/s)	Mass flowrate
Fdiff	(kg/s)	Diffusive mass flowrate
g	(m/s^2)	Gravity
gamma	(-)	Flow direction
h	(J/kg)	Enthalpy
heat_diff_coef	(m^2s)	Thermal conductivity coefficient
hhom	(J/kg)	Homogeneous enthalpy
htot	(J/kg)	Total enthalpy
j	(Kg/m^2s)	Mass flux
k		Mass transfer coefficient
K	(-)	Equilibrium constant
Latent_heat	(J/kg)	Latent heat
M	(kg)	Mass holdup
Mass_diff_coef	(m^2/s)	Mass diffusion coefficient
MTRANS	(kg/m^3s)	Internal rate of mass transfer per unit volume
n	(Kg/m^2s)	Total mass flux
P	(Pa)	Pressure
Pw	(m)	Wetted perimeter
Q	(J/s)	Energy flowrate
Qdiff	(J/s)	Diffusive energy flowrate
qflux	(J/m^2s)	Heat flux
QTRANS	(J/m^3s)	Internal rate of heat transfer per unit volume
R	(kg/m^3)	Reaction source term
rb	(m)	Bubble radius
s	(J/kg.K)	Entropy
spv	(m^3/kg)	Specific volume
surften	(N/m)	Surface tension
T	(K)	Temperature
TAUWALL	(Pa)	Wall shear stress
u	(J/kg)	Internal energy
uhom	(J/kg)	Homogenous internal energy
utot	(J/kg)	Total internal energy
V	(m^3)	Volume
vdj	(m/s)	Drift flux velocity
veloc	(m/s)	Velocity

Vfrac	(-)	Volume fraction
visc	(Pa.s)	Viscosity
vsup	(m/s)	Superficial velocity
W	(J/s)	Work done
x	(-)	Mass fraction
z	(m)	Axial position

Appendix B

The No Extent Zone Model Class

The *No Extent Zone Model Class* is used to mathematically characterise a *Zone* that represents a region of the process whose hold-up is negligible. Such a *Zone* can be treated as a point (or line), to which algebraic equations can be assigned, to relate the output state variables to the input state variables. It should be noted that only one phase is allowed to flow through each *Zone* of no extent. The *Model Subclasses* for the *No Extent Zone Model Class* correspond to the flowrate of material through the *Zone*, the composition, the pressure and the temperature. Since the *Zone* of no extent contains no matter, there is no natural object to assign the equations to (e.g. a phase). To overcome this problem, all equations are expressed with respect to the state variables of the *Zone's Ports*.

In addition to structuring the equations according to the *model subclasses*, the equations are also structured according to whether the *Zone* has a single input and a single output or multiple inputs and outputs. For the case where there is just one input and one output, the *Zone* permits a change in the pressure between the input and the output *Ports*. Consequently the temperature of the outlet, the work that the *Zone* does against the environment and the heat exchanged between the *Zone* and its environment can be deduced from the first law of thermodynamics. For the case where the *Zone* has multiple input or output streams, the model class imposes the constraint that the pressures, temperatures and compositions of the outlet streams are the same. Furthermore, the constraint that there is no pressure drop across the *Zone* is enforced, so that all input streams must also have the same pressure.

B.1 The Port Subclass

To support the generation of the equations of state for each *Port*, it is necessary to relate the temperature of each *Port* (T^P) to the *Port* enthalpy (h^P). The generic equation is generated in the form:

$$h^P = f\left(T^P, P^P, x_i^P\right) \qquad [B.1]$$

where x_i^P is the mass fraction of component i at *Port* P and P^P is the *Port* pressure. The *Port* enthalpy is determined from the appropriate *Connection*.

B.2 The Mass Flowrate Subclass

Since the *Zone* of no extent represents a point, no material can accumulate within the *Zone* and hence it is necessary that the net mass flowrate of material into and out of the *Zone* is zero.

$$\sum_{P=1}^{NP} F^P = 0 \qquad [B.2]$$

In Eq. B.2, F^P is the mass flowrate through *Port* P and NP is the number of *Ports*. It should be noted that F^P is negative for outward flow.

Single Input / Single Output: For a single input/single output *Zone* of no extent, the physical behaviour of the *Zone* may dictate the mass flowrate through the *Zone*. For such cases it may be necessary to relate the mass flowrate through the *Zone* to the pressure drop across the *Zone* (eg. pipe flow) or to specify the mass flowrate explicitly.

$$F_{in}^P = f\left(P_{in}^P, P_{out}^P, Z_{in}^P, Z_{out}^P\right) \qquad [B.3]$$

In Eq. B.3, Z^P is the height of each *Port* and subscripts in and out refer to the inlet and outlet *Ports* respectively. If material flows through the *Zone* at a specific mass flowrate:

$$F_{in}^P = const \qquad [B.4]$$

If the *Zone* is in series with another *Zone* of no extent, it is possible that the overall mass flowrate through the two *Zones* in series is dictated by the physical behaviour of the second *Zone*. In such cases Eq. B.2 is sufficient to relate the output mass flowrate to the input mass flowrate.

Multiple Input / Multiple Output: For a multiple input / multiple output *Zone* of no extent, only Eq B.2 is generated. The mass flowrate through each *Connection* that is linked to a Multiple Input

/ Multiple Output *Zone* of no extent, may be determined through the use of an additional single input/single output *Zone* of no extent.

B.3 The Composition Subclass

The principle of mass conservation applied to each component across the *Zone* dictates that:

$$0 = \sum_{P=1}^{NP} F^P x_i^P \qquad [B.5]$$

Multiple Output: If the *Zone* is linked to two or more *Connections* that transfer material away from the *Zone*, it is assumed that perfect mixing occurs within the *Zone*, and hence the composition at each outlet *Port* is the same.

$$x_i^P = x_{i,out} \qquad \text{if } F^P < 0 \qquad [B.6]$$

where $x_{i,out}$ is the mass fraction of component i at each outlet *Port*.

B.4 The Pressure Subclass

The equations for Single Input / Single Output *Zones* and Multiple Input / Multiple Output *Zones* are different.

Single Input / Single Output: The model generation engine uses the following equation to relate the outlet pressure to the inlet pressure and the pressure drop across the *Zone*.

$$P_{out}^P = P_{in}^P - \Delta P^Z \qquad [B.7]$$

Eq. B.7 is sufficiently versatile to enable the following to be modelled:

1) No pressure drop across the *Zone*, ($\Delta P^Z = 0$).
2) A fixed pressure drop across the *Zone*, determined through the use of any of a number of empirical correlations, which relate the pressure drop to the flowrate through the *Zone*.
3) A pressure drop determined from the pressure at the inlet and outlet *Ports*.

Multiple Input / Multiple Output Zone: For a multiple input, multiple output *Zone*, it is assumed firstly that the pressures of all the input and output *Ports* are the same and consequently that there is no pressure drop across the *Zone*.

$$P^P = P^Z \qquad [B.8]$$

B.5 The Temperature Subclass:

The first law of thermodynamics is used to relate the enthalpy of the *Zone's* outlet *Ports* to the enthalpy of the *Zone's* inlet *Ports*, the work that the *Zone* does against the surrounding environment and the energy that is transferred between the *Zone* and its surrounding environment (specified through an *Energy Connection*).

$$0 = \sum_{P=1}^{NP} F^P \left[h^P + \frac{1}{2}\left(v^P\right)^2 \right] + Q^Z - W^Z \qquad [B.9]$$

In Eq. B.9, v^P is the velocity of material flowing through the *Port*, Q is the flow of heat into the *Zone* from its environment and W^Z is the rate of work that the *Zone* does against the surrounding environment. It should be noted that as the *Zone* represents a point in space, there is no work against gravity. If the kinetic energy of the material flowing through each *Port* cannot be neglected, Eqs. [B.10] and [B.11] must be included within the model to relate the velocity of the material at each *Port* to the respective mass flowrate.

$$F^P = \rho^P v^P A^P \qquad [B.10]$$

$$\rho^P = f\left(T^P, P^P, x_i^P\right) \qquad [B.11]$$

Single Input / Single Output: The following are additional assumptions that may need to be made for a Single Input / Single Output *Zone* if the work that the *Zone* does against its *Environment* cannot be neglected.

Isothermal Process: If the process is isothermal, both the pressure and the temperature are known at both the inlet and the outlet *Ports*.

$$T_{out}^{P} = T_{in}^{P} \qquad [B.12]$$

Isentropic Process: An isentropic system represents a section of the process where the entropy (s) of the system remains constant.

$$s_{in}^{P} = s_{out}^{P} \qquad [B.13]$$

For such a system, the principle of continuity, the component mass balance, the first law of thermodynamics and the entropy constraint can be solved simultaneously to relate the pressure, composition and temperature of the outlet *Port* to the respective state variables of the inlet *Port* and the work that the *Zone* does against its environment. As for the *Temperature Subclass* it is also necessary to relate the entropy at the respective *Ports* to the *Port's* state variables.

$$s_{in}^{P} = f\left(T_{in}^{P}, P_{in}^{P}, x_{i,in}^{P}\right) \qquad [B.14]$$

$$s_{out}^{P} = f\left(T_{out}^{P}, P_{out}^{P}, x_{out}^{P}\right) \qquad [B.15]$$

Polytropic Process: In many real processes it has been found that the thermodynamic states during an expansion or compression can be approximately described by the following relationship.

$$P_{in}^{P}\left(v_{in}^{P}\right)^{n} = P_{out}^{P}\left(v_{out}^{P}\right)^{n} \qquad [B.16]$$

where v is the specific volume of the *Phase* and n is a constant called the polytropic coefficient or the index of expansion and compression. The specific volume of the phase at each *Port* can be related to the appropriate state variables through the following equations.

$$v_{in}^{P} = f\left(T_{in}^{P}, P_{in}^{P}, x_{in}^{P}\right) \qquad [B.17]$$

$$v_{out}^{P} = f\left(T_{out}^{P}, P_{out}^{P}, x_{out}^{P}\right) \qquad [B.18]$$

Multiple Input / Multiple Output Zone: As for the Composition Subclass it is assumed that perfect mixing occurs within the *Zone* and that consequently the temperature of the each outlet *Port* will be the same.

$$T_{out}^{P} = T_{out} \qquad [B.19]$$

Appendix C

The Single-Phase Lumped Zone Model Class

The *Single Phase Lumped Zone Model Class* is used to mathematically characterise a *Zone* that encompasses material which is in just one stable thermodynamic state and whose state is approximated to be uniform across the extent of the *Zone*. *Model subclasses* exist for the Port entities in addition to the extent, the pressure, the temperature and the composition of the *Zone*. Each *model subclass* permits the relevant thermodynamic property either to remain constant throughout the simulation or to vary in time. If the property remains constant, the user must declare the value of that property. If the property varies, a suitable set of equations is formulated to determine its value throughout the time horizon.

The equations within the *Single-Phase Lumped Zone Model Class* are structured according to the state of the phase within the *Zone*. For the particular case where the *Zone* represents a gaseous phase, the model building approach equates the extent of the *Zone* to the total available space deduced from the volumetric constraints of the *Vessel* and the other *Zones* that are contained within it. The pressure, if of interest to the user, is deduced from both the extent of the *Zone* and from the total mass of material within the *Zone* through an equation of state. The liquid and solid *Zone* types are grouped together and treated in the same way. The extent of material within a *Zone* of this type is deduced from the total mass of material within the *Zone* and from the density of the phase. The pressure, if of interest, is deduced, not from the *Zone* itself, but from the *Zone's* environment, through a *Connection*.

In addition to specifying the phase type of the *Zone*, the user may also specify the compressibility of the phase. If the phase is declared as being compressible the pressure of the *Zone* is related to the phase density. If the phase is declared as being incompressible, the phase density is independent of the pressure.

C.1 The Port Subclass

For each *Port* connected to a Single Phase Open Connection that transfers material away from the *Zone*, it is necessary to relate the mass fraction (x_i) and the enthalpy (h) of the *Port* to the respective state variables of the *Zone* (superscript Z).

$$x_i^P = x_i^Z \qquad h^P = h^Z \qquad \text{[C.1] - [C.2]}$$

Further, regardless of the direction of flow, it is also necessary to relate the pressure at the *Port* to the pressure of the *Zone*. Depending on the physical description provided by the user, this may or may not include the hydrostatic pressure of the *Zone*.

$$P^P = P^Z + \rho^Z g\left(\frac{V^Z}{A^Z} - Z^P\right) \qquad \text{[C.3]}$$

where ρ is the density of material within the Zone, g is the gravitational acceleration, V is the volume of the *Zone*, A is the cross sectional area of the Zone and Z^P is the height of the *Port* relative to the *Zone's* datum. For each *Port* linked to a *Closed Material Connection* and for each component species transfer by the *Connection* it is necessary to relate the mass concentrations at the *Port* to the mass concentrations within the *Zone*.

$$\rho_i^P = \frac{M^Z x_i^Z}{V^Z} \qquad \text{[C.4]}$$

For each *Port* linked to a *Closed Energy Connection*, it is necessary to relate the temperature of the *Port* (T^P) to the temperature of the *Zone* (T^Z).

$$T^P = T^Z \qquad \text{[C.5]}$$

C.2 The Extent Subclass

The extent of the *Zone* may be assigned a constant value or can be determined as part of the global solution.

Case 1 – Constant Extent:

$$V^Z = const \qquad [C.6]$$

Case 2 – Variable Extent, Gaseous *Zone*: If the material is a gas, the extent of the *Zone* is determined from the volume of the other *Zones* within the *Vessel* and from the volumetric constraints of the *Vessel*. The equation for the volumetric constraint of the *Vessel* is generated within the *Vessel Model Class* (Section 5.2), so no further equations are needed.

Case 2a – Variable Extent, Liquid and Solid *Zones*: If the *Zone* encompasses material that is either a liquid or a solid, the extent of the *Zone* can be determined from the total mass of material within the *Zone* and from the density of the material.

$$V^Z = \frac{M^Z}{\rho^Z} \qquad [C.7]$$

$$\rho^Z = f\left(T^Z, P^Z, x_i{}^Z\right) \qquad [C.8]$$

In Eq. C.7, M refers to the mass of material.

If the *Zone* encompasses a single-component phase, the mass of material within the *Zone* can be determined from an overall lumped mass balance. The overall lumped mass balance is derived by applying the principle of conservation to the total mass of material that is contained within the *Zone*. In the absence of nuclear reactions, mass can neither be created nor destroyed.

$$\frac{dM^Z}{dt} = \sum_{P=1}^{NPB} F^P + \sum_{P=1}^{NPD}\sum_{i=1}^{NC} F_i^P \qquad [C.9]$$

In Eq. C.9 NPB and NPD refers to the number of *Ports* that experience bulk flow of material (i.e. are linked to an *Open Material Connection*) and diffusive flow of material (i.e. are linked to a *Closed Material Connection*) respectively. F_i^P is the mass flowrate of component i through *Port* P and NC is the number of component species that are transferred by the *Closed Connection*. If the

material within the *Zone* contains more than one component, Eq.C.9 is not used, since the total mass of the *Zone* is determined within the *Composition Subclass*.

C.3 The Composition Subclass

The composition of the phase within the *Zone* can be assigned a constant set of values or can be determined as part of the global solution.

Case 1 – Constant Composition: For the particular case where the composition of the phase does not vary with time, the mass fraction of each component species should be declared.

$$x_i^{\,z} = const \qquad [C.10]$$

Case 2 – Variable Composition: For a system that encompasses a multi-component phase, the law of conservation can be applied to each component within the system. The mass balance for each component within the *Zone* is:

$$\frac{dM_i^{\,z}}{dt} = \sum_{P=1}^{NPB} F^P x_i^P + \sum_{P=1}^{NPD} F_i^P + V^z R_i^{\,z} \qquad [C.11]$$

where R_i is the volumetric reaction source term for component i.

Constitutive Equations: The constitutive equations expand certain terms in the conservation equations by expressing them as functions of the state variables of the system. For Eq. C.11 the source term, which represents the rate of reaction, is expressed in terms of the product of the individual reaction rates and the stoichiometric coefficients.

$$R_i^{\,z} = \sum_{j=1}^{NR} \upsilon_{i,j} r_j \qquad [C.12]$$

where υ is the stoichiometric coefficient, r_j is the rate of reaction for reaction j, subscript i refers to component i and NR represent the number of reactions. It is assumed that the user will declare both the stoichiometric coefficients and the rate equations for each reaction. The mass fraction of

each component is determined from the component mass of each species and the total mass of material within the *Zone*.

$$x_i^{\,z} = \frac{M_i^{\,z}}{M^z} \qquad [C.13]$$

The total mass of material within a multicomponent *Zone* is determined by summing the masses of each component within the *Zone*.

$$M^z = \sum_{i=1}^{NCM} M_i^{\,z} \qquad [C.14]$$

In Eq. C.14, NCM refers to the number of components.

C.4 The Temperature Subclass

The temperature of the phase within the *Zone* can be declared as a constant value or can be determined as part of the solution.

Case 1: Constant Temperature (Isothermal): For this case, the user is required to provide the temperature of the *Zone* as part of the physical description.

$$T^z = const \qquad [C.15]$$

Since outlet *Connections* require information concerning the enthalpy of the phase within the *Zone*, it is necessary to relate the enthalpy (h) to the temperature, pressure and composition of the *Zone*.

$$h^z = f\left(T^z, P^z, x_i^{\,z}\right) \qquad [C.16]$$

Case 2: Variable Temperature: For the case where the temperature of the *Zone* varies, a lumped energy balance is used to calculate the temperature of the *Zone*. To ensure that a proper treatment of the energy balance occurs, all forms in which the material may hold energy are considered.

These are; the internal energy of the material resulting from the random molecular movement and the molecular interactions of the material, the kinetic energy, which represents the bulk movement of the material, and the potential energy that any external force field exerts on the material. In addition to the convective input and output, energy can also be added or removed from the system. This form of energy transfer may arise from heat that is transferred to the *Zone* or as work that the *Zone* does against the surrounding environment. It should be noted that any heat generated by a reaction is accounted for by the change in the internal energy of the phase. The energy balance can be formulated by combining the terms together so that the conservation law holds and by assuming that the accumulation of kinetic and potential energy is negligible compared with the accumulation of thermal energy:

$$\frac{d}{dt}M^Z u^Z = \sum_{P=1}^{NPB} F^P \left[h^P + \frac{1}{2}\left(v^P\right)^2 + gZ^P \right] + \sum_{P=1}^{NPD} Q^P + Q^Z - W^Z \qquad [C.17]$$

In Eq. C.17, u is the internal energy, v is the velocity, Z^P is the vertical position of Port P relative to the datum, Q is the heat added to the Zone, either directly (superscript Z) or through a Port (superscript P), and W is the total work done on the *Zone*. NPB refers to the number of *Ports* that experience the bulk flow of material (i.e. are linked to a Single Phase Open Connection) and NPD refer to the number of *Ports* that experience the diffusive flow of energy (i.e. are linked to a Closed Energy Connection). It is beneficial to separate the total work term into two contributions: the first contribution arises from the motion of the *Zone's* boundary against the force exerted on the *Zone's* surface by the surroundings; the second contribution arises from the mechanical work that is transmitted from the *Zone* (W^M).

$$\frac{d}{dt}M^Z u^Z = \sum_{P=1}^{P=NPB} F^P \left[h^P + \frac{1}{2}\left(v^P\right)^2 + gZ^P \right] + \sum_{P=1}^{NPD} Q^P + Q^Z - W^M - P^Z \frac{dV^Z}{dt} \qquad [C.18]$$

For a *Zone* of fixed extent the volume of the *Zone* remains constant and hence the right hand term can be omitted from Eq. C.18.

$$\frac{d}{dt}M^Z u^Z = \sum_{P=1}^{P=NPB} F^P \left[h^P + \frac{1}{2}\left(v^P\right)^2 + gZ^P \right] + \sum_{P=1}^{NPD} Q^P + Q^Z - W^M \qquad [C.19]$$

For a *Zone* of variable extent, the work term can be combined with the accumulation of internal energy to yield:

$$\frac{d}{dt}M^Z h^Z - V^Z \frac{d}{dt}P^Z = \sum_{P=1}^{P=NPB} F^P\left[h^P + \frac{1}{2}\left(v^P\right)^2 + gZ^P\right] + \sum_{P=1}^{NPD} Q^P + Q^Z - W^M \qquad [C.20]$$

Model Completeness: Since outlet *Connections* require information concerning the enthalpy of the phase within the *Zone*, for the case where the extent of the *Zone* is fixed, it is necessary to use an equation of state to relate the enthalpy of the *Phase* to its state variables.

$$h^Z = f\left(T^Z, P^Z, x_i{}^Z\right) \qquad [C.21]$$

For the case where the *Zone* consists of either a gaseous phase of constant pressure or a liquid or solid phase of both constant extent and constant pressure, it is necessary to determine the density of the phase from its three state variables.

$$\rho^Z = f(T^Z, P^Z, x_i{}^Z) \qquad [C.22]$$

For the particular case where the extent, the composition and the pressure of the *Zone* are constant throughout the simulation it is also necessary to calculate the total mass of the *Zone* from the *Zone's* extent.

$$M^Z = \rho^Z V^Z \qquad [C.23]$$

If the kinetic energy transfer at the *Connections* must be considered, then it is necessary to relate the velocity at each *Port* to the mass flowrate of the *Port*, the cross sectional area of the *Port* and the material density of the phase.

$$F^P = \rho^P v^P A^P \qquad [C.24]$$

$$\rho^P = f\left(T^P, P^P, x_i^P\right) \qquad [C.25]$$

Constitutive Equations: Constitutive equations are needed to describe the work done by (or on) the *Zone*. It is proposed that a library of the most common forms (e.g. a compressor) should be used. The heat that is transferred between the *Zone* and its environment will be determined through the *Connection* entity.

Equations of State: To complete the model, a constitutive equation is needed to relate the internal energy to the state variables.

$$u^z = f\left(T^z, P^z, x_i^z\right) \qquad [C.26]$$

C.5 The Pressure Subclass

The pressure of the *Zone* may be assigned a constant value or can be determined as part of the global solution.

Case 1: Constant Pressure:

$$P^z = const \qquad [C.27]$$

Case 2: Variable Pressure – Gaseous Phase: If the *Phase* within the *Zone* is gaseous, an equation of state can be used to determine the pressure of the *Zone* from the density of the *Zone*. The density can be determined from knowledge of the total mass of material within the *Zone* and from the *Zone's* extent.

$$\rho^z = f(T^z, P^z, x_i{}^z) \qquad [C.28]$$

$$\rho^z = \frac{M^z}{V^z} \qquad [C.29]$$

If the material within the *Zone* consists of a single component, a mass balance applied to the total mass of the single-component phase is required to evaluate M^Z.

$$\frac{dM^z}{dt} = \sum_{P=1}^{NPB} F^P + \sum_{P=1}^{NPD}\sum_{i=1}^{NC} F_i^P \qquad [C.30]$$

Model Completeness: If the *Vessel* is closed and contains more than one *Zone*, it is necessary to equate the pressures of each compressible *Zone* within the vessel through the *Pressure Connection*.

Case 3: Variable Pressure – Liquid and Solid Phases: For each *Zone* that encompasses material that is incompressible, the pressure of the *Zone* is determined from the pressure of the *Zone's* environment established through a *Pressure Connection*.

Appendix D

The Single-Phase, One-dimensional, Duct Zone Model Class

The *Single-phase, One-dimensional, Duct Zone Model Class* is used to model a *Zone* where the thermodynamic state of the material within the *Zone* varies over the length of the *Zone*. Similar to the *Single-phase Lumped Zone Model Class*, the equations within the *Model Class* are structured to enable each state variable either to be specified explicitly or to be determined as part of the global solution. A *model subclass* exists for each key state variable and contains all the necessary equations and model logic to determine that state variable.

The current prototype software is restricted to modelling *Zones* of constant cross sectional area, however the length of the *Zone* may vary with time. To permit the variation in the length of the *Zone* with time, the generic conservation equations are expressed in their normalised form. Certain terms in the normalised conservation equations are zero for *Zones* of fixed size. The derivation of the normalising transformations can be found in Appendix G. The cross sectional shape of the duct is arbitrary and is characterized by its area A and wetted perimeter P_w. The ends of the duct are referred to as axial boundaries and the sides as a single radial boundary.

D.1 The Port Subclass

The interface between the *Zone* and its environment is specified through the *Zone's Boundaries*. If there is an exchange of either mass or energy across this *Boundary*, it is necessary to attach a *Port* to the *Boundary* to enable the exchange of mass and energy to be specified through a *Connection*. If there is no *Port* attached to the *Boundary*, the framework assumes that the *Boundary* is fully insulated and consequently there is no exchange of either mass or energy across it.

The criteria for establishing the *Port* equations are such that full consideration is given to both the bulk flow of material and the diffusive transfer of material and energy through the *Port*. To enable the various combinations of bulk flow and diffusive transport to be considered, the bulk flow and the diffusive transfer components at each *Port* are determined independently. If there is bulk flow through the *Zone*, there is a clear direction to the flow of material and consequently there is an inlet *Port* associated with the flow of material into the *Zone* and an outlet *Port* associated with the flow of material leaving the *Zone*. For each outlet *Port* subject to the bulk flow of material,

$$F^P = A^Z v^Z \rho^Z \qquad x_i^P = x_i^Z \qquad h^P = h^Z \qquad \text{[D.1] - [D.3]}$$

where F^P is the total bulk flow of mass through *Port* P, A is the cross sectional area, v is the axial velocity, x_i is the mass fraction of component i and h is the enthalpy. If the *Port* is attached to a *Closed Material Connection* or a *Closed Energy Connection*, Eqs. D.4 and D.5 are generated respectively. This is applicable to all *Ports*, regardless of whether the *Port* is located on an axial or a radial boundary.

$$\rho_i^P = \rho^Z x_i^Z \qquad \text{[D.4]}$$

$$T^P = T^Z \qquad \text{[D.5]}$$

where ρ is the density and T is the temperature. For cases where the *Port* is subject to either the diffusive transfer of mass or energy, the diffusive transfer can be deduced from the appropriate flux at the boundary and the boundary cross sectional area.

$$F_{D,i}^P = A^Z j_{i,z}^Z \qquad Q_D^P = A^Z q_{i,z}^Z \qquad \text{[D.6] - [D.7]}$$

It should be noted that to ensure the generation of a well posed model, Eqs. D.6 and D.7 are generated only for axial *Ports* that experience thermal or material diffusion and lie upstream of the *Connection*. The exception to this rule is the case where material or energy flows from a *Zone* that experiences thermal or material diffusion through an *Open Material Connection* into a second *Zone* that also experiences thermal or material diffusion. For this case Eqs. D.6 and D.7 are also generated for the downstream *Port*.

To maintain full continuity in the pressure field, the pressure of each *Port* (P^P) attached to an axial boundary is equated to the pressure of the *Zone* (P^Z) at the boundary.

$$P^P = P^Z \qquad [D.8]$$

D.2 The Core Equation Subclass

The *Core Equation Subclass* contains the equations that are common to a number of the other *subclasses*. To prevent the repetition of the same equation, it is prudent to automatically generate the appropriate equation once within this section.

The Phase Density: The density of the phase is related to the state variables of the phase through an equation of state.

$$\rho^Z = f\left(T^Z, P^Z, x_i^Z\right) \qquad [D.9]$$

D.3 The Extent Subclass

The *Zone's* extent is determined from the *Zone's* cross sectional area and the *Zone's* axial boundaries (Z_1 and Z_2).

$$V^Z = A^Z\left(Z_2 - Z_1\right) \qquad [D.10]$$

It is assumed that the user will specify the cross sectional area as a model parameter. The axial boundary positions may be assigned a fixed co-ordinate value or the user may provide an equation that can be used to determine the position of the axial boundary as a function of time.

D.4 The Composition Subclass

The composition of the *Zone* may be declared as remaining constant or determined as part of the global solution.

Case 1, Constant Composition: If the composition (x_i) of the *Zone* remains constant throughout the time horizon, the user should declare the composition of the phase.

$$x_i^Z = const \qquad [D.11]$$

Case 2, Variable Composition: If the composition of the phase varies, the normalised, component-mass conservation equation is generated.

$$\left[\frac{\hat{z}-1}{(Z_2-Z_1)}\frac{dZ_1}{dt}-\frac{\hat{z}}{(Z_2-Z_1)}\frac{dZ_2}{dt}\right]\frac{\partial\rho_i^Z}{\partial\hat{z}}+\frac{\partial\rho_i^Z}{\partial t}+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial\hat{z}}v^Z\rho_i^Z+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial\hat{z}}j_i^Z-R_i^Z-\frac{P_W\cdot\Gamma_i^P}{A^Z}=0 \qquad [D.12]$$

In Eq. D.12, ρ_i is the mass concentration of component i, v is the axial velocity, R_i is the rate of reaction, j_i is the diffusive mass flux of component i and Γ_i^P is the radial mass flux of component i through *Port* P.

The Constitutive Equations: Constitutive equations are needed to evaluate the terms for both the reaction source and the diffusive mass flux. For the diffusive mass flux term, it is assumed that Fick's first law can be used.

$$j_i^Z=\frac{-D_i}{(Z_2-Z_1)}\frac{\partial\rho_i^Z}{\partial\hat{z}} \qquad [D.13]$$

In Eq. D.13, D_i is the diffusion coefficient for component i. The total source term for chemical reaction (generation/depletion) can be expressed as the sum of the individual reaction source terms for that component. For each reaction that occurs, the source term is equal to the product of the stoichiometric coefficient and the rate of reaction.

$$R_i^Z=\sum_{j=1}^{j=NR}\upsilon_{ij}r_j \qquad [D.14]$$

The user must declare the reaction kinetics for each reaction that occurs. The mass fraction of each species can be calculated by dividing the mass concentration of each species by the total density.

$$x_i^z = \frac{\rho_i^z}{\rho^z} \qquad [D.15]$$

The Boundary Conditions: If there is bulk flow of material through the *Zone*, the inlet *Boundary* corresponds to the point at which material enters the *Zone* and the outlet boundary corresponds to the boundary at which material leaves the *Zone*. The inlet boundary condition arises by applying the principle of mass conservation to the inlet. If the mass flowrate of component i, associated with *Port* P is F_i^P, then conservation of mass at the inlet gives:

$$F_i^P = A^Z v^Z \rho_i^Z + A^Z j_i^Z \qquad [D.16]$$

$$F_i^P = F^P x_i^P + F_{D,i}^P \qquad [D.17]$$

By disregarding terms within Eqs. 16 and 17, the appropriate boundary condition can be formulated for the case where there is no diffusion ($j_i = 0$, $F^P{}_{D,i} = 0$), and no bulk flow or diffusion ($v = 0$, $F^P = 0$, $j_i = 0$, $F^P{}_{D,I} = 0$). In certain instances it is possible that a specific component may be present within the *Zone* but not within a *Connection* that has been linked to the *Zone*. If this *Connection* is linked to an inlet *Boundary*, then it is necessary to declare the inlet mass flowrate of the component to be zero.

$$F_i^P = 0 \qquad [D.18]$$

The output boundary conditions are only required if there is diffusive mass transfer. If a *Port* is connected to the outlet boundary, the *Port Subclass* will formulate the boundary condition. Alternatively, if there is no *Port* attached to an axial boundary, the following boundary condition is generated.

$$j_i^z = 0 \qquad [D.19]$$

D.5 The Temperature Subclass:

The temperature of the *Zone* may be declared as remaining constant or can be determined as part of the global solution.

Case 1, Constant Temperature: If the temperature of a specific phase remains constant throughout the time horizon, the user can declare the temperature explicitly.

$$T^z = const \qquad [D.20]$$

For the bulk flow of material between two *Zones*, the enthalpy of the upstream *Zone* must be known. For this reason, an equation of state is generated to ensure that, where necessary, the appropriate *Connection* has access to all the required thermodynamic data.

$$h^z = f(P^z, T^z, x_i^z) \qquad [D.21]$$

Case 2, Variable Temperature: The energy balance is derived by applying the first law of thermodynamics to an open and unsteady *system*. The energy balance considers both the kinetic energy, associated with the fluid motion, and the internal energy, associated with the random translational and internal motion of the molecules. The kinetic energy associated with turbulent eddies is neglected and the potential energy is not treated as a conserved quantity but instead is included within the work term. The term representing the work done consists of two parts, the work against the volume forces (gravity) and the work against the surface forces (Pressure). The latter is included in the enthalpy flux term. For a 1-D Zone, the heat added to the *System* occurs through the *Vessel* wall and is represented by a wall heat flux.

The 1-D energy balance, expressed over a normalised domain is:

$$\left[\frac{\hat{z}-1}{(Z_2-Z_1)}\frac{dZ_1}{dt}-\frac{\hat{z}}{(Z_2-Z_1)}\frac{dZ_2}{dt}\right]\frac{\partial \rho^z E^z}{\partial \hat{z}}+\frac{\partial \rho^z E^z}{\partial t}+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}\left[\rho^z v^z H^z\right]+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}q^z-\frac{q^P P_w^z}{A^z}-v^z\rho^z gb=0 \qquad [D.22]$$

Here E is the total energy per unit volume, H is the combined enthalpy and kinetic energy, q is the axial heat flux, q^P is the radial heat flux through *Port* P, P_w is the wetted perimeter and A is the cross sectional area. The term b is used to account for the inclination of the *Zone*. The total energy per unit volume represents both the kinetic energy and the internal energy of the phase.

Constitutive Equations: Two algebraic equations are required to decompose the total internal energy / enthalpy into the internal energy and kinetic energy components.

$$E^z = u^z + \frac{v^{z^2}}{2} \qquad \text{and} \qquad H^z = h^z + \frac{v^{z^2}}{2} \qquad \text{[D23] – [D24]}$$

where u is the internal energy and h is the enthalpy.

Fourier's equation is used to express the diffusive heat flux in terms of the temperature gradient.

$$q^z = -\frac{k}{(Z_2 - Z_1)} \frac{\partial T^z}{\partial \hat{z}} \qquad \text{[D25]}$$

where k is the thermal conductivity of the material. The scalar term (b) describing the inclination of the *Zone* can be expressed in terms of the angle of the *Zone* to the horizontal.

$$b = \sin \vartheta \qquad \text{[D26]}$$

where θ is the angle of inclination relative to the horizontal plane.

Equations of State: Equations of state are needed to relate both the internal energy and the enthalpy of the phase to the pressure, temperature and composition.

$$u^z = f(T^z, P^z, x_i^z) \qquad \text{[D.27]}$$

$$h^z = f(T^z, P^z, x_i^z) \qquad \text{[D.28]}$$

The Boundary Conditions: The inlet boundary condition arises by applying the principle of energy conservation to the inlet. If the energy flowrate associated with the Port is Q^P, then conservation of energy at the inlet gives.

$$Q^P = \rho^z v^z A^z \left(h^z + \frac{(v^z)^2}{2} \right) + A^z q^z \qquad \text{[D.29]}$$

$$Q^P = F^P h^P + Q_D^P \qquad \text{[D.30]}$$

Where Q^P is the flow of energy through the *Port*. By disregarding terms within Eqs. D.29 and D.30, an appropriate boundary condition can be formulated for the case where there is no diffusion ($q = 0$, $Q_D{}^P = 0$) and for the case where there is no flow ($v = 0$, $F^P = 0$). The output boundary conditions are required only if there is diffusive heat transfer. If a *Port* is connected to the outlet boundary, the Port Subclass will formulate the appropriate boundary condition. Alternatively, if there is no *Port* attached to the outlet axial boundary, the following boundary condition is generated.

$$q^z = 0 \qquad [D.31]$$

It should be noted that for the case where there is no bulk flow, the direction of energy flow is unknown and hence the boundaries cannot be identified as "inlet" or "outlet". For this case the first boundary is treated as the "inlet" and the second boundary is treated as the "outlet".

D.6 The Pressure and Velocity Subclass

Since the equations required to calculate the pressure and the velocity of a compressible phase are nonlinear and coupled, it is necessary to determine these two key attributes simultaneously within the Pressure and Velocity Subclass. Within the *Pressure and Velocity Subclass* the equations are structured according to a number of specific cases that correspond to the physical description provided by the user. The cases are as follows.

Case 1: Material is Stationary: For the material within the Zone to be fully stationary, there must be no bulk flow of material (forced flow) and no net (diffusive) flow of component mass through the *Zone*. For this case the velocity of the phase can be set to zero and the bulk flow terms within the appropriate mass and energy balances can be removed from the respective conservation equations.

$$v^z = 0 \qquad [D.32]$$

If the material is stationary, the pressure may be declared by the user or determined with a (hydrostatic) momentum balance.

$$P^z = \text{const} \qquad [D.33]$$

$$\frac{1}{(Z_2 - Z_1)} \frac{\partial P^Z}{\partial \hat{z}} - \rho^Z gb = 0 \qquad [D.34]$$

Case 2: Diffusive Flow of Component Mass and no Bulk Flow: As Fick's law predicts the diffusive mass flux relative to the mixture velocity (Bird et al., 2002), it is necessary to determine the mixture velocity for the case where there is diffusion yet no net bulk flow of mass through the *Zone*. In this particular instance, the mixture velocity is the sum of the mass averaged velocity for each component within the *Zone*. The mass averaged velocity for an arbitrary component is not the instantaneous molecular velocity of a molecule of the component but the arithmetic average of the velocities of all the molecules of the component within a volumetric element. The mixture velocity is determined from:

$$v^Z = \sum_{i=1}^{i=NC} x_i^Z v_i^Z = \frac{\sum_{i=1}^{i=NC} n_i^Z}{\rho^Z} \qquad [D.35]$$

where n_i is the combined mass flux of component i. The combined mass flux of component i can be expressed in terms of the diffusive mass flux and the convective mass flux.

$$n_i^Z = j_i^Z + \rho_i^Z v^Z \qquad [D.36]$$

In many instances the mass flux associated with the mass averaged velocity is several orders of magnitude less than the diffusive mass flux. If this is the case, then the mass averaged velocity can be set to zero by declaring the material to be stationary. As in case 1, the pressure may be declared by the user or determined with a momentum balance. In the latter case, Eq. D.34 is again.

Case 3: Bulk Flow of Material: If there is bulk flow of material through the *Zone* it is necessary to couple the principle of momentum conservation with the principle of mass conservation. It is also necessary to distinguish between a compressible phase and an incompressible phase to ensure that the pressure and density are not coupled for the case where there is no interdependency.

Case 3a: Incompressible Zone: For an incompressible phase, the velocity of the phase can be deduced from the continuity requirements of the *Zone*. For instances where the density variation

along the Zone's axis can be ignored the velocity at any point within the *Zone* is equal to the superficial velocity.

$$v^Z = \frac{F^P}{A^Z \rho^P} \qquad [D.37]$$

$$\rho^P = f\left(T^P, P^P, x_i^P\right) \qquad [D.38]$$

$$h^P = f\left(T^P, P^P, x_i^P\right) \qquad [D.39]$$

If there is a considerable density gradient within the *Zone*, the velocity can be deduced from the principle of mass conservation. If the composition of the phase remains constant, and has been declared by the user, the density of the phase cannot vary with time and hence the principle of mass conservation dictates that:

$$\frac{1}{(Z_2 - Z_1)} \frac{\partial}{\partial \hat{z}} v^Z \rho^Z - \frac{P_W \Gamma_i^P}{A} = 0 \qquad [D.40]$$

The following boundary condition is generated.

$$F_B^P = A^Z v^Z \rho^Z \qquad [D.41]$$

If the composition does vary, the density of the phase (determined within the core subclass) is related to the sum of the mass component concentrations. This makes one of the component mass balances redundant and consequently this component mass balance can be used to determine the velocity profile through the *Zone*.

$$\rho^Z = \sum_{i=1}^{NC} \rho_i^Z \qquad [D.42]$$

where NC is the number of components. Since the pressure is independent of the density of the phase, the pressure may either be declared by the user:

$$P^Z = \text{const} \qquad [D.43]$$

or determined with a momentum balance that is based on the same quasi-steady state assumption used by Barber (2000).

$$\frac{\partial}{\partial z}\rho^{Z}\left(v^{Z}\right)^{2}+\frac{1}{\left(Z_{2}-Z_{1}\right)}\frac{\partial P^{Z}}{\partial \hat{z}}-\rho^{Z}gb+\frac{\tau_{w}P_{w}}{A^{Z}}=0 \qquad \text{[D.44]}$$

where τ_w is the wall shear stress. The current scope of the framework requires the user to specify a suitable correlation for the wall shear stress. The boundary condition for Eq. D.44 has already been formulated within the *Port Subclass*.

Case 3a: Compressible Zone: If the *Zone* encompasses material that is compressible, the velocity of the *Zone* can be deduced from the momentum balance.

$$\left[\frac{\hat{z}-1}{\left(Z_{2}-Z_{1}\right)}\frac{dZ_{1}}{dt}-\frac{\hat{z}}{\left(Z_{2}-Z_{1}\right)}\frac{dZ_{2}}{dt}\right]\frac{\partial\rho^{Z}v^{Z}}{\partial\hat{z}}+\frac{\partial\rho^{Z}v^{Z}}{\partial t}+\frac{1}{\left(Z_{2}-Z_{1}\right)}\frac{\partial}{\partial\hat{z}}\left[\rho^{Z}v^{Z}v^{Z}\right]-\rho^{Z}gb+\frac{1}{\left(Z_{2}-Z_{1}\right)}\frac{\partial P^{Z}}{\partial\hat{z}}+\frac{\tau_{w}P_{w}}{A^{Z}}=0 \qquad \text{[D.45]}$$

For instances where the composition of the phase varies, the density of the phase is related to the sum of the component mass concentrations through Eq. D.42. If the composition of the *Zone* remains constant, the density of the phase has yet to be determined and it is therefore necessary to determine the density with an overall mass balance.

$$\left[\frac{\hat{z}-1}{\left(Z_{2}-Z_{1}\right)}\frac{dZ_{1}}{dt}-\frac{\hat{z}}{\left(Z_{2}-Z_{1}\right)}\frac{dZ_{2}}{dt}\right]\frac{\partial\rho^{Z}}{\partial\hat{z}}+\frac{\partial\rho^{Z}}{\partial t}+\frac{1}{\left(Z_{2}-Z_{1}\right)}\frac{\partial}{\partial\hat{z}}\rho^{Z}v^{Z}-\frac{P_{W}\Gamma_{i}^{P}}{A^{Z}}=0 \qquad \text{[D.46]}$$

Boundary Conditions: Since the pressure at each Axial Boundary is equated to the pressure of the appropriate *Port*, the boundary condition for Eq. D.45 has already been formulated. The boundary condition for Eq. D.46 arises by applying the principle of continuity to the total mass of the phase at the inlet.

$$F_{B}^{P}=A^{Z}v^{Z}\rho^{Z} \qquad \text{[D.47]}$$

Appendix E

The Two-phase, Lumped Zone Model Class

The *Two-phase, Lumped Zone Model Class* is used to model a *Zone* that encompasses material that is in two stable thermodynamic conditions. Further, the state variables and the volume fraction are uniform over the extent of the *Zone*. The *subclasses* for the *Two-phase, Lumped Zone Model Class* describe the thermodynamic properties of the two phases (composition, temperature, pressure), the volume fraction and the velocity of each phase and the extent of the two-phase *Zone*.

To enable the framework to permit the state variables of a specific phase to be determined either independently or collectively with the second phase, conservation equations are written either for each phase (heterogeneous balance equations) or for the mixture as a whole (homogeneous balance equation). Since each heterogeneous balance equation is used to determine a specific state variable for one of the phases, two heterogeneous balance equations are required to model a particular key aspect of the *Zone* (e.g. temperature, composition). For the homogeneous balance the two phases are modelled as a single mixture or pseudo phase, represented by a single set of state variables. Consequently only one conservation equation is needed. To enable the use of either method to determine the appropriate state variable(s) both sets of equations are contained within the same *subclass*. For the particular case where the state variable does not vary through the spatial and time domains, each *subclass* permits the appropriate balance equation to be omitted, and a value assigned for the key attribute.

E.1 The Port Subclass

For each *Port* connected to a *Single Phase Open Connection* that transfers material in phase k away from the *Zone*, it is necessary to relate the mass fraction (x_i) and the enthalpy (h) of the *Port* to the respective state variables of the *Zone*.

$$x_i^P = x_{k,i}^Z \qquad h^P = h_k^Z \qquad \text{[E.1] – [E.2]}$$

Further, regardless of the direction of flow, it is also necessary to relate the pressure at the *Port* (P^P) to the pressure of the *Zone* (P^Z). Depending on the physical description provided by the user, this may or may not include the hydrostatic pressure of the *Zone*.

$$P^P = P^Z + \rho_m^Z g\left(\frac{V^Z}{A^Z} - Z^P\right) \qquad \text{[E.3]}$$

where ρ is the density, g is the gravitational acceleration and Z is the height of the port relative to the pressure datum for the *Zone*. For each *Port* linked to a *Closed Material Connection* it is necessary to relate the mass concentrations of the component species at the *Port* to the mass concentrations within the *Zone*.

$$\rho_i^P = \rho_k^Z x_{k,i}^Z \qquad \text{[E.4]}$$

For each *Port* linked to a *Closed Energy Connection*, it is necessary to relate the temperature of the *Port* (T^P) to the temperature of the appropriate phase within the *Zone* (T^Z).

$$T^P = T_k^Z \qquad \text{[E.5]}$$

If there is relative motion between the two phases within the *Zone*, the *Port* entity can provide the necessary equations to determine the mass flowrate of a specific phase through the *Port*. The equations used to determine the mass flowrate at a *Port* consider both the relative motion of the phases and the motion of the *Boundary* that the phase is flowing through. Since the framework does not require the *Boundary* of the *Zone* to be defined, the motion of the *Port* is declared instead. The current scope of the framework only permits relative movement between the two phases for a vertically orientated *Zone*, and consequently Eq. E.6 is applied only if the *Zone* orientation has been declared as vertical.

$$F^P = \varepsilon_k \rho_k \left| v_k - v^P \right| A^Z \qquad \text{[E.6]}$$

In Eq. E.6, F^P is the mass flowrate associated with *Port* P, ε_k is the volume fraction of phase k, v_k is the velocity of phase k, v^P is the velocity of the moving *Port* and A^Z is the cross sectional area of the *Zone*. In the vertical co-ordinate system the downward direction is positive and the upward direction is negative. More information concerning the coordinate system can be found in section E.8. The velocity of the *Port* may be declared to be zero, equated to the velocity of the dispersed phase (v_d) or determined with the use of a user-defined subroutine.

E.2 The Core Equations Subclass

To streamline the construction of the equations, those that occur within several *subclasses* have been extracted from the appropriate *subclasses* and are declared within this section.

Calculating the Mass of each Phase:

Case 1, Variable Composition: If the composition of the phase varies, the total mass of the phase (M_k) is determined by summing the component mass hold-ups ($M_{k,i}$) determined within the *Composition Subclass*.

$$M_k^Z = \sum_{i=1}^{NCM} M_{k,i}^Z \qquad [E.7]$$

Case 2, Constant Composition, Constant Extent, Constant Volume Fraction and the phases within the *Zone* are fully incompressible: For this case, the total mass of the phase remains constant throughout the time horizon and can be determined from Eq. E.8.

$$M_k^Z = \varepsilon_k^Z \rho_k^Z V^Z \qquad [E.8]$$

Case 3: Constant Composition, Pressure / Temperature Isotherm: For the case where both phases are at the boiling temperature, any change in the pressure or temperature of the *Zone* will result in either evaporation or condensation of one of the phases. As it is very difficult to model the transfer of mass that results from either condensation or evaporation, the mass balance is written for the two-phase mixture as a whole.

$$\frac{dM^Z}{dt} = \sum_{P=1}^{NP} F^P \qquad [E.9]$$

The second equation that is generated for Case 3, is the boiling isotherm. Coupling the boiling isotherm with the energy balance and Eq. E.9, allows the mass of each phase to be determined.

$$P^Z = f(T^Z) \qquad [E.10]$$

Case 4, For all other cases: For all other cases it is necessary to determine the total hold-up of the phase with the use of an overall mass balance.

$$\frac{dM_k}{dt} = \sum_{P=1}^{NPB} F^P + \sum_{P=1}^{NPD}\sum_{i=1}^{NC} F_i^P \pm V^Z \Gamma_{IC}^Z \qquad [E.11]$$

where Γ_{IC} is the rate of interphase mass transfer per unit volume. Depending on the direction of mass transfer specified by the user, the interfacial mass transfer is either positive or negative. NPD refers to the number of *Ports* that experience diffusive flow of material (i.e. are linked to a *Closed Material Connection*) respectively. NC refers to the number of components that are transported through Port P. NPD refers to the number of ports subject to diffusive mass transfer.

Calculating The Mass Fraction of Each Phase: The mass fraction is calculated straightforwardly.

$$M^Z = \sum_{k=1}^{2} M_k^Z \qquad X_k^Z = \frac{M_k^Z}{M^Z} \qquad [E.12] - [E.13]$$

Calculating the Density of Each Phase: The density of the phase is expressed as a function of the state variables for that phase.

$$\rho_k^Z = f(T_k^Z, P_k^Z, x_{k,i}^Z) \qquad [E.14]$$

E.3 The Extent Subclass

The extent of the two-phase *Zone* may be assigned a constant value or can be determined as part of the global solution.

Case 1: Constant Extent

$$V^z = const \qquad [E.15]$$

Case 2: Variable Extent: If the extent of the *Zone* varies it is necessary to relate the extent of the *Zone* to the density and the total mass of each phase.

$$V^z = \sum_{k=1}^{2} \frac{M_k^z}{\rho_k^z} \qquad [E.16]$$

E.4 The Composition Subclass

For each phase, the composition may be declared to remain constant, in which case the composition of the phase should be declared by the user, or may vary during the time horizon.

Case 1, Constant Composition: If the composition of a specific phase remains constant throughout the time horizon, the user should declare the composition.

$$x_{k,i}^z = const \qquad [E.17]$$

Case 2, Variable Composition – No Chemical Equilibrium: If there is no chemical equilibrium between the two phases within the *Zone*, the composition of each phase can be determined independently by applying the law of conservation to each component within the phase.

$$\frac{dM_{k,i}^z}{dt} = \sum_{P=1}^{NPB} F^P x_i^P + \sum_{P=1}^{NPD} F_i^P + V^z \varepsilon_k^z R_{k,i}^z \pm V^z \Gamma_{IC,i}^z \qquad [E.18]$$

where $R_{k,i}$ is the reaction source term for component i within phase k. NPB refers to the number of *Ports* that are linked to the bulk flow of phase k and NPD refers to the number of *Ports* that are linked to the diffusive flow of phase k.

Constitutive Equations: If a reaction occurs within phase k, the source term representing the rate of reaction is expressed in terms of the product of the individual reaction rates and the stoichiometric coefficients.

$$R_{k,i}^{Z} = \sum_{j=1}^{NR} \upsilon_{k,i,j} r_{k,j} \qquad [E.19]$$

where $\upsilon_{k,i,j}$ is the stoichiometric coefficient of component i within phase k for reaction j, $r_{k,j}$ is the rate of reaction for reaction j within phase k and NR refers to the number of reactions. Subscript i refers to the component, i. It is assumed that the user will declare both the stoichiometric coefficients and the rate equations for each reaction. To enable the equations of state to be used, it is necessary to calculate the mass fraction of each species, within each phase.

$$x_{k,i}^{Z} = \frac{M_{k,i}^{Z}}{M_{k}^{Z}} \qquad [E.20]$$

Case 3, Variable Composition – Chemical Equilibrium: For the specific case where there is chemical equilibrium between the two phases within the *Zone*, the principle of component mass conservation is applied to the two-phase mixture as a whole rather than to each phase. The composition of the two phases can be related through the use of equilibrium constants. With this information and with knowledge of the total mass of each component within the two-phase mixture, the composition of each phase can be deduced. It should be noted that, for this case, each phase within the two-phase *Zone* must have the same set of components.

$$\frac{dM_{i}^{Z}}{dt} = \sum_{P=1}^{NPB} F^{P} x_{i}^{P} + \sum_{P=1}^{NPD} F_{i}^{P} + \sum_{k=1}^{2} V^{Z} \varepsilon_{k}^{Z} R_{k,i}^{Z} \qquad [E.21]$$

$$M_{i}^{Z} = \sum_{k=1}^{k=2} M_{k}^{Z} x_{k,i} \qquad [E.22]$$

$$x_{1,i}^{Z} = K_{i} x_{2,i}^{Z} \qquad [E.23]$$

$$x_{k,i}^{Z} = \frac{M_{k,i}^{Z}}{M_{k}^{Z}} \qquad [E.24]$$

where K_i is the equilibrium coefficient of component i. Subscript 1 corresponds to phase 1 and subscript 2 to phase 2. The equilibrium constants can be expressed as functions of the temperature, the pressure and the composition of either of the two phases.

$$K_i = f\left(T^Z, P^Z, x_{k,i}\right) \qquad [E.25]$$

If a reaction occurs within phase *k*, it is necessary to express the reaction source term in terms of the products of the individual reaction rates and the stoichiometric coefficients using Eq. E.19.

E.5 The Temperature Subclass

The *Temperature Subclass* permits the temperature of each phase to be declared to have a constant value or to be calculated with the use of an energy balance. To allow the temperature of a specific phase to be determined either independently or coupled with the second phase, an energy balance can be generated for each phase (heterogeneous balance equations) or for the mixture as a whole (homogeneous balance).

Case 1: Heterogeneous Phase Temperatures:

Case 1a: Declared Phase Temperature

$$T_k^Z = const \qquad [E.26]$$

To ensure that any outlet *Connection* can be connected to the *Zone*, it is necessary to determine the enthalpy of each phase within the *Zone*.

$$h_k^Z = f\left(T_k^Z, P^Z, x_{k,i}{}^Z\right) \qquad [E.27]$$

Case 1b: Variable Phase Temperature: The heterogeneous-phase energy balance is used to determine the temperature of each phase over the time horizon. The basic energy balance is the same as the single-phase energy balance. However, the interfacial transfer of heat, declared through the internal connection (IC) is also considered (Λ_{IC}). The interfacial transfer of heat is defined as the heat transfer from phase 2 to phase 1 and consequently Λ_{IC} is expressed as a

positive term for phase 1 and a negative term for phase 2. The energy balance is an expression of the first law of thermodynamics with consideration given to the heat that is transferred to the *Phase* from the environment (Q^P and Q^Z) and the work that the phase does against the environment (W). The energy balance is formulated by assuming that the accumulation of kinetic and potential energy is negligible compared with the accumulation of thermal energy. If the extent of the *Zone* is constant Eq. E.28 is generated. Alternatively if the extent of the *Zone* varies, Eq. E.29 is generated, with h_k replacing u_k.

$$\frac{d}{dt}\left(M_k{}^Z u_k^Z\right) = \sum_{P=1}^{NPB} F^P\left[h^P + \frac{1}{2}\left(v^P\right)^2 + gZ^P\right] + \sum_{P=1}^{NPD} Q^P + Q^Z \pm V^Z \Lambda^{IC} \pm V^Z \Gamma_{IC} h_k - W_k^Z \qquad \text{[E.28]}$$

$$\frac{d}{dt}\left(M_k{}^Z h_k^Z\right) - V^Z \frac{d}{dt} P^Z = \sum_{P=1}^{NPB} F^P\left[h^P + \frac{1}{2}\left(v^P\right)^2 + gZ^P\right] + \sum_{P=1}^{NPD} Q^P + Q^Z \pm V^Z \Lambda^{IC} \pm V^Z \Gamma_{IC} h_k - W_k^Z \qquad \text{[E.29]}$$

In Eqs. [E.28] and [E.29] NPB refers to the number of *Ports* linked to the bulk transfer of phase k and NPD refers to the number of *Ports* linked to the diffusive flow of heat into phase k.

Constitutive Equations: Constitutive equations are needed to describe the work done on the *Zone*. It is proposed that a library of the most common forms should be used. The heat that is transferred between the phase and its environment will be determined through the *Connection* entity. The velocity of the material at each *Port* can be calculated as follows:

$$v^P = \frac{F^P}{\rho^P A^P} \qquad \text{[E.30]}$$

Equations of State: To complete the model, an equation of state is required for each phase to relate the enthalpy or the internal energy (for the case where the extent of the *Zone* is constant) to the state variables of the phase. To ensure that all of the interlinked *Connections* have access to the appropriate information of the *Zone* entity, Eq. E.31 is always generated, even if the extent of the *Zone* is constant.

$$h_k^Z = f(T_k^Z, P^Z, x_{k,i}^Z) \qquad \text{[E.31]}$$

$$u_k^Z = f(T_k^Z, P^Z, x_{k,i}^Z) \qquad \text{[E.32]}$$

Case 2: Homogeneous Mixture Temperature:

Case 2a: Declared mixture temperature:

$$T^Z = const \qquad [E.33]$$

where T^Z is the temperature of the Zone (two-phase mixture). To ensure that any outlet *Connection* can be connected to the *Zone*, it is necessary to determine the enthalpy of each phase within the *Zone* by using Eq. 27, with the phase temperature equated to the temperature of the *Zone*.

Case 2b: Variable Mixture Temperature: The homogenous energy balance is used to determine the temperature of the two-phase mixture at all points within the *Zone*. The homogeneous energy balance can be derived by summing the heterogeneous energy balances for the two phases. If the extent of the *Zone* is constant Eq. E.34 is generated. Alternatively if the extent of the Zone varies, Eq. E.35 is generated, with h^Z replacing u^Z

$$\frac{d}{dt}\left(M^Z u^Z\right) = \sum_{P=1}^{NPB} F^P\left[h^P + \frac{1}{2}\left(v^P\right)^2 + gZ^P\right] + \sum_{P=1}^{NPD} Q^P + Q^Z - W^Z \qquad [E.34]$$

$$\frac{d}{dt}\left(M^Z h^Z\right) - V^Z \frac{d}{dt} P^Z = \sum_{P=1}^{NPB} F^P\left[h^P + \frac{1}{2}\left(v^P\right)^2 + gZ^P\right] + \sum_{P=1}^{NPD} Q^P + Q^Z - W^Z \qquad [E.35]$$

where u^Z is the internal energy of the two phase mixture, NPB is the total number of *Ports* linked to the bulk transfer of material into or out of the *Zone* (either phase) and NPD is the total number of *Ports* linked to the diffusive transfer of heat into or out of the *Zone*.

Constitutive Equations: The internal energy and the enthalpy of the mixture can be related to the respective phase terms through the following equations.

$$u^Z = \sum_{k=1}^{2} X_k^Z u_k^Z \qquad [E.36]$$

$$h^Z = \sum_{k=1}^{2} X_k^Z h_k^Z \qquad [E.37]$$

As for the heterogeneous case, Eq. E.30 is generated if the kinetic energy of the material flowing into the *Zone* needs to be considered. The internal energy and the enthalpy of each phase are expressed in terms of the pressure and temperature of the *Zone* and the mass fraction of the phase through Eqs. E.31 and E.32, with the phase temperature equal to the temperature of the *Zone*.

E.6 The Pressure Subclass

If the two-phase *Zone* encompasses a gaseous compressible phase and is enclosed within a closed *Vessel,* the pressure of the two-phase *Zone* needs to be determined as part of the global solution. For this case, Eq. E.38 can be used to determine the density of the phase.

$$V^Z = \sum_{k=1}^{2} \frac{M_k^Z}{\rho_k^Z} \qquad [E.38]$$

Then the equation of state can be used to relate the pressure to the density. For all other cases, the pressure of the *Zone* can be deduced from the appropriate *Connection*. It should be noted that if the *Zone* is of variable extent, Eq. E.16 (which is identical to Eq. E.38) will be generated in addition to the volumetric constraint of the vessel and hence there is no need to specify Eq. E.38.

E.7 The Volume Fraction Subclass

If the ratio of the two volume fractions does not change, they can be assigned a constant value (Eq. E.39). Alternatively, if the volume fraction changes, the phase volume fraction can be determined from the volume of the phase and the total volume of the *Zone*.

$$\varepsilon_k^Z = const \qquad [E.39]$$

$$\varepsilon_k^Z = \frac{M_k^Z}{\rho_k^Z V^Z} \qquad [E.40]$$

E.8 The Velocity Subclass

The *Velocity Subclass* has been devised so that *COMPASS* is capable of handling a variety of flow scenarios. The equations within the *Velocity Subclass* have been formulated so that the two phases may be stationary or in a state of movement. Further, the two phases may flow counter-currently or co-currently at different or identical velocities. To structure the model-generating algorithms for the velocity subclass, six different categories are used to characterise the velocities. The six different categories are shown in Table E.1.

Scenario	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
Dispersed phase is moving	✗	✔	✔	✔	✔	✗
Continuous phase is moving	✗	✔	✔	✔	✗	✔
Overall movement of the mixture	✗	✔	✔	✗	✔	✔
Relative motion between the phases	✗	✗	✔	✔	✔	✔

Table E.1: The Different Flow Scenarios

Case 1: - Both Phases are Stationary: If both phases are stationary the velocity of each phase can be declared as zero.

$$v_k^Z = 0 \qquad [E.41]$$

Case 2: - Both Phases Travel at the Same Velocity: For this case, both phases travel in the same direction at identical velocities. Consequently the velocity of each phase is equal to the velocity of the mixture (J^Z).

$$v_k^Z = J^Z \qquad [E.42]$$

The velocity of the mixture is deduced from the mass flowrate at the inlet *Ports*.

$$J^Z = \sum_{k=1}^{k=2} \sum_{P=1}^{P=NP} \frac{F_k^{P(inlet)}}{\rho_k^{P(inlet)} A^Z} \qquad [E.43]$$

Case 3 - Relative Velocity between the Two Phases: Two key factors are considered when determining the velocity of each phase within the *Zone*: the average velocity of the mixture as a whole and the relative velocity between the two phases. As the direction in which each phase flows cannot be deduced from the position of the respective *Ports*, it is necessary for the user to specify whether the flow is co- or counter- current. If the flow is co-current, the framework treats the two entrance *Ports* as coexisting on the same plane and requires additional information from the user concerning whether the two-phase material flows downwards (in the direction of gravity) or upwards. For the case of counter-current flow, the relative density of the two phases is used to determine both the direction in which each phase travels and the relative positions of the two inlet and the two outlet *Ports*. The direction of flow is represented by the tag λ, which is negative for upward flow and positive for downward flow.

For Downward Flow: $\lambda_k = 1$ [E.44]

For Upward Flow: $\lambda_k = -1$ [E.45]

Just as in Case 2, the average velocity of the mixture is calculated from the mass flow rates at the two inlet *Ports*.

$$J^Z = \sum_{k=1}^{k=2} \sum_{n=1}^{n=NP} \frac{\lambda_k F_k^{P(inlet)}}{\rho_k^{P(inlet)} A^Z} \qquad [E.46]$$

The modelling of a system where there is a relative velocity between the two phases can be simplified by assuming that the relative velocity between the phases can be determined algebraically in terms of the local condition. This approach is frequently referred to as the drift-flux model (Zuber and Findlay, 1965; Ishii, 1977). The drift-flux model considers the movement of the mixture as a whole, represented by the mixture mass and momentum balances (for distributed systems). The movement of the dispersed phase is then superimposed onto the movement of the mixture with the use of an additional kinematic correlation, capable of predicting the relative movement of the dispersed phase with respect to the movement of the mixture. The additional kinematic correlation (drift flux relationship) allows the slip (relative velocity) as well as the positioning of the phase perpendicular to the direction of flow (in the second dimension) to be considered. This is advantageous for two-phase flows where there is a non-uniform distribution of the two phases. Both the slip and the non-uniformity of the

distribution are highly dependent on the configuration of the two phases and must be determined with the use of semi-theoretical correlations for each of the appropriate flow regimes.

Within the context of this work, one phase must be identified as the dispersed phase and the other phase is called the continuous phase. Although a dispersed phase exists only for the particle, bubble and slug flow regimes, this approach has been applied, with reasonable success, to the other flow regimes in upward vertical pipe flow (figure 2.2). To enable the drift flux velocity to be used, the user must state which phase is dispersed (d) and which phase is continuous (c).

$$v_c^Z = J^Z \pm \frac{\varepsilon_d^Z v_{dj}^Z}{\varepsilon_c^Z} \qquad [E.47]$$

$$v_d^Z = J^Z \pm v_{dj}^Z \qquad [E.48]$$

Where v_{dj} is the drift velocity, v_d is the velocity of the dispersed phase and v_c is the velocity of the continuous phase. The sign of the drift velocity depends on the density difference. A huge number of closure equations exist to determine the drift velocity and it is proposed that a detailed library of such correlations should be compiled if this research is to be followed up. Closure correlations are required not only for the drift velocity but also to deduce the flow regime that occurs. With knowledge of the flow regime, the appropriate form of the drift flux equation can be substituted into the model, thereby avoiding any need to specify the flow regime (cf. Borkowski, 1992).

Clearly, creating a framework that ensures that the appropriate form of the closure equation is used is not a trivial task. In addition to providing a physical description of a given two-phase process, the user must also make decisions concerning the validity of the closure equations and the selection of the appropriate correlations. If the user in not an expert in the field, the framework must assist the user in this task by providing a substantial amount of information concerning the validity of each correlation. Such an intelligent system is beyond the scope of the current work. Nevertheless, some elementary correlations have been implemented within the framework to illustrate how a solvable two-phase model can be formulated from a purely physical description. Table E.2 contains the correlations that have been used. All of these correlations have been embedded with the model-generating engine for the specific case of vertically upwards pipe flow. At the present stage of development, the user is required to state the flow regime.

Flow Regime	Drift Flux Velocity	Mass Distribution Parameter	Dispersed Phase
Dispersed	$v_{dj} = \frac{2}{9} r_d^2 g \frac{(\Delta\rho)}{\mu_m}(1-\varepsilon_d)^2$	$C_o = 1.2 - 0.2\sqrt{\rho_d / \rho_c}$	Gas, liquid or solid
Slug	$v_{dj} = 0.35\left(gD\frac{(\Delta\rho)}{\rho_c}\right)^{\frac{1}{2}}$	$C_o = 1.2$	Gas
Churn	$v_{dj} = \sqrt{2}\left(g\sigma\frac{(\Delta\rho)}{\rho_c^2}\right)^{\frac{1}{4}}$	$C_o = 1.2$	Gas
Annular	$v_{dj} = \left(\frac{1-\varepsilon_d}{\varepsilon_d + 4\sqrt{\frac{\rho_d}{\rho_c}}}\right)\sqrt{\frac{(\Delta\rho)gD(1-\varepsilon_d)}{0.015\rho_c}}$	$C_0 = 1$	Gas

Table E.2: Correlations for the Drift Flux Velocity and the Mass Distribution Parameter for the Different Flow Regimes as Proposed by Ishii (1977).

In Table E.2, r_d is the bubble or drop radius, ρ is the density, μ is the viscosity, σ is the surface tension, D is the pipe diameter and g is the gravitational acceleration. The subscript c refers to the continuous phase and the subscript d refers to the dispersed phase, and m refers to the mixture. The mass distribution parameter is used in modelling distributed two-phase systems. Constitutive equations are required for the viscosity of the mixture as well as for the radius of the bubble / drops. At this stage in the development, the user should either specify appropriate values or closure equations for these terms.

Case 4: Relative Velocity between the Two Phases, No Bulk Movement of the Two-phase Mixture: The algorithm used to formulate the mathematical model for Case 4 is the same as for Case 3 except that the velocity of the mixture is set to zero.

$$J^z = 0 \qquad [E.49]$$

Case 5: Continuous Phase is Stationary, Dispersed Phase is Moving: For this case the velocity of the continuous phase can be set to zero, and Eq. [E.50] can be used to determine the velocity of the dispersed phase.

$$v_d^Z = \frac{v_{dj}}{\left(1-\varepsilon_d^Z\right)} \qquad \text{[E.50]}$$

$$v_c^Z = 0 \qquad \text{[E.51]}$$

Case 6: Dispersed Phase is Stationary, Continuous Phase is Moving: The algorithm for Case 6 sets the velocity of the dispersed phase to zero and generates Eq. [E.53] to determine the velocity of the continuous phase.

$$v_d^Z = 0 \qquad \text{[E.52]}$$

$$v_c^Z = \frac{F_c^{P(inlet)}}{\rho_k^{P(inlet)} A^Z \varepsilon_k^Z} \qquad \text{[E.53]}$$

This case is most commonly used for fixed beds.

Appendix F

The Two-phase One-dimensional, Duct Zone Model Class

The *One-dimensional, Two-phase Duct Zone Model Class* is used to model a *Zone* which encompasses material that is in two stable thermodynamic conditions. Further, the state variables, the volume fraction or both attributes are distributed over the axial length of the *Zone*. The *model subclasses* correspond to the composition, temperature, pressure, velocity and volume fraction of each of the two phases and the extent of the *Zone*. Certain equations that occur within more than one *model subclass* are extracted from the appropriate *model subclass* and placed within the *Core Model Subclass*.

Within each *model subclass*, the key attributes (pressure, temperature, volume fraction) of each phase may be assigned a fixed value or be determined as part of the overall model. Further, for certain key attributes, homogeneity can be assumed and these attributes can be determined for the mixture as a whole rather than for each phase. As for the *Single-Phase, One-dimensional, Duct Zone Model Class*, the conservation equations are developed for a duct of constant cross sectional area. The equations are also expressed in their normalised form, allowing either *Zones* of fixed or variable size to be modelled. The reader should refer to Appendix G for the derivation of the normalising transformations. For the *Pressure Subclass*, the surface tension between the two phases is assumed to be small, and hence the pressure difference between the two phases is zero at each point in space and time.

F.1 The Port Subclass

The *Port* equations for the *Two Phase, 1-D Duct Zone Model Class* are similar to those for the *Single Phase, 1-D Duct Zone Model Class*, with full consideration given to both the bulk flow of

material and the diffusive flow of material and energy through the *Port*. Further, to permit the various combinations of bulk and diffusive flow to be considered, the bulk and diffusive flow components are calculated separately. The sole difference between the single phase, 1-D *Port* equations and the two-phase, 1-D *Port* equations arises from the need to specify the selective transfer of one of the two phases through the *Port*. That is to say each *Port* transmits only one of two phases. The following equations are generated for each outlet *Port* that experiences the bulk flow of material.

$$F^P = A^Z v_k^Z \varepsilon_k^Z \rho_k^Z \qquad x_{k,i}^P = x_{k,i}^Z \qquad h^P = h_k^Z \qquad \text{[F.1] – [F.3]}$$

Where F^P is the total bulk flow of mass (of phase k) through *Port* P, A is the cross sectional area of the *Zone* and ε_k, v_k and ρ_k are respectively the volume fraction, velocity and the density of phase k. If there is diffusive flow of mass or energy through the *Port*, either in conjunction with bulk flow or as the sole form of mass or energy transfer, it is necessary to deduce the diffusive flowrate of material ($F^P_{D,I}$) and energy (Q^P_D) from the respective diffusive components. For each outlet *Port* subject to either the diffusive flow of mass or energy.

$$F_{D,i}^P = A^Z \varepsilon_k^Z j_{k,i,z}^Z \qquad Q_D^P = A^Z \varepsilon_k^Z q_k^Z \qquad \text{[F.4] – [F.5]}$$

To maintain full continuity in the pressure field, the pressure of each *Port* is equated to the pressure of the boundary.

$$P^P = P^Z \qquad \text{[F.6]}$$

F.2 The Core Equations Subclass

The following equations for the mixture and phase densities are automatically generated within the *Core Subclass*.

$$\rho_m^Z = \sum_{k=1}^{2} \varepsilon_k^Z \rho_k^Z \qquad \text{[F.7]}$$

$$\rho_k^Z = f\left(T_k^Z, P^Z, x_{k,i}^Z\right) \qquad \text{[F.8]}$$

F.3 The Extent Subclass

The *Zone's* extent is determined from its cross sectional area and its axial boundaries (Z_1 and Z_2).

$$V^Z = A^Z(Z_2 - Z_1) \qquad [F.9]$$

F.4 The Composition Subclass

The *Composition Subclass* is designed to allow the composition of each phase to be determined separately. The composition of each phase may be determined with the use of a component mass balance or may be assigned a constant value.

Case 1, Constant Composition: If the composition of phase k within the *Zone* remains constant throughout the time horizon, the user should declare the mass fraction of each component ($x_{k,i}$).

$$x_{k,i}^Z = const \qquad [F.10]$$

Case 2, Variable Composition: If the composition of the phase varies, the normalised, component-mass conservation equation is generated. The normalised component mass balance for the i^{th} component of the k^{th} phase within a two-phase mixture is:

$$\left[\frac{\hat{z}-1}{(Z_2-Z_1)}\frac{dZ_1}{dt} - \frac{\hat{z}}{(Z_2-Z_1)}\frac{dZ_2}{dt}\right]\frac{\partial \varepsilon_k^Z \rho_{k,i}^Z}{\partial \hat{z}} + \frac{\partial \varepsilon_k^Z \rho_{k,i}^Z}{\partial t} + \frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}} v_k^Z \varepsilon_k^Z \rho_{k,i}^Z + \frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}} \varepsilon_k^Z j_{k,i}^Z - \varepsilon_k^Z R_{k,i}^Z - \Gamma_{IC,i} = 0 \qquad [F.11]$$

where $\rho_{k,i}$ is the mass concentration of component i within phase k, $j_{k,i}$ is the diffusive mass flux of component i in phase k and $\Gamma_{IC,i}$ is the interfacial rate of component mass transfer per unit volume of the *Zone*, specified by an *Internal Connection*. The rate source term ($R_{k,i}$) in Eq. F.11 has been specified per unit volume of the phase.

The Constitutive Equations: Constitutive equations are needed to expand the reaction sink term and the diffusive mass flux term. In should be noted that the interfacial rate of mass transfer is determined as part of the *Internal Connection*. Assuming that Fick's first law can be used to

describe the diffusive mass flux, the diffusive mass flux of component i can be expressed in terms of the diffusion coefficient ($D_{k,i}$) and the concentration gradient.

$$j_{k,i}^{Z} = \frac{-D_{k,i}}{(Z_2 - Z_1)} \frac{\partial \rho_{k,i}^{Z}}{\partial \hat{z}} \qquad [F.12]$$

The total source term for chemical reaction (generation/depletion) is equal to the sum of the products of the stoichiometric coefficients and the individual reactions rates.

$$R_{k,i}^{Z} = \sum_{j=1}^{NR} \upsilon_{k,i,j} r_{k,j} \qquad [F.13]$$

The mass fraction of each component within phase k is determined from the mass concentration of the component and the total density of the phase.

$$x_{k,i}^{Z} = \frac{\rho_{k,i}^{Z}}{\rho_{k}^{Z}} \qquad [F.14]$$

The Boundary Conditions: If there is bulk flow of material through the *Zone*, the inlet *Boundary* corresponds to the point at which material enters the *Zone* and the outlet boundary corresponds to the point at which material leaves the *Zone*. The inlet boundary condition is specified by applying the principle of mass conservation to the inlet. If the mass flowrate of component i, associated with *Port* P is F_i^P, then conservation of mass at the inlet gives:

$$F_i^P = A^Z v_k^Z \varepsilon_k^Z \rho_{k,i}^Z + A^Z \varepsilon_k^Z j_{k,i}^Z \qquad [F.15]$$

$$F_i^P = F^P x_i^P + F_{D,i}^P \qquad [F.16]$$

In certain instances it is possible that a specific component may be present within the *Zone* yet not present within a *Connection* that has been linked to the *Zone*. If this *Connection* is linked to an inlet *Boundary*, then it is necessary to declare the inlet mass flowrate of this component to be zero.

$$F_i^P = 0 \qquad [F.17]$$

The outlet boundary conditions are only required if there is diffusive mass transfer. Applying the principle of mass conservation to the outlet gives Eq. F.18 for the case where the *Boundary* is attached to a *Port* and Eq. F.19 for the case where there is no *Port* attached to the *Boundary*.

$$A^Z \varepsilon_k^Z j_{k,i}^Z = F_{D,i}^P \qquad [F.18]$$

$$j_{k,i}^Z = 0 \qquad [F.19]$$

F.5 The Temperature Subclass

The *Temperature Subclass* is designed to allow the temperature of the two phases to be the same (homogenous) or different (heterogeneous). In addition, the *Temperature Subclass* permits the temperature of each phase to be declared as a constant value or to be calculated. If the temperature needs to be computed, an energy balance is used.

Case 1: Heterogeneous Phase Temperatures

The user may declare the temperature of each phase to remain constant or specify that the phase temperature should be determined as part of the simulation.

Case 1a: Declared Phase Temperature

$$T_k^Z = const \qquad [F.20]$$

Case 1b: Variable Phase Temperature: The heterogeneous phase-energy balance is used to determine the temperature of the phase, which varies over both the spatial domain and the time domain and is different from the temperature of the second phase. The basis behind the energy balance is the same as that for the single-phase energy balance, however the partial occupation of the control volume by each phase and the interfacial exchange of energy associated with the transfer of both heat and mass are also considered. The heterogeneous energy balance is formulated with explicit consideration of the wall heat flux within the balance equations. This formulation assumes that the wall is completely covered by one phase, and hence the term should

either be included or omitted depending on whether or not the wall is in contact with the phase. The 1-D, normalised heterogeneous energy balance is[13]:

$$\left[\frac{\hat{z}-1}{(Z_2-Z_1)}\frac{dZ_1}{dt}-\frac{\hat{z}}{(Z_2-Z_1)}\frac{dZ_2}{dt}\right]\frac{\partial \rho_k^Z \varepsilon_k^Z E_k^Z}{\partial \hat{z}}+\frac{\partial \rho_k^Z \varepsilon_k^Z E_k^Z}{\partial t}$$
$$+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}\left[\rho_k^Z \varepsilon_k^Z v_k^Z H_k^Z\right]+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}\varepsilon_k^Z q_k^Z-\frac{Q_w P_w}{A}-\varepsilon_k^Z v_k^Z \rho_k^Z gb-\Gamma_{IC}^Z H_k-\Lambda_{IC}=0 \qquad [F.21]$$

In Eq. F.21, Λ_{IC} is the interfacial exchange of heat. All other terms have been defined previously. It should be noted that the last two terms on the left hand side of Eq. F.21 are specified through an *Internal Connection*.

Constitutive Equations: Constitutive equations are required to determine the heat transferred through thermal conduction and to express the total energy and enthalpy of each phase in terms of the thermal and kinetic components. In an identical manner to the *Single-phase 1-D Duct Zone Model Class*, Fourier's law is used to determine the diffusive heat flux.

$$q_k^Z=-\frac{k_k}{(Z_2-Z_1)}\frac{\partial T_k^Z}{\partial \hat{z}} \qquad [F.22]$$

Where k_k is the thermal conductivity of phase k. The total energy and total enthalpy of each phase are expressed in terms of the kinetic and thermal components.

$$E_k^Z=u_k^Z+\frac{v_k^{Z^2}}{2} \qquad \text{and} \qquad H_k^Z=h_k^Z+\frac{v_k^{Z^2}}{2} \qquad [F.23] - [F.24]$$

Equations of State: Equations of state are needed to relate the internal energy and the enthalpy of the phase to the pressure, temperature and composition of each phase.

---

[13] As the average of a product of variables is not the same as the product of the average variables it is necessary to introduce a distribution parameter to account for this discrepancy. For the energy balance an enthalpy distribution parameter should be used that is defined as $\frac{\langle \varepsilon H_k v_k^2 \rangle}{\langle \varepsilon_k \rangle \langle v_k \rangle \langle H_k \rangle}$ where the square brackets refer to the area averaging procedure. Research has shown that the enthalpy distribution parameter can be approximated to be unity and for this reason the parameter has been omitted from the energy balance (Ishii, 1977).

$$u_k^Z = f(T_k^Z, P^Z, x_{k,i}^Z) \qquad \text{and} \qquad h_k^Z = f(T_k^Z, P^Z, x_{k,j}^Z) \qquad \text{[F.25] - [F.26]}$$

The Boundary Conditions: The inlet boundary condition is determined by applying the principle of energy conservation to the inlet. If the flow of energy at the *Port* is Q^P, then conservation of energy at the inlet gives.

$$Q^P = \rho_k^Z \varepsilon_k^Z v_k^Z A^Z H_k^Z + A^Z \varepsilon_k^Z q_k^Z \qquad \text{[F.27]}$$

$$Q^P = F^P h^P + Q_D^P \qquad \text{[F.28]}$$

Where Q^P is the flow of energy through the *Port*. The outlet boundary condition is required for the case where there is axial heat diffusion through the *Zone*.

$$Q_D^P = A^Z \varepsilon_k^Z q_k^Z \qquad \text{[F.29]}$$

It should be noted that for the case where there is no bulk flow, the direction of energy flow is arbitary and hence it is not possible to identify "inlet" and "outlet" boundaries. For this case Eqs. F.27 and F.28 are applied to the first axial boundary and Eq. F.29 is applied to the second boundary (so the two boundary conditions are equivalent). If there is no *Port* attached to the axial boundary and there is conduction of energy through the *Zone*, the following boundary condition is generated.

$$q_k^Z = 0 \qquad \text{[F.30]}$$

Case 2: Homogeneous Mixture Temperature

If the temperature of each phase is the same throughout the *Zone*, the temperature can be computed for the mixture as a whole. As with the heterogeneous phase temperature, the temperature of the mixture may either be declared or calculated with the use of a homogeneous energy balance.

Case 2a: Declared Phase Temperature

$$T_m^Z = const \qquad [F.31]$$

Case 2b: Variable Phase Temperature: The homogenous energy balance is used to determine the temperature of the two-phase mixture at any point within the *Zone*. The homogeneous energy balance can be derived by combining the heterogeneous energy balances for the two phases. It should be noted that the bulk flow of energy for each phase is considered separately to facilitate consideration of the specific cases where one of the two phases is stationary.

$$\left[\frac{\hat{z}-1}{(Z_2-Z_1)}\frac{dZ_1}{dt}-\frac{\hat{z}}{(Z_2-Z_1)}\frac{dZ_2}{dt}\right]\frac{\partial \rho_m^Z E_m^Z}{\partial \hat{z}}+\frac{\partial \rho_m^Z E_m^Z}{\partial t}$$
$$+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}\left[\varepsilon_d^Z \rho_d^Z v_d^Z H_d^Z+\varepsilon_c^Z \rho_c^Z v_c^Z H_c^Z\right]+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}q_m^Z-\frac{Q_w P_w}{A}-v_m^{\prime} \rho_m^{\prime} g h=0 \qquad [F.32]$$

Constitutive Equations: Constitutive equations are needed to determine the heat transfer through thermal conduction and to express the total enthalpy and the total internal energy of the mixture in terms of the respective properties of each phase. Fourier's law (Eq. F.22) is used to determine the conductive heat flux, with Eq. F.33 used to provide an approximation for the thermal conductivity of the mixture.

$$k_m = \sum \varepsilon_k^Z k_k \qquad [F.33]$$

The total thermal energy (internal energy and kinetic energy) is related to the internal energy of each phase through the following equations.

$$E_m^Z = \sum_{k=1}^{2} X_k^Z E_k^Z \qquad [F.34]$$

$$X^Z = \frac{\rho_k^Z \varepsilon_k^Z}{\rho_m^Z} \qquad [F.35]$$

The total internal energy and enthalpy of each phase can be further decomposed into a kinetic and a thermal component, using Eqs. F.23 and F.24. Equations of state (Eqs. F.25 and F.26) are required to relate the internal energy and the enthalpy of each phase to the local temperature, pressure and phase compositions of the *Zone*.

The Boundary Conditions: The boundary condition generated for the inlet has been formulated so that full consideration is given to the possibility of having two or more inlet *Ports* on the same *Boundary* with sufficient mixing occurring at the inlet to ensure that the temperature of the two phases is homogeneous.

$$Q^{B} = \rho_c^Z \varepsilon_c^Z v_c^Z A^Z H_c^Z + \rho_d^Z \varepsilon_d^Z v_d^Z A^Z H_d^Z + A^Z q_m^Z \qquad [F.36]$$

$$Q^B = \sum_{NP=1}^{TNP} Q^P \qquad [F.37]$$

$$Q^P = F^P h^P + Q_D^P \qquad [F.38]$$

where Q^B is the flow of energy at the Boundary and TNP is the total number of *Ports* at the *Boundary*. The outlet boundary condition is required for the case where there is axial heat diffusion through the *Zone*.

$$Q_D^P = A^Z q_m^Z \qquad [F.39]$$

If there is no bulk flow, Eqs. F.36, F.37 and F.38 are applied to the first axial boundary and Eq. F.39 is applied to the second boundary. If there is no *Port* attached to the axial boundary and there is conduction of energy through the *Zone*, the following boundary condition is generated.

$$q_m^Z = 0 \qquad [F.40]$$

F.6 The Pressure and Velocity Subclass

Because of the high degree of coupling between the pressure and the velocity for a compressible mixture, it is necessary to determine these two key attributes collectively within the *Pressure and Velocity Subclass*. The *Pressure and Velocity Subclass* is formulated to enable the mixture to be declared to be compressible or incompressible, stationary or flowing. Furthermore the two phases may travel at the same velocity or at different velocities, and in directions that are either co-current or counter-current. The movement of the two phases within the *Zone* is categorised according to the six different cases introduced in Appendix E (Table E.1).

Case 1: - Both Phases are Stationary: If both phases are stationary the velocity of each phase can be declared to be zero.

$$v_k^Z = 0 \qquad [F.41]$$

The pressure may be assigned a constant value (Eq. F.42) or determined with the use of a momentum balance (Eq. F.43). Equation F.43 is written for the general case. When both phase are stationary the momentum balance can be simplified for the specific (hydrostatic) case where the phase velocities, the mixture velocity and the wall shear stress are all zero. It should be noted that the mixture density is determined within the *Core Subclass* and the boundary condition for Eq. F.43 is generated within the *Port Subclass*.

$$P^Z = const \qquad [F.42]$$

$$\left[\frac{\hat{z}-1}{(Z_2-Z_1)}\frac{dZ_1}{dt} - \frac{\hat{z}}{(Z_2-Z_1)}\frac{dZ_2}{dt}\right]\frac{\partial \rho_m^Z v_m^Z}{\partial \hat{z}} + \frac{\partial \rho_m^Z v_m^Z}{\partial t} + \frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}\left[C_{vc}\varepsilon_c^Z \rho_c^Z v_c^{Z^2} + C_{vd}\varepsilon_d^Z \rho_d^Z v_d^{Z^2}\right]$$
$$-\rho_m^Z gb + \frac{1}{(Z_2-Z_1)}\frac{\partial P^Z}{\partial \hat{z}} + \frac{\tau_w P_w}{A} = 0$$

[F.43]

where τ_W is the wall shear stress and C_{vk} is the momentum distribution parameter. The momentum distribution parameter is used to account for the difference between the average of the product of the squared phase velocity and the volume fraction and the product of the averaged volume fraction and the squared averaged phase velocity[14]. In the absence of better information, $C_{v,k}$ is usually set to unity.

Constitutive Equations: A constitutive equation is needed for the wall shear stress. The current scope of the framework assumes that the user will supply an appropriate correlation to determine the wall shear stress (e.g. Lockhart-Martinelli; Chexal and Lellouch, 1986).

The Boundary Condition(s): As the pressure at each axial *Boundary* is equated to the pressure of the appropriate *Port*, the boundary condition for Eq. F.43 has already been formulated.

---

[14] $C_{vk} = \dfrac{\langle \varepsilon_k v_k^{\,2} \rangle}{\langle \varepsilon_k \rangle \langle v_k \rangle^2}$

Case 2: - Both Phases Travel at the Same Velocity: For this case, the velocities of the two phases are equal and can be represented by a two-phase mixture velocity (v_m). The velocity of the mixture can be calculated in an identical manner to the velocity of a single-phase system, except that mixture properties must be used. The appropriate equations are categorised according to whether the mixture is compressible or incompressible. If the mixture is compressible, the velocity and the pressure are coupled, preventing the two key attributes from being treated independently. For an incompressible mixture, the equations are no longer coupled and consequently the key attributes can be treated independently.

Case 2a: Compressible Mixture: For a compressible mixture, where the two phases travel at an equal velocity, the homogeneous (mixture) mass and momentum equations (Eq. F.43) must be solved simultaneously with an equation of state (Eq. F.8), which is used to relate the density of the compressible phase to the pressure at any point within the *Zone*. If the composition of the compressible phase varies with time, the density of the compressible phase is related to the sum of the mass concentrations.

NCM

$$\rho_k^Z = \sum_{i=1} \rho_{k,i}^Z \qquad [F.44]$$

For the case where the user has declared the composition of the compressible phase, the velocity of the mixture is determined by applying the principle of mass conservation to the total mass of the mixture.

$$\left[\frac{\hat{z}-1}{(Z_2-Z_1)}\frac{dZ_1}{dt}-\frac{\hat{z}}{(Z_2-Z_1)}\frac{dZ_2}{dt}\right]\frac{\partial \rho_m^Z}{\partial \hat{z}}+\frac{\partial \rho_m^Z}{\partial t}+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}\left[v_m^Z \rho_m^Z\right]=0 \qquad [F.45]$$

Constitutive Equations: A constitutive equation is needed for the wall shear stress. To ensure that the phase velocity can be used within the component mass balance and the energy balance, the velocity of each phase is equated to the velocity of the mixture.

$$v_k^Z = v_m^Z \qquad [F.46]$$

The Boundary Condition(s): For the case where the compressible phase consists of a single component, the boundary condition is applied at the *Port* that the continuous phase flows through[15].

$$F^P = A^Z v_c^Z \rho_c^Z \varepsilon_c^Z \qquad [F.47]$$

Case 2b: Incompressible Mixture: If both phases within the mixture are incompressible, the velocity of the mixture can be determined from the principle of conservation and a homogenous momentum balance can be used to determined the pressure. Since the two equations are uncoupled, each key attribute can be treated independently. The pressure may be assigned a constant value (Eq. F.42) or determined with the use of a momentum balance (Eq. F.43). If the wall shear stress cannot be neglected, the user should provide a correlation for the wall shear stress.

Case 3: The Two Phases are Move at Different Velocities, Relative Motion of the Mixture: As with the *Two-Phase Lumped Zone Model Class* the drift-flux model is used to compute the two phases velocities. The drift flux approach is based around the velocity of the mixture, so it is necessary to determine the velocity of the mixture using the same equations that were generated for Case 2. The difference between the two sets of equations arises from the constitutive equations relating the mixture velocity to the velocity of each phase. The velocity of each phase is related to the velocity of the mixture through the following set of equations.

$$v_m^Z = \frac{\sum_{k=1}^{2} \varepsilon_k^Z \rho_k^Z v_k^Z}{\rho_m^Z} \qquad [F.48]$$

$$v_d^Z = v_{dj}^Z + C_o J^Z \qquad [F.49]$$

$$J^Z = \varepsilon_d^Z v_d^Z + \varepsilon_c^Z v_c^Z \qquad [F.50]$$

Where v_{dj} is the drift flux velocity and J is the volumetric flux of the mixture. The mass distribution parameter (C_o) accounts for the non-uniform distribution of the dispersed phase in the plane perpendicular to the flow.

---

[15] Note: The connection corresponding to the dispersed phase is used as a boundary condition in the Volume Fraction Subclass.

Constitutive Equations: Constitutive equations are required for the drift velocity (Table E.2), the mass distribution parameter (Table E.2) and the momentum distribution parameter. Ishii (1977) proposed that the following correlations could be used to determine the momentum distribution parameter.

For the liquid phase (continuous): $$C_{vc} = 1 + 1.5(C_o - 1) \qquad [F.51]$$

For the gas phase (dispersed): $$C_{vd} \approx 1 + 0.5(C_o - 1) \qquad [F.52]$$

For annular flow, the momentum distribution parameter is the same for each phase:

$$C_{vk} = \begin{cases} 1.02 \text{ (turbulent flow)} \\ 1.33 \text{ (laminar flow)} \end{cases} \qquad [F.53] - [F.54]$$

Case 4: - Both Phases are Moving, No Overall Movement of the Mixture: If the movement of one phase displaces the other phase in the opposite direction, and the overall mixture is motionless, the downward volumetric flux must equal the upward volumetric flux. In such circumstances, the volumetric flux of the mixture (J) is zero. The velocity of the dispersed phase is therefore equal to the drift velocity and the velocity of the continuous phase can be found from the constraint that the volumetric flux is zero. The velocities of the two phases are calculated through Eqs. F.55 and F.56 with a suitable correlation selected (from Table E.2) for the drift velocity.

$$v_d^Z = v_{dj}^Z \qquad [F.55]$$

$$v_c^Z = -\frac{\varepsilon_d^Z v_d^Z}{\varepsilon_c^Z} \qquad [F.56]$$

The pressure of the mixture can either be assigned a fixed value (Eq. F.42) or can be computed with the use of the momentum balance (Eq. F.43).

Case 5: Continuous Phase is Stationary, Dispersed Phase is Moving: Once again the pressure and the velocity are not coupled and hence can be treated independently. If the continuous phase is stationary, (e.g. the bubbling of a gas stream through a liquid pool) the velocity of the dispersed phase can be deduced from the drift flux correlation with the velocity of the continuous phase set to zero.

$$v_c^Z = 0 \qquad \text{[F.57]}$$

$$v_d^Z = v_{dj}^Z + C_o J^Z \qquad \text{[F.58]}$$

$$J^Z = \varepsilon_d^Z v_d^Z \qquad \text{[F.59]}$$

The pressure may be assigned a constant value (Eq. F.42) or can be determined with the use of a momentum balance (Eq. F.43). In the latter case, the continuous phase velocity term within the momentum balance is equated to zero.

Case 6: Dispersed Phase is Stationary:

If the stationary phase is dispersed (packed beds, catalytic plug flow reactors), the velocity of the continuous phase can be deduced from the continuity requirements of the phase. If the composition of the continuous phase varies, Eq. F.44 is generated. Alternatively, if the composition has been declared by the user, Eq. F.60 is generated.

$$\left[\frac{\hat{z}-1}{(Z_2-Z_1)}\frac{dZ_1}{dt} - \frac{\hat{z}}{(Z_2-Z_1)}\frac{dZ_2}{dt}\right]\frac{\partial \varepsilon_c^Z \rho_c^Z}{\partial \hat{z}} + \frac{\partial \varepsilon_c^Z \rho_c^Z}{\partial t} + \frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}\left[\varepsilon_c^Z v_c^Z \rho_c^Z\right] \pm \Gamma_{IC}^Z = 0 \qquad \text{[F.60]}$$

Boundary Condition: The boundary condition is derived by applying the principle of conservation to the inlet *Port* that the continuous phase flows through.

$$F^P = A^Z \rho_c^Z \varepsilon_c^Z v_c^Z \qquad \text{[F.61]}$$

Alternatively, if the density of the continuous phase and the volume fraction of either phase remain constant along the axial length of the *Zone* and the interfacial exchange of mass is neglected, the superficial velocity can be used.

$$v_c^Z = \frac{F^P}{A^Z \varepsilon_c^Z \rho_c^Z} \qquad \text{[F.62]}$$

In many instances, the two-phase system will involve a gas or liquid stream flowing through a region that is filled with a matrix of stationary solid objects, (solid particles or a solid mesh). Although it is, in principle, possible to write a rigorous momentum balance for the flowing phase, the equation is not justified in view of the uncertainties in the understanding of the frictional and drag forces that exist between the two phases. Instead, semi-empirical correlations, such as the Ergun equation, are used to determine the pressure drop for the flowing phase. Within this *subclass*, the user may either equate the pressure to a constant value, or select an appropriate semi-empirical correlation to determine the pressure drop, such as Eq. F.63.

$$-\frac{\partial P^Z}{\partial z} = 150\frac{\left(1-\varepsilon_d^Z\right)^2}{\varepsilon_d^{Z^3}}\frac{\mu_c^Z v_c^Z}{d_d^{Z^2}} + 1.75\frac{1-\varepsilon_d^Z}{\varepsilon_d^{Z^3}}\frac{\rho_c^Z v_c^{Z^2}}{d_d^Z} \qquad \text{[F.63]}$$

where d_d represents the diameter of the dispersed phase particles.

Boundary Condition: The boundary condition has already been formulated within the *Port Subclass*.

F.7 The Volume Fraction Subclass

Since the volume fractions of the phases must sum to unity, only one volume fraction term needs to be calculated. For the *Two-phase, One-dimensional, Duct Zone Model Class*, the volume fraction may be assigned a constant value or can be found from one of the phase continuity equations.

Case 1: Constant Volume Fraction: For the case where the volume fraction remains constant over both the axial length and the time horizon, a fixed value can be assigned to the volume fraction of one phase.

$$\varepsilon_k^Z = \text{constant} \qquad \text{[F.64]}$$

Case 2: Variable Volume Fraction: If the volume fractions of the two phases vary over the extent of the *Zone* or over the time horizon, the volume fraction must be deduced by applying the principle of conservation to one of the two phases. For dispersed two-phase flow, the dispersed phase continuity equation is generally used, (Azberl, 1981; Lapin and Lubert, 1994). The

dispersed phase corresponds to the bubbles in bubbly flow, the slugs in slug flow and the droplets in droplet flow. For some flow regimes there is no dispersed phase as such, but the same approach can be applied to either of the two phases.

$$\left[\frac{\hat{z}-1}{(Z_2-Z_1)}\frac{dZ_1}{dt}-\frac{\hat{z}}{(Z_2-Z_1)}\frac{dZ_2}{dt}\right]\frac{\partial \varepsilon_k^Z \rho_k^Z}{\partial \hat{z}}+\frac{\partial \varepsilon_k^Z \rho_k^Z}{\partial t}+\frac{1}{(Z_2-Z_1)}\frac{\partial}{\partial \hat{z}}\left[\varepsilon_k^Z v_k^Z \rho_k^Z\right]-\Gamma_{IC}^Z=0 \qquad \text{[F.65]}$$

The volume fraction of the continuous phase is determined from the constraint that the sum of the volume fractions is unity.

$$\sum \varepsilon_k^Z = 1 \qquad \text{[F.66]}$$

<u>Boundary Conditions:</u> The principle of continuity can be applied at the inlet boundary to give.

At the Inlet; $\qquad F^P = \rho_d^Z \varepsilon_d^Z v_d^Z A^Z \qquad$ [F.67]

where F^P is the total mass flowrate of the dispersed phase through *Port* P. If there is no Port at the inlet boundary, Eq. F.68 is generated.

$$\varepsilon_d^Z(0) = 0 \qquad \text{[F.68]}$$

Appendix G

Derivation of the Normalising Transformations

To allow COMPASS to handle distributed domains of both constant and variable size it is necessary to express all the partial differential equations in their normalised form. By normalising the conservation equations, the differential terms are expressed over a domain of unit length, and a mapping factor (characteristic length) is used to map the normalised domain to the actual *Zone* length. The derivation of the normalising transformation for the axial domain is given in this appendix.

To normalise the conservation equation, the conservation equations are written as functions of the normalised length $\left(\hat{z}\right)$ (defined by Eq. G.1) and time $\left(\hat{t}\right)$ (defined by Eq. G.2).

$$\hat{z} = \frac{z - z_1(t)}{L(t)} \qquad [G.1]$$

$$\hat{t} = t \qquad [G.2]$$

Where z is the original co-ordinate, referring to the actual geometry of the domain, $z_1(t)$ is the moving inlet boundary and L(t) is the length of the domain ($z_2(t)$ $z_1(t)$). To normalise the conservation equations, the partial derivatives (with respect to z and t) are reformulated and expressed in terms of the normalised length $\left(\hat{z}\right)$ and normalised time $\left(\hat{t}\right)$. If T is any variable of interest,

$$dT = \frac{\partial T}{\partial \hat{z}} d\hat{z} + \frac{\partial T}{\partial \hat{t}} d\hat{t} \qquad [G.3]$$

Eq. G.3 can be differentiated with respect to z to give Eq. G.4, and with respect to t to give Eq. G.5.

$$\frac{\partial T}{\partial z}=\frac{\partial T}{\partial \hat{z}}\frac{\partial \hat{z}}{\partial z}+\frac{\partial T}{\partial \hat{t}}\frac{\partial \hat{t}}{\partial z} \qquad \text{Eq. 4} \qquad \frac{\partial T}{\partial t}=\frac{\partial T}{\partial \hat{z}}\frac{\partial \hat{z}}{\partial t}+\frac{\partial T}{\partial \hat{t}}\frac{\partial \hat{t}}{\partial t} \qquad \text{[G.4] – [G.5]}$$

Eqs. G.1 and G.2 can be differentiated with respect to z to give

$$\frac{\partial \hat{z}}{\partial z}=\frac{1}{L(t)} \qquad \frac{\partial \hat{t}}{\partial z}=0 \qquad \text{[G.6] – [G.7]}$$

Substituting Eq. G.6 into Eq. G.4 gives:

$$\frac{\partial T}{\partial z}=\frac{1}{z_2(t)-z_1(t)}\frac{\partial T}{\partial \hat{z}} \qquad \text{[G.8]}$$

Similarly, Eq. G.1 and G.2 can be differentiated with respect to time to give:

$$\frac{\partial \hat{z}}{\partial t}=\frac{-\hat{z}}{L(t)}\frac{dL}{dt} \qquad \frac{\partial \hat{t}}{\partial t}=1 \qquad \text{[G.9] – [G.10]}$$

Substituting Eq G.9 into Eq. G.5 gives

$$\frac{\partial T}{\partial t}=\frac{-\hat{z}}{z_1(t)}\frac{\partial T}{\partial \hat{z}}\frac{dz_1}{dt}+\frac{\partial T}{\partial \hat{t}} \qquad \text{[G.11]}$$

Which can be written in the form

$$\frac{\partial T}{\partial t}=\left(\frac{\hat{z}-1}{z_2(t)-z_1(t)}\frac{dz_1}{dt}-\frac{\hat{z}}{z_2(t)-z_1(t)}\frac{dz_2}{dt}\right)\frac{\partial T}{\partial \hat{z}}+\frac{\partial T}{\partial \hat{t}} \qquad \text{[G.12]}$$

G.8 and G.12 can be used to normalise all of the one-dimensional conservation equations that are expressed over an axial domain. For all the 1-D conservation equations, Eq. G.8 is used to replace the axial derivatives and Eq. G.12 is used to replace the time derivatives.

Appendix H

Flex Input Specification

```
%{
/* LEX INPUT SPECIFICATION */
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include "vessel.h"
#include "phase.h"
#include "process.h"
#include "reservoir.h"
#include "zone.h"
#include "noextentzone.h"
#include "node.h"
#include "globalterms.h"
#include "singlephaselumpedzone.h"
#include "parser.tab.h"

int yylineno = 1;
%}

/* REGULAR DEFINITIONS */

comment         "#".*
endofline       [\n]
delimiter       [ \t]
whitespace      {delimiter}+
letter          [A-Za-z]
underscore      "_"
string          {letter}({underscore})*
digit           [0-9]
number          (("-")?)({digit}+|({digit}*\.{digit}+)([eE][-+]?{digit}+)?)
id              {string}({string}|{digit})*
arrow           "->"|"<->"


%%

{comment}       { }
{whitespace}    { }
{endofline}     {yylineno++;}


1-D                             {return OneD;}
ABSORPTION                      {return ABSORPTION;}
ANNULAR                         {return ANNULAR;}
AXIAL                           {return AXIAL;}
BOILING                         {return BOILING;}
BOILING_ISOTHERM                {return BOILING_ISOTHERM;}
BOUNDARY                        {return BOUNDARY;}
BULK_FLOW                       {return BULK_FLOW;}
CARTESIAN                       {return CARTESIAN;}
CHURN                           {return CHURN;}
CLOSED                          {return CLOSED;}
COMPONENT                       {return COMPONENT;}
COMPRESSIBLE                    {return COMPRESSIBLE;}
CONNECTION                      {return CONNECTION;}
CONTINUITY                      {return CONTINUITY;}
CONTINUOUS_PHASE                {return CONTINUOUS_PHASE;}
COORDINATES                     {return COORDINATES;}
DEFINE_BOUNDARIES               {return DEFINE_BOUNDARIES;}
```

```
DEFINE_COMPONENTS               {return DEFINE_COMPONENTS;}
DEFINE_CONNECTIONS              {return DEFINE_CONNECTIONS;}
DEFINE_FLOWSHEET                {return DEFINE_FLOWSHEET;}
DEFINE_PHASES                   {return DEFINE_PHASES;}
DEFINE_PHYSICOCHEMICAL_BEHAVIOUR      {return DEFINE_PHYSICOCHEMICAL_BEHAVIOUR;}
DEFINE_PORTS                    {return DEFINE_PORTS;}
DEFINE_RESERVOIRS               {return DEFINE_RESERVOIRS;}
DEFINE_VESSELS                  {return DEFINE_VESSELS;}
DEFINE_ZONES                    {return DEFINE_ZONES;}
DISPERSED_PHASE                 {return DISPERSED_PHASE;}
DIFFUSION                       {return DIFFUSION;}
DIFFUSIVE_FLOW                  {return DIFFUSIVE_FLOW;}
DISPERSED                       {return DISPERSED;}
DOWNWARD_FLOW                   {return DOWNWARD_FLOW;}
DYNAMIC                         {return DYNAMIC;}
END                             {return END;}
ENERGY                          {return ENERGY;}
EQUILIBRIUM                     {return EQUILIBRIUM;}
EQUATE_PRESSURE                 {return EQUATE_PRESSURE;}
EXACT                           {return EXACT;}
EXTERNALFORCE                   {return EXTERNALFORCE;}
EXTERNALWORK                    {return EXTERNALWORK;}
FIXED                           {return XXFIXEDXX;}
FLOW_REGIME                     {return FLOW_REGIME;}
GAS                             {return GAS;}
GEOMETRY                        {return GEOMETRY;}
GRAVITY                         {return GRAVITY        ;}
HEATFLUX                        {return HEATFLUX;}
HORIZONTAL                      {return HORIZONTAL;}
INCLUDE_HYDROSTATIC_PRESSURE    {return INCLUDE_HYDROSTATIC_PRESSURE;}
INCOMPRESSIBLE                  {return INCOMPRESSIBLE;}
INTERFACIAL_HEAT_TRANSFER       {return INTERFACIAL_HEAT_TRANSFER; }
INTERFACIAL_MASS_TRANSFER       {return INTERFACIAL_MASS_TRANSFER; }
INTERNAL_RELATIVE_MOTION        {return INTERNAL_RELATIVE_MOTION;}
ISENTROPIC                      {return ISENTROPIC;}
ISOTHERMAL                      {return ISOTHERMAL;}
LIQUID                          {return LIQUID;}
LUMPED                          {return LUMPED;}
MATERIAL                        {return MATERIAL;}
MIXTURE_BULK_FLOW               {return MIXTURE_BULK_FLOW;}
NEGLECT_KE                      {return NEGLECT_KE;}
NEGLECT_PE                      {return NEGLECT_PE;}
NEGLECT_WALL_SHEAR              {return NEGLECT_WALL_SHEAR;}
NO_BULK_FLOW                    {return NO_BULK_FLOW;}
NOEXTENT                        {return NOEXTENT;}
NONE                            {return NONE;}
NOT_SPECIFIED                   {return NOT_SPECIFIED;}
OPEN                            {return OPEN;}
ORIENTATION                     {return ORIENTATION;}
PHASE                           {return PHASE;}
PHASE_ASSUMPTIONS               {return PHASE_ASSUMPTIONS;}
PHASE_COMPOSITION               {return PHASE_COMPOSITION;}
PHASE_DEPLETION                 {return PHASE_DEPLETION;}
PHASE_TEMPERATURE               {return PHASE_TEMPERATURE;}
PHASE_VOLUME_FRACTION           {return PHASE_VOLUME_FRACTION;}
POLYTROPIC                      {return POLYTROPIC;}
PORT_MASS_FLOWRATE              {return PORT_MASS_FLOWRATE;}
PORT                            {return PORT;}
PRESSURE_CONNECTION             {return PRESSURE_CONNECTION;}
PRESSURE_DRIVEN                 {return PRESSURE_DRIVEN;}
PROCESS                         {return PROCESS;}
RADIAL                          {return RADIAL;}
REACTION                        {return REACTION;}
RELATIVE_PHASE_MOTION           {return RELATIVE_PHASE_MOTION;}
RESERVOIR                       {return RESERVOIR;}
RESERVOIR_COMPOSITION           {return RESERVOIR_COMPOSITION;}
RESERVOIR_PRESSURE              {return RESERVOIR_PRESSURE;}
RESERVOIR_TEMPERATURE           {return RESERVOIR_TEMPERATURE;}
SINGLEPHASE_OPEN                {return SINGLEPHASE_OPEN;}
SLUG                            {return SLUG;}
SOLID                           {return SOLID;}
```

```
STATIONARY                  {return STATIONARY;}
STEADY_STATE                {return STEADY_STATE;}
SUPERFICIAL_VELOCITY        {return SUPERFICIAL_VELOCITY;}
TWOPHASE_OPEN               {return TWOPHASE_OPEN;}
UPWARD_FLOW                 {return UPWARD_FLOW;}
USER_DEFINED_SUBMODEL       {return USER_DEFINED_SUBMODEL;}
VARIABLE                    {return VARIABLE;}
VERTICAL                    {return VERTICAL;}
VESSEL                      {return VESSEL;}
VOLUME                      {return VOLUME;}
VOLUME_GAIN                 {return VOLUME_GAIN;}
ZERO                        {return ZERO;}
ZONE                        {return ZONE;}
ZONE_ASSUMPTIONS            {return ZONE_ASSUMPTIONS;}
ZONE_COMPOSITION            {return ZONE_COMPOSITION;}
ZONE_DESCRIPTION            {return ZONE_DESCRIPTION;}
ZONE_DYNAMICS               {return ZONE_DYNAMICS;}
ZONE_EXTENT                 {return ZONE_EXTENT;}
ZONE_FLOWRATE               {return ZONE_FLOWRATE;}
ZONE_HEAT_AND_WORK          {return ZONE_HEAT_AND_WORK;}
ZONE_ORIENTATION            {return ZONE_ORIENTATION;}
ZONE_PRESSURE               {return ZONE_PRESSURE;}
ZONE_PRESSURE_DROP          {return ZONE_PRESSURE_DROP;}
ZONE_TEMPERATURE            {return ZONE_TEMPERATURE;}
ZONE_VELOCITY               {return ZONE_VELOCITY;}
ZONE_VOLUME_FRACTION        {return ZONE_VOLUME_FRACTION;}

{number}        {yylval.dval = atof(yytext);
                 return NUMBER;}

{id}            {yylval.sval = (char *) strdup(yytext);
                 return IDENTIFIER;}

{arrow}         {return ARROW;}
{underscore}    {printf("Error line %d: leading or lone underscore\n", yylineno);
                 printf("***ABORTED***\n");
                 exit(0);}

\;              {return ';';}
\:              {return ':';}
\[              {return '[';}
\]              {return ']';}
\,              {return ',';}
\.              {return '.';}
\=              {return '=';}

.               {printf("Warning line %d: `%s' unrecognised\n", yylineno, yytext);
                 printf("***ABORTED***\n");
                 exit(0);}
%%

int yywrap()
{
  return(1);
}

void yyerror(const char *s)
{
  fprintf(stdout, "Error line %d: %s at or near '%s'\n", yylineno, s, yytext);
  cout << "***ABORTED***\n" ;
  exit(1);
}
```

Appendix I

Parser Input Specification

```
/* Definition Section */
%{
#include <stdlib.h>
#include <string.h>
#include <fstream.h>
#include <iostream.h>
using namespace std ;

#include "process.h"
#include "reservoir.h"
#include "zone.h"
#include "vessel.h"
#include "globalterms.h"
#include "singlephaselumpedzone.h"
#include "twophaselumpedzone.h"
#include "twophaseductzone.h"
#include "port.h"
#include "zonedimensions.h"

static string currentname ;
static string phasename1 ;
static string phasename2 ;
static string floatphasename ;
static string vesselname ;
static string zonename ;
static string reservoirnm ;
Phase::Phase_type T1 ;
Phase::Comp C1 ;
Vessel::VesselProperties V1 ;
Zone::SteadyState ZSS ;
Zone::ThermodynamicProcess NETA ;
Zone::Heatflux NEHF ;
Zone::Workdone  NEWD ;
Zone::Kineticenergy NEKE ;
Zone::PressureDropType NEPD ;
Zone::Flowrate NEFR ;
Zone::Extent SLEX ;
Zone::Reaction SLRE ;
Zone::MassDiffusion SPMD ;
Zone::Composition_set SLCS ;
Zone::Temperature_set SLTS ;
Zone::Heatflux SLHF ;
Zone::Workdone  SLWD ;
Zone::Kineticenergy SLKE ;
Zone::Potentialenergy SLPE ;
Zone::ThermalDiffusion SPTD ;
Zone::Pressure_set SLPS ;
Zone::WallShear SDWS ;
Zone::Velocity SPVD ;
Zone::InternalMassTransfer IMT ;
Zone::InternalHeatTransfer IHT ;
Zone::VolumeFraction ZVF ;
Zone::TwoPhaseZoneVelocity ZTPV ;
Zone::RelativeMotion ZRM ;
Zone::FlowRegime ZFR ;
Zone::FlowDirection ZFD ;
Zone::BulkMovement ZMBM ;
Zone::SuperficialVelocity ZSV ;
```

```
Port::MotionType ZPV ;
Port::HydrostaticPressure PSH ;
Zone::PhaseMotionType PMT ;
SinglePhaseOpenConnection::EquatePressure CPS ;
ZoneBoundary::BoundaryPosition ZBP ;
Zone::Orientation ZO ;

extern Process* P1 ;

%}

%union {
  int ival;
  double dval;
  char* sval;
  int SS_set ;
}

%token          PROCESS
%token          DEFINE_PHASES
%token          PHASE
%token          LIQUID
%token          SOLID
%token          GAS
%token          INCOMPRESSIBLE
%token          COMPRESSIBLE
%token          COMPONENT
%token          DEFINE_VESSELS
%token          VESSEL
%token          OPEN
%token          CLOSED
%token          VOLUME
%token          DEFINE_RESERVOIRS
%token          RESERVOIR
%token          DEFINE_COMPONENTS
%token          DEFINE_CONNECTIONS
%token          CONNECTION
%token          PRESSURE_CONNECTION
%token          SINGLEPHASE_OPEN
%token          EQUATE_PRESSURE
%token          ENERGY
%token          MATERIAL
%token          TWOPHASE_OPEN
%token          DEFINE_ZONES
%token          ZONE
%token          ORIENTATION
%token          VERTICAL
%token          HORIZONTAL
%token          DEFINE_BOUNDARIES
%token          BOUNDARY
%token          AXIAL
%token          RADIAL
%token          DEFINE_PORTS
%token          PORT
%token          INCLUDE_HYDROSTATIC_PRESSURE
%token          DEFINE_FLOWSHEET
%token          ZONE_DESCRIPTION
%token          ZONE_DYNAMICS
%token          STEADY_STATE
%token          DYNAMIC
%token          GEOMETRY
%token          NOEXTENT
%token          LUMPED
%token          OneD
%token          CARTESIAN
%token          COORDINATES
%token          SINGLEPHASELUMPED
%token          SINGLEPHASE_OneD_DUCT
%token          DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
%token          ZONE_FLOWRATE
%token          PRESSURE_DRIVEN
```

```
%token          CONTINUITY
%token          ZONE_ASSUMPTIONS
%token          PHASE_ASSUMPTIONS
%token          ZONE_TEMPERATURE
%token          PHASE_TEMPERATURE
%token          RESERVOIR_TEMPERATURE
%token          ZONE_HEAT_AND_WORK
%token          EXTERNALWORK
%token          HEATFLUX
%token          ISOTHERMAL
%token          ISENTROPIC
%token          POLYTROPIC
%token          NEGLECT_KE
%token          NEGLECT_PE
%token          BOILING_ISOTHERM
%token          INTERFACIAL_HEAT_TRANSFER
%token          ZONE_PRESSURE
%token          NEGLECT_WALL_SHEAR
%token          RESERVOIR_PRESSURE
%token          ZONE_PRESSURE_DROP
%token          NOT_SPECIFIED
%token          NONE
%token          ZERO
%token          USER_DEFINED_SUBMODEL
%token          GRAVITY
%token          EXTERNALFORCE
%token          RESERVOIR_COMPOSITION
%token          ZONE_COMPOSITION
%token          PHASE_COMPOSITION
%token          REACTION
%token          DIFFUSION
%token          INTERFACIAL_MASS_TRANSFER
%token          EQUILIBRIUM
%token          BOILING
%token          ABSORPTION
%token          ZONE_EXTENT
%token          XXFIXEDXX
%token          VARIABLE
%token          ZONE_VOLUME_FRACTION
%token          PHASE_VOLUME_FRACTION
%token          PHASE_COMPOSITION
%token          ZONE_VELOCITY
%token          SUPERFICIAL_VELOCITY
%token          DIFFUSIVE_FLOW
%token          EXACT
%token          INTERNAL_RELATIVE_MOTION
%token          ZONE_ORIENTATION
%token          RELATIVE_PHASE_MOTION
%token          UPWARD_FLOW
%token          DOWNWARD_FLOW
%token          MIXTURE_BULK_FLOW
%token          BULK_FLOW
%token          NO_BULK_FLOW
%token          PORT_MASS_FLOWRATE
%token          PHASE_DEPLETION
%token          VOLUME_GAIN
%token          CONTINUOUS_PHASE
%token          DISPERSED_PHASE
%token          STATIONARY
%token          FLOW_REGIME
%token          DISPERSED
%token          SLUG
%token          CHURN
%token          ANNULAR
%token          END

%token <sval> IDENTIFIER
%token <dval> NUMBER
%token        ARROW
%token        ';'
%token        ':'
```

```
%token          '['
%token          ']'
%token          ','
%token          '.'
%token          '='


%type<SS_set>           yyDynamicsBlock

%%

/* GRAMMAR DEFINITION */

/* A Complete Input */

yyProcess               : yyProcessBlock yyPhaseBlock yyVesselBlock yyReservoirBlock
                                         yyConnectionBlock yyFlowsheetBlock
                        | yyProcessBlock yyPhaseBlock yyVesselBlock yyConnectionBlock
                                         yyFlowsheetBlock
                        ;

yyProcessBlock          : PROCESS ':' IDENTIFIER ';'
                        ;

/* PhaseBlock */

yyPhaseBlock            : DEFINE_PHASES  yyPhaseCollection END
                        ;

/* PhaseCollection */

yyPhaseCollection       :  yySinglePhase
                        |  yyPhaseCollection yySinglePhase
                        ;

/* SinglePhase */

yySinglePhase           : yySinglePhaseBlock  yyComponentBlock
                        ;

yySinglePhaseBlock      : PHASE ':' IDENTIFIER ':' SOLID ':' INCOMPRESSIBLE ';'
                        | PHASE ':' IDENTIFIER ':' LIQUID ':' INCOMPRESSIBLE ';'
                        | PHASE ':' IDENTIFIER ':' GAS ':' COMPRESSIBLE ';'
                        | PHASE ':' IDENTIFIER ':' GAS ':' INCOMPRESSIBLE ';'
                        ;

/* Component Block */

yyComponentBlock        :  DEFINE_COMPONENTS yyMultipleComponents END
                        ;

yyMultipleComponents    : yySingleComponent
                        | yyMultipleComponents yySingleComponent
                        ;

/* Single Component */

yySingleComponent       : COMPONENT ':' IDENTIFIER ':' IDENTIFIER ';'
                        | COMPONENT ':' IDENTIFIER ';'
                        ;

/* VesselBlock */

yyVesselBlock           : DEFINE_VESSELS yyVesselCollection END
                        ;

/* VesselCollection */

yyVesselCollection      : yySingleVessel
                        |  yyVesselCollection yySingleVessel
```

```
                        ;

yySingleVessel          : yySingleVesselBlock yyZoneBlock
                        ;

yySingleVesselBlock     : VESSEL ':' IDENTIFIER ':' IDENTIFIER ':' CLOSED ':' NUMBER ';'
                        | VESSEL ':' IDENTIFIER ':' IDENTIFIER ':' CLOSED ';'
                        | VESSEL ':' IDENTIFIER ':' IDENTIFIER  ':' OPEN ';'
                        | VESSEL ':' IDENTIFIER ':' IDENTIFIER  ';'
                        ;

yyZoneBlock             : DEFINE_ZONES yyZoneCollection END
                        ;

yyZoneCollection        : yySingleZoneBlock
                        | yyZoneCollection yySingleZoneBlock
                        ;

yySingleZoneBlock       : yyNoExtentZoneBlock
                        | yySinglePhaseLumpedZoneBlock
                        | yySinglePhaseOneDDuctZoneBlock
                        | yyTwoPhaseLumpedZoneBlock
                        | yyTwoPhaseDuctZoneBlock
                        ;

/*-------------------------------------------------------------------------*/
 /*                          No Extent Zone Description                     *
/*-------------------------------------------------------------------------*/

yyNoExtentZoneBlock     : yyNoExtentZoneDetailsBlock yyNoExtentPhyicalBehaviour
                        ;

yyNoExtentZoneDetailsBlock   : yyNoExtentZoneIdentifierBlock DEFINE_PORTS yyPortBlock END
                             ;

yyNoExtentZoneIdentifierBlock : ZONE ':' IDENTIFIER ';' GEOMETRY ':' NOEXTENT ';'
                              ;

/*-------------------------------------------------------------------------*/
 /*                      Single Phase Lumped Zone Description               */
/*-------------------------------------------------------------------------*/

yySinglePhaseLumpedZoneBlock            : yySinglePhaseLumpedZoneDetailsBlock
                                          yySinglePhaseLumpedPhyicalBehaviour
                                        ;

yySinglePhaseLumpedZoneDetailsBlock     : yySinglePhaseLumpedZoneIdentifierBlock
                                              DEFINE_PORTS yyPortBlock END
                                        ;

yySinglePhaseLumpedZoneIdentifierBlock : ZONE ':' IDENTIFIER ';' GEOMETRY ':' LUMPED ';'
                                         PHASE ':' IDENTIFIER ';'
                                       ;

/*-------------------------------------------------------------------------*/
 /*                      Single Phase OneD Duct Zone Description           *
/*-------------------------------------------------------------------------*

yySinglePhaseOneDDuctZoneBlock          : yySinglePhaseOneDDuctZoneDetailsBlock
                                          yySinglePhaseOneDDuctPhyicalBehaviour
                                        ;

yySinglePhaseOneDDuctZoneDetailsBlock : yySinglePhaseOneDDuctZoneIdentifierBlock
                                        DEFINE_BOUNDARIES yyOneDDuctBoundaryBlock END
                                        DEFINE_PORTS yyDistributedPortBlock END
                                      ;

yySinglePhaseOneDDuctZoneIdentifierBlock : yySinglePhaseOneDZoneIDBlock
                                           yyOrientationBlock
                                         ;
```

```
yySinglePhaseOneDZoneIDBlock : ZONE ':' IDENTIFIER ';' GEOMETRY ':' OneD ';' COORDINATES
                                ':' CARTESIAN ';' PHASE ':' IDENTIFIER ';'
                             ;

yyOrientationBlock           : ORIENTATION ':' NUMBER ';'
                             ;

yyOneDDuctBoundaryBlock      : yyOneDDuctBoundaryBlock yyOneDDuctBoundary
                             | yyOneDDuctBoundary
                             ;

yyOneDDuctBoundary           : BOUNDARY ':' IDENTIFIER ':' AXIAL ':'  NUMBER ';'
                             | BOUNDARY ':' IDENTIFIER ':' AXIAL ':' VARIABLE ';'
                             | BOUNDARY ':' IDENTIFIER ':' RADIAL ':' NUMBER ';'
                             ;

/*-------------------------------------------------------------------------*/
 /*                        Two Phase Lumped Zone Description                */
/*-------------------------------------------------------------------------*/

yyTwoPhaseLumpedZoneBlock          :     yyTwoPhaseLumpedZoneDetailsBlock
yyTwoPhaseLumpedPhysicalBehaviour
                                   ;

yyTwoPhaseLumpedZoneDetailsBlock   :     yyTwoPhaseZoneIdentifierBlock  DEFINE_PORTS
yyPortBlock END
                                   ;

yyTwoPhaseZoneIdentifierBlock      :     ZONE ':' IDENTIFIER ';' GEOMETRY ':' LUMPED
';' PHASE ':' IDENTIFIER ',' IDENTIFIER ';'
                                   ;

/*-------------------------------------------------------------------------*/
 /*                        Two Phase OneD Duct Zone Description             */
/*-------------------------------------------------------------------------*/

yyTwoPhaseDuctZoneBlock            :  yyTwoPhaseDuctZoneDetailsBlock
                                      yyTwoPhaseDuctPhysicalBehaviour
                                   ;

yyTwoPhaseDuctZoneDetailsBlock     :  yyTwoPhaseDuctZoneIdentifierBlock
                                      DEFINE_BOUNDARIES yyOneDDuctBoundaryBlock END
                                      DEFINE_PORTS yyDistributedPortBlock END
                                   ;

yyTwoPhaseDuctZoneIdentifierBlock  :     yyTwoPhaseDuctZoneIDBlock yyOrientationBlock
                                   ;

yyTwoPhaseDuctZoneIDBlock          :     ZONE ':' IDENTIFIER ';' GEOMETRY ':' OneD
';' COORDINATES ':' CYLINDRICAL ';' PHASE ':' IDENTIFIER ',' IDENTIFIER ';'
                                   ;

/*-------------------------------------------------------------------------*/
 /*                        Port Description                                 */
/*-------------------------------------------------------------------------*/

yyPortBlock     : yyPortBlock yySinglePort
                | yySinglePort
                ;

yySinglePort    : PORT ':' IDENTIFIER ';'
                | PORT ':' IDENTIFIER ':' INCLUDE_HYDROSTATIC_PRESSURE ';'
                | PORT ':' IDENTIFIER ':' IDENTIFIER ';'
                | PORT ':' IDENTIFIER ':' IDENTIFIER ':' PORT_MASS_FLOWRATE ':' ZERO ';'
                | PORT ':' IDENTIFIER ':' IDENTIFIER ':' PORT_MASS_FLOWRATE ':'
                  PHASE_DEPLETION ';'
                | PORT ':' IDENTIFIER ':' IDENTIFIER ':' VOLUME_GAIN ':' IDENTIFIER ';'
                ;

yyDistributedPortBlock  : yyDistributedPortBlock yySingleDistributedPort
                        | yySingleDistributedPort
```

```
                        ;

yySingleDistributedPort : PORT ':' IDENTIFIER ':' IDENTIFIER ';'
                        | PORT ':' IDENTIFIER ':' IDENTIFIER ':' IDENTIFIER ';'
                        ;

yyNoExtentPhyicalBehaviour              :  DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
yyNoExtentFlowrateBlock yyNoExtentExternalFactorsBlock yyNoExtentPressureDropBlock
yyNoExtentAssumptionsBlock END
                                        ;

yySinglePhaseLumpedPhyicalBehaviour   :  DEFINE_PHYSICOCHEMICAL_BEHAVIOUR yyDynamicsBlock
yyLumpedExtentBlock yySinglePhaseLumpedCompositionBlock yySinglePhaseLumpedTempBlock
yyLumpedPressureBlock END
                                        ;

yySinglePhaseOneDDuctPhyicalBehaviour : DEFINE_PHYSICOCHEMICAL_BEHAVIOUR yyDynamicsBlock
yySinglePhaseOneDDuctCompositionBlock yySinglePhaseOneDDuctTempBlock
yyOneDDuctPressureBlock yySinglePhaseOneDDuctVelocityBlock END
                                        ;

yyTwoPhaseLumpedPhysicalBehaviour     : DEFINE_PHYSICOCHEMICAL_BEHAVIOUR yyDynamicsBlock
yyLumpedExtentBlock yyTwoPhaseLumpedCompositionBlock yyTwoPhaseLumpedTemperatureBlock
yyLumpedPressureBlock yyTwoPhaseVolumeFractionBlock yyTwoPhaseLumpedVelocityBlock END
                                        ;

yyTwoPhaseDuctPhysicalBehaviour                 : DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
yyDynamicsBlock yyTwoPhaseDuctCompositionBlock yyTwoPhaseDuctTemperatureBlock
yyOneDDuctPressureBlock yyTwoPhaseVolumeFractionBlock yyTwoPhaseDuctVelocityBlock END
                                        ;

/*-------------------------------------------------------------------------*/
 /*                    No Extent Physical Behaviour Description             */
/*-------------------------------------------------------------------------*/

yyNoExtentFlowrateBlock          :  ZONE_FLOWRATE '=' NUMBER ';'
                                 |  ZONE_FLOWRATE ':' PRESSURE_DRIVEN ';'
                                 |  ZONE_FLOWRATE ':' CONTINUITY ';'
                                 ;

yyNoExtentExternalFactorsBlock   :  ZONE_HEAT_AND_WORK ':' HEATFLUX ':' EXTERNALWORK ';'
                                 |  ZONE_HEAT_AND_WORK ':' EXTERNALWORK ':' HEATFLUX  ';'
                                 |  ZONE_HEAT_AND_WORK ':' HEATFLUX ';'
                                 |  ZONE_HEAT_AND_WORK ':' EXTERNALWORK ';'
                                 |  ZONE_HEAT_AND_WORK ':' NONE ';'
                                 ;

yyNoExtentPressureDropBlock      : ZONE_PRESSURE_DROP '=' NUMBER ';'
                                 | ZONE_PRESSURE_DROP ':' NOT_SPECIFIED ';'
                                 | ZONE_PRESSURE_DROP ':' USER_DEFINED_SUBMODEL ';'
                                 ;

yyNoExtentAssumptionsBlock       : ZONE_ASSUMPTIONS ':' ISOTHERMAL ':' NEGLECT_KE ';'
                                 | ZONE_ASSUMPTIONS ':' ISOTHERMAL ';'
                                 | ZONE_ASSUMPTIONS ':' ISENTROPIC ':' NEGLECT_KE ';'
                                 | ZONE_ASSUMPTIONS ':' ISENTROPIC ';'
                                 | ZONE_ASSUMPTIONS ':' POLYTROPIC ':' NUMBER ':'
                                   NEGLECT_KE ';'
                                 | ZONE_ASSUMPTIONS ':' POLYTROPIC ':' NUMBER ';'
                                 | ZONE_ASSUMPTIONS ':' NEGLECT_KE ';'
                                 | ZONE_ASSUMPTIONS ':' NONE ';'
                                 ;

/*-------------------------------------------------------------------------*/
 /*                  Single Phase Lumped Behaviour Description              */
/*-------------------------------------------------------------------------*/

yyDynamicsBlock                  : ZONE_DYNAMICS ':' STEADY_STATE ';'
                                 | ZONE_DYNAMICS ':' DYNAMIC ';'
                                 ;
```

```
yySinglePhaseLumpedCompositionBlock   : ZONE_COMPOSITION ':' FIXED ';'
                                          yySinglePhaseCompositionDeclarationBlock
                                      | ZONE_COMPOSITION ':' VARIABLE ':' REACTION ';'
                                      | ZONE_COMPOSITION ':' VARIABLE ';'
                                      ;

yySinglePhaseCompositionDeclarationBlock  :  yySinglePhaseCompositionDeclarationBlock
                                             yySinglePhaseCompositionDeclaration
                                          |  yySinglePhaseCompositionDeclaration
                                          ;

yySinglePhaseCompositionDeclaration       : IDENTIFIER '=' NUMBER ';'
                                          ;

yySinglePhaseLumpedTempBlock  : ZONE_TEMPERATURE '=' NUMBER ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' EXTERNALWORK ':' NEGLECT_KE ':'
          NEGLECT_PE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' EXTERNALWORK ':' NEGLECT_KE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':'HEATFLUX ':' EXTERNALWORK ':' NEGLECT_PE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':'HEATFLUX ':' EXTERNALWORK ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' NEGLECT_KE ':' NEGLECT_PE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX  ':' NEGLECT_KE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' NEGLECT_PE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' EXTERNALWORK ':' NEGLECT_KE ':' NEGLECT_PE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' EXTERNALWORK ':' NEGLECT_KE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' EXTERNALWORK ':' NEGLECT_PE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' EXTERNALWORK ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' NEGLECT_KE ':' NEGLECT_PE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' NEGLECT_KE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ':' NEGLECT_PE ';'
        | ZONE_TEMPERATURE ':' VARIABLE ';'
                           ;

yyLumpedExtentBlock           : ZONE_EXTENT '=' NUMBER ';'
                              | ZONE_EXTENT ':' VARIABLE ';'
                              ;

yyLumpedPressureBlock         : ZONE_PRESSURE '=' NUMBER ';'
                              | ZONE_PRESSURE ':' VARIABLE ';'
                              ;

/*-------------------------------------------------------------------------*/
/*                    Single Phase 1-D Duct Behaviour Description          */
/*-------------------------------------------------------------------------*/

yySinglePhaseOneDDuctCompositionBlock : ZONE_COMPOSITION ':' FIXED ';'
                                          yySinglePhaseCompositionDeclarationBlock
              | ZONE_COMPOSITION ':' VARIABLE ':' DIFFUSION ':' REACTION ';'
              | ZONE_COMPOSITION ':' VARIABLE ':' REACTION ':' DIFFUSION  ';'
              | ZONE_COMPOSITION ':' VARIABLE ':' DIFFUSION ';'
              | ZONE_COMPOSITION ':' VARIABLE ':' REACTION ';'
              | ZONE_COMPOSITION ':' VARIABLE ';'
              ;


yySinglePhaseOneDDuctTempBlock : ZONE_TEMPERATURE '=' NUMBER ';'
                       | PHASE_TEMPERATURE ':' BOILING_ISOTHERM ';'
                       | ZONE_TEMPERATURE ':' VARIABLE ':' DIFFUSION ';'
                       | ZONE_TEMPERATURE ':' VARIABLE ':' DIFFUSION ':' NEGLECT_KE ';'
                       | ZONE_TEMPERATURE ':' VARIABLE ';'
                       | ZONE_TEMPERATURE ':' VARIABLE ':' NEGLECT_KE ';'
                       ;

yyOneDDuctPressureBlock       : ZONE_PRESSURE '=' NUMBER ';'
                              | ZONE_PRESSURE ':' VARIABLE ':' NEGLECT_WALL_SHEAR ';'
                              | ZONE_PRESSURE ':' VARIABLE ';'
                              ;

yySinglePhaseOneDDuctVelocityBlock    : ZONE_VELOCITY ':' BULK_FLOW ';'
```

```
                                    | ZONE_VELOCITY ':' SUPERFICIAL_VELOCITY ';'
                                    | ZONE_VELOCITY ':' DIFFUSIVE_FLOW ';'
                                    | ZONE_VELOCITY '=' NUMBER ';'
                                    ;
/*-------------------------------------------------------------------------*
 /*                    Two Phase Lumped Behaviour Description               *
/*-------------------------------------------------------------------------*

yyTwoPhaseLumpedCompositionBlock    : ZONE_COMPOSITION ';'
                      yyTwoPhaseLumpedCompositionBlock2 yyTwoPhaseLumpedCompositionBlock2
                      yyInternalMassTransfer
                                    | ZONE_COMPOSITION ';'
                      yyTwoPhaseLumpedCompositionBlock2 yyTwoPhaseLumpedCompositionBlock2
                                    ;

yyTwoPhaseLumpedCompositionBlock2   : yySinglePhaseLumpedCompDeclarationBlock
                                      yySinglePhaseCompositionDeclarationBlock
                                    | yySinglePhaseLumpedCompDeclarationBlock
                                    ;

yySinglePhaseLumpedCompDeclarationBlock : PHASE_COMPOSITION ':' IDENTIFIER ':' FIXED
                                        ';'
                      | PHASE_COMPOSITION ':' IDENTIFIER  ':' VARIABLE ':' REACTION ';'
                      | PHASE_COMPOSITION ':' IDENTIFIER ':' VARIABLE ';'
                                        ;

yyInternalMassTransfer              : INTERFACIAL_MASS_TRANSFER ':' EQUILIBRIUM ';'
        | INTERFACIAL_MASS_TRANSFER ':' BOILING ':' IDENTIFIER ARROW IDENTIFIER ';'
        | INTERFACIAL_MASS_TRANSFER ':' ABSORPTION ':' IDENTIFIER ARROW IDENTIFIER ':'
          yyComponentDeclaration ';'
        | INTERFACIAL_MASS_TRANSFER ':' NONE ';'
                                    ;

yyComponentDeclaration : yyComponentDeclaration ',' yysinglecomponentdeclaration
                       | yysinglecomponentdeclaration
                       ;

yysinglecomponentdeclaration        : IDENTIFIER
                                    ;

yyTwoPhaseLumpedTemperatureBlock    : ZONE_TEMPERATURE ':' yyInternalHeatTransfer
yyHeterogeneousTemperatureBlock yyHeterogeneousTemperatureBlock
                                    | ZONE_TEMPERATURE ';'
yyHeterogeneousTemperatureBlock yyHeterogeneousTemperatureBlock
                                    | yyHomogeneousTemperatureBlock
                                    ;

yyHeterogeneousTemperatureBlock     : PHASE_TEMPERATURE ':' IDENTIFIER '=' NUMBER ';'
                                    | PHASE_TEMPERATURE ':' BOILING_ISOTHERM ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' HEATFLUX ':' EXTERNALWORK ':'
      NEGLECT_KE ':' NEGLECT_PE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' HEATFLUX ':' EXTERNALWORK ':'
      NEGLECT_KE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':'HEATFLUX ':' EXTERNALWORK ':'
      NEGLECT_PE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':'HEATFLUX ':' EXTERNALWORK ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' HEATFLUX ':' NEGLECT_KE ':'
      NEGLECT_PE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' HEATFLUX ':' NEGLECT_KE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' HEATFLUX ':' NEGLECT_PE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' HEATFLUX ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' EXTERNALWORK ':' NEGLECT_KE ':'
      NEGLECT_PE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' EXTERNALWORK ':' NEGLECT_KE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' EXTERNALWORK ':' NEGLECT_PE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' EXTERNALWORK ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' NEGLECT_KE ':' NEGLECT_PE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' NEGLECT_KE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ':' NEGLECT_PE ';'
    | PHASE_TEMPERATURE ':' IDENTIFIER ':' VARIABLE ';'
```

```
yyInternalHeatTransfer                  : INTERFACIAL_HEAT_TRANSFER ';'
                                        ;

yyHomogeneousTemperatureBlock           : ZONE_TEMPERATURE '=' NUMBER ';'
                                        | ZONE_TEMPERATURE ':' BOILING_ISOTHERM ';'

    | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' EXTERNALWORK ':' NEGLECT_KE ':'
      NEGLECT_PE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' EXTERNALWORK ':' NEGLECT_KE ';'
    | ZONE_TEMPERATURE ':'VARIABLE ':' HEATFLUX ':' EXTERNALWORK ':' NEGLECT_PE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' EXTERNALWORK ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' NEGLECT_KE ':' NEGLECT_PE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' NEGLECT_KE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ':' NEGLECT_PE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' HEATFLUX ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' EXTERNALWORK ':' NEGLECT_KE ':' NEGLECT_PE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' EXTERNALWORK ':' NEGLECT_KE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' EXTERNALWORK ':' NEGLECT_PE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' EXTERNALWORK ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' NEGLECT_KE ':' NEGLECT_PE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' NEGLECT_KE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ':' NEGLECT_PE ';'
    | ZONE_TEMPERATURE ':' VARIABLE ';'
                                        ;

yyTwoPhaseVolumeFractionBlock   : PHASE_VOLUME_FRACTION ':' FIXED ';' IDENTIFIER
                                  '=' NUMBER ';' IDENTIFIER '=' NUMBER ';'
                                | PHASE_VOLUME_FRACTION ':' VARIABLE ';'
                                ;

yyTwoPhaseLumpedVelocityBlock   : ZONE_VELOCITY ':' STATIONARY ';'
                                | ZONE_VELOCITY ';' yyLumpedTwoPhaseOrientation
      yyMixtureBulkFlow yyInternalRelativeMotion yyFlowRegimeBlock  yyPhaseMotionBlock
                                 yyPhaseMotionBlock
                                | ZONE_VELOCITY ';' yyMixtureBulkFlow
      yyInternalRelativeMotion yyFlowRegimeBlock  yyPhaseMotionBlock yyPhaseMotionBlock
                                ;

yyMixtureBulkFlow               : MIXTURE_BULK_FLOW ':' BULK_FLOW ';'
                                | MIXTURE_BULK_FLOW ':' NO_BULK_FLOW ';'
                                ;

yyInternalRelativeMotion        : INTERNAL_RELATIVE_MOTION ':' NONE ';'
                                | INTERNAL_RELATIVE_MOTION ':' RELATIVE_PHASE_MOTION ';'
                                ;

yyFlowRegimeBlock               : FLOW_REGIME ':' BUBBLE ';'
                                | FLOW_REGIME ':' SLUG ';'
                                | FLOW_REGIME ':' CHURN ';'
                                | FLOW_REGIME ':' ANNULAR ';'
                                ;

yyLumpedTwoPhaseOrientation     : ZONE_ORIENTATION ':' VERTICAL ';'
                                | ZONE_ORIENTATION ':' HORIZONTAL ';'
                                ;

yyPhaseMotionBlock              : CONTINUOUS_PHASE ':' IDENTIFIER ';'
                                | CONTINUOUS_PHASE ':' IDENTIFIER ':' DOWNWARD_FLOW ';'
                                | CONTINUOUS_PHASE ':' IDENTIFIER ':' UPWARD_FLOW ';'
                                | CONTINUOUS_PHASE ':' IDENTIFIER ':' STATIONARY ';'
                                | DISPERSED_PHASE ':' IDENTIFIER ';'
                                | DISPERSED_PHASE ':' IDENTIFIER ':' DOWNWARD_FLOW ';'
                                | DISPERSED_PHASE ':' IDENTIFIER ':' UPWARD_FLOW ';'
                                | DISPERSED_PHASE ':' IDENTIFIER ':' STATIONARY ';'
                                ;

/*-------------------------------------------------------------------------*
 /*                  Two Phase 1-D Duct Behaviour Description
*/
/*-------------------------------------------------------------------------*
```

```
yyTwoPhaseDuctCompositionBlock  : ZONE_COMPOSITION ';' yyTwoPhaseDuctCompositionBlock2
                                  yyTwoPhaseDuctCompositionBlock2  yyInternalMassTransfer
                                | ZONE_COMPOSITION ';' yyTwoPhaseDuctCompositionBlock2
                                  yyTwoPhaseDuctCompositionBlock2
                                ;

yyTwoPhaseDuctCompositionBlock2       : yySinglePhaseDuctCompDeclarationBlock
                                        yySinglePhaseCompositionDeclarationBlock
                                      | yySinglePhaseDuctCompDeclarationBlock
                                      ;

yySinglePhaseDuctCompDeclarationBlock : PHASE_COMPOSITION ':' IDENTIFIER ':' FIXED ';'
        | PHASE_COMPOSITION ':' IDENTIFIER  ':' VARIABLE ':' DIFFUSION ':' REACTION ';'
        | PHASE_COMPOSITION ':' IDENTIFIER  ':' VARIABLE ':' DIFFUSION ';'
        | PHASE_COMPOSITION ':' IDENTIFIER  ':' VARIABLE ':' REACTION ';'
        | PHASE_COMPOSITION ':' IDENTIFIER ':' VARIABLE ';'
                                ;

yyTwoPhaseDuctTemperatureBlock        : ZONE_TEMPERATURE ':' yyInternalHeatTransfer
              yyHeterogeneousDuctTemperatureBlock yyHeterogeneousDuctTemperatureBlock
                                      | ZONE_TEMPERATURE ';'
              yyHeterogeneousDuctTemperatureBlock yyHeterogeneousDuctTemperatureBlock
                                      | yyHomogeneousDuctTemperatureBlock
                                      ;

yyHeterogeneousDuctTemperatureBlock   : PHASE_TEMPERATURE '=' NUMBER ';'
        | PHASE_TEMPERATURE ':' BOILING_ISOTHERM ';'
        | PHASE_TEMPERATURE ':' VARIABLE ':' DIFFUSION ';'
        | PHASE_TEMPERATURE ':' VARIABLE ':' DIFFUSION ':' NEGLECT_KE ';'
        | PHASE_TEMPERATURE ':' VARIABLE ';'
        | PHASE_TEMPERATURE ':' VARIABLE ':' NEGLECT_KE ';'
                                ;

yyHomogeneousDuctTemperatureBlock     : ZONE_TEMPERATURE '=' NUMBER ';'
        | ZONE_TEMPERATURE ':' BOILING_ISOTHERM ';'
        | ZONE_TEMPERATURE  ':' VARIABLE ':' DIFFUSION ';'
        | ZONE_TEMPERATURE  ':' VARIABLE ':' DIFFUSION ':' NEGLECT_KE ';'
        | ZONE_TEMPERATURE  ':' VARIABLE  ';'
        | ZONE_TEMPERATURE  ':' VARIABLE  ';' ZONE_ASSUMPTIONS ':' NEGLECT_KE ';'
                                ;

yyTwoPhaseDuctVelocityBlock           : ZONE_VELOCITY ':' STATIONARY ';'
                                      | ZONE_VELOCITY ';' yyMixtureBulkFlow
                                        yyInternalRelativeMotion
                                      | ZONE_VELOCITY ';' yyMixtureBulkFlow
yyInternalRelativeMotion  yyDistributedPhaseMotionBlock yyDistributedPhaseMotionBlock
                                      | ZONE_VELOCITY ';' yyMixtureBulkFlow
yyInternalRelativeMotion  yyDistributedPhaseMotionBlock yyDistributedPhaseMotionBlock
yyFlowRegimeBlock
                                ;

yyDistributedPhaseMotionBlock         : CONTINUOUS_PHASE ':' IDENTIFIER ';'
                        | CONTINUOUS_PHASE ':' IDENTIFIER ':' STATIONARY ';'
                        | CONTINUOUS_PHASE ':' IDENTIFIER ':' BULK_FLOW ';'
                        | CONTINUOUS_PHASE ':' IDENTIFIER ':' SUPERFICIAL_VELOCITY ';'
                        | DISPERSED_PHASE ':' IDENTIFIER ';'
                        | DISPERSED_PHASE ':' IDENTIFIER ':' STATIONARY ';'
                                ;

/* Reservoir Block */

yyReservoirBlock                : DEFINE_RESERVOIRS yyReservoirCollection END
                                ;

/* Reservoir Collection */

yyReservoirCollection           : yySingleReservoir
                                | yyReservoirCollection yySingleReservoir
                                ;
```

```
/* Single Reservoir */

yySingleReservoir              : yyReservoirIdentifierBlock yyReservoirPhyicalBehaviour
                               ;

yyReservoirIdentifierBlock     : RESERVOIR ':' IDENTIFIER ':' PHASE ':' IDENTIFIER  ';'
                               | RESERVOIR ':' IDENTIFIER ';'
                               ;

yyReservoirPhyicalBehaviour    : DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                                 yyReservoirAttributeBlock  END
                               ;

yyReservoirAttributeBlock      : yyReservoirAttributeBlock yySingleReservoirAttribute
                               | yySingleReservoirAttribute
                               ;

yySingleReservoirAttribute     : yyRerservoirComposition
                               | yyReservoirTemperatureBlock
                               | yyReservoirPressureBlock
                               ;

yyRerservoirComposition        : RESERVOIR_COMPOSITION ';' yyReservoirCompositionBlock
                               ;

yyReservoirCompositionBlock    : yyReservoirCompositionBlock
                                 yyReservoirCompositionDeclaration
                               | yyReservoirCompositionDeclaration
                               ;

yyReservoirCompositionDeclaration     : IDENTIFIER '=' NUMBER ';'
                               ;

yyReservoirTemperatureBlock    : RESERVOIR_TEMPERATURE '=' NUMBER ';'
                               ;

yyReservoirPressureBlock       : RESERVOIR_PRESSURE '=' NUMBER ';'
                               ;

/* Connection Block */

yyConnectionBlock              : DEFINE_CONNECTIONS yyConnectionCollection END
                               ;

/* Collection Connection */

yyConnectionCollection         : yySingleConnection
                               | yyConnectionCollection yySingleConnection
                               ;

yySingleConnection             : CONNECTION ':' IDENTIFIER ':' PRESSURE_CONNECTION ';'
                | CONNECTION ':' IDENTIFIER ':' OPEN ':' MATERIAL ':' IDENTIFIER ':'
                  EQUATE_PRESSURE ';'
                | CONNECTION ':' IDENTIFIER ':' OPEN ':' MATERIAL ':' IDENTIFIER ';'
                | CONNECTION ':' IDENTIFIER ':' OPEN ':' ENERGY ':' NUMBER  ';'
                | CONNECTION ':' IDENTIFIER ':' CLOSED ':' ENERGY ';'
                | CONNECTION ':' IDENTIFIER ':' CLOSED ':' MATERIAL ':' IDENTIFIER ';'
                | CONNECTION ':' IDENTIFIER ':' TWOPHASE_OPEN ':' IDENTIFIER ','
                               ;

yyFlowsheetBlock               : DEFINE_FLOWSHEET  yyMultipleConnections END
                               ;

yyMultipleConnections          : yySingleConnection
                               | yyMultipleConnections yySingleConnection
                               ;

/* Single Connection */

yySingleConnection             : CONNECTION ':' IDENTIFIER ':' IDENTIFIER '.' IDENTIFIER
                                 ARROW IDENTIFIER '.' IDENTIFIER ';'
```

```
        | CONNECTION ':' IDENTIFIER ':' IDENTIFIER ARROW IDENTIFIER '.' IDENTIFIER ';'
        | CONNECTION ':' IDENTIFIER ':' IDENTIFIER '.' IDENTIFIER ARROW IDENTIFIER ';'
                                ;
%%
```

Appendix J

The *COMPASS* Input Files

J.1 The *COMPASS* Input File for the Buffer Tank Simulation

```
PROCESS : Buffer_Tank_system ;

DEFINE_PHASES
        PHASE : product_stream : LIQUID : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Methanol : a ;
                        COMPONENT : Water : b ;
                END #component_section
        END #phase section

DEFINE_VESSELS
        VESSEL : InletPipe : P101  ;
        DEFINE_ZONES
                ZONE : Z1 ;
                GEOMETRY : NOEXTENT ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_FLOWRATE = 50 ;
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
        END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
END #zone section

VESSEL : Tank : T101  ;
DEFINE_ZONES
        ZONE : Z2 ;
        GEOMETRY : LUMPED ;
        PHASE : product_stream ;
        DEFINE_PORTS
                PORT : P1 ;
                PORT : P2 : INCLUDE_HYDROSTATIC_PRESSURE ;
        END #DEFINE_PORTS
        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                ZONE_DYNAMICS : DYNAMIC ;
                ZONE_EXTENT : VARIABLE ;
                ZONE_COMPOSITION : FIXED ;
                        Methanol = 0.5 ;
                        Water = 0.5 ;
                ZONE_TEMPERATURE = 293 ;
                ZONE_PRESSURE = 1.013e+5 ;
        END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION
END #zone section

VESSEL : OutletPipe1 : P102  ;
DEFINE_ZONES
        ZONE : Z3 ;
        GEOMETRY : NOEXTENT ;
        DEFINE_PORTS
                PORT : P1 ;
                PORT : P2 ;
        END #DEFINE_PORTS
        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
```

```
                ZONE_FLOWRATE : PRESSURE_DRIVEN ;
                ZONE_HEAT_AND_WORK : NONE ;
                ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                ZONE_ASSUMPTIONS : NEGLECT_KE ;
        END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
END #zone section

VESSEL : Pump : X101  ;
        DEFINE_ZONES
                ZONE : Z4 ;
                GEOMETRY : NOEXTENT ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_FLOWRATE : PRESSURE_DRIVEN ;
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP = -8.5E5 ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

VESSEL : OutletPipe2 : P103  ;
        DEFINE_ZONES
                ZONE : Z5 ;
                GEOMETRY : NOEXTENT ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_FLOWRATE : PRESSURE_DRIVEN ;
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

VESSEL : Splitter : S101  ;
        DEFINE_ZONES
                ZONE : Z6 ;
                        GEOMETRY : NOEXTENT ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                        PORT : P3 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_FLOWRATE : CONTINUITY  ;
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP = 0 ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

VESSEL : OutletPipe3 : P104  ;
        DEFINE_ZONES
                ZONE : Z7 ;
                        GEOMETRY : NOEXTENT ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_FLOWRATE : PRESSURE_DRIVEN ;
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
END #zone section
```

```
VESSEL : OutletPipe4 : P105  ;
        DEFINE_ZONES
                ZONE : Z8 ;
                GEOMETRY : NOEXTENT ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_FLOWRATE : PRESSURE_DRIVEN ;
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                        END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
                END #zone section

END #vessel section

DEFINE_RESERVOIRS

        RESERVOIR : R101 : PHASE : product_stream ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_COMPOSITION ;
                                Methanol = 0.5 ;
                                Water = 0.5 ;
                        RESERVOIR_TEMPERATURE = 293 ;
                        RESERVOIR_PRESSURE = 1.013e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

        RESERVOIR : R102 : PHASE : product_stream ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_PRESSURE = 1.013e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

        RESERVOIR : R103 : PHASE : product_stream ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_PRESSURE = 1.013e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

END #reservoir section

DEFINE_CONNECTIONS

        CONNECTION : C101 : OPEN : MATERIAL : product_stream ;
        CONNECTION : C102 : OPEN : MATERIAL : product_stream ;
        CONNECTION : C103 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;
        CONNECTION : C104 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;
        CONNECTION : C105 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;
        CONNECTION : C106 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;
        CONNECTION : C107 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;
        CONNECTION : C108 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;
        CONNECTION : C109 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;
        CONNECTION : C110 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;

END #connection section

DEFINE_FLOWSHEET

        CONNECTION : C101 : R101 -> Z1.P1 ;
        CONNECTION : C102 : Z1.P2 -> Z2.P1 ;
        CONNECTION : C103 : Z2.P2 -> Z3.P1 ;
        CONNECTION : C104 : Z3.P2 -> Z4.P1 ;
        CONNECTION : C105 : Z4.P2 -> Z5.P1 ;
        CONNECTION : C106 : Z5.P2 -> Z6.P1 ;
        CONNECTION : C107 : Z6.P2 -> Z7.P1 ;
        CONNECTION : C108 : Z7.P2 -> R102 ;
        CONNECTION : C109 : Z6.P3 -> Z8.P1 ;
        CONNECTION : C110 : Z8.P2 -> R103 ;

END #flow sheet section
```

J.2 The *COMPASS* Input File for the Gas Phase Continuous Stirred Tank Reactor Simulation

```
PROCESS : Gas_CSTR_system ;

DEFINE_PHASES
        PHASE : raw_materials : GAS : COMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Cyclopropane : a ;
                END #component_section

        PHASE : product_stream : GAS : COMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Cyclopropane : a ;
                        COMPONENT : Propylene : b ;
                END #component_section
END #phase section

DEFINE_VESSELS
        VESSEL : InletPipe : P101 ;
        DEFINE_ZONES
                ZONE : Z1 ;
                        GEOMETRY : NOEXTENT ;
                        DEFINE_PORTS
                                PORT : P1 ;
                                PORT : P2 ;
                        END #DEFINE_PORTS
                        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                                ZONE_FLOWRATE = 1.0e-2 ;
                                ZONE_HEAT_AND_WORK : NONE ;
                                ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                                ZONE_ASSUMPTIONS : NEGLECT_KE ;
                        END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
                END #zone section

VESSEL : Reactor : R101 : CLOSED ;
        DEFINE_ZONES
                ZONE : Z2 ;
                GEOMETRY : LUMPED ;
                PHASE : product_stream ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_DYNAMICS : DYNAMIC ;
                        ZONE_EXTENT = 2 ;
                        ZONE_COMPOSITION : VARIABLE : REACTION ;
                        ZONE_TEMPERATURE : VARIABLE : NEGLECT_KE : NEGLECT_PE ;
                        ZONE_PRESSURE : VARIABLE ;
                END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

VESSEL : OutletPipe1 : P102  ;
        DEFINE_ZONES
                ZONE : Z3 ;
                GEOMETRY : NOEXTENT ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_FLOWRATE : PRESSURE_DRIVEN ;
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
END #zone section
```

```
END #vessel section


DEFINE_RESERVOIRS

        RESERVOIR : R101 : PHASE : raw_materials ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_COMPOSITION ;
                                Cyclopropane = 1.0 ;
                        RESERVOIR_TEMPERATURE = 780 ;
                        RESERVOIR_PRESSURE = 1.0e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

        RESERVOIR : R102 : PHASE : product_stream ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_PRESSURE = 0.9e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

END #reservoir section

DEFINE_CONNECTIONS

        CONNECTION : C101 : OPEN : MATERIAL : raw_materials ;
        CONNECTION : C102 : OPEN : MATERIAL : raw_materials ;
        CONNECTION : C103 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;
        CONNECTION : C104 : OPEN : MATERIAL : product_stream : EQUATE_PRESSURE ;


END #connection section

DEFINE_FLOWSHEET

        CONNECTION : C101 : R101 -> Z1.P1 ;
        CONNECTION : C102 : Z1.P2 -> Z2.P1 ;
        CONNECTION : C103 : Z2.P2 -> Z3.P1 ;
        CONNECTION : C104 : Z3.P2 -> R102 ;

END #flow sheet section
```

J.3 The *COMPASS* Input File for the Flash Vessel Simulation

```
PROCESS : Flash_in_equilibrium ;

DEFINE_PHASES
        PHASE : GP : GAS : COMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Propane : a ;
                        COMPONENT : Pentane : b ;
                END #component_section

        PHASE : LP : LIQUID : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Propane : a ;
                        COMPONENT : Pentane : b ;
                END #component_section
END #phase section

DEFINE_VESSELS
        VESSEL : InletPipe : P101  ;
                DEFINE_ZONES
                        ZONE : Z1 ;
                        GEOMETRY : NOEXTENT ;
                        DEFINE_PORTS
                                PORT : P1 ;
                                PORT : P2 ;
                        END #DEFINE_PORTS
                        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                                ZONE_FLOWRATE = 10 ;
```

```
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

VESSEL : Flash_Vessel : FV101 : CLOSED : 5 ;
        DEFINE_ZONES
                ZONE : Z2 ;
                GEOMETRY : LUMPED ;
                PHASE : GP, LP ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                        PORT : P3 : INCLUDE_HYDROSTATIC_PRESSURE ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_DYNAMICS : DYNAMIC ;
                        ZONE_EXTENT = 5 ;
                        ZONE_COMPOSITION  ;
                                PHASE_COMPOSITION : LP : VARIABLE ;
                                PHASE_COMPOSITION : GP : VARIABLE ;
                                INTERFACIAL_MASS_TRANSFER : EQUILIBRIUM ;
                        ZONE_TEMPERATURE : VARIABLE : NEGLECT_KE : NEGLECT_PE  ;
                        ZONE_PRESSURE : VARIABLE ;
                        PHASE_VOLUME_FRACTION : VARIABLE ;
                        ZONE_VELOCITY : STATIONARY ;
                END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

VESSEL : Outlet_Gas_Pipe : P102  ;
        DEFINE_ZONES
                ZONE : Z3 ;
                GEOMETRY : NOEXTENT ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_FLOWRATE : PRESSURE_DRIVEN ;
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

VESSEL : Outlet_Liquid_Pipe : P103  ;
        DEFINE_ZONES
                ZONE : Z4 ;
                GEOMETRY : NOEXTENT ;
                DEFINE_PORTS
                        PORT : P1 ;
                        PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_FLOWRATE : PRESSURE_DRIVEN ;
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
                END #zone section

END #vessel section

DEFINE_RESERVOIRS

        RESERVOIR : R101 : PHASE : LP ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_COMPOSITION ;
                                Propane = 0.5 ;
                                Pentane = 0.5 ;
```

```
                RESERVOIR_TEMPERATURE = 293 ;
                RESERVOIR_PRESSURE = 10.13e5 ;
        END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

RESERVOIR : R102 : PHASE : GP ;
        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                RESERVOIR_PRESSURE = 1.013e5 ;
        END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

RESERVOIR : R103 : PHASE : LP ;
        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                RESERVOIR_PRESSURE = 1.013e5 ;
        END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

END #reservoir section

DEFINE_CONNECTIONS

        CONNECTION : C101 : OPEN : MATERIAL : LP : EQUATE_PRESSURE ;
        CONNECTION : C102 : OPEN : MATERIAL : LP : EQUATE_PRESSURE ;
        CONNECTION : C103 : OPEN : MATERIAL : GP : EQUATE_PRESSURE ;
        CONNECTION : C104 : OPEN : MATERIAL : LP : EQUATE_PRESSURE ;
        CONNECTION : C105 : OPEN : MATERIAL : GP : EQUATE_PRESSURE ;
        CONNECTION : C106 : OPEN : MATERIAL : LP : EQUATE_PRESSURE ;

END #connection section

DEFINE_FLOWSHEET

        CONNECTION : C101 : R101 -> Z1.P1 ;
        CONNECTION : C102 : Z1.P2 -> Z2.P1 ;
        CONNECTION : C103 : Z2.P2 -> Z3.P1  ;
        CONNECTION : C104 : Z2.P3 -> Z4.P1 ;
        CONNECTION : C105 : Z3.P2 -> R102 ;
        CONNECTION : C106 : Z4.P2 -> R103 ;

END #flow sheet section
```

J.4 The *COMPASS* Input File for the Plug Flow Reactor Simulation

```
PROCESS : Gas_PFR ;

DEFINE_PHASES
        PHASE : Product_stream : GAS : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Butene : a ;
                        COMPONENT : Butyne : b ;
                        COMPONENT : Water : c ;
                END #component_section

        PHASE : StainlessSteel : SOLID : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : StainlessSteel : a ;
                END #component_section
END #phase section

DEFINE_VESSELS
        VESSEL : InletPipe1 : P101  ;
                DEFINE_ZONES
                        ZONE : Z1 ;
                        GEOMETRY : NOEXTENT ;
                        DEFINE_PORTS
                                PORT : P1 ;
                                PORT : P2 ;
                        END #DEFINE_PORTS
                        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                                ZONE_FLOWRATE = 0.002 ;
```

```
                        ZONE_HEAT_AND_WORK : NONE ;
                        ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                        ZONE_ASSUMPTIONS : NEGLECT_KE ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

VESSEL : Reactor1 : P101  ;
        DEFINE_ZONES
                ZONE : Z2 ;
                GEOMETRY : 1-D ;
                COORDINATES : CARTESIAN ;
                PHASE : Product_stream ;
                ORIENTATION : 0 ;
                DEFINE_BOUNDARIES
                        BOUNDARY : Inlet  : AXIAL  : 0 ;
                        BOUNDARY : Outlet : AXIAL  : 10 ;
                        BOUNDARY : Wall   : RADIAL : 0.3 ;
                END #DEFINE_BOUNDARIES
                DEFINE_PORTS
                        PORT : P1 : Inlet ;
                        PORT : P2 : Outlet ;
                        PORT : P3 : Wall ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_DYNAMICS : DYNAMIC ;
                        ZONE_COMPOSITION : VARIABLE : REACTION ;
                        ZONE_TEMPERATURE : VARIABLE :  NEGLECT_KE ;
                        ZONE_PRESSURE = 1.0E5 ;
                        ZONE_VELOCITY : BULK_FLOW ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION

ZONE : Z3 ;
        GEOMETRY : LUMPED ;
        PHASE : StainlessSteel ;
        DEFINE_PORTS
                PORT : P1 ;
                PORT : P2 ;
                END #DEFINE_PORTS
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        ZONE_DYNAMICS : DYNAMIC ;
                        ZONE_EXTENT = 0.09734 ; #the volume of the tube wall
                        ZONE_COMPOSITION : FIXED ;
                                StainlessSteel = 1 ;
                        ZONE_TEMPERATURE : VARIABLE : NEGLECT_KE : NEGLECT_PE ;
                        ZONE_PRESSURE = 1.013e+5 ;
                END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section
END #vessel section

DEFINE_RESERVOIRS
        RESERVOIR : R101 : PHASE : Product_stream ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_COMPOSITION ;
                                Butene = 0.1 ;
                                Butyne = 0.0 ;
                                Water = 0.9 ;
                        RESERVOIR_TEMPERATURE = 820 ;
                        RESERVOIR_PRESSURE = 1.013e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

        RESERVOIR : R102  ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_PRESSURE = 1.013e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

        RESERVOIR : R103 ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_TEMPERATURE = 800 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
END #reservoir section
```

```
DEFINE_CONNECTIONS

        CONNECTION : C101 : OPEN : MATERIAL : Product_stream : EQUATE_PRESSURE  ;
        CONNECTION : C102 : OPEN : MATERIAL : Product_stream : EQUATE_PRESSURE ;
        CONNECTION : C103 : OPEN : MATERIAL : Product_stream ;
        CONNECTION : C104 : CLOSED : ENERGY ;
        CONNECTION : C105 : CLOSED : ENERGY ;

END #connection section

DEFINE_FLOWSHEET

        CONNECTION : C101 : R101 -> Z1.P1  ;
        CONNECTION : C102 : Z1.P2 -> Z2.P1 ;
        CONNECTION : C103 : Z2.P2 -> R102 ;
        CONNECTION : C104 : Z2.P3 -> Z3.P1 ;
        CONNECTION : C105 : Z3.P2 -> R103 ;

END #flow sheet section
```

J.5 The *COMPASS* Input File for the Drum Boiler Simulation

```
PROCESS : DRUM_BOILER ;

DEFINE_PHASES
        PHASE : Water : LIQUID : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : H2O : a ;
                END #component_section

        PHASE : Steam : GAS : COMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : H2O : a ;
                END #component_section
END #phase section

DEFINE_VESSELS
        VESSEL : InletPipe : P101  ;
        DEFINE_ZONES
                ZONE : Z1 ;
                        GEOMETRY : NOEXTENT ;
                        DEFINE_PORTS
                                PORT : PA1 ;
                                PORT : PA2 ;
                        END #DEFINE_PORTS
                        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                                ZONE_FLOWRATE = 31.41 ;
                                ZONE_HEAT_AND_WORK : NONE ;
                                ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                                ZONE_ASSUMPTIONS : NEGLECT_KE ;
                        END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

        VESSEL : DRUM : V101 : CLOSED : 20.36 ;
        DEFINE_ZONES
                ZONE : Z2 ;
                        GEOMETRY : LUMPED ;
                                PHASE : Water, Steam ;
                        DEFINE_PORTS
                                PORT : PB1 ;
                                PORT : PB2 ;
                                PORT : PB3 ;
                                PORT : PB4 ;
                                PORT : PB5 ;
                        END #DEFINE_PORTS
                        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                                ZONE_DYNAMICS : DYNAMIC ;
                                ZONE_EXTENT = 20.36 ;
                                ZONE_COMPOSITION ;
```

```
                        PHASE_COMPOSITION : Water : FIXED ;
                                H2O = 1 ;
                        PHASE_COMPOSITION : Steam : FIXED ;
                                H2O = 1 ;
                        ZONE_TEMPERATURE : BOILING_ISOTHERM ;
                        ZONE_PRESSURE : VARIABLE ;
                        PHASE_VOLUME_FRACTION : VARIABLE ;
                        ZONE_VELOCITY : STATIONARY ;
                END# PHYSICOCHEMICAL_BEHAVIOUR_SECTION
END #DEFINE ZONES

VESSEL : Boiling_Tube : P104 ;
DEFINE_ZONES
        ZONE : Z3 ;
        GEOMETRY : 1-D ;
        COORDINATES : CARTESIAN ;
        PHASE : Water, Steam ;
        ORIENTATION : 90 ;
        DEFINE_BOUNDARIES
                BOUNDARY : Inlet : AXIAL  : 0  ;
                BOUNDARY : Outlet : AXIAL : 25 ;
                BOUNDARY : Wall : RADIAL  : 0.02504 ;
        END #DEFINE_BOUNDARIES
        DEFINE_PORTS
                PORT : PD1 : Inlet ;
                PORT : PD2 : Outlet : Water ;
                PORT : PD3 : Outlet : Steam ;
        END #DEFINE_PORTS
        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                ZONE_DYNAMICS : STEADY_STATE ;
                ZONE_COMPOSITION ;
                        PHASE_COMPOSITION : Water : FIXED ;
                                H2O = 1 ;
                        PHASE_COMPOSITION : Steam : FIXED ;
                                H2O = 1 ;
                INTERFACIAL_MASS_TRANSFER : BOILING : Water -> Steam ;
                ZONE_TEMPERATURE : BOILING_ISOTHERM ;
                ZONE_PRESSURE : VARIABLE ;
                PHASE_VOLUME_FRACTION : VARIABLE ;
                ZONE_VELOCITY ;
                        MIXTURE_BULK_FLOW : BULK_FLOW ;
                        INTERNAL_RELATIVE_MOTION : NONE ;
                        CONTINUOUS_PHASE : Water  ;
                        DISPERSED_PHASE : Steam ;
                END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
END #zone section

VESSEL : Downcomer : P103  ;
DEFINE_ZONES
        ZONE : Z4 ;
                GEOMETRY : 1-D ;
                COORDINATES : CARTESIAN ;
                PHASE : Water ;
                ORIENTATION : 90 ;
                DEFINE_BOUNDARIES
                        BOUNDARY : Inlet  : AXIAL : 0  ;
                        BOUNDARY : Outlet : AXIAL : 25 ;
                        BOUNDARY : Wall : RADIAL : 1 ;
                END #DEFINE_BOUNDARIES
                DEFINE_PORTS
                PORT : PE1 : Inlet ;
                PORT : PE2 : Outlet ;
END #DEFINE_PORTS
DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                ZONE_DYNAMICS : STEADY_STATE ;
                ZONE_COMPOSITION : FIXED ;
                        H2O = 1 ;
                ZONE_TEMPERATURE : VARIABLE : NEGLECT_KE  ;
                ZONE_PRESSURE : VARIABLE : NEGLECT_WALL_SHEAR ;
                ZONE_VELOCITY : BULK_FLOW ;
END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
```

```
 END #zone section

VESSEL : OutletSteamPipe : P102  ;
DEFINE_ZONES
        ZONE : Z5 ;
        GEOMETRY : NOEXTENT ;
        DEFINE_PORTS
                PORT : PF1 ;
                PORT : PF2 ;
        END #DEFINE_PORTS
        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                ZONE_FLOWRATE = 27.778 ;
                ZONE_HEAT_AND_WORK : NONE ;
                ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                ZONE_ASSUMPTIONS : NEGLECT_KE ;
            END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section
END #vessel section

DEFINE_RESERVOIRS

        RESERVOIR : R101 : PHASE : Water ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_COMPOSITION ;
                                H2O = 1 ;
                        RESERVOIR_TEMPERATURE = 571.3 ;
                        RESERVOIR_PRESSURE = 8.4e6 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

        RESERVOIR : R102  ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_PRESSURE = 1.0e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

END #reservoir section

DEFINE_CONNECTIONS

# Inlet Water Feed Connections
        CONNECTION : C101 : OPEN : MATERIAL : Water ;
        CONNECTION : C102 : OPEN : MATERIAL : Water ;
# Downcomer Connections
        CONNECTION : C103 : OPEN : MATERIAL : Water : EQUATE_PRESSURE ;
        CONNECTION : C104 : OPEN : MATERIAL : Water : EQUATE_PRESSURE ;
# Connections to link the outlet of the Riser to the Drum
        CONNECTION : C105 : OPEN : MATERIAL : Water : EQUATE_PRESSURE ;
        CONNECTION : C106 : OPEN : MATERIAL : Steam ;
# Outlet steam Connection
        CONNECTION : C107 : OPEN : MATERIAL : Steam ;
        CONNECTION : C108 : OPEN : MATERIAL : Steam ;

END #connection section

DEFINE_FLOWSHEET

        CONNECTION : C101 : R101 -> Z1.PA1  ;
        CONNECTION : C102 : Z1.PA2 -> Z2.PB1 ;
        CONNECTION : C103 : Z2.PB2 -> Z4.PE1 ;
        CONNECTION : C104 : Z4.PE2 -> Z3.PD1 ;
        CONNECTION : C105 : Z3.PD2 -> Z2.PB3 ;
        CONNECTION : C106 : Z3.PD3 -> Z2.PB4 ;
        CONNECTION : C107 : Z2.PB5 -> Z5.PF1 ;
        CONNECTION : C108 : Z5.PF2 -> R102 ;

END #flow sheet section
```

J.6 The *COMPASS* Input File for the Adsorptive Reactor Simulation

```
PROCESS : Adsorptive_reactor ;

DEFINE_PHASES
        PHASE : Raw_materials : GAS : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Methylcyclohexane : a ;
                        COMPONENT : Inert : b ;
                END #component_section

        PHASE : Product : GAS : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Methylcyclohexane : a ;
                        COMPONENT : Toluene : b ;
                        COMPONENT : Hydrogen : c ;
                        COMPONENT : Inert : d ;
                END #component_section

        PHASE : Solid : SOLID : INCOMPRESSIBLE ;
                DEFINE_COMPONENTS
                        COMPONENT : Catalyst : a ;
                        COMPONENT : Toluene : b ;.
                END #component_section
END #phase section

DEFINE_VESSELS
        VESSEL : InletPipe1 : P101  ;
        DEFINE_ZONES
                ZONE : Z1 ;
                        GEOMETRY : NOEXTENT ;
                        DEFINE_PORTS
                                PORT : P1 ;
                                PORT : P2 ;
                        END #DEFINE_PORTS
                        DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                                ZONE_FLOWRATE = 0.001 ;
                                ZONE_HEAT_AND_WORK : NONE ;
                                ZONE_PRESSURE_DROP : NOT_SPECIFIED ;
                                ZONE_ASSUMPTIONS : NEGLECT_KE ;
                        END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
END #zone section

VESSEL : Reactor1 : P101  ;
DEFINE_ZONES
        ZONE : Z2 ;
                GEOMETRY : 1-D ;
                COORDINATES : CARTESIAN ;
                PHASE : Product, Solid ;
                ORIENTATION : 0 ;
                DEFINE_BOUNDARIES
                        BOUNDARY : Inlet : AXIAL : 0  ;
                        BOUNDARY : Outlet : AXIAL : 1 ;
                        BOUNDARY : Wall : RADIAL : 0.15 ;
                END #DEFINE BOUNDARY
                DEFINE_PORTS
                        PORT : P1 : Inlet : Product ;
                        PORT : P2 : Outlet ;
END #DEFINE_BOUNDARY
DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
        ZONE_DYNAMICS : DYNAMIC ;
        ZONE_COMPOSITION  ;
                PHASE_COMPOSITION : Product : VARIABLE : REACTION ;
                PHASE_COMPOSITION : Solid : VARIABLE  ;
        INTERFACIAL_MASS_TRANSFER : ABSORPTION : Product -> Solid :
                Toluene  ;
        ZONE_TEMPERATURE : VARIABLE : DIFFUSION : NEGLECT_KE ;
        ZONE_PRESSURE = 2.0E+5 ;
                PHASE_VOLUME_FRACTION : FIXED ;
```

```
                                        Product = 0.3;
                                        Solid = 0.7 ;
                                ZONE_VELOCITY ;
                                        MIXTURE_BULK_FLOW : BULK_FLOW ;
                                        INTERNAL_RELATIVE_MOTION : RELATIVE_PHASE_MOTION ;
                                        CONTINUOUS_PHASE : Product : SUPERFICIAL_VELOCITY ;
                                        DISPERSED_PHASE : Solid : STATIONARY ;
                        END # PHYSICOCHEMICAL_BEHAVIOUR_SECTION
        END #zone section

END #vessel section


DEFINE_RESERVOIRS

        RESERVOIR : R1 : PHASE : Raw_materials ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_COMPOSITION ;
                                Methylcyclohexane = 0.1 ;
                                Inert = 0.9 ;
                        RESERVOIR_TEMPERATURE = 300 ;
                        RESERVOIR_PRESSURE = 2.0e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

        RESERVOIR : R2 : PHASE : Raw_materials ;
                DEFINE_PHYSICOCHEMICAL_BEHAVIOUR
                        RESERVOIR_PRESSURE = 2.0e5 ;
                END # DEFINE_PHYSICOCHEMICAL_BEHAVIOUR

END #reservoir section

DEFINE_CONNECTIONS

        CONNECTION : C101 : OPEN : MATERIAL : Raw_materials : EQUATE_PRESSURE  ;
        CONNECTION : C102 : OPEN : MATERIAL : Raw_materials : EQUATE_PRESSURE ;
        CONNECTION : C103 : OPEN : MATERIAL : Product  ;

END #connection section

DEFINE_FLOWSHEET

        CONNECTION : C101 : R1 -> Z1.P1  ;
        CONNECTION : C102 : Z1.P2 -> Z2.P1 ;
        CONNECTION : C103 : Z2.P2 -> R2 ;

END #flow sheet section
```

Appendix K

Generated Models Written in TEXT

Within this appendix, the generated models for the six case studies in Chapter 7 are given. For the first three examples (the buffer-tank system, gas-phase CSTR and he adiabatic flash vessel), the *COMPASS* generated model is presented in text format. For the remaining three examples (a gaseous phase plug flow reactor, a drum boiler and a two-phase adsorptive reactor), the *COMPASS* generated gPROMS input file is given. The models presented in this chapter are copies of the generated models and no editing has been made other than reducing the font size and removing superfluous line breaks.

K.1 The Buffer Tank System Model (TEXT)

```
Process: Buffer_Tank_system

-------------------------------------------------------------
Phase Information
-------------------------------------------------------------

Buffer_Tank_system consists of 1 phases: product_stream

Phase name:        product_stream:
Phase Type:        Liquid
Compressibility:   Incompressibile
Component list:    Methanol which is represented by a
                   Water which is represented by b

-------------------------------------------------------------
Connection Information:
-------------------------------------------------------------

Buffer_Tank_system consists of 10 connections: C101, C102, C103, C104, C105, C106, C107, C108, C109,
C110

--------------------------
Connection: C101
--------------------------
Connection Type:                    Open Material Connection. Port / Reservoir Pressures are not
Equated
Transfers the following phases:     product_stream
Is linked to the following Zones:   R101, Z1
Upstream Zone/Reservoir:            R101
Downstream Zone/Reservoir:          Z1

The Model

Variable List:
Parameter List:

h(R101) = h(Z1,P1)
```

x(R101,a) = x(Z1,P1,a)

x(R101,b) = x(Z1,P1,b)

Connection: C102

Connection Type: Open Material Connection. Port / Reservoir Pressures are not Equated
Transfers the following phases: product_stream
Is linked to the following Zones: Z1, Z2
Upstream Zone/Reservoir: Z1
Downstream Zone/Reservoir: Z2

The Model

Variable List:
Parameter List:

h(Z1,P2) = h(Z2,P1)

F(Z1,P2) = F(Z2,P1)

x(Z1,P2,a) = x(Z2,P1,a)

x(Z1,P2,b) = x(Z2,P1,b)

Connection: C103

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: product_stream
Is linked to the following Zones: Z2, Z3
Upstream Zone/Reservoir: Z2
Downstream Zone/Reservoir: Z3

The Model

Variable List:
Parameter List:

h(Z2,P2) = h(Z3,P1)

F(Z2,P2) = F(Z3,P1)

x(Z2,P2,a) = x(Z3,P1,a)

x(Z2,P2,b) = x(Z3,P1,b)

P(Z2,P2) = P(Z3,P1)

Connection: C104

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: product_stream
Is linked to the following Zones: Z3, Z4
Upstream Zone/Reservoir: Z3
Downstream Zone/Reservoir: Z4

The Model

Variable List:
Parameter List:

h(Z3,P2) = h(Z4,P1)

F(Z3,P2) = F(Z4,P1)

x(Z3,P2,a) = x(Z4,P1,a)

x(Z3,P2,b) = x(Z4,P1,b)

P(Z3,P2) = P(Z4,P1)

Connection: C105

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: product_stream
Is linked to the following Zones: Z4, Z5
Upstream Zone/Reservoir: Z4
Downstream Zone/Reservoir: Z5

The Model

Variable List:
Parameter List:

h(Z4,P2) = h(Z5,P1)

F(Z4,P2) = F(Z5,P1)

x(Z4,P2,a) = x(Z5,P1,a)

x(Z4,P2,b) = x(Z5,P1,b)

P(Z4,P2) = P(Z5,P1)

Connection: C106

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: product_stream
Is linked to the following Zones: Z5, Z6
Upstream Zone/Reservoir: Z5
Downstream Zone/Reservoir: Z6

The Model

Variable List:
Parameter List:

h(Z5,P2) = h(Z6,P1)

F(Z5,P2) = F(Z6,P1)

x(Z5,P2,a) = x(Z6,P1,a)

x(Z5,P2,b) = x(Z6,P1,b)

P(Z5,P2) = P(Z6,P1)

Connection: C107

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: product_stream
Is linked to the following Zones: Z6, Z7
Upstream Zone/Reservoir: Z6
Downstream Zone/Reservoir: Z7

The Model

Variable List:
Parameter List:

h(Z6,P2) = h(Z7,P1)

F(Z6,P2) = F(Z7,P1)

x(Z6,P2,a) = x(Z7,P1,a)

x(Z6,P2,b) = x(Z7,P1,b)

P(Z6,P2) = P(Z7,P1)

Connection: C108

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: product_stream
Is linked to the following Zones: Z7, R102
Upstream Zone/Reservoir: Z7
Downstream Zone/Reservoir: R102

The Model

Variable List:
Parameter List:

P(R102) = P(Z7,P2)

Connection: C109

Connection Type:	Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases:	product_stream
Is linked to the following Zones:	Z6, Z8
Upstream Zone/Reservoir:	Z6
Downstream Zone/Reservoir:	Z8

The Model

Variable List:
Parameter List:

h(Z6,P3) = h(Z8,P1)

F(Z6,P3) = F(Z8,P1)

x(Z6,P3,a) = x(Z8,P1,a)

x(Z6,P3,b) = x(Z8,P1,b)

P(Z6,P3) = P(Z8,P1)

Connection: C110

Connection Type:	Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases:	product_stream
Is linked to the following Zones:	Z8, R103
Upstream Zone/Reservoir:	Z8
Downstream Zone/Reservoir:	R103

The Model

Variable List:
Parameter List:

P(R103) = P(Z8,P2)

Model Entity Information

Buffer_Tank_system consists of 11 model objects: InletPipe, Tank, OutletPipe1, Pump, OutletPipe2, Splitter, OutletPipe3, OutletPipe4, R101, R102, R103

Vessel: InletPipe

Reference name: P101
Number of Zones: 1
Zone Information:

Zone: Z1

Zone_Type:	, No-extent
Connections:	C101, C102
Ports:	P1 is linked to Connection C101. P2 is linked to Connection C102.
Physical Description:	The mass flowrate through the Zone is 50Kg/m3 The pressure drop can be determined from the pressure of the neighbouring
Zones	
	The Process is adiabatic There is no work done between the environment and the Zone The kinetic energy of the Connections is negligible

The Model:

Variable List: h(Z1,P1), x(Z1,P1,a), x(Z1,P1,b), h(Z1,P2), x(Z1,P2,a), x(Z1,P2,b), F(Z1,P1), F(Z1,P2)
Parameter List: Fref(Z1)

The Flowrate Subclass:
---

0 = F(Z1,P1) - F(Z1,P2)

F(Z1,P1) = Fref(Z1)

The Composition Subclass:

The Principle of Conservation for Components: Methanol, Water

0 = F(Z1,P1) * x(Z1,P1,a) - F(Z1,P2) * x(Z1,P2,a)

0 = F(Z1,P1) * x(Z1,P1,b) - F(Z1,P2) * x(Z1,P2,b)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z1,P1) * h(Z1,P1) - F(Z1,P2) * h(Z1,P2)

Vessel: Tank

Reference name: T101
Number of Zones: 1
Zone Information:

Zone: Z2

Zone_Type: Single-phase, Lumped
Phases: product_stream
Connections: C102, C103
Ports: P1 is linked to Connection C102.
P2 is linked to Connection C103.
The Port's Hydrostatic Pressure will be Modelled.

Physical Description: Physical Properties of the Zone vary with time.
The Zone's Extent varies during the simulation:
Composition of the Zone is known throughout the simulation:
Component: Methanol Mass Fraction: 0.5
Component: Water Mass Fraction: 0.5
Temperature has been assigned a value of: 293 degree K
Pressure has been assigned a value of: 100000

The Model:

Variable List: P(Z2,P1), P(Z2), h(Z2,P1), x(Z2,P1,a), x(Z2,P1,b), x(Z2,P2,a), x(Z2,a), x(Z2,P2,b), x(Z2,b), h(Z2,P2), h(Z2), P(Z2,P2), den(Z2), V(Z2), M(Z2), T(Z2), F(Z2,P1), F(Z2,P2)
Parameter List: g(Z2), A(Z2)

The Portal Equations for Port: P1

P(Z2,P1) = P(Z2)

The Portal Equations for Port: P2
---

x(Z2,P2,a) = x(Z2,a)

x(Z2,P2,b) = x(Z2,b)

h(Z2,P2) = h(Z2)

P(Z2,P2) = P(Z2) + den(Z2) * g(Z2) * (V(Z2) / A(Z2))

The Extent Subclass:

V(Z2) = M(Z2) / den(Z2)

den(Z2) = f(T(Z2), P(Z2), x(Z2,a), x(Z2,b))

d/dt(M(Z2)) = F(Z2,P1) - F(Z2,P2)

The Composition Subclass:

The Mass Fraction for Components: Methanol, Water

x(Z2,a) = 0.5

x(Z2,b) = 0.5

The Temperature Subclass:

T(Z2) = 293

The Equation of State for the Enthalpy of Phase: product_stream

h(Z2) = f(T(Z2), P(Z2), x(Z2,a), x(Z2,b))

The Pressure Subclass:

P(Z2) = 100000

Vessel: OutletPipe1

Reference name: P102
Number of Zones: 1
Zone Information:

Zone: Z3

Zone_Type: , No-extent
Connections: C103, C104
Ports: P1 is linked to Connection C103.
P2 is linked to Connection C104.

Physical Description: The mass flowrate through the Zone is related to the pressure drop across the Zone
The pressure drop can be determined from the pressure of the neighbouring Zones
The Process is adiabatic
There is no work done between the environment and the Zone
The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z3,P1), h(Z3,P1), x(Z3,P1,a), x(Z3,P1,b), P(Z3,P2), h(Z3,P2), x(Z3,P2,a), x(Z3,P2,b), F(Z3,P1), F(Z3,P2)
Parameter List:

The Flowrate Subclass:
---

0 = F(Z3,P1) - F(Z3,P2)

F(Z3,P1) = f(P(Z3,P1),P(Z3,P2))

The Composition Subclass:

The Principle of Conservation for Components: Methanol, Water

0 = F(Z3,P1) * x(Z3,P1,a) - F(Z3,P2) * x(Z3,P2,a)

0 = F(Z3,P1) * x(Z3,P1,b) - F(Z3,P2) * x(Z3,P2,b)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z3,P1) * h(Z3,P1) - F(Z3,P2) * h(Z3,P2)

Vessel: Pump

Reference name: X101
Number of Zones: 1
Zone Information:

Zone: Z4

Zone_Type: , No-extent
Connections: C104, C105
Ports: P1 is linked to Connection C104.
P2 is linked to Connection C105.

Physical Description: The mass flowrate through the Zone is related to the pressure drop across the Zone
There is a pressure drop of -850000 atm across the Zone
The Process is adiabatic
There is no work done between the environment and the Zone
The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z4,P1), h(Z4,P1), x(Z4,P1,a), x(Z4,P1,b), P(Z4,P2), h(Z4,P2), x(Z4,P2,a), x(Z4,P2,b), F(Z4,P1), F(Z4,P2)
Parameter List:

The Flowrate Subclass:
---

0 = F(Z4,P1) - F(Z4,P2)

F(Z4,P1) = f(P(Z4,P1),P(Z4,P2))

The Composition Subclass:

The Principle of Conservation for Components: Methanol, Water

0 = F(Z4,P1) * x(Z4,P1,a) - F(Z4,P2) * x(Z4,P2,a)

0 = F(Z4,P1) * x(Z4,P1,b) - F(Z4,P2) * x(Z4,P2,b)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z4,P1) * h(Z4,P1) - F(Z4,P2) * h(Z4,P2)

The Pressure Subclass:

P(Z4,P1) = P(Z4,P2) + -850000

Vessel: OutletPipe2

Reference name: P103
Number of Zones: 1
Zone Information:

Zone: Z5

Zone_Type: , No-extent
Connections: C105, C106
Ports: P1 is linked to Connection C105.
P2 is linked to Connection C106.

Physical Description: The mass flowrate through the Zone is related to the pressure drop across the Zone
The pressure drop can be determined from the pressure of the neighbouring Zones
The Process is adiabatic
There is no work done between the environment and the Zone
The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z5,P1), h(Z5,P1), x(Z5,P1,a), x(Z5,P1,b), P(Z5,P2), h(Z5,P2), x(Z5,P2,a), x(Z5,P2,b), F(Z5,P1), F(Z5,P2)
Parameter List:

The Flowrate Subclass:
---

0 = F(Z5,P1) - F(Z5,P2)

F(Z5,P1) = f(P(Z5,P1),P(Z5,P2))

The Composition Subclass:

The Principle of Conservation for Components: Methanol, Water

0 = F(Z5,P1) * x(Z5,P1,a) - F(Z5,P2) * x(Z5,P2,a)

0 = F(Z5,P1) * x(Z5,P1,b) - F(Z5,P2) * x(Z5,P2,b)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z5,P1) * h(Z5,P1) - F(Z5,P2) * h(Z5,P2)

Vessel: Splitter

Reference name: S101
Number of Zones: 1
Zone Information:

Zone: Z6

Zone_Type: , No-extent
Connections: C106, C107, C109
Ports: P1 is linked to Connection C106.
P2 is linked to Connection C107.
P3 is linked to Connection C109.

Physical Description: The mass flowrate through the Zone is dictated by either an upstream or a downstream No Extent Zone
There is a pressure drop of 0 atm across the Zone
The Process is adiabatic
There is no work done between the environment and the Zone
The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z6,P1), h(Z6,P1), x(Z6,P1,a), x(Z6,P1,b), T(Z6,P2), h(Z6,P2), P(Z6,P2), x(Z6,P2,a), x(Z6,P2,b), T(Z6,P3), h(Z6,P3), P(Z6,P3), x(Z6,P3,a), x(Z6,P3,b), F(Z6,P1), F(Z6,P2), F(Z6,P3), xref(Methanol), xref(Water), Tref, presref
Parameter List:

The Portal Equations for Port: P2

T(Z6,P2) = f(h(Z6,P2), P(Z6,P2), x(Z6,P2,a), x(Z6,P2,b))

The Portal Equations for Port: P3

T(Z6,P3) = f(h(Z6,P3), P(Z6,P3), x(Z6,P3,a), x(Z6,P3,b))

The Flowrate Subclass:
---

0 = F(Z6,P1) - F(Z6,P2) - F(Z6,P3)

The Composition Subclass:

The Principle of Conservation for Components: Methanol, Water

0 = F(Z6,P1) * x(Z6,P1,a) - F(Z6,P2) * x(Z6,P2,a) - F(Z6,P3) * x(Z6,P3,a)

0 = F(Z6,P1) * x(Z6,P1,b) - F(Z6,P2) * x(Z6,P2,b) - F(Z6,P3) * x(Z6,P3,b)

The Perfect Mixing Constraints for the Outlets: P2, P3

x(Z6,P2,a) = xref(Methanol)

x(Z6,P2,b) = xref(Water)

x(Z6,P3,a) = xref(Methanol)

x(Z6,P3,b) = xref(Water)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z6,P1) * h(Z6,P1) - F(Z6,P2) * h(Z6,P2) - F(Z6,P3) * h(Z6,P3)

T(Z6,P2) = Tref

T(Z6,P3) = Tref

The Pressure Subclass:

P(Z6,P1) = presref

P(Z6,P2) = presref

P(Z6,P3) = presref

Vessel: OutletPipe3

Reference name: P104
Number of Zones: 1
Zone Information:

Zone: Z7

Zone_Type:	, No-extent
Connections:	C107, C108
Ports:	P1 is linked to Connection C107. P2 is linked to Connection C108.
Physical Description:	The mass flowrate through the Zone is related to the pressure drop across the
Zone	The pressure drop can be determined from the pressure of the neighbouring
Zones	The Process is adiabatic There is no work done between the environment and the Zone The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z7,P1), h(Z7,P1), x(Z7,P1,a), x(Z7,P1,b), P(Z7,P2), h(Z7,P2), x(Z7,P2,a), x(Z7,P2,b), F(Z7,P1), F(Z7,P2)
Parameter List:

The Flowrate Subclass:
---

0 = F(Z7,P1) - F(Z7,P2)

F(Z7,P1) = f(P(Z7,P1),P(Z7,P2))

The Composition Subclass:

The Principle of Conservation for Components: Methanol, Water

0 = F(Z7,P1) * x(Z7,P1,a) - F(Z7,P2) * x(Z7,P2,a)

0 = F(Z7,P1) * x(Z7,P1,b) - F(Z7,P2) * x(Z7,P2,b)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z7,P1) * h(Z7,P1) - F(Z7,P2) * h(Z7,P2)

Vessel: OutletPipe4

Reference name: P105
Number of Zones: 1
Zone Information:

Zone: Z8

Zone_Type: , No-extent
Connections: C109, C110
Ports: P1 is linked to Connection C109.
P2 is linked to Connection C110.

Physical Description: The mass flowrate through the Zone is related to the pressure drop across the Zone
The pressure drop can be determined from the pressure of the neighbouring Zones
The Process is adiabatic
There is no work done between the environment and the Zone
The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z8,P1), h(Z8,P1), x(Z8,P1,a), x(Z8,P1,b), P(Z8,P2), h(Z8,P2), x(Z8,P2,a), x(Z8,P2,b), F(Z8,P1), F(Z8,P2)
Parameter List:

The Flowrate Subclass:
---

0 = F(Z8,P1) - F(Z8,P2)

F(Z8,P1) = f(P(Z8,P1),P(Z8,P2))

The Composition Subclass:

The Principle of Conservation for Components: Methanol, Water

0 = F(Z8,P1) * x(Z8,P1,a) - F(Z8,P2) * x(Z8,P2,a)

0 = F(Z8,P1) * x(Z8,P1,b) - F(Z8,P2) * x(Z8,P2,b)

```
The Temperature Subclass:
-------------------------

The No Extent Energy Balance:

0 = F(Z8,P1) * h(Z8,P1) - F(Z8,P2) * h(Z8,P2)

-----------------------------------------------------------
Reservoir: R101
-----------------------------------------------------------

Phases: product_stream
Connections:            C101
Physical Description:   Composition:
                              Component: Methanol Mass Fraction: 0.5
                              Component: Water    Mass Fraction: 0.5
                        Temperature: 293
                        Pressure:    101300

The Model:

Variable List:  T(R101), P(R101), x(R101,a), x(R101,b), h(R101)
Parameter List:

T(R101) = 293

P(R101) = 101300

x(R101,a) = 0.5

x(R101,b) = 0.5

The Equation of State for the Enthalpy of Phase: product_stream

h(R101) = f(T(R101), P(R101), x(R101,a), x(R101,b))

-----------------------------------------------------------
Reservoir: R102
-----------------------------------------------------------

Phases: product_stream
Connections:            C108
Physical Description:   Pressure:    101300

The Model:

Variable List:  P(R102)
Parameter List:

P(R102) = 101300

-----------------------------------------------------------
Reservoir: R103
-----------------------------------------------------------

Phases: product_stream
Connections:            C110
Physical Description:   Pressure:    101300

The Model:

Variable List:  P(R103)
Parameter List:

P(R103) = 101300
```

K.2 The Gas Phase Continuous Stirred Tank Reactor System Model (TEXT)

```
Process: Gas_CSTR_system

-----------------------------------------------------------
Phase Information
-----------------------------------------------------------

Gas_CSTR_system consists of 2 phases: raw_materials, product_stream

Phase name:       raw_materials:
Phase Type:       Gas
Compressibility:  Compressibile
```

```
Component list:   Cyclopropane which is represented by a

Phase name:       product_stream:
Phase Type:       Gas
Compressibility:  Compressibile
Component list:   Cyclopropane which is represented by a
                  Propylene which is represented by b

-----------------------------------------------------------
Connection Information:
-----------------------------------------------------------

Gas_CSTR_system consists of 4 connections: C101, C102, C103, C104

--------------------------
Connection: C101
--------------------------

Connection Type:                    Open Material Connection. Port / Reservoir Pressures are not
Equated
Transfers the following phases:     raw_materials
Is linked to the following Zones:   R101, Z1
Upstream Zone/Reservoir:            R101
Downstream Zone/Reservoir:          Z1


The Model

Variable List:
Parameter List:

h(R101) = h(Z1,P1)

x(R101,a) = x(Z1,P1,a)


--------------------------
Connection: C102
--------------------------

Connection Type:                    Open Material Connection. Port / Reservoir Pressures are not
Equated
Transfers the following phases:     raw_materials
Is linked to the following Zones:   Z1, Z2
Upstream Zone/Reservoir:            Z1
Downstream Zone/Reservoir:          Z2


The Model

Variable List:
Parameter List:

h(Z1,P2) = h(Z2,P1)

F(Z1,P2) = F(Z2,P1)

x(Z1,P2,a) = x(Z2,P1,a)


--------------------------
Connection: C103
--------------------------

Connection Type:                    Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases:     product_stream
Is linked to the following Zones:   Z2, Z3
Upstream Zone/Reservoir:            Z2
Downstream Zone/Reservoir:          Z3


The Model

Variable List:
Parameter List:

h(Z2,P2) = h(Z3,P1)

F(Z2,P2) = F(Z3,P1)

x(Z2,P2,a) = x(Z3,P1,a)
```

x(Z2,P2,b) = x(Z3,P1,b)

P(Z2,P2) = P(Z3,P1)

Connection: C104

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: product_stream
Is linked to the following Zones: Z3, R102
Upstream Zone/Reservoir: Z3
Downstream Zone/Reservoir: R102

The Model

Variable List:
Parameter List:

P(R102) = P(Z3,P2)

Model Entity Information

Gas_CSTR_system consists of 5 model objects: InletPipe, Reactor, OutletPipe1, R101, R102

Vessel: InletPipe

Reference name: P101
Number of Zones: 1
Zone Information:

Zone: Z1

Zone_Type: , No-extent
Connections: C101, C102
Ports: P1 is linked to Connection C101.
P2 is linked to Connection C102.

Physical Description: The mass flowrate through the Zone is 0.01Kg,m3
The pressure drop can be determined from the pressure of the neighbouring
Zones
The Process is adiabatic
There is no work done between the environment and the Zone
The kinetic energy of the Connections is negligible

The Model:

Variable List: h(Z1,P1), x(Z1,P1,a), h(Z1,P2), x(Z1,P2,a), F(Z1,P1), F(Z1,P2)
Parameter List: Fref(Z1)

The Flowrate Subclass:

---

0 = F(Z1,P1) - F(Z1,P2)

F(Z1,P1) = Fref(Z1)

The Composition Subclass:

The Principle of Conservation for Components: Cyclopropane

0 = F(Z1,P1) * x(Z1,P1,a) - F(Z1,P2) * x(Z1,P2,a)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z1,P1) * h(Z1,P1) - F(Z1,P2) * h(Z1,P2)

Vessel: Reactor

Reference name: R101
Vessel Type: Closed
Number of Zones: 1
Zone Information:

Zone: Z2

Zone_Type:	Single-phase, Lumped
Phases:	product_stream
Connections:	C102, C103
Ports:	P1 is linked to Connection C102.
	P2 is linked to Connection C103.
Physical Description:	Physical Properties of the Zone vary with time.
	The Zone's Extent is fixed with volume: 2 m3
	Composition varies through the simulation and must be determined with reaction XXXX.
	Temperature must be determined as part of the global solution with:
	No External Heat Flux
	No External Work
	Potential Energy of the Connections is Neglected
	Kinetic Energy of the Connections is Neglected
	Pressure must be determined as part of the global solution.

The Model:

Variable List: P(Z2,P1), P(Z2), h(Z2,P1), x(Z2,P1,a), x(Z2,P2,a), x(Z2,a), x(Z2,P2,b), x(Z2,b), h(Z2,P2), h(Z2), P(Z2,P2), V(Z2), M(Z2,a), F(Z2,P1), F(Z2,P2), R(Z2,a), M(Z2,b), R(Z2,b), r(Z2,j), M(Z2), u(Z2), T(Z2), den(Z2)
Parameter List: Ypsilon(a,j), Ypsilon(b,j)

The Portal Equations for Port: P1

P(Z2,P1) = P(Z2)

The Portal Equations for Port: P2

---

x(Z2,P2,a) = x(Z2,a)

x(Z2,P2,b) = x(Z2,b)

h(Z2,P2) = h(Z2)

P(Z2,P2) = P(Z2)

The Extent Subclass:

V(Z2) = 2

The Composition Subclass:

The Lumped Single Phase Composition Mass Balances for: Cyclopropane, Propylene

d/dt(M(Z2,a)) = F(Z2,P1) * x(Z2,P1,a) - F(Z2,P2) * x(Z2,P2,a) + V(Z2) * R(Z2,a)

d/dt(M(Z2,b)) = - F(Z2,P2) * x(Z2,P2,b) + V(Z2) * R(Z2,b)

The Reaction Kinetic Equations for: Cyclopropane, Propylene

R(Z2,a) = SIGMA(j)(Ypsilon(a,j) * r(Z2,j))

R(Z2,b) = SIGMA(j)(Ypsilon(b,j) * r(Z2,j))

The Mass Fraction for Components: Cyclopropane, Propylene

x(Z2,a) = M(Z2,a) / M(Z2)

x(Z2,b) = M(Z2,b) / M(Z2)

M(Z2) = M(Z2,a) + M(Z2,b)

The Temperature Subclass:

The Lumped Energy Conservation Equation:

d/dt(M(Z2) * u(Z2)) = F(Z2,P1) * h(Z2,P1) - F(Z2,P2) * h(Z2,P2)

The Equation of State for the Internal Energy of Phase: product_stream

u(Z2) = f(T(Z2), P(Z2), x(Z2,a), x(Z2,b))

The Equation of State for the Enthalpy of Phase: product_stream

h(Z2) = f(T(Z2), P(Z2), x(Z2,a), x(Z2,b))

The Pressure Subclass:
---

den(Z2) = f(T(Z2), P(Z2), x(Z2,a), x(Z2,b))

V(Z2) = M(Z2) / den(Z2)

Vessel: OutletPipe1

Reference name: P102
Number of Zones: 1
Zone Information:

Zone: Z3

Zone_Type: , No-extent
Connections: C103, C104
Ports: P1 is linked to Connection C103.
P2 is linked to Connection C104.

Physical Description: The mass flowrate through the Zone is related to the pressure drop across the Zone
The pressure drop can be determined from the pressure of the neighbouring Zones
The Process is adiabatic
There is no work done between the environment and the Zone
The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z3,P1), h(Z3,P1), x(Z3,P1,a), x(Z3,P1,b), P(Z3,P2), h(Z3,P2), x(Z3,P2,a), x(Z3,P2,b), F(Z3,P1), F(Z3,P2)
Parameter List:

The Flowrate Subclass:
---

0 = F(Z3,P1) - F(Z3,P2)

F(Z3,P1) = f(P(Z3,P1),P(Z3,P2))

The Composition Subclass:

The Principle of Conservation for Components: Cyclopropane, Propylene

0 = F(Z3,P1) * x(Z3,P1,a) - F(Z3,P2) * x(Z3,P2,a)

0 = F(Z3,P1) * x(Z3,P1,b) - F(Z3,P2) * x(Z3,P2,b)

```
The Temperature Subclass:
-------------------------

The No Extent Energy Balance:

0 = F(Z3,P1) * h(Z3,P1) - F(Z3,P2) * h(Z3,P2)

-----------------------------------------------------------
Reservoir: R101
-----------------------------------------------------------

Phases: raw_materials
Connections:            C101
Physical Description:   Composition:
                          Component: Cyclopropane     Mass Fraction: 1
                        Temperature: 780
                        Pressure:    101300

The Model:

Variable List:  T(R101), P(R101), x(R101,a), h(R101)
Parameter List:

T(R101) = 780

P(R101) = 101300

x(R101,a) = 1

The Equation of State for the Enthalpy of Phase: raw_materials

h(R101) = f(T(R101), P(R101), x(R101,a))

-----------------------------------------------------------
Reservoir: R102
-----------------------------------------------------------

Phases: product_stream
Connections:            C104
Physical Description:   Pressure:    90000

The Model:

Variable List:  P(R102)
Parameter List:

P(R102) = 90000
```

K.3 The Adiabatic Flash Vessel Model (TEXT)

```
Process: Flash_in_equilibrium

-----------------------------------------------------------
Phase Information
-----------------------------------------------------------

Flash_in_equilibrium consists of 2 phases: GP, LP

Phase name:       GP:
Phase Type:       Gas
Compressibility:  Compressibile
Component list:   Propane which is represented by a
                  Pentane which is represented by b

Phase name:       LP:
Phase Type:       Liquid
Compressibility:  Incompressibile
Component list:   Propane which is represented by a
                  Pentane which is represented by b

-----------------------------------------------------------
Connection Information:
-----------------------------------------------------------

Flash_in_equilibrium consists of 6 connections: C101, C102, C103, C104, C105, C106
```

Connection: C101

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: LP
Is linked to the following Zones: R101, Z1
Upstream Zone/Reservoir: R101
Downstream Zone/Reservoir: Z1

The Model

Variable List:
Parameter List:

h(R101) = h(Z1,P1)

P(R101) = P(Z1,P1)

x(R101,a) = x(Z1,P1,a)

x(R101,b) = x(Z1,P1,b)

Connection: C102

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: LP
Is linked to the following Zones: Z1, Z2
Upstream Zone/Reservoir: Z1
Downstream Zone/Reservoir: Z2

The Model

Variable List:
Parameter List:

h(Z1,P2) = h(Z2,P1)

F(Z1,P2) = F(Z2,P1)

x(Z1,P2,a) = x(Z2,P1,a)

x(Z1,P2,b) = x(Z2,P1,b)

P(Z1,P2) = P(Z2,P1)

Connection: C103

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: GP
Is linked to the following Zones: Z2, Z3
Upstream Zone/Reservoir: Z2
Downstream Zone/Reservoir: Z3

The Model

Variable List:
Parameter List:

h(Z2,P2) = h(Z3,P1)

F(Z2,P2) = F(Z3,P1)

x(Z2,P2,a) = x(Z3,P1,a)

x(Z2,P2,b) = x(Z3,P1,b)

P(Z2,P2) = P(Z3,P1)

Connection: C104

Connection Type: Open Material Connection. Port / Reservoir Pressures are Equated
Transfers the following phases: LP
Is linked to the following Zones: Z2, Z4

Upstream Zone/Reservoir: Z2
Downstream Zone/Reservoir: Z4

The Model

Variable List:
Parameter List:

h(Z2,P3) = h(Z4,P1)

F(Z2,P3) = F(Z4,P1)

x(Z2,P3,a) = x(Z4,P1,a)

x(Z2,P3,b) = x(Z4,P1,b)

P(Z2,P3) = P(Z4,P1)

Connection: C105

Connection Type: Open Material Connection. Port - Reservoir Pressures are Equated
Transfers the following phases: GP
Is linked to the following Zones: Z3, R102
Upstream Zone/Reservoir: Z3
Downstream Zone/Reservoir: R102

The Model

Variable List:
Parameter List:

P(R102) = P(Z3,P2)

Connection: C106

Connection Type: Open Material Connection. Port - Reservoir Pressures are Equated
Transfers the following phases: LP
Is linked to the following Zones: Z4, R103
Upstream Zone/Reservoir: Z4
Downstream Zone/Reservoir: R103

The Model

Variable List:
Parameter List:

P(R103) = P(Z4,P2)

Model Entity Information

Flash_in_equilibrium consists of 7 model objects: InletPipe, Flash_Vessel, Outlet_Gas_Pipe, Outlet_Liquid_Pipe, R101, R102, R103

Vessel: InletPipe

Reference name: P101
Number of Zones: 1
Zone Information:

Zone: Z1

Zone_Type: , No-extent
Connections: C101, C102
Ports: P1 is linked to Connection C101.
P2 is linked to Connection C102.

Physical Description: The mass flowrate through the Zone is 10[illegible] m3
The pressure drop can be determined from the pressure of the neighbouring Zones
The Process is adiabatic
There is no work done between the environment and the Zone

The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z1,P1), h(Z1,P1), x(Z1,P1,a), x(Z1,P1,b), P(Z1,P2), h(Z1,P2), x(Z1,P2,a), x(Z1,P2,b), F(Z1,P1), F(Z1,P2)
Parameter List: Fref(Z1)

The Flowrate Subclass:
---

0 = F(Z1,P1) - F(Z1,P2)

F(Z1,P1) = Fref(Z1)

The Composition Subclass:

The Principle of Conservation for Components: Propane, Pentane

0 = F(Z1,P1) * x(Z1,P1,a) - F(Z1,P2) * x(Z1,P2,a)

0 = F(Z1,P1) * x(Z1,P1,b) - F(Z1,P2) * x(Z1,P2,b)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z1,P1) * h(Z1,P1) - F(Z1,P2) * h(Z1,P2)

Vessel: Flash_Vessel

Reference name: FV101
Vessel Type: Closed
Vessel Volume: 11.5m3
Number of Zones: 1
Zone Information:

Zone: Z2

Zone_Type: Two-phase, Lumped
Phases: GP, LP
Connections: C102, C103, C104
Ports: P1 is linked to Connection C102.
P2 is linked to Connection C103.
P3 is linked to Connection C104.
The Port's Hydrostatic Pressure will be Modelled.

Physical Description: Physical Properties of the Zone vary with time.
.....
The Zone's Extent is fixed with volume: 5 m3
.....
Composition of phase GP varies and must be determined throughout the simulation.
Composition of phase LP varies and must be determined throughout the simulation.
The two phases are in chemical equilibrium.
.....
The Temperature of each Phase will be determined collectively.
The Zone temperature must be determined as part of the global solution with:
No External Heat Flux
No External Work
Potential Energy of the Connections is Neglected
Kinetic Energy of the Connections is Neglected
.....
Pressure must be determined as part of the global solution:
.....
The volume fraction of each Phase varies during the simulation.
.....
The velocity of the material can be neglected.

The Model:

Variable List: P(Z2,P1), P(Z2), h(Z2,P1), x(Z2,P1,a), x(Z2,P1,b), P(Z2,P2), h(Z2,P2), h(Z2,GP), x(Z2,P2,a), x(Z2,GP,a), x(Z2,P2,b), x(Z2,GP,b), P(Z2,P3), den(Z2,LP), V(Z2), Vfrac(Z2,LP), h(Z2,P3), h(Z2,LP), x(Z2,P3,a), x(Z2,LP,a), x(Z2,P3,b), x(Z2,LP,b), M(Z2,GP), M(Z2,GP,a), M(Z2,GP,b), M(Z2,LP), M(Z2,LP,a), M(Z2,LP,b), x(Z2,GP), M(Z2), x(Z2,LP), den(Z2,GP), T(Z2), M(Z2,a), F(Z2,P1), F(Z2,P2), F(Z2,P3), M(Z2,b), K(Z2,a), K(Z2,b), u(Z2), u(Z2,GP), u(Z2,LP), Vfrac(Z2,GP)
Parameter List: g(Z2), A(Z2)

The Portal Equations for Port: P1

P(Z2,P1) = P(Z2)

The Portal Equations for Port: P2

P(Z2,P2) = P(Z2)

h(Z2,P2) = h(Z2,GP)

x(Z2,P2,a) = x(Z2,GP,a)

x(Z2,P2,b) = x(Z2,GP,b)

The Portal Equations for Port: P3

P(Z2,P3) = P(Z2) + den(Z2,LP) * g(Z2) * (V(Z2) * Vfrac(Z2,LP) / A(Z2))

h(Z2,P3) = h(Z2,LP)

x(Z2,P3,a) = x(Z2,LP,a)

x(Z2,P3,b) = x(Z2,LP,b)

The Core Equations:

The Total Mass of Phases GP, LP

M(Z2,GP) = M(Z2,GP,a) + M(Z2,GP,b)

M(Z2,LP) = M(Z2,LP,a) + M(Z2,LP,b)

The Mass Fraction of Phases GP, LP

x(Z2,GP) = M(Z2,GP) / M(Z2)

x(Z2,LP) = M(Z2,LP) / M(Z2)

The Total Mass of the Zone

M(Z2) = M(Z2,GP) + M(Z2,LP)

The Density of Phases: GP, LP

den(Z2,GP) = f(T(Z2), P(Z2), x(Z2,GP,a), x(Z2,GP,b))

den(Z2,LP) = constant

The Extent Subclass:

V(Z2) = 5

The Composition Subclass:

The Total Mass Balance and Equilibrium Equations for Components: Propane, Pentane

d/dt(M(Z2,a)) = F(Z2,P1) * x(Z2,P1,a) - F(Z2,P2) * x(Z2,P2,a) - F(Z2,P3) * x(Z2,P3,a)

d/dt(M(Z2,b)) = F(Z2,P1) * x(Z2,P1,b) - F(Z2,P2) * x(Z2,P2,b) - F(Z2,P3) * x(Z2,P3,b)

x(Z2,GP,a) = K(Z2,a) * x(Z2,LP,a)

x(Z2,GP,b) = K(Z2,b) * x(Z2,LP,b)

The Total Mass of Components Propane, Pentane within the Zone.

M(Z2,a) = M(Z2,GP) * x(Z2,GP,a) + M(Z2,LP) * x(Z2,LP,a)

M(Z2,b) = M(Z2,GP) * x(Z2,GP,b) + M(Z2,LP) * x(Z2,LP,b)

The Equations of State for the Equilibrium Constants of Components Propane, Pentane

K(Z2,a) = f(T(Z2), P(Z2), x(Z2,GP,a), x(Z2,GP,b), x(Z2,LP,a), x(Z2,LP,b), x(Z2,LP,a), x(Z2,LP,b))

K(Z2,b) = f(T(Z2), P(Z2), x(Z2,GP,a), x(Z2,GP,b), x(Z2,LP,a), x(Z2,LP,b), x(Z2,LP,a), x(Z2,LP,b))

The Mass Fraction for Components: Propane, Pentane within Phase: GP

x(Z2,GP,a) = M(Z2,GP,a) / M(Z2,GP)

x(Z2,GP,b) = M(Z2,GP,b) / M(Z2,GP)

The Mass Fraction for Components: Propane, Pentane within Phase: LP

x(Z2,LP,a) = M(Z2,LP,a) / M(Z2,LP)

x(Z2,LP,b) = M(Z2,LP,b) / M(Z2,LP)

The Temperature Subclass:

The Lumped, Homogeneous, Two-Phase Energy Conservation Equation:

d/dt(M(Z2) * u(Z2)) = F(Z2,P1) * h(Z2,P1) - F(Z2,P2) * h(Z2,P2) - F(Z2,P3) * h(Z2,P3)

u(Z2) = x(Z2,GP) * u(Z2,GP) + x(Z2,LP) * u(Z2,LP)

The Equation of State for the Internal Energy of Phases GP, LP

u(Z2,GP) = f(T(Z2), P(Z2), x(Z2,GP,a), x(Z2,GP,b))

u(Z2,LP) = f(T(Z2), P(Z2), x(Z2,LP,a), x(Z2,LP,b))

The Equation of State for the Enthalpy of Phases GP, LP

h(Z2,GP) = f(T(Z2), P(Z2), x(Z2,GP,a), x(Z2,GP,b))

h(Z2,LP) = f(T(Z2), P(Z2), x(Z2,LP,a), x(Z2,LP,b))

The Pressure Subclass:

V(Z2) = M(Z2,GP) / den(Z2,GP) + M(Z2,LP) / den(Z2,LP)

The Volume Fraction Subclass:

M(Z2,GP) = Vfrac(Z2,GP) * den(Z2,GP) * V(Z2)

M(Z2,LP) = Vfrac(Z2,LP) * den(Z2,LP) * V(Z2)

Vessel: Outlet_Gas_Pipe

Reference name: P102
Number of Zones: 1
Zone Information:

Zone: Z3

Zone_Type: , No-extent
Connections: C103, C105
Ports: P1 is linked to Connection C103.
P2 is linked to Connection C105.

Physical Description: The mass flowrate through the Zone is related to the pressure drop across the Zone

The pressure drop can be determined from the pressure of the neighbouring Zones

The Process is adiabatic
There is no work done between the environment and the Zone
The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z3,P1), h(Z3,P1), x(Z3,P1,a), x(Z3,P1,b), P(Z3,P2), h(Z3,P2), x(Z3,P2,a), x(Z3,P2,b), F(Z3,P1), F(Z3,P2)
Parameter List:

The Flowrate Subclass:
---

0 = F(Z3,P1) - F(Z3,P2)

F(Z3,P1) = f(P(Z3,P1),P(Z3,P2))

The Composition Subclass:

The Principle of Conservation for Components: Propane, Pentane

0 = F(Z3,P1) * x(Z3,P1,a) - F(Z3,P2) * x(Z3,P2,a)

0 = F(Z3,P1) * x(Z3,P1,b) - F(Z3,P2) * x(Z3,P2,b)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z3,P1) * h(Z3,P1) - F(Z3,P2) * h(Z3,P2)

Vessel: Outlet_Liquid_Pipe

Reference name: P103
Number of Zones: 1
Zone Information:

Zone: Z4

Zone_Type: , No-extent
Connections: C104, C106
Ports: P1 is linked to Connection C104.
P2 is linked to Connection C106.

Physical Description: The mass flowrate through the Zone is related to the pressure drop across the Zone

The pressure drop can be determined from the pressure of the neighbouring Zones

The Process is adiabatic
There is no work done between the environment and the Zone
The kinetic energy of the Connections is negligible

The Model:

Variable List: P(Z4,P1), h(Z4,P1), x(Z4,P1,a), x(Z4,P1,b), P(Z4,P2), h(Z4,P2), x(Z4,P2,a), x(Z4,P2,b), F(Z4,P1), F(Z4,P2)
Parameter List:

The Flowrate Subclass:
---

0 = F(Z4,P1) - F(Z4,P2)

F(Z4,P1) = f(P(Z4,P1),P(Z4,P2))

The Composition Subclass:

The Principle of Conservation for Components: Propane, Pentane

0 = F(Z4,P1) * x(Z4,P1,a) - F(Z4,P2) * x(Z4,P2,a)

0 = F(Z4,P1) * x(Z4,P1,b) - F(Z4,P2) * x(Z4,P2,b)

The Temperature Subclass:

The No Extent Energy Balance:

0 = F(Z4,P1) * h(Z4,P1) - F(Z4,P2) * h(Z4,P2)

Reservoir: R101

Phases: LP
Connections: C101
Physical Description: Composition:
Component: Propane Mass Fraction: 0.5
Component: Pentane Mass Fraction: 0.5
Temperature: 293
Pressure: 1.013e+06

The Model:

Variable List: T(R101), P(R101), x(R101,a), x(R101,b), h(R101)
Parameter List:

T(R101) = 293

P(R101) = 1.013e+06

x(R101,a) = 0.5

x(R101,b) = 0.5

The Equation of State for the Enthalpy of Phase: LP

h(R101) = f(T(R101), P(R101), x(R101,a), x(R101,b))

Reservoir: R102

Phases: GP
Connections: C105
Physical Description: Pressure: 101300

The Model:

Variable List: P(R102)
Parameter List:

P(R102) = 101300

Reservoir: R103

Phases: LP
Connections: C106
Physical Description: Pressure: 101300

The Model:

Variable List: P(R103)
Parameter List:

P(R103) = 101300

K.4 The Gas Phase Plug Flow Reactor System Model (gPROMS)

```
#-------------------------------------------------------------
# The gPROMS Input File For Process: Gas_PFR
#-------------------------------------------------------------

DECLARE
  TYPE

    Area              =   1       :   0.0      :   1e5      UNIT = "m2"
    NoType            =   1       :   -1e200   :   1e200   UNIT = "-"
    MassFlowrate      =   1       :   -1e20    :   1e20    UNIT = "kg/s"
    MassFlux          =   1e3     :   -1e3     :   1e8     UNIT = "kg/m2s"
    Velocity          =   1       :   -1e3     :   1e3     UNIT = "m/s"
    Density           =   1e3     :   -1e-5    :   1e10    UNIT = "kg/m3"
    MassHoldup        =   1       :   -1e-3    :   1e20    UNIT = "kg"
    VolumetricHoldup  =   1       :   0        :   1e5     UNIT = "m3"
    ReactionSink      =   1e3     :   -1e10    :   1e10    UNIT = "kg/m3s"
    ReactionRate      =   1       :   -1e10    :   1e10    UNIT = "kg/s"
    MassFraction      =   0.5     :   -1e-3    :   1.001   UNIT = "-"
    Pressure          =   1e5     :   1        :   1e20    UNIT = "kN/m2"
    Temperature       =   300     :   10       :   5e4     UNIT = "K"
    InternalEnergy    =   1e3     :   -1E20    :   1e20    UNIT = "kJ/kg"
    Enthalpy          =   1e2     :   -1E20    :   1e20    UNIT = "kJ/kg"
    HeatTerm          =   1       :   -1e50    :   1e50    UNIT = "kJ/s"
    HeatFlux          =   1e3     :   -1e10    :   1e10    UNIT = "kJ/m2s"
    HeatTransferCoef  =   1e3     :   -1e3     :   1e8     UNIT = "m-2s-1"
    Length            =   1       :   -0       :   1e5     UNIT = "m"

END

#-------------------------------------------------------------
# The gPROMS Model For Zone: ZONE_MODEL1
#-------------------------------------------------------------

MODEL Zone_MODEL1

  PARAMETER

        Fref                            AS REAL

  VARIABLE

        P_P1                            AS Pressure
        h_P1                            AS Enthalpy
        x_P1_a                          AS MassFraction
        x_P1_b                          AS MassFraction
        x_P1_c                          AS MassFraction
        P_P2                            AS Pressure
        h_P2                            AS Enthalpy
        x_P2_a                          AS MassFraction
        x_P2_b                          AS MassFraction
        x_P2_c                          AS MassFraction
        F_P1                            AS MassFlowrate
        F_P2                            AS MassFlowrate

EQUATION

# The Flowrate Subclass:
#-----------------------

0 = F_P1 - F_P2  ;

F_P1 = Fref  ;

# The Composition Subclass:
#--------------------------

# The Principle of Conservation for Components: Butene, Butyne, Water

0 = F_P1 * x_P1_a - F_P2 * x_P2_a  ;

0 = F_P1 * x_P1_b - F_P2 * x_P2_b  ;

0 = F_P1 * x_P1_c - F_P2 * x_P2_c  ;

# The Temperature Subclass:
```

```
#-------------------------

# The No Extent Energy Balance:

0 = F_P1 * h_P1 - F_P2 * h_P2  ;


END #ZONE MODEL Z1


#-----------------------------------------------------------
# The gPROMS Model For Zone: Z2
#-----------------------------------------------------------

MODEL Zone_MODEL2

PARAMETER

      Ypsilon(a,j)                AS REAL
      Ypsilon(b,j)                AS REAL
      Ypsilon(c,j)                AS REAL
      g                           AS REAL
      Pw                          AS REAL

DISTRIBUTION_DOMAIN
     Axial          AS ( 0 : 1)

VARIABLE

      F_P1_a                      AS MassFlowrate
      F_P1                        AS MassFlowrate
      x_P1_a                      AS MassFraction
      F_P1_b                      AS MassFlowrate
      x_P1_b                      AS MassFraction
      F_P1_c                      AS MassFlowrate
      x_P1_c                      AS MassFraction
      P_P1                        AS Pressure
      Q_P1                        AS HeatTerm
      h_P1                        AS Enthalpy
      x_P2_a                      AS MassFraction
      x_P2_b                      AS MassFraction
      x_P2_c                      AS MassFraction
      h_P2                        AS Enthalpy
      F_P2                        AS MassFlowrate
      A                           AS Area
      P_P2                        AS Pressure
      V                           AS VolumetricHoldup
      Z_Inlet                     AS Length
      Z_Outlet                    AS Length
      b                           AS NoType
      T_P3                        AS DISTRIBUTION(Axial) OF Temperature
      T                           AS DISTRIBUTION(Axial) OF Temperature
      den                         AS DISTRIBUTION(Axial) OF Density
      P                           AS DISTRIBUTION(Axial) OF Pressure
      x_a                         AS DISTRIBUTION(Axial) OF MassFraction
      x_b                         AS DISTRIBUTION(Axial) OF MassFraction
      x_c                         AS DISTRIBUTION(Axial) OF MassFraction
      C_a                         AS DISTRIBUTION(Axial) OF Density
      veloc                       AS DISTRIBUTION(Axial) OF Velocity
      R_a                         AS DISTRIBUTION(Axial) OF ReactionSink
      C_b                         AS DISTRIBUTION(Axial) OF Density
      R_b                         AS DISTRIBUTION(Axial) OF ReactionSink
      C_c                         AS DISTRIBUTION(Axial) OF Density
      R_c                         AS DISTRIBUTION(Axial) OF ReactionSink
      r(j)                        AS DISTRIBUTION(Axial) OF ReactionRate
      utot                        AS DISTRIBUTION(Axial) OF InternalEnergy
      htot                        AS DISTRIBUTION(Axial) OF Enthalpy
      qflux_P3                    AS DISTRIBUTION(Axial) OF HeatFlux
      h                           AS DISTRIBUTION(Axial) OF Enthalpy
      u                           AS DISTRIBUTION(Axial) OF InternalEnergy

EQUATION

# The Portal Equations for Port: P1
#---------------------------------

F_P1_a = F_P1 * x_P1_a  ;

F_P1_b = F_P1 * x_P1_b  ;

F_P1_c = F_P1 * x_P1_c  ;
```

```
P_P1 = P(0)  ;

Q_P1 = F_P1 * h_P1  ;


# The Portal Equations for Port: P2
#----------------------------------

x_P2_a = x_a(1)  ;

x_P2_b = x_b(1)  ;

x_P2_c = x_c(1)  ;

h_P2 = h(1)  ;

F_P2 = A * veloc(1) * den(1)  ;

P_P2 = P(1)  ;

# The Portal Equations for Port: P3
#----------------------------------

FOR znorm := 0 TO 1 DO

        T_P3(znorm) = T(znorm)  ;

END #FOR


# The Core Subclass:
#-------------------------------

# The Density Equation of State for Phase: Product_stream

FOR znorm := 0 TO 1 DO

        den(znorm) = f(T(znorm), P(znorm), x_a(znorm), x_b(znorm), x_c(znorm))  ;

END #FOR


# The Extent Subclass:
#-------------------------------

V = A * (Z_Outlet - Z_Inlet)  ;

Z_Inlet = 0  ;

Z_Outlet = 5  ;

b = sin(0)  ;


# The Composition Subclass:
#---------------------------

# The Lumped Single Phase Composition Mass Balances for: Butene, Butyne, Water

# The 1-D Duct Component Balance for Component Butene

FOR znorm := 0|+ TO 1 DO

        $(C_a(znorm)) + 1 / (Z_Outlet - Z_Inlet) * PARTIAL((veloc(znorm) * C_a(znorm)), Axial) -
R_a(znorm) = 0  ;

END #FOR

# The 1-D Duct Component Balance for Component Butyne

FOR znorm := 0|+ TO 1 DO

        $(C_b(znorm)) + 1 / (Z_Outlet - Z_Inlet) * PARTIAL((veloc(znorm) * C_b(znorm)), Axial) -
R_b(znorm) = 0  ;

END #FOR

# The 1-D Duct Component Balance for Component Water

FOR znorm := 0|+ TO 1 DO
```

```
        $(C_c(znorm)) + 1 / (Z_Outlet - Z_Inlet) * PARTIAL((veloc(znorm) * C_c(znorm)), Axial) -
R_c(znorm) = 0  ;

END #FOR

# The Boundary Condition at znorm = 0
F_P1_a = A * veloc(0) * C_a(0)  ;

# The Boundary Condition at znorm = 0
F_P1_b = A * veloc(0) * C_b(0)  ;

# The Boundary Condition at znorm = 0
F_P1_c = A * veloc(0) * C_c(0)  ;
```

```
# The Reaction Kinetic Equations for: Butene, Butyne, Water

FOR znorm := 0 TO 1 DO

        R_a(znorm) = SIGMA(j)(Ypsilon(a,j) * r(j,znorm))  ;

END #FOR

FOR znorm := 0 TO 1 DO

        R_b(znorm) = SIGMA(j)(Ypsilon(b,j) * r(j,znorm))  ;

END #FOR

FOR znorm := 0 TO 1 DO

        R_c(znorm) = SIGMA(j)(Ypsilon(c,j) * r(j,znorm))  ;

END #FOR

# The Mass Fraction for Components: Butene, Butyne, Water

FOR znorm := 0 TO 1 DO

        x_a(znorm) = C_a(znorm) / den(znorm)  ;

END #FOR

FOR znorm := 0 TO 1 DO

        x_b(znorm) = C_b(znorm) / den(znorm)  ;

END #FOR

FOR znorm := 0 TO 1 DO

        x_c(znorm) = C_c(znorm) / den(znorm)  ;

END #FOR


# The Temperature Subclass:
#--------------------------

# The 1-D Duct Energy Balance

FOR znorm := 0|+ TO 1 DO

        $(den(znorm) * utot(znorm)) + 1 / (Z_Outlet - Z_Inlet) * PARTIAL((den(znorm) * veloc(znorm) *
htot(znorm)), Axial) - den(znorm) * g * b * veloc(znorm) - (Pw * qflux_P3(znorm)) / A = 0  ;

END #FOR

# The Boundary Condition at znorm = 0
Q_P1 = A * veloc(0) * den(0) * htot(0)  ;

FOR znorm := 0 TO 1 DO

        htot(znorm) = h(znorm)  ;

END #FOR

FOR znorm := 0 TO 1 DO
```

```
        utot(znorm) = u(znorm)   ;
END #FOR
# The Equation of State for the Enthalpy of Phase: Product_stream
FOR znorm := 0 TO 1 DO
        h(znorm) = f(T(znorm), P(znorm), x_a(znorm), x_b(znorm), x_c(znorm))   ;
END #FOR
# The Equation of State for the Internal Energy of Phase:Product_stream
FOR znorm := 0 TO 1 DO
        u(znorm) = f(T(znorm), P(znorm), x_a(znorm), x_b(znorm), x_c(znorm))   ;
END #FOR

# The Pressure and Velocity Subclass:
#-----------------------------------

# The Phase Density
FOR znorm := 0 TO 1 DO

        den(znorm) = C_a(znorm) + C_b(znorm) + C_c(znorm)   ;

END #FOR

# The Zone Pressure

FOR znorm := 0 TO 1 DO

        P(znorm) = 100000   ;

END #FOR


END #ZONE MODEL Z2
#-----------------------------------------------------------
# The gPROMS Model For Zone: Z3
#-----------------------------------------------------------

MODEL Zone_MODEL3

  VARIABLE

        T_P1                    AS Temperature
        T                       AS Temperature
        T_P2                    AS Temperature
        V                       AS VolumetricHoldup
        x_a                     AS MassFraction
        u                       AS InternalEnergy
        M                       AS MassHoldup
        Q_P1                    AS HeatTerm
        Q_P2                    AS HeatTerm
        P                       AS Pressure
        h                       AS Enthalpy
        den                     AS Density

EQUATION

# The Portal Equations for Port: P1
#----------------------------------

T_P1 = T   ;

# The Portal Equations for Port: P2
#----------------------------------

T_P2 = T   ;


# The Extent Subclass:
#---------------------------------

V = 0.2   ;
```

```
# The Composition Subclass:
#--------------------------

# The Mass Fraction for Components: StainlessSteel

x_a = 1   ;


# The Temperature Subclass:
#--------------------------

# The Lumped Energy Conservation Equation:

$(M * u) = Q_P1 + Q_P2   ;

# The Equation of State for the Internal Energy of Phase:StainlessSteel

u = f(T, P, x_a)   ;

# The Equation of State for the Enthalpy of Phase: StainlessSteel

h = f(T, P, x_a)   ;

# The Density Equation of State for Phase: StainlessSteel

den = f(T, P, x_a)   ;

V = M / den   ;


# The Pressure Subclass:
#-----------------------

P = 100000   ;


END #ZONE MODEL Z3
```

```
#-------------------------------------------------------------
# The gPROMS Model For Reservoir: R101
#-------------------------------------------------------------

MODEL RESERVOIR_MODEL1

  VARIABLE

        T                              AS Temperature
        P                              AS Pressure
        x_a                            AS MassFraction
        x_b                            AS MassFraction
        x_c                            AS MassFraction
        h                              AS Enthalpy

EQUATION

T = 920   ;

P = 100000   ;

x_a = 0.25   ;

x_b = 0   ;

x_c = 0.75   ;

# The Equation of State for the Enthalpy of Phase: Product_stream

h = f(T, P, x_a, x_b, x_c)   ;

END #RESERVOIR MODEL R101

#-------------------------------------------------------------
# The gPROMS Model For Reservoir: R102
#-------------------------------------------------------------

MODEL RESERVOIR_MODEL2
```

```
  VARIABLE

        P                               AS Pressure

EQUATION

P = 100000  ;

END #RESERVOIR MODEL R102

#-------------------------------------------------------------
# The gPROMS Model For Reservoir: R103
#-------------------------------------------------------------

MODEL RESERVOIR_MODEL3

  VARIABLE

        T                               AS Temperature

EQUATION

T = 800  ;

END #RESERVOIR MODEL R103

#-------------------------------------------------------------
# The gPROMS Model For All The Connections
# And The Vessel Constraint Equations:
# And The Equations for the Boundary and Port Velocities:
#-------------------------------------------------------------

MODEL FLOWSHEET

UNIT     Z1       AS ZONE_MODEL1
UNIT     Z2       AS ZONE_MODEL2
UNIT     Z3       AS ZONE_MODEL3
UNIT     R101     AS RESERVOIR_MODEL1
UNIT     R102     AS RESERVOIR_MODEL2
UNIT     R103     AS RESERVOIR_MODEL3

PARAMETER
        r                               AS REAL
        pi                              AS REAL

VARIABLE

        alpha_C104                      AS HeatTransferCoef
        A_C105                          AS Area
        alpha_C105                      AS HeatTransferCoef

EQUATION

R101.h = Z1.h_P1  ;

R101.P = Z1.P_P1  ;

R101.x_a = Z1.x_P1_a  ;

R101.x_b = Z1.x_P1_b  ;

R101.x_c = Z1.x_P1_c  ;

Z1.h_P2 = Z2.h_P1  ;

Z1.F_P2 = Z2.F_P1  ;

Z1.x_P2_a = Z2.x_P1_a  ;

Z1.x_P2_b = Z2.x_P1_b  ;

Z1.x_P2_c = Z2.x_P1_c  ;

Z1.P_P2 = Z2.P_P1  ;

Z2.qflux_P3 = alpha_C104 * (Z3.T_P1 - Z2.T_P3)  ;

Z3.Q_P1 =  - Integral( znorm := 0:1 ; 2 * pi * r * Z2.qflux_P3(znorm))  ;
```

```
Z3.Q_P2 =  - A_C105 * alpha_C105 * (Z3.T_P2 - R103.T)  ;

END  #MODEL FLOWSHEET
```

K.5 The Drum Boiler Model (gPROMS)

```
#-----------------------------------------------------------
# The gPROMS Input File For Process: DRUM_BOILER
#-----------------------------------------------------------

DECLARE
  TYPE

    Area              =  1       :   0.0      :   1e5      UNIT = "m2"
    NoType            =  1       :   -1e200   :    1e200   UNIT = "-"
    MassFlowrate      =  1       :   -1e20    :   1e20     UNIT = "kg/s"
    MassFlux          =  1e3     :   -1e3     :   1e8      UNIT = "kg/m2s"
    Velocity          =  1       :   -1e3     :   1e3      UNIT = "m/s"
    Density           =  1e3     :   -1e-5    :   1e10     UNIT = "kg/m3"
    MassHoldup        =  1       :   -1e-3    :   1e20     UNIT = "kg"
    VolumetricHoldup  =  1       :   0        :   1e5      UNIT = "m3"
    MassFraction      =  0.5     :   -1e-3    :   1.001    UNIT = "-"
    VolumeFraction    =  0.5     :   -1e-3    :   1.001    UNIT = "-"
    Pressure          =  1e5     :   1        :   1e20     UNIT = "kN/m2"
    Temperature       =  300     :   10       :   5e4      UNIT = "K"
    InternalEnergy    =  1e3     :   -1E20    :   1e20     UNIT = "kJ/kg"
    Enthalpy          =  1e2     :   -1E20    :   1e20     UNIT = "kJ/kg"
    HeatTerm          =  1       :   -1e50    :   1e50     UNIT = "kJ/s"
    HeatFlux          =  1e3     :   -1e10    :   1e10     UNIT = "kJ/m2s"
    Length            =  1       :   -0       :   1e5      UNIT = "m"
    Stress            =  1e3     :   0        :   1e10     UNIT = "N/m"

END

#-----------------------------------------------------------
# The gPROMS Model For Zone: ZONE_MODEL1
#-----------------------------------------------------------

MODEL Zone_MODEL1

  PARAMETER

        Fref                            AS REAL

  VARIABLE

        h_PA1                           AS Enthalpy
        x_PA1_a                         AS MassFraction
        h_PA2                           AS Enthalpy
        x_PA2_a                         AS MassFraction
        F_PA1                           AS MassFlowrate
        F_PA2                           AS MassFlowrate

EQUATION

# The Flowrate Subclass:
#-----------------------

0 = F_PA1 - F_PA2  ;

F_PA1 = Fref  ;

# The Composition Subclass:
#--------------------------

# The Principle of Conservation for Components: H2O

0 = F_PA1 * x_PA1_a - F_PA2 * x_PA2_a  ;


# The Temperature Subclass:
#--------------------------

# The No Extent Energy Balance:
```

```
0 = F_PA1 * h_PA1 - F_PA2 * h_PA2  ;

END #ZONE MODEL Z1

#-----------------------------------------------------------
# The gPROMS Model For Zone: Z2
#-----------------------------------------------------------

MODEL Zone_MODEL2

  VARIABLE

        P_PB1                          AS Pressure
        P                              AS Pressure
        h_PB1                          AS Enthalpy
        x_PB1_a                        AS MassFraction
        P_PB2                          AS Pressure
        h_PB2                          AS Enthalpy
        h_Water                        AS Enthalpy
        x_PB2_a                        AS MassFraction
        x_Water_a                      AS MassFraction
        P_PB3                          AS Pressure
        h_PB3                          AS Enthalpy
        x_PB3_a                        AS MassFraction
        P_PB4                          AS Pressure
        h_PB4                          AS Enthalpy
        x_PB4_a                        AS MassFraction
        P_PB5                          AS Pressure
        h_PB5                          AS Enthalpy
        h_Steam                        AS Enthalpy
        x_PB5_a                        AS MassFraction
        x_Steam_a                      AS MassFraction
        M                              AS MassHoldup
        F_PB1                          AS MassFlowrate
        F_PB2                          AS MassFlowrate
        F_PB3                          AS MassFlowrate
        F_PB4                          AS MassFlowrate
        F_PB5                          AS MassFlowrate
        x_Water                        AS MassFraction
        M_Water                        AS MassHoldup
        x_Steam                        AS MassFraction
        M_Steam                        AS MassHoldup
        den_Water                      AS Density
        den_Steam                      AS Density
        T                              AS Temperature
        V                              AS VolumetricHoldup
        u                              AS InternalEnergy
        u_Water                        AS InternalEnergy
        u_Steam                        AS InternalEnergy
        Vfrac_Water                    AS VolumeFraction
        Vfrac_Steam                    AS VolumeFraction

  EQUATION

  # The Portal Equations for Port: PB1
  #-----------------------------------

  P_PB1 = P  ;


  # The Portal Equations for Port: PB2
  #-----------------------------------

  P_PB2 = P  ;

  h_PB2 = h_Water  ;

  x_PB2_a = x_Water_a  ;


  # The Portal Equations for Port: PB3
  #-----------------------------------

  P_PB3 = P  ;


  # The Portal Equations for Port: PB4
  #-----------------------------------

  P_PB4 = P  ;
```

```
# The Portal Equations for Port: PB5
#-----------------------------------

P_PB5 = P  ;

h_PB5 = h_Steam  ;

x_PB5_a = x_Steam_a  ;


# The Core Equations:
#--------------------

# The Total Mass of Phases Water, Steam

$(M) = F_PB1 - F_PB2 + F_PB3 + F_PB4 - F_PB5  ;

T = f(P)  ;

# The Mass Fraction of Phases Water, Steam

x_Water = M_Water / M  ;

x_Steam = M_Steam / M  ;

# The Total Mass of the Zone

M = M_Water + M_Steam  ;

# The Density of Phases: Water, Steam

den_Water = constant  ;

# The Density Equation of State for Phase: Steam

den_Steam = f(T, P, x_Steam_a)  ;


# The Extent Subclass:
#---------------------------------

V = 20.36  ;


# The Composition Subclass:
#--------------------------

# The Composition of Phase: Water
# The Mass Fraction for Components: H2O within Phase: Water

x_Water_a = 1  ;


# The Composition of Phase: Steam
# The Mass Fraction for Components: H2O within Phase: Steam

x_Steam_a = 1  ;



# The Temperature Subclass:
#--------------------------

# The Lumped, Homogeneous, Two-Phase Energy Conservation Equation:

$(M * u) = F_PB1 * h_PB1 - F_PB2 * h_PB2 + F_PB3 * h_PB3 + F_PB4 * h_PB4 - F_PB5 * h_PB5  ;

#

u = x_Water * u_Water + x_Steam * u_Steam  ;

# The Equation of State for the Internal Energy of Phases Water, Steam
```

```
u_Water = f(T, P, x_Water_a)  ;

u_Steam = f(T, P, x_Steam_a)  ;

# The Equation of State for the Enthalpy of Phases Water, Steam

h_Water = f(T, P, x_Water_a)  ;

h_Steam = f(T, P, x_Steam_a)  ;

# The Pressure Subclass:
#-----------------------

V = M_Water / den_Water + M_Steam / den_Steam  ;

# The Volume Fraction Subclass:
#------------------------------

M_Water = Vfrac_Water * den_Water * V  ;

M_Steam = Vfrac_Steam * den_Steam * V  ;

# The Relative Motion Subclass:
#------------------------------


END #ZONE MODEL Z2


#-------------------------------------------------------------
# The gPROMS Model For Zone: Z3
#-------------------------------------------------------------

MODEL Zone_MODEL3

  PARAMETER

        g                               AS REAL
        latent_heat                     AS REAL


DISTRIBUTION_DOMAIN
      Axial          AS ( 0 : 1)


VARIABLE

        F_PD1_a                         AS MassFlowrate
        F_PD1                           AS MassFlowrate
        x_PD1_a                         AS MassFraction
        P_PD1                           AS Pressure
        h_PD1                           AS Enthalpy
        x_PD2_a                         AS MassFraction
        h_PD2                           AS Enthalpy
        F_PD2                           AS MassFlowrate
        A                               AS Area
        P_PD2                           AS Pressure
        x_PD3_a                         AS MassFraction
        h_PD3                           AS Enthalpy
        F_PD3                           AS MassFlowrate
        P_PD3                           AS Pressure
        V                               AS VolumetricHoldup
        Z_Inlet                         AS Length
        Z_Outlet                        AS Length
        b                               AS NoType
        Pw                              AS Length
        den                             AS DISTRIBUTION(Axial) OF Density
        den_Water                       AS DISTRIBUTION(Axial) OF Density
        den_Steam                       AS DISTRIBUTION(Axial) OF Density
        Vfrac_Water                     AS DISTRIBUTION(Axial) OF VolumeFraction
        Vfrac_Steam                     AS DISTRIBUTION(Axial) OF VolumeFraction
        T                               AS DISTRIBUTION(Axial) OF Temperature
        P                               AS DISTRIBUTION(Axial) OF Pressure
        x_Water_a                       AS DISTRIBUTION(Axial) OF MassFraction
        x_Steam_a                       AS DISTRIBUTION(Axial) OF MassFraction
        veloc                           AS DISTRIBUTION(Axial) OF Velocity
        TAUWALL                         AS DISTRIBUTION(Axial) OF Stress
```

```
        veloc_Water                     AS DISTRIBUTION(Axial) OF Velocity
        veloc_Steam                     AS DISTRIBUTION(Axial) OF Velocity
        MTRANS                          AS DISTRIBUTION(Axial) OF MassFlux
        qflux                           AS DISTRIBUTION(Axial) OF HeatFlux
        Pw                              AS Length
        A                               AS Area
        g                               AS REAL

EQUATION

# The Portal Equations for Port: PD1
#-----------------------------------

F_PD1_a = F_PD1 * x_PD1_a  ;

P_PD1 = P(0)  ;

# The Portal Equations for Port: PD2
#-----------------------------------

x_PD2_a = x_Water_a(1)  ;

h_PD2 = h_Water(1)  ;

F_PD2 = A * veloc_Water(1) * den_Water(1) * Vfrac_Water(1)  ;

P_PD2 = P(1)  ;

# The Portal Equations for Port: PD3
#-----------------------------------

x_PD3_a = x_Steam_a(1)  ;

h_PD3 = h_Steam(1)  ;

F_PD3 = A * veloc_Steam(1) * den_Steam(1) * Vfrac_Steam(1)  ;

P_PD3 = P(1)  ;

# The Core Subclass:
#-----------------------------------

# The Density of the Mixture

FOR znorm := 0 TO 1 DO

        den(znorm) = Vfrac_Water(znorm) * den_Water(znorm) + Vfrac_Steam(znorm) * den_Steam(znorm)  ;

END #FOR

# The Density of Phases: Water, Steam

# The Density Equation of State for Phase: Water

FOR znorm := 0 TO 1 DO

        den_Water(znorm) = f(T(znorm), P(znorm), x_Water_a(znorm))  ;

END #FOR

# The Density Equation of State for Phase: Steam

FOR znorm := 0 TO 1 DO

        den_Steam(znorm) = f(T(znorm), P(znorm), x_Steam_a(znorm))  ;

END #FOR

# The Extent Subclass:
#-----------------------------------

V = A * (Z_Outlet - Z_Inlet)  ;

Z_Inlet = 0  ;
```

```
Z_Outlet = 25  ;

b = sin(90)  ;


# The Composition Subclass:
#--------------------------

# The Composition of Phase: Water
# The Mass Fraction for Components: H2O within Phase: Water

FOR znorm := 0 TO 1 DO

        x_Water_a(znorm) = 1  ;

END #FOR


# The Composition of Phase: Steam
# The Mass Fraction for Components: H2O within Phase: Steam

FOR znorm := 0 TO 1 DO

        x_Steam_a(znorm) = 1  ;

END #FOR



# The Temperature Subclass:
#--------------------------

FOR znorm := 0 TO 1 DO

        T(znorm) = f(P(znorm))  ;

END #FOR


# The Pressure and Velocity Subclass:
#------------------------------------

# The 1-D Duct Momentum Balance

FOR znorm := 0 TO 1|- DO

       1 / (Z_Outlet - Z_Inlet) * PARTIAL((den(znorm) * veloc(znorm) * veloc(znorm)), Axial) + 1 /
(Z_Outlet - Z_Inlet) * PARTIAL((P(znorm)), Axial) + (TAUWALL(znorm) * Pw / A) - den(znorm) * g * b = 0
;

END #FOR

# The 1-D Duct Total Mass Balance

FOR znorm := 0|+ TO 1 DO

        + 1 /  ( Z_Outlet - Z_Inlet )  * PARTIAL((veloc(znorm) * den(znorm)), Axial) = 0  ;

END #FOR

# The Boundary Condition at znorm = 0
F_PD1 = A * Vfrac_Water(0) * veloc_Water(0) * den_Water(0)  ;

FOR znorm := 0 TO 1 DO

       veloc_Water(znorm) = veloc(znorm)  ;

END #FOR

FOR znorm := 0 TO 1 DO

       veloc_Steam(znorm) = veloc(znorm)  ;

END #FOR


# The Volume Fraction Subclass:
#------------------------------

# The 1-D Duct Phase Balance for Phase Steam
```

```
FOR znorm := 0|+ TO 1 DO

        1 / (Z_Outlet - Z_Inlet) * PARTIAL((veloc_Steam(znorm) * Vfrac_Steam(znorm) *
den_Steam(znorm)), Axial) - MTRANS(znorm) = 0   ;

END #FOR

# The Boundary Condition at znorm = 0
Vfrac_Steam(0) = 0   ;

FOR znorm := 0 TO 1 DO

        Vfrac_Steam(znorm) + Vfrac_Water(znorm) = 1   ;

END #FOR
```

```
The Internal Connection Subclass:
---------------------------------

Zone Domain = [0, 1]
MTRANS(Z3) = ((qflux(Z3) * Pw(Z3) / A(Z3)) + den(Z3) * g) / latent_heat


END #ZONE MODEL Z3


#-------------------------------------------------------------
# The gPROMS Model For Zone: Z4
#-------------------------------------------------------------

MODEL Zone_MODEL4

  PARAMETER

        g                                   AS REAL


  DISTRIBUTION_DOMAIN
        Axial           AS ( 0 : 1)


  VARIABLE

        F_PE1_a                             AS MassFlowrate
        F_PE1                               AS MassFlowrate
        x_PE1_a                             AS MassFraction
        P_PE1                               AS Pressure
        Q_PE1                               AS HeatTerm
        h_PE1                               AS Enthalpy
        x_PE2_a                             AS MassFraction
        h_PE2                               AS Enthalpy
        F_PE2                               AS MassFlowrate
        A                                   AS Area
        P_PE2                               AS Pressure
        V                                   AS VolumetricHoldup
        Z_Inlet                             AS Length
        Z_Outlet                            AS Length
        b                                   AS NoType
        den                                 AS DISTRIBUTION(Axial) OF Density
        T                                   AS DISTRIBUTION(Axial) OF Temperature
        P                                   AS DISTRIBUTION(Axial) OF Pressure
        x_a                                 AS DISTRIBUTION(Axial) OF MassFraction
        veloc                               AS DISTRIBUTION(Axial) OF Velocity
        htot                                AS DISTRIBUTION(Axial) OF Enthalpy
        h                                   AS DISTRIBUTION(Axial) OF Enthalpy
        utot                                AS DISTRIBUTION(Axial) OF InternalEnergy
        u                                   AS DISTRIBUTION(Axial) OF InternalEnergy


  EQUATION

  # The Portal Equations for Port: PE1
  #-----------------------------------

  F_PE1_a = F_PE1 * x_PE1_a   ;

  P_PE1 = P(0)   ;

  Q_PE1 = F_PE1 * h_PE1   ;
```

```
# The Portal Equations for Port: PE2
#-----------------------------------

x_PE2_a = x_a(1)  ;

h_PE2 = h(1)  ;

F_PE2 = A * veloc(1) * den(1)  ;

P_PE2 = P(1)  ;


# The Core Subclass:
#-------------------------------

# The Density Equation of State for Phase: Water

FOR znorm := 0 TO 1 DO

        den(znorm) = f(T(znorm), P(znorm), x_a(znorm))  ;

END #FOR


# The Extent Subclass:
#-------------------------------

V = A * (Z_Outlet - Z_Inlet)  ;

Z_Inlet = 0  ;

Z_Outlet = 25  ;

b = sin(90)  ;


# The Composition Subclass:
#--------------------------

# The Mass Fraction for Components: H2O

FOR znorm := 0 TO 1 DO

        x_a(znorm) = 1  ;

END #FOR


# The Temperature Subclass:
#--------------------------

# The 1-D Duct Energy Balance

FOR znorm := 0|+ TO 1 DO

        1 / (Z_Outlet - Z_Inlet) * PARTIAL((den(znorm) * veloc(znorm) * htot(znorm)), Axial) -
den(znorm) * g * b * veloc(znorm) = 0  ;

END #FOR

# The Boundary Condition at znorm = 0
Q_PE1 = A * veloc(0) * den(0) * htot(0)  ;

FOR znorm := 0 TO 1 DO

        htot(znorm) = h(znorm)  ;

END #FOR

FOR znorm := 0 TO 1 DO

        utot(znorm) = u(znorm)  ;

END #FOR

# The Equation of State for the Enthalpy of Phase: Water

FOR znorm := 0 TO 1 DO

        h(znorm) = f(T(znorm), P(znorm), x_a(znorm))  ;
```

```
END #FOR

# The Equation of State for the Internal Energy of Phase:Water

FOR znorm := 0 TO 1 DO

        u(znorm) = f(T(znorm), P(znorm), x_a(znorm))   ;

END #FOR


# The Pressure and Velocity Subclass:
#-------------------------------------

# The 1-D Duct Total Mass Balance

FOR znorm := 0|+ TO 1 DO

         + 1 /  ( Z_Outlet - Z_Inlet )  * PARTIAL((veloc(znorm) * den(znorm)), Axial) = 0   ;

END #FOR

# The Boundary Condition at znorm = 0
F_PE1 = A * veloc(0) * den(0)   ;

# The 1-D Duct Momentum Balance

FOR znorm := 0 TO 1|- DO

        1 / (Z_Outlet - Z_Inlet) * PARTIAL((den(znorm) * veloc(znorm) * veloc(znorm)), Axial) + 1 /
(Z_Outlet - Z_Inlet) * PARTIAL((P(znorm)), Axial) - den(znorm) * g * b = 0   ;

END #FOR


END #ZONE MODEL Z4


#-----------------------------------------------------------
# The gPROMS Model For Zone: ZONE_MODEL5
#-----------------------------------------------------------

MODEL Zone_MODEL5

  PARAMETER

        Fref                            AS REAL

  VARIABLE

        h_PF1                           AS Enthalpy
        x_PF1_a                         AS MassFraction
        h_PF2                           AS Enthalpy
        x_PF2_a                         AS MassFraction
        F_PF1                           AS MassFlowrate
        F_PF2                           AS MassFlowrate

EQUATION

# The Flowrate Subclass:
#-----------------------

0 = F_PF1 - F_PF2   ;

F_PF1 = Fref   ;


# The Composition Subclass:
#--------------------------

# The Principle of Conservation for Components: H2O

0 = F_PF1 * x_PF1_a - F_PF2 * x_PF2_a   ;


# The Temperature Subclass:
#--------------------------

# The No Extent Energy Balance:
```

```
0 = F_PF1 * h_PF1 - F_PF2 * h_PF2  ;

END #ZONE MODEL Z5

#-----------------------------------------------------------
# The gPROMS Model For Reservoir: R101
#-----------------------------------------------------------

MODEL RESERVOIR_MODEL1

  VARIABLE

        T                               AS Temperature
        P                               AS Pressure
        x_a                             AS MassFraction
        h                               AS Enthalpy

EQUATION

T = 571.3  ;

P = 8.4e+06  ;

x_a = 1  ;

# The Equation of State for the Enthalpy of Phase: Water

h = f(T, P, x_a)  ;

END #RESERVOIR MODEL R101

#-----------------------------------------------------------
# The gPROMS Model For Reservoir: R102
#-----------------------------------------------------------

MODEL RESERVOIR_MODEL2

  VARIABLE

        P                               AS Pressure

EQUATION

P = 100000  ;

END #RESERVOIR MODEL R102

#-----------------------------------------------------------
# The gPROMS Model For All The Connections
# And The Vessel Constraint Equations:
# And The Equations for the Boundary and Port Velocities:
#-----------------------------------------------------------

MODEL FLOWSHEET

UNIT     Z1       AS ZONE_MODEL1
UNIT     Z2       AS ZONE_MODEL2
UNIT     Z3       AS ZONE_MODEL3
UNIT     Z4       AS ZONE_MODEL4
UNIT     Z5       AS ZONE_MODEL5
UNIT     R101     AS RESERVOIR_MODEL1
UNIT     R102     AS RESERVOIR_MODEL2

EQUATION

R101.h = Z1.h_PA1  ;

R101.x_a = Z1.x_PA1_a  ;

Z1.h_PA2 = Z2.h_PB1  ;

Z1.F_PA2 = Z2.F_PB1  ;

Z1.x_PA2_a = Z2.x_PB1_a  ;

Z2.h_PB2 = Z4.h_PE1  ;

Z2.F_PB2 = Z4.F_PE1  ;
```

```
Z2.x_PB2_a = Z4.x_PE1_a  ;

Z2.P_PB2 = Z4.P_PE1  ;

Z4.P_PE2 = Z3.P_PD1  ;

Z3.h_PD2 = Z2.h_PB3  ;

Z3.F_PD2 = Z2.F_PB3  ;

Z3.x_PD2_a = Z2.x_PB3_a  ;

Z3.P_PD2 = Z2.P_PB3  ;

Z3.h_PD3 = Z2.h_PB4  ;

Z3.F_PD3 = Z2.F_PB4  ;

Z3.x_PD3_a = Z2.x_PB4_a  ;

Z2.h_PB5 = Z5.h_PF1  ;

Z2.F_PB5 = Z5.F_PF1  ;

Z2.x_PB5_a = Z5.x_PF1_a  ;

END  #MODEL FLOWSHEET
```

K.6 An Absorptive Reactor (gPROMS input file)

```
#-------------------------------------------------------------
# The gPROMS Input File For Process: Adsorptive_reactor
#-------------------------------------------------------------

DECLARE
  TYPE

    Area              =   1       :   0.0     :   1e5     UNIT = "m2"
    NoType            =   1       :   -1e200  :   1e200   UNIT = "-"
    MassFlowrate      =   1       :   -1e20   :   1e20    UNIT = "kg/s"
    MassFlux          =   1e3     :   -1e3    :   1e8     UNIT = "kg/m2s"
    Velocity          =   1       :   -1e3    :   1e3     UNIT = "m/s"
    Density           =   1e3     :   -1e-5   :   1e10    UNIT = "kg/m3"
    VolumetricHoldup  =   1       :   0       :   1e5     UNIT = "m3"
    ReactionSink      =   1e3     :   -1e10   :   1e10    UNIT = "kg/m3s"
    ReactionRate      =   1       :   -1e10   :   1e10    UNIT = "kg/s"
    MassFraction      =   0.5     :   -1e-3   :   1.001   UNIT = "-"
    VolumeFraction    =   0.5     :   -1e-3   :   1.001   UNIT = "-"
    Pressure          =   1e5     :   1       :   1e20    UNIT = "kN/m2"
    Temperature       =   300     :   10      :   5e4     UNIT = "K"
    HeatDiffusionCoef =   1e3     :   -1e3    :   1e8     UNIT = "m-1s-1"
    InternalEnergy    =   1e3     :   -1E20   :   1e20    UNIT = "kJ/kg"
    Enthalpy          =   1e2     :   -1E20   :   1e20    UNIT = "kJ/kg"
    HeatTerm          =   1       :   -1e50   :   1e50    UNIT = "kJ/s"
    HeatFlux          =   1e3     :   -1e10   :   1e10    UNIT = "kJ/m2s"
    Length            =   1       :   -0      :   1e5     UNIT = "m"

END

#-------------------------------------------------------------
# The gPROMS Model For Zone: ZONE_MODEL1
#-------------------------------------------------------------

MODEL Zone_MODEL1

PARAMETER

      Fref                          AS REAL

VARIABLE

      P_P1                          AS Pressure
      h_P1                          AS Enthalpy
      x_P1_a                        AS MassFraction
      x_P1_b                        AS MassFraction
      P_P2                          AS Pressure
      h_P2                          AS Enthalpy
```

```
        x_P2_a                          AS MassFraction
        x_P2_b                          AS MassFraction
        F_P1                            AS MassFlowrate
        F_P2                            AS MassFlowrate

EQUATION

# The Flowrate Subclass:
#-----------------------

0 = F_P1 - F_P2  ;

F_P1 = Fref  ;


# The Composition Subclass:
#--------------------------

# The Principle of Conservation for Components: Methylcyclohexane, Inert

0 = F_P1 * x_P1_a - F_P2 * x_P2_a  ;

0 = F_P1 * x_P1_b - F_P2 * x_P2_b  ;


# The Temperature Subclass:
#--------------------------

# The No Extent Energy Balance:

0 = F_P1 * h_P1 - F_P2 * h_P2  ;


END #ZONE MODEL Z1


#-------------------------------------------------------------
# The gPROMS Model For Zone: Z2
#-------------------------------------------------------------

MODEL Zone_MODEL2

PARAMETER

        Ypsilon(Product,a,j)            AS REAL
        Ypsilon(Product,b,j)            AS REAL
        Ypsilon(Product,c,j)            AS REAL
        Ypsilon(Product,d,j)            AS REAL
        g                               AS REAL
        heat_diff_coef_Product          AS REAL
        heat_diff_coef_Solid            AS REAL


DISTRIBUTION_DOMAIN
      Axial          AS ( 0 : 1)


VARIABLE

        F_P1_a                          AS MassFlowrate
        F_P1                            AS MassFlowrate
        x_P1_a                          AS MassFraction
        F_P1_b                          AS MassFlowrate
        F_P1_c                          AS MassFlowrate
        F_P1_d                          AS MassFlowrate
        x_P1_d                          AS MassFraction
        P_P1                            AS Pressure
        Q_P1                            AS HeatTerm
        h_P1                            AS Enthalpy
        Qdiff_P1                        AS HeatTerm
        x_P1_b                          AS MassFraction
        x_P2_a                          AS MassFraction
        x_P2_b                          AS MassFraction
        x_P2_c                          AS MassFraction
        x_P2_d                          AS MassFraction
        h_P2                            AS Enthalpy
        F_P2                            AS MassFlowrate
        A                               AS Area
        P_P2                            AS Pressure
        Qdiff_P2                        AS HeatTerm
        V                               AS VolumetricHoldup
```

```
Z_Inlet                 AS Length
Z_Outlet                AS Length
b                       AS NoType
den                     AS DISTRIBUTION(Axial) OF Density
den_Product             AS DISTRIBUTION(Axial) OF Density
den_Solid               AS DISTRIBUTION(Axial) OF Density
Vfrac_Product           AS DISTRIBUTION(Axial) OF VolumeFraction
Vfrac_Solid             AS DISTRIBUTION(Axial) OF VolumeFraction
T                       AS DISTRIBUTION(Axial) OF Temperature
P                       AS DISTRIBUTION(Axial) OF Pressure
x_Product_a             AS DISTRIBUTION(Axial) OF MassFraction
x_Product_b             AS DISTRIBUTION(Axial) OF MassFraction
x_Product_c             AS DISTRIBUTION(Axial) OF MassFraction
x_Product_d             AS DISTRIBUTION(Axial) OF MassFraction
x_Solid_a               AS DISTRIBUTION(Axial) OF MassFraction
x_Solid_b               AS DISTRIBUTION(Axial) OF MassFraction
C_Product_a             AS DISTRIBUTION(Axial) OF Density
veloc_Product           AS DISTRIBUTION(Axial) OF Velocity
R_Product_a             AS DISTRIBUTION(Axial) OF ReactionSink
C_Product_b             AS DISTRIBUTION(Axial) OF Density
R_Product_b             AS DISTRIBUTION(Axial) OF ReactionSink
MTRANS_b                AS DISTRIBUTION(Axial) OF MassFlux
C_Product_c             AS DISTRIBUTION(Axial) OF Density
R_Product_c             AS DISTRIBUTION(Axial) OF ReactionSink
C_Product_d             AS DISTRIBUTION(Axial) OF Density
R_Product_d             AS DISTRIBUTION(Axial) OF ReactionSink
r(Z2,Product,j)         AS DISTRIBUTION(Axial) OF ReactionRate
C_Solid_a               AS DISTRIBUTION(Axial) OF Density
C_Solid_b               AS DISTRIBUTION(Axial) OF Density
MTRANS                  AS DISTRIBUTION(Axial) OF MassFlux
utot                    AS DISTRIBUTION(Axial) OF InternalEnergy
htot_Product            AS DISTRIBUTION(Axial) OF Enthalpy
qflux                   AS DISTRIBUTION(Axial) OF HeatFlux
heat_diff_coef          AS DISTRIBUTION(Axial) OF HeatDiffusionCoef
x_Product               AS DISTRIBUTION(Axial) OF MassFraction
x_Solid                 AS DISTRIBUTION(Axial) OF MassFraction
h_Product               AS DISTRIBUTION(Axial) OF Enthalpy
utot_Product            AS DISTRIBUTION(Axial) OF InternalEnergy
u_Product               AS DISTRIBUTION(Axial) OF InternalEnergy
utot_Solid              AS DISTRIBUTION(Axial) OF InternalEnergy
u_Solid                 AS DISTRIBUTION(Axial) OF InternalEnergy
EQUATION

# The Portal Equations for Port: P1
#----------------------------------

F_P1_a = F_P1 * x_P1_a  ;

F_P1_b = 0  ;

F_P1_c = 0  ;

F_P1_d = F_P1 * x_P1_d  ;

P_P1 = P(0)  ;

Q_P1 = F_P1 * h_P1 + Qdiff_P1  ;


# The Portal Equations for Port: P2
#----------------------------------

x_P2_a = x_Product_a(1)  ;

x_P2_b = x_Product_b(1)  ;

x_P2_c = x_Product_c(1)  ;

x_P2_d = x_Product_d(1)  ;

h_P2 = h_Product(1)  ;

F_P2 = A * veloc_Product(1) * den_Product(1) * Vfrac_Product(1)  ;

P_P2 = P(1)  ;

Qdiff_P2 = A * qflux(1)  ;

# The Core Subclass:
```

```
#---------------------------------

# The Density of the Mixture

FOR znorm := 0 TO 1 DO

        den(znorm) = Vfrac_Product(znorm) * den_Product(znorm) + Vfrac_Solid(znorm) * den_Solid(znorm)
;

END #FOR

# The Density Equation of State for Phase: Product

FOR znorm := 0 TO 1 DO

        den_Product(znorm) = f(T(znorm), P(znorm), x_Product_a(znorm), x_Product_b(znorm),
        x_Product_c(znorm), x_Product_d(znorm))  ;

END #FOR

# The Density Equation of State for Phase: Solid

FOR znorm := 0 TO 1 DO

        den_Solid(znorm) = f(T(znorm), P(znorm), x_Solid_a(znorm), x_Solid_b(znorm))  ;

END #FOR


# The Extent Subclass:
#---------------------------------

V = A * (Z_Outlet - Z_Inlet)  ;

Z_Inlet = 0  ;

Z_Outlet = 1  ;

b = sin(0)  ;


# The Composition Subclass:
#-------------------------

# The Composition of Phase: Product
# The Two Phase Duct Mass Balances for: Methylcyclohexane, Toluene, Hydrogen, Inert

# The 1-D Duct Component Balance for Component Methylcyclohexane

FOR znorm := 0|+ TO 1 DO

        $(Vfrac_Product(znorm) * C_Product_a(znorm)) + 1 / (Z_Outlet - Z_Inlet) *
PARTIAL((veloc_Product(znorm) * Vfrac_Product(znorm) * C_Product_a(znorm)), Axial) -
Vfrac_Product(znorm) * R_Product_a(znorm) = 0  ;

END #FOR

# The 1-D Duct Component Balance for Component Toluene

FOR znorm := 0|+ TO 1 DO

        $(Vfrac_Product(znorm) * C_Product_b(znorm)) + 1 / (Z_Outlet - Z_Inlet) *
PARTIAL((veloc_Product(znorm) * Vfrac_Product(znorm) * C_Product_b(znorm)), Axial) -
Vfrac_Product(znorm) * R_Product_b(znorm) + MTRANS_b(znorm) = 0  ;

END #FOR

# The 1-D Duct Component Balance for Component Hydrogen

FOR znorm := 0|+ TO 1 DO

        $(Vfrac_Product(znorm) * C_Product_c(znorm)) + 1 / (Z_Outlet - Z_Inlet) *
PARTIAL((veloc_Product(znorm) * Vfrac_Product(znorm) * C_Product_c(znorm)), Axial) -
Vfrac_Product(znorm) * R_Product_c(znorm) = 0  ;

END #FOR

# The 1-D Duct Component Balance for Component Inert

FOR znorm := 0|+ TO 1 DO
```

```
        $(Vfrac_Product(znorm) * C_Product_d(znorm)) + 1 / (Z_Outlet - Z_Inlet) *
PARTIAL((veloc_Product(znorm) * Vfrac_Product(znorm) * C_Product_d(znorm)), Axial) -
Vfrac_Product(znorm) * R_Product_d(znorm) = 0  ;

END #FOR

# The Boundary Condition at znorm = 0
F_P1_a = A * veloc_Product(0) * Vfrac_Product(0) * C_Product_a(0)  ;

# The Boundary Condition at znorm = 0
F_P1_b = A * veloc_Product(0) * Vfrac_Product(0) * C_Product_b(0)  ;

# The Boundary Condition at znorm = 0
F_P1_c = A * veloc_Product(0) * Vfrac_Product(0) * C_Product_c(0)  ;

# The Boundary Condition at znorm = 0
F_P1_d = A * veloc_Product(0) * Vfrac_Product(0) * C_Product_d(0)  ;


# The Reaction Kinetic Equations for: Methylcyclohexane, Toluene, Hydrogen, Inert

FOR znorm := 0 TO 1 DO

        R_Product_a(znorm) = SIGMA(j)(Ypsilon(Product,a,j) * r_Product_j)  ;

END #FOR

FOR znorm := 0 TO 1 DO

        R_Product_b(znorm) = SIGMA(j)(Ypsilon(Product,b,j) * r_Product_j)  ;

END #FOR

FOR znorm := 0 TO 1 DO

        R_Product_c(znorm) = SIGMA(j)(Ypsilon(Product,c,j) * r_Product_j)  ;

END #FOR

FOR znorm := 0 TO 1 DO

        R_Product_d(znorm) = SIGMA(j)(Ypsilon(Product,d,j) * r_Product_j)  ;

END #FOR

# The Mass Fraction for Components: Methylcyclohexane, Toluene, Hydrogen, Inert

FOR znorm := 0 TO 1 DO

        x_Product_a(znorm) = C_Product_a(znorm) / den_Product(znorm)  ;

END #FOR

FOR znorm := 0 TO 1 DO

        x_Product_b(znorm) = C_Product_b(znorm) / den_Product(znorm)  ;

END #FOR

FOR znorm := 0 TO 1 DO

        x_Product_c(znorm) = C_Product_c(znorm) / den_Product(znorm)  ;

END #FOR

FOR znorm := 0 TO 1 DO

        x_Product_d(znorm) = C_Product_d(znorm) / den_Product(znorm)  ;

END #FOR


# The Composition of Phase: Solid
# The Two Phase Duct Mass Balances for: Catalyst, Toluene

# The 1-D Duct Component Balance for Component Catalyst

FOR znorm := 0 TO 1 DO
```

```
        $(Vfrac_Solid(znorm) * C_Solid_a(znorm)) = 0  ;
END #FOR
# The 1-D Duct Component Balance for Component Toluene
FOR znorm := 0 TO 1 DO
        $(Vfrac_Solid(znorm) * C_Solid_b(znorm)) - MTRANS_b(znorm) = 0  ;
END #FOR

# The Mass Fraction for Components: Catalyst, Toluene
FOR znorm := 0 TO 1 DO
        x_Solid_a(znorm) = C_Solid_a(znorm) / den_Solid(znorm)  ;
END #FOR
FOR znorm := 0 TO 1 DO
        x_Solid_b(znorm) = C_Solid_b(znorm) / den_Solid(znorm)  ;
END #FOR

# The Total Interfacial Exchange of Mass:
#----------------------------------------
FOR znorm := 0 TO 1 DO
        MTRANS(znorm) = MTRANS_b(znorm)  ;
END #FOR


# The Temperature Subclass:
#--------------------------
# The 1-D Duct Energy Balance
FOR znorm := 0|+ TO 1|- DO
        $(den(znorm) * utot(znorm)) + 1 / (Z_Outlet - Z_Inlet) * PARTIAL((Vfrac_Product(znorm) *
den_Product(znorm) * veloc_Product(znorm) * htot_Product(znorm)), Axial) + 1 / (Z_Outlet - Z_Inlet) *
PARTIAL((qflux(znorm)), Axial) - den(znorm) * g * b * veloc_Product(znorm) = 0  ;
END #FOR
# The Boundary Condition at znorm = 0
Q_P1 = A * Vfrac_Product(0) * veloc_Product(0) * den_Product(0) * htot_Product(0) + A * qflux(0)  ;
FOR znorm := 0 TO 1 DO
        qflux(znorm) =  - (heat_diff_coef(znorm) / (Z_Outlet - Z_Inlet)) * PARTIAL((T(znorm)), Axial)
;
END #FOR
FOR znorm := 0 TO 1 DO
        heat_diff_coef(znorm) = Vfrac_Product(znorm) * heat_diff_coef_Product + Vfrac_Solid(znorm) *
heat_diff_coef_Solid  ;
END #FOR

# The Mixture's Thermodynamic Properties
FOR znorm := 0 TO 1 DO
        utot(znorm) = x_Product(znorm) * utot_Product(znorm) + x_Solid(znorm) * utot_Solid(znorm)  ;
END #FOR
FOR znorm := 0 TO 1 DO
        x_Product(znorm) = den_Product(znorm) * Vfrac_Product(znorm) / den(znorm)  ;
```

```
END #FOR

FOR znorm := 0 TO 1 DO

        x_Solid(znorm) = den_Solid(znorm) * Vfrac_Solid(znorm) / den(znorm)   ;

END #FOR

# The Total Enthalpy and Internal Energy of Phases Product, Solid

FOR znorm := 0 TO 1 DO

        htot_Product(znorm) = h_Product(znorm)   ;

END #FOR

FOR znorm := 0 TO 1 DO

        utot_Product(znorm) = u_Product(znorm)   ;

END #FOR

FOR znorm := 0 TO 1 DO

        utot_Solid(znorm) = u_Solid(znorm)   ;

END #FOR

# The Equation of State for the Internal Energy of Phases Product, Solid

FOR znorm := 0 TO 1 DO

        u_Product(znorm) = f(T(znorm), P(znorm), x_Product_a(znorm), x_Product_b(znorm),
x_Product_c(znorm), x_Product_d(znorm))   ;

END #FOR

FOR znorm := 0 TO 1 DO

        u_Solid(znorm) = f(T(znorm), P(znorm), x_Solid_a(znorm), x_Solid_b(znorm))   ;

END #FOR

# The Equation of State for the Enthalpy of Phases Product, Solid

FOR znorm := 0 TO 1 DO

        h_Product(znorm) = f(T(znorm), P(znorm), x_Product_a(znorm), x_Product_b(znorm),
x_Product_c(znorm), x_Product_d(znorm))   ;

END #FOR

# The Pressure and Velocity Subclass:
#-----------------------------------

# The Superficial Velocity
F_P1 = A * den_Product(znorm) * veloc_Product(znorm) * Vfrac_Product(znorm)   ;

# The Zone Pressure

FOR znorm := 0 TO 1 DO

        P(znorm) = 200000   ;

END #FOR

# The Volume Fraction Subclass:
#------------------------------

FOR znorm := 0 TO 1 DO

        Vfrac_Product(znorm) = 0.3   ;

END #FOR
```

```
FOR znorm := 0 TO 1 DO

        Vfrac_Solid(znorm) = 0.7   ;

END #FOR


The Internal Connection Subclass:
---------------------------------

User supplied equation required.

END #ZONE MODEL Z2


#-----------------------------------------------------------
# The gPROMS Model For Reservoir: R1
#-----------------------------------------------------------

MODEL RESERVOIR_MODEL1

  VARIABLE

        T                               AS Temperature
        P                               AS Pressure
        x_a                             AS MassFraction
        x_b                             AS MassFraction
        h                               AS Enthalpy

EQUATION

T = 623   ;

P = 200000   ;

x_a = 0.1   ;

x_b = 0.9   ;

# The Equation of State for the Enthalpy of Phase: Raw_materials

h = f(T, P, x_a, x_b)   ;

END #RESERVOIR MODEL R1

#-----------------------------------------------------------
# The gPROMS Model For Reservoir: R2
#-----------------------------------------------------------

MODEL RESERVOIR_MODEL2

  VARIABLE

        P                               AS Pressure

EQUATION

P = 200000   ;

END #RESERVOIR MODEL R2

#-----------------------------------------------------------
# The gPROMS Model For All The Connections
# And The Vessel Constraint Equations:
# And The Equations for the Boundary and Port Velocities:
#-----------------------------------------------------------

MODEL FLOWSHEET

UNIT     Z1       AS ZONE_MODEL1
UNIT     Z2       AS ZONE_MODEL2
UNIT     R1       AS RESERVOIR_MODEL1
UNIT     R2       AS RESERVOIR_MODEL2

EQUATION

R1.h = Z1.h_P1   ;

R1.P = Z1.P_P1   ;

R1.x_a = Z1.x_P1_a   ;
```

```
R1.x_b = Z1.x_P1_b  ;

Z1.h_P2 = Z2.h_P1  ;

Z1.F_P2 = Z2.F_P1  ;

Z1.x_P2_a = Z2.x_P1_a  ;

Z1.x_P2_b = Z2.x_P1_d  ;

Z1.P_P2 = Z2.P_P1  ;

Z2.Qdiff_P1 = 0  ;

Z2.Qdiff_P2 = 0  ;
```

END #MODEL FLOWSHEET

BIBL. LONDON IV.