An integrated framework for representing design history

This item was submitted to Loughborough University's Institutional Repository by the/an author.

Additional Information:


Metadata Record: [https://dspace.lboro.ac.uk/2134/32242](https://dspace.lboro.ac.uk/2134/32242)

Publisher: © Roger Goodwin

Rights: This work is made available according to the conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) licence. Full details of this licence are available at: [https://creativecommons.org/licenses/by-nc-nd/4.0/](https://creativecommons.org/licenses/by-nc-nd/4.0/)

Please cite the published version.
An Integrated Framework for Representing Design History

by
Roger Goodwin

A Doctoral Thesis
Submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy of Loughborough University

January 1997

© Roger Goodwin 1997
ABSTRACT

Design is a difficult and complex process requiring: creativity, experience, domain knowledge, and problem solving skills. Much of the information that is used and generated during the design process is rarely explicitly recorded. This includes the reasons why design decisions were made. This information is commonly referred to as design rationale (DR). As a result many of the tasks that are performed during the design process are still poorly understood and modifications to designs can have unforeseen and possibly dangerous consequences.

This thesis reports on previous and current research in the field of capturing DR. The techniques that have been applied in the past have suffered from several limitations. First, DR is often captured in isolation and not explicitly recorded with the artefact. Second, the research has concentrated on recording the design deliberation and little support has been provided for recording other aspects of DR. Finally little consideration has been given to the access and management aspects of recording such information.

This thesis presents a novel approach to recording DR. In this approach, DR is viewed as a by product of the design process that consists of several different aspects of design, namely deliberation, design space, constraints, functionality, and objectives. A framework is presented that supports the recording and integration of these aspects in a single environment while providing support for traditional design tasks. These aspects of DR are explicitly recorded with the design artefact. The environment presented also supports the access and management of the DR captured and provides support for the management of the design process as a whole.

Keywords: computer aided design, design rationale, integration, design information.
ACKNOWLEDGEMENTS

I would like to thank my supervisor, Dr Paul W. H. Chung, and director of research, Professor Frank P. Lees, for their invaluable guidance and encouragement throughout the course of this work.

I would also like to thank Professor Alan Duxbury for his assistance in the first case study and Colin Pinder for his time and effort in the second. The efforts of Dick Treece, my British Gas supervisor, and the staff at the Gas Research Centre who took part in the third and fourth case studies is also much appreciated.

Many thanks to all my friends and colleagues in the Chemical Engineering Department at Loughborough for all their support, encouragement and entertainment. I would specially like to thank James Soutter for all his effort and enthusiasm in solving irritating computing problems.

Finally I would like to thank my family for their support through all my years of study. Especially Barbara for her support and love through the good times and bad.

This work was funded by British Gas, through a British Gas Research Scholarship.
## CONTENTS

### CHAPTER 1  INTRODUCTION

1.1 Motivation
1.2 Contributions
1.3 Structure of the Thesis

### CHAPTER 2  COMPUTER AIDED DESIGN

2.1 The Process of Design
2.2 Types of Design
2.3 Computer Aided Design Tools
2.4 Strengths and Weaknesses of Computer Aided Design Tools
2.5 Summary

### CHAPTER 3  DESIGN RATIONALE

3.1 Overview of Design Rationale Research
3.2 Design Deliberation
   3.2.1 Toulmin Form
   3.2.2 Rhetorical Structure Theory
   3.2.3 Semi-Structured Message Templates
   3.2.4 Issue Based Information Systems
3.3 Design Constraints
   3.3.1 Mathematical Representations
   3.3.2 Graphical Representations
   3.3.3 Frame Representations
   3.3.4 Rule Representations
3.4 Design Space
   3.4.1 Tree Representations
   3.4.2 Network Representations
3.5 Design Functionality
   3.5.1 Semantic Representations
   3.5.2 Mathematical Representations
   3.5.3 Tree Representations
3.6 Relationships Between Design Objects
3.7 Requirements
   3.7.1 Semantic Representations
   3.7.2 Tree Representations
   3.7.3 Logical Representations
3.8 Integrating Different Aspects of Design Rationale
3.9 Conclusions
CHAPTER 4  AN OVERVIEW OF AN INTEGRATED DESIGN INFORMATION SYSTEM

4.1 An Integrated Design Information System
4.1.1 Design Space
4.1.2 Design Diagrams
4.1.3 Design Deliberation
4.1.4 Design Constraints
4.1.5 Design Functionality
4.1.6 Design Changes

4.2 Recording and Accessing Design Information
4.2.1 Design Diagrams and Alternatives
4.2.2 Design Deliberation
4.2.3 Design Constraints
4.2.4 Design Parameters and Functionality
4.2.5 Design Changes
4.2.6 Project Management

4.3 Summary

CHAPTER 5  IMPLEMENTATION OF IDIS

5.1 System Components
5.2 Frame Based Representation
5.3 Data Structures
5.3.1 Viewpoint Mechanism
5.3.2 Design Diagrams and Sketches
5.3.3 Issue Base
5.3.4 Rule Base

5.4 User Interface and Component Integration
5.4.1 User Interface
5.4.2 Integrating the Components

5.5 Exporting Design Information
5.6 Hardware and Software Requirements

5.7 Summary

CHAPTER 6  USING IDIS

6.1 Starting a New Project
6.2 Creating a New Issue Base
6.3 Exploring the Design Space
6.4 Creating Detailed Design Diagrams
6.5 Adding and Checking Design Constraints
6.6 Exporting the Design Information to Other Packages

6.7 Summary
CHAPTER 7  EVALUATION OF IDIS

7.1  Issue Base Case Study  123
7.2  Reconstruction Case Study  130
7.3  Off-Line Case Study  134
7.4  On-Line Case Study With Multiple Users  136
7.5  Summary  139

CHAPTER 8  CONCLUSIONS

8.1  Contributions of IDIS  141
  8.1.1  Integration  142
  8.1.2  Maintaining the Integrity of an Issue Base  142
  8.1.3  Identifying the Differences Between Designs  143
  8.1.4  Managing Design Information and the Design Project  144
8.2  Limitations of IDIS  144
8.3  Future Work in the Field of Design Rationale Capture and Use  146

REFERENCES  149

APPENDICES

Appendix A  A List of the Units in IDIS
Appendix B  Software and Hardware Requirements of IDIS
Appendix C  IDIS user manual
Appendix D  Issue Bases From First Case Study
Appendix E  Issue Bases From Second Case Study
## Index Of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Some of the research in design rationale since 1985</td>
<td>17</td>
</tr>
<tr>
<td>3.2</td>
<td>The Toulmin Form Micro Argument</td>
<td>18</td>
</tr>
<tr>
<td>3.3</td>
<td>The Extended Toulmin Form model</td>
<td>19</td>
</tr>
<tr>
<td>3.4</td>
<td>The structure of an argument in Rhetorical Structure Theory</td>
<td>20</td>
</tr>
<tr>
<td>3.5</td>
<td>The structure of the Issue Based Information System representation</td>
<td>23</td>
</tr>
<tr>
<td>3.6</td>
<td>Potts and Bruns representation of design deliberation and design artefacts</td>
<td>27</td>
</tr>
<tr>
<td>3.7</td>
<td>Constraint graph representation of constraints f1 to f3 above</td>
<td>33</td>
</tr>
<tr>
<td>3.8</td>
<td>A frame representation of constraints applied to a pump</td>
<td>34</td>
</tr>
<tr>
<td>3.9</td>
<td>DRCS Representation of an Aeroplane</td>
<td>34</td>
</tr>
<tr>
<td>3.10</td>
<td>Linear representation</td>
<td>37</td>
</tr>
<tr>
<td>3.11</td>
<td>Tree representation</td>
<td>38</td>
</tr>
<tr>
<td>3.12</td>
<td>Network representation</td>
<td>38</td>
</tr>
<tr>
<td>3.13</td>
<td>Requiremental and functional space representation</td>
<td>41</td>
</tr>
<tr>
<td>3.14</td>
<td>Sub-components of an aeroplane represented in DRCS notation</td>
<td>49</td>
</tr>
<tr>
<td>3.15</td>
<td>The structure of a nitric acid cooler (NAC)</td>
<td>50</td>
</tr>
<tr>
<td>3.16</td>
<td>A FR representation of the above nitric acid cooler</td>
<td>50</td>
</tr>
<tr>
<td>3.17</td>
<td>A comparison of the IBIS and DRL structures</td>
<td>54</td>
</tr>
<tr>
<td>3.18</td>
<td>DMS representation of design goals</td>
<td>55</td>
</tr>
<tr>
<td>3.19</td>
<td>Clibbon and Edmonds representation of requirements and design objects</td>
<td>56</td>
</tr>
<tr>
<td>4.1</td>
<td>Different aspects of DR grouped together at a single point in the design space</td>
<td>63</td>
</tr>
<tr>
<td>5.1</td>
<td>The AutoCAD screen with IDIS menus</td>
<td>93</td>
</tr>
<tr>
<td>6.1</td>
<td>Bill adds Sue as a member of the project team.</td>
<td>100</td>
</tr>
<tr>
<td>6.2</td>
<td>Bill adds the first issue to the project</td>
<td>101</td>
</tr>
<tr>
<td>6.3</td>
<td>Bill links position 1 to issue 1</td>
<td>102</td>
</tr>
<tr>
<td>6.4</td>
<td>Sue views all the nodes she has not seen</td>
<td>103</td>
</tr>
<tr>
<td>6.5</td>
<td>Sue views the contents of issue 1 and the structure of the issue base</td>
<td>104</td>
</tr>
<tr>
<td>6.6</td>
<td>Sue lists the other members of the design team</td>
<td>105</td>
</tr>
</tbody>
</table>
6.7 Bill views the issue base structure
6.8 Bill views John’s contribution to the project to date
6.9 Bill adds a new viewpoint to the design space.
6.10 A schematic diagram of the acid catalysed liquid phase esterification of ethanol and acetic acid process
6.11 Viewpoint 4 after it had been modified
6.12 Sue views the changes that had been made to viewpoint4
6.13 The completed flow sheet diagram
6.14 Sue sets the parameters for one of the distillation columns
6.15 John loads the rule template in order to create a new rule
6.16 Bill views the constraints that have been recorded and the results of the check on the design
6.17 The final structure of the design space in the Ethyl Acetate Project

7.1 The initial issue base
7.2 The addition of the combinedWith link
7.3 The addition of issue 2
7.4 The removal of the combinedWith links
7.5 The final issue base
7.6 The use of the FollowUp link
## Index Of Tables

<table>
<thead>
<tr>
<th>Table Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Explicit and implicit information used and generated in design</td>
<td>10</td>
</tr>
<tr>
<td>2.2</td>
<td>Some of the commercial CAD tools available</td>
<td>11</td>
</tr>
<tr>
<td>3.1</td>
<td>An example of Lees' decision matrix</td>
<td>28</td>
</tr>
<tr>
<td>3.2</td>
<td>Constraint C2 represented using Debenham's notation</td>
<td>31</td>
</tr>
<tr>
<td>3.3</td>
<td>A comparison of some tools reviewed and aspects of DR they address</td>
<td>57</td>
</tr>
<tr>
<td>4.1</td>
<td>A representation of the constraint</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>“the pressure of heater2 should not exceed 24 bar”</td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>Possible conditions and connection available in IDIS</td>
<td>119</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

This thesis addresses the problem of recording and accessing design rationale (DR) within a framework that supports the collaborative design of chemical plants.

Any design can be described has having two distinct parts, the product of the design and the process by which the product is achieved. The product of the design refers to the final documentation containing aspects such as: the detailed design diagrams, flow sheets, operating and manufacturing procedures, reports, etc. The means by which these are achieved can be referred to as the process of the design. Recording the process by which the design evolves provides information about how the designers produced the final product. It is effectively a history of how the design was created.

The final documentation typically contains the results of the design process but rarely records the history of how these results were achieved. Information such as: the alternatives explored, constraints, design modifications, functionality of the design components and the requirements, are rarely explicitly recorded. The failure to record the history means that a considerable amount of the understanding about the design is lost. This may have serious implications on subsequent modifications and maintenance of chemical plants.

However, the most important and elusive aspect of design history is the rationale behind the decisions made. Understanding why designers explored, accepted and/or rejected certain alternatives is a fundamental part of the design history. An explicit
record of design rationale is vital in providing a better understanding of the final design. This thesis considers novel ways of representing, recording and accessing design rationale within a single integrated system.

1.1 Motivation

Existing designs are regularly reused and redesigned. A deeper understanding of the original design will reduce the chances of dangerous modifications made to it when it is redesigned. One way to avoid serious incidents caused by modifications to chemical plants is to explicitly record the original design assumptions, constraints and the reasons behind any of the decisions made. So, when a modification is proposed, it is possible to check all the relevant factors and assess the impact of the change. The reasoning behind design decisions is commonly referred to as design rationale and an explicit record of it can be used to:

- justify design decisions,
- provide information for the final design documents,
- prevent accidents occurring by making design assumptions explicit,
- assist in modifications to existing designs,
- prevent designers from ‘reinventing the wheel’,
- help develop and adapt cases for Case-Based Reasoning design systems,
- bridge the gap between novice and expert designers.

In this thesis, DR is viewed as a by-product of the design process and a major component of design history that consists of various different types of information. One of the major components is the deliberation that takes place between the designers during the design process. It has been used extensively to record the DR. However, a record of the deliberation in isolation provides only part of the DR. The design alternatives that have been explored, the constraints that have been applied to the design, the requirements and the functionality of the design artefacts are all important aspects of DR. The research in the field of DR has developed tools and techniques to capture these different aspects. The importance of integrating these different aspects has also been identified and techniques have been developed to support integration. A review is given in Chapter 3.
DR tools should support the recording of most, if not all, of these aspects and integrate the information in a single environment with the design artefact. It is argued in this thesis that the current DR capture tools do not integrate a substantial number of these different aspects. This thesis addresses this problem and presents a design tool that overcomes some of the limitations of existing tools.

An Integrated Design Information System (IDIS) is presented that integrates the: deliberation, alternatives, constraints, modifications and functionality of the components with the design artefacts. IDIS provides an integrated framework that supports the design process and records DR within a single environment. Case studies that have been used to test IDIS are presented and the findings are discussed.

1.2 Contributions

IDIS provides several advantages and novel features that are not supported by current DR capture tools. These features are described throughout the thesis. The major contributions made by IDIS are:

A novel integration model is used to integrate several different aspects of DR with the design artefacts. A viewpoint mechanism is used to record alternatives that are explored in the design space during the design process. This mechanism provides a structure that supports the integration of the: deliberation, alternatives, constraints, modifications, functionality of the components and the design artefacts themselves.

An Issue Based Information System is used to support and record the design deliberation in IDIS. As the deliberation is intended to provide an understanding of the design decisions the integrity of the information upon which the decision is based must be maintained. If information is added in an unstructured manner to an issue base that has been resolved it can result in a loss of integrity of the existing information. Changes to an issue base that causes a loss of integrity or result in an incorrect structure are
referred to as a violation of temporal integrity of the issue base. By conserving the
temporal integrity of an issue base it is possible to examine the issue base structure to
see how it had evolved. IDIS is unique in its ability to maintain the temporal integrity
of an issue base.

Often small changes between two designs cannot be easily identified and may have
significant and unforeseen consequences. IDIS provides designers with a novel feature
that can be used to easily identify the differences between two design diagrams. This is
achieved by recording the modifications that have been made to each design and a
representation of how they are linked in the design space.

Finally IDIS provides features for managing the information recorded and the design
project. Search and automatic identification of new information facilities are provided
to support the access of the information recorded in the system. Management features
allow project leaders to monitor the progress of the project against deadlines and to
view the contribution of the members of the design team. The project leader also has
control over who can access and modify the information recorded in the system. These
management features are unique to IDIS.

1.3 Structure of the Thesis

Chapter 2 describes different types of design and processes involved in design. The
range and scope of the Computer Aided Design (CAD) packages are presented and the
benefits and limitations of these are discussed.

Chapter 3 provides a detailed review of the research into DR. The different aspects of
DR are identified. The techniques and tools reviewed are presented according to the
different aspects of DR which they address.

Chapters 4, 5 and 6 relate to the development of IDIS, the Integrated Design
Information System which has been developed as part of this project. Chapter 4
discusses the design issues relating to the development of IDIS. Chapter 5 discusses 
the implementation details of the system. Chapter 6 describes how the system is used 
with the aid of one of the case studies.

Chapter 7 describes four different case studies conducted during the development of 
IDIS. Each case study tested a different aspect of the system. The lessons learnt from 
these case studies and the modifications made to IDIS as a result of the feedback are 
discussed.

Chapter 8 is a summary of the research and describes how IDIS has provided a new 
tool that helps capture DR. While the unique features of IDIS (the ability to integrate 
and manage information, identify differences between two designs and maintain the 
integrity of an issue base) enhance the capture of DR there are limitations to this 
system. These are summarised and the areas for future work are identified.
Chapter 2

Computer Aided Design

Computers are now used in almost every area of industrial design. It is hard to imagine any aspect of design being performed without the aid of some computer package. The term computer aided design (CAD) can be applied to almost any package which is used in the design process. However, this can cause confusion and in this thesis the term is used primarily for packages that have specifically been developed to aid the design process.

Standard computing packages have been used throughout the design process. For example: spreadsheets were used to perform large and complex calculations; databases were used to record information about the design and/or manufacturer information; word processing packages were used to produce the design documentation and drawing and drafting packages were used to reduce the time it took to produce design diagrams.

The role of computers in the design process has expanded rapidly. This is partly because the cost of computers has fallen and also because software technology has developed. Many of the standard packages are still being used but new tools have also been developed to provide more design specific support. Current packages provide functions such as three dimensional viewing and modelling, flow analysis, support for concurrent engineering and Computer Aided Design Manufacture Integration. The developments in artificial intelligence (AI) techniques mean that new tools are also being developed which provide a more active role in the design process.
To understand how computers can be used in the design process it is necessary to have an understanding of the processes involved in design as well as the different types of design. These topics are covered briefly in sections 2.1 and 2.2. In section 2.3 some of the existing CAD packages are presented and compared. Section 2.4 concludes by discussing the aspects of design which computers do and do not support well and identifies possible areas for future research. One such area is the capture and use of DR which will provide an insight into some of the less well understood processes of design.

2.1 The Process of Design

To manufacture any item successfully, it is necessary to have a design for that item. For a relatively simple item the designer and manufacturer may be the same person and the design may consist of only a simple sketch. However, in industry it is rare that the designer and manufacturer are the same person. When large and complex artefacts are designed (e.g. buildings, process plants, aircraft) the designers are often different from the manufactures or constructors. Indeed, they may work for separate companies. In such cases the designers must specify every detail of the design in the documentation that is produced for the manufacturers and users. Cross (1989) describes the aim of a design project as the production of a clear, concise and complete design document that describes the item and the means by which it can be manufactured.

The documentation includes: the artefact drawings, building or manufacturing specifications, commissioning and testing criteria and the operating procedures. The design documentation should also indicate the limits which apply to the design as a whole or any of its sub-components.

The task of the designer is to produce the above documentation for a given set of requirements and constraints. The means by which the designer achieves this aim can be referred to as the design process. The design process has traditionally been viewed
as a combination of problem solving and search processes in AI design research, (Brown and Chandrasekaran 1989, Mittal and Araya 1992, Smithers et al. 1990). Indeed the design process has been viewed as a search process within the design community since the 1960's, (Gregory 1966). Smithers et al. (1990) describes the design process as an exploration of the design space that is bound by constraints. In the rest of this section the design process is described in terms of this definition. The exploration model of design can be viewed as having three main activities:

- identify the objectives,
- generate and explore possible designs,
- evaluate the possible designs and choose the best solution.

The first stage of any design is to examine the requirements and identify the objectives of the design. The requirement specification is not always well defined and the designer's first task is to identify the important characteristics and any limitations of the required artefact. It is also vital to establish which aspects of the artefact are required and those which are only desirable. Good communication between members of the design team and the customer is vital during this stage. Having identified the required functions of the artefact, the designer must also identify the constraints which apply to the design. These may have a significant bearing on the types of design that are possible. This done, the designer has a good idea of the required goal and also knows the limits of the design space within which it must be achieved.

During the next stage, the designer attempts to produce a design which satisfies the requirements and constraints specified by the first stage. In most large scale designs the problem is decomposed into smaller more manageable sub-problems each of which may address a different aspect of the design. These sub-problems are then tackled separately and merged to provide an overall solution to the problem. During the resolution of these sub-problems the designer must ensure that the solutions are compatible. For each problem the designer may explore more than one possible solution. Partial solutions may be generated and each may have different modifications and suggest alternative ways of meeting the requirements. As this process continues
the designer explores different paths through the design space in search of an acceptable solution.

In the final design stage, each possible solution must be evaluated to ascertain if it meets the specified requirements. Where several solutions are acceptable the designer may use some other criteria to choose between them. These may include the cost, ease of manufacture, environmental friendliness, or the designer may choose his or her preferred method. If the design requirements and/or the constraints have not been correctly satisfied the customer may choose the most acceptable or suitable design. In some cases, the customer may reject the design leading to a redefining of the specification and the repeating of the whole process.

Once the process has been completed the designer will have at least one possible solution which meets the initial requirements and constraints. The design diagrams, manufacturing information, etc. are all included in the final design documentation. These diagrams and documents can be thought of as the product of the design process.

During the design process many different types of information are used and generated. Design information can be divided into two main categories: that which is *used* during the design process and that which is *generated* during the design process. The different sources of information used include: personal expertise, regulations and standards, company policies and information produced by other team members. These various sources are all used as input to the design process and are used by the designer to generate and evaluate design alternatives. The information generated includes: design ideas and concepts, design diagrams, sketches, minutes from meetings, manufacturing and operating instructions and other such documents.

The information used and generated can be sub-divided again into two different sets of information, namely explicit and implicit. The different types of information used and generated are shown in Table 2.1. Explicit information includes regulations and standards, company policy, design documentation and drawings. Implicit information includes the designer's expertise, the designer's intent underlying decisions made and
design ideas and concepts. When a project is completed the main tangible product of the design is the final design documentation. In many cases much of the implicit design information is lost as the implicit input may not be recorded. As a result the final documentation is not a complete record of the design. Without this complete record of the design it is not always possible to identify all the consequences of modifying the design. Within the process industry there are numerous examples of disasters that have been caused due to unforeseen consequences of modifications to existing designs (Kletz 1988, Lees 1980).

Table 2.1 Explicit and implicit information used and generated in design

<table>
<thead>
<tr>
<th>Used</th>
<th>Generated</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explicit Information</strong></td>
<td><strong>Generated</strong></td>
</tr>
<tr>
<td>Design Standards</td>
<td>Diagrams</td>
</tr>
<tr>
<td>Company Policies</td>
<td>Manufacturing instructions</td>
</tr>
<tr>
<td></td>
<td>Minutes from meetings</td>
</tr>
<tr>
<td><strong>Implicit Information</strong></td>
<td><strong>Generated</strong></td>
</tr>
<tr>
<td>Expertise</td>
<td>Design concepts and ideas</td>
</tr>
<tr>
<td>Negotiation with other designers</td>
<td>Design rationale</td>
</tr>
</tbody>
</table>

2.2 Types of Design

Design can be categorised in different ways. Process plant design is commonly divided into process and mechanical design (Coulson and Richardson 1983). Process design refers to the design of the processes used to produce the final product and mechanical design refers to the design of the physical plant that will perform these processes.

There are two basic types of design that are accepted within the design community as a whole; these are creative and routine design (Tong and Sriram 1992). Routine design is more common than creative design because many of the tasks the designer is required to perform are modifications to existing designs. Creative design is less common and is more difficult and time consuming, it is required when new processes or products are developed. Most industrial designs have financial constraints and therefore creative designs are rare. Manufacturers are much more likely to opt for a tried and tested design, or one that modifies a known operating design thus reducing risk factors such as costs.
Tools have been developed to support these different types of design. One of the difficulties with developing such tools is that many of the processes involved are still poorly understood. This is especially true for creative design. In the next section some of the design tools used in the process industry are presented. These tools provide support for both the process and mechanical design of process plants. Routine design tasks are supported by many of the tools but little if any support is provided for creative design.

2.3 Computer Aided Design Tools

CAD systems are very useful and valuable tools and are now used in many different design fields (engineering, fashion, architecture, etc.). Large software companies have developed off the peg general CAD programs that can be applied to any field, e.g. AutoCAD (Autodesk Ltd.) and INTERGRAPH (Intergraph electronics). These off the peg systems have helped to promote the rapid growth in the use of CAD in a wide range of industries. Table 2.2 shows some of the more common tools used in CAD.

<table>
<thead>
<tr>
<th>Product Type</th>
<th>Packages</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D design and draughting packages</td>
<td>ChoiceCAD, Visio</td>
<td>Used to create 2D detailed design diagrams.</td>
</tr>
<tr>
<td>3D design and draughting packages</td>
<td>Topcad, ADAMS, AutoCAD</td>
<td>Used to create 3D detailed design diagrams</td>
</tr>
<tr>
<td>Computational Fluid Dynamics</td>
<td>ADINA, Parallel-PHENICS</td>
<td>Supports computational fluid and heat transfer calculations</td>
</tr>
<tr>
<td>Piping Design</td>
<td>AutoPlant, PipeCAD</td>
<td>Provides support for piping design in petrochemical industries</td>
</tr>
<tr>
<td>Process Design and Simulation</td>
<td>ASPEN</td>
<td>Calculates detailed mass and energy balances</td>
</tr>
<tr>
<td>Concurrent Engineering</td>
<td>ADINA, Podium</td>
<td>Supports design teams working concurrently on the same project</td>
</tr>
</tbody>
</table>
In industries where design plays an important role (aeronautical, automotive, petrochemical, etc.) many of the larger companies have developed in house CAD systems to develop their own specific products. These are necessary because the designer sometimes require special facilities not available in commercial systems.

Although many different aspects of the design are supported by these tools the functions they offer are normally performed in isolation. Some packages can be combined to improve their functionality, others allow the designer to export information (e.g. 2D or 3D CAD models, documentation, component and material information) to a file that other packages can read. The industry is addressing the problem of exchanging information between systems and has introduced a standard exchange format, STEP (standard for the exchange of product model data). However, integrating the wide range of CAD tools is still a problem and many consultancy firms provide services and tools to help individual users develop tailor made integrated packages.

2.4 Strengths and Weaknesses of Computer Aided Design Tools

CAD systems provide many advantages for a busy designer. They are ideal for the large number of repetitive numerical calculations required, and often perform these tasks better than the designer. Draughting packages support the production and modifications of design diagrams. Some of these draughting packages also allow the designer to view the design in 3D providing a more realistic view of the artefact being designed. Large amounts of information are used and produced during the design process and databases and document management packages support the recording and access of this information. The use of component catalogues and costing packages supports the monitoring of the cost of the design and the individual components that make up the design artefacts. CAD CAM tools also allow the designer to manufacture prototypes of various artefacts relatively easily.
The above benefits highlight some of the areas where the designer can save time during the design process. These tools allow the designer to spend more time on the more difficult aspects of design. These include the creative processes of generating and evaluating design alternatives.

However, there are limitations with existing CAD packages. Before any process can be represented on a computer the process itself must be fully understood. Traditional CAD systems can only perform processes that are well understood and can be broken down into a symbolic or mathematical representation. With the development of knowledge based systems, object oriented languages and new representation techniques it is now possible to represent and reason with more complex information. However, to represent such information the designer still has to fully understand the processes involved, what information and knowledge is used as well as how it is used. Unfortunately such an understanding is not available for many of the processes involved.

"Design is a highly creative activity involving diverse problem solving techniques, and many kinds of knowledge. Clearly, as we don’t know many of the problem solving components of general design, and as we poorly understand those components that we know about, a comprehensive, detailed model of design is currently out of reach."

(Brown and Chandrasekaran, 1992: p244)

As Brown and Chandrasekaran state, design is by nature a creative process. During the design process the designer uses general design knowledge, domain specific knowledge and personal and professional experience. Although computers can emulate some of the routine processes which are well understood, they cannot provide any creative help. Also, they cannot provide any help to such areas as aesthetic judgement which is also regarded as one of the most important aspects of the design process. Computers cannot provide any support in these areas but the support they provide in the other areas allows the designer to spend more time on these more difficult tasks.
New computer technology and software engineering techniques allow designers to use computers in many more complex ways than was possible in the past. Current research in the field of design is exploring many different aspects of the design process. Some of this research includes: the use of Case-Base Reasoning in design (Maher and Balachandran 1994, Raphael and Kumar 1996), Qualitative Modelling (Chung 1993, Vianna and McGreavy 1995) and the use of Neural Networks in design (Cauvin 1995).

The main problem in developing tools that support more of the design processes is the lack of understanding of some of the processes involved. One of the reasons why these processes are not understood is because much of the reasoning behind the decisions made is rarely explicitly recorded. This reasoning is commonly referred to as design rationale (DR). An explicit representation of DR will provide an insight into some of the reasoning that occurs during the design process and hence lead to a better understanding of the design process itself. It can also be used directly in fields such as Case-Base Reasoning (Kolodner 1993). The record of the DR may be used to help create new example cases with a deeper understanding of the design. This will help to improve the retrieval of past cases and may also assist in the adaptation of cases. This thesis addresses the problem of explicitly recording DR.

2.5 Summary

In this chapter the tasks involved in the design process and different types of design were briefly discussed. Some of the existing commercial CAD tools were presented and the types of tasks they perform were described. These tools are often available in independent packages and address one particular aspect of design. The integration of these tools is a problem that remains to be resolved. Existing CAD tools are passive and primarily support the routine design tasks, storage and access of information. A better understanding of the processes involved may enable researchers in the future to develop CAD tools that support some of the more difficult aspects of the design process. A better understanding of the processes involved may be improved by explicitly recording DR.
Chapter 3

Design Rationale

Design rationale (DR) can be defined as the fundamental reason for the course of actions taken during the design of an artefact. It is a by-product of the design process and is distinct from the final design documents which represent the product of design. For many design projects, the DR is rarely recorded completely and explicitly. If it is, it is often difficult to retrieve or reuse. DR capture and reuse is a popular current research topic (Morran and Carroll 1996, Lee 1992) because many benefits can be obtained from explicitly recording DR. Some of the benefits are:

- justify design decisions,
- provide information for the final design documents,
- prevent accidents occurring by making design assumptions explicit,
- assist in subsequent modifications to existing designs,
- prevent designers 'reinventing the wheel'.

DR is a representation of the designer’s understanding of the design as it evolves and is not represented by any single aspect of the design. A designer will have an understanding or a belief about different aspects of the design and each of these may be expressed in different forms. As a result DR needs to be constructed or captured from different sources of design information. Aspects of design that can be used to represent DR include:

- design requirements,
- design deliberation,
- design constraints,
- the design space explored,
- the functionality,
- the relationships and interaction between design objects.
The requirements of a design provide a record of what goals the designer is trying to achieve. By understanding the goal of the designer one will have a better understanding of the actions that are performed in order to meet that goal. A record of the design space provides an insight into what areas and issues have been explored and in conjunction with the chronological history of the design provides a record of how the designer arrived at that point. A record of the functionality of the objects provides an insight into the intended function of a design component. Closely related to this is a record of the relationships between the different design components which provides an understanding of how the components interact.

This chapter presents an overview of the research to date as well as the techniques and tools that have been developed to record DR. These are discussed in the context of the different aspects listed above.

3.1 Overview of Design Rationale Research

Some of the research that has been carried out over the last ten years is presented in Figure 3.1. The figure shows that much of the early work in the area of capturing DR concentrated on recording the design deliberation and explicitly recording design constraints.

In the late 1970s and early 1980s knowledge based systems were developed in many different domains to solve a variety of different problems. Within the design research community, knowledge based systems were developed to record design constraints and check designs for violations of these recorded constraints. Design deliberation tools were also developed to support collaborative computer working and to record and provide access to useful information. The design deliberation is a rich source of DR as it is the means by which designers discuss and explain their understanding of the design with colleagues. Shum and Hammond (1994) provide a comprehensive survey of the work conducted in this area. They conclude that the deliberation is a good source of DR but also identify the need to provide closer integration of DR with
the design artefacts and other aspects of the design process. The deliberation represents only one of the mediums through which designers express their understanding of the design and, therefore, it cannot provide a complete representation of DR on its own.

Design deliberation is still seen as one of the most important aspects of DR and is recorded in many of the current systems. Research since the late 1980's and 1990's has addressed other areas such as the design space, functionality, requirements and the relationships between the design components. Some systems developed have addressed more than one aspect of DR. Many of these aspects overlap and are captured via different sources of information including: minutes from meetings, design documentation, notes, sketches, e-mail messages, design diagrams. Some of the systems that have been developed to cope with this wealth of information are discussed below.
3.2 Design Deliberation

Deliberation occurs in many stages of the design process when members of the team discuss the layout, specification, constraints, components parameters, etc. Any representation used to capture such information should support the deliberation process and provide a clear structure of the discussion that has taken place. There are several techniques that can be used to represent design deliberation: Toulmin Form, Rhetorical Structure Theory, semi-structured message templates and Issue Based Information Systems. These techniques are discussed in this section.

3.2.1 Toulmin Form

Toulmin Form (Toulmin 1958) was one of the earliest techniques that was developed to represent the structure of an argument. This semi-formal representation is based on the assumption that every component of an argument can be represented using a simple syntax. An argument is split into sub components called micro arguments. The basic components of micro arguments are: a claim, data that supports the claim and a warrant that justifies the use of the data to support the claim. The complete micro argument has six components with a backing for the warrant, and qualifier and rebuttal nodes that indicate exceptions to the arguments. The structure of a Toulmin Form micro argument is shown in Figure 3.2.

![Figure 3.2. The Toulmin Form Micro Argument](image)

The Toulmin representation has been used in several domains including: policy making (Storrs 1991, Ball 1994) and safety argumentation (Forder et al. 1993). Forder et al. (1993) extended this representation and used it to explicitly record the structure of safety arguments for the design of safety critical systems. The Extended Toulmin
Form (ETF) includes an argument node as a sub-component. In ETF the rebuttal node is attached to this central argument node rather than the qualifier to make it clear that it is the validity of the micro arguments that is being questioned and not the claim itself. This new structure is shown in Figure 3.3. Their system using ETF makes the argumentation embodied in text explicit, thereby capturing the rationale and making it easier to see if an argument relating to the safety of a system is correct. Although Toulmin Form was one of the earliest methods of representing the structure of an argument it has not been a popular method used to capture design deliberation. The fundamental problem with this representation is that it has its origins in text analysis rather than open discussion. As a result it provides poor support for the process of deliberation and argumentation.

![Figure 3.3. The Extended Toulmin Form model](image)

3.2.2 Rhetorical Structure Theory

Rhetorical Structure Theory (RST) (Mann and Thompson 1987) is another method developed to explicitly describe the relationships between different components of English text. This theory views text as sets of hierarchically organised clauses and groups of clauses that are related to one another in various ways. The underlying arguments in a piece of text are captured by representing the components of the text as clauses linked by functional relationships. An example presented in Mann and Thompson (1987) demonstrates how an invitation can be represented as three clauses:
As members of the University staff you are cordially invited to attend the 1983 Annual Staff Breakfast presented by President James Zumberge and Staff Assembly.

This is an opportunity to meet some of the other staff members affiliated with the University, as well as the Staff Assembly representatives and President Zumberge.

The continental breakfast and get-together will be held in the Town and Gown Auditorium (on Main Campus) at 8:30 AM on Thursday 11/3.

Each clause of the invitation has a definite role: clause 1 is the invitation (action), clause 2 is the reason for the action (motivation) and clause 3 is the means by which the action can be performed (enablement). This is shown diagrammatically in Figure 3.4.

![Diagram of Rhetorical Structure Theory](image)

**Figure 3.4. The structure of an argument in Rhetorical Structure Theory**

Figure 3.4 shows how the invitation is split into three clauses labelled 1 to 3, the relationships between these are represented by the links to the action (1) from the motivation (2) and enablement (3). Apart from the motivation and enablement links the model includes: solutionhood, elaboration, background, purpose and concession links. The components in a piece of text can be represented in a semi-formal manner by splitting the important aspects of the text into clauses and linking them with the appropriate relationship link. This method has been employed by Ndumu et al. (1996) to record the structure of design plans and by Fawcett and Davies (1992) to support planning discourse. However, it has not been used to record deliberation during the design process. To record the deliberation that takes place during the design process a
representation must be able to support the deliberation process and capture the information as it is produced. RST was developed to represent only the structure of arguments in existing text documents and does not provide this support.

3.2.3 Semi-Structured Message Templates

Semi-structured message template representations have been proposed by Glicksman et al. (1992) and Malone et al. (1987). Semi-structured message templates were developed by Malone et al. (1987) to support computer based communication and are defined as:

"... messages as identifiable types, with each type containing a known set of fields, but with some of the fields containing unstructured text or other information."

(Malone et al. 1987: p116)

An example of a template for a seminar announcement would include fields for time, place, speaker and topic. The free text relating to the different aspects of the announcement are added to the appropriate slots in the template. Malone et al. (1987) argue that by recording messages using structured templates computers will be able to automatically process the information more easily than if it was recorded in an unstructured manner. They claim that as people already use similar structures to store and process information, their representation does not add an unnecessary burden to the users.

The templates can be arranged in a hierarchical structure to group particular types of messages. The general message types are stored at the higher levels of the network than the more specific message types. However, unlike RST and Toulmin Form, Semi-Structured Templates do not provide a means of explicitly representing the relationships between the different components. Each template is also specific to different tasks so this method is not suitable for domains that deal with general problems. While this method has been used by Malone et al. (1987) to support computer based communication, it has not been used to capture design deliberation.
3.2.4 Issue Based Information Systems

Issue Based Information Systems (IBIS) is the only technique developed to record and support discussion which is an inherent part of any deliberation process. It was developed by Kunz and Rittle (1970) to support deliberation amongst government administration and planning groups. The IBIS representation, and extensions of it, have been the most popular methods used to capture design deliberation and are still used today.

The IBIS representation, like the TF and RST, is basically a node and link type representation. The deliberation is represented as a network of nodes with each node representing a different aspect of the deliberation. The basic nodes in the IBIS representation are: issue, position, argument, and decision. The network is hierarchical in nature with an issue node being represented as the root. An issue is any question or problem that requires a decision. The relationships between these nodes are represented by the type of link between them. The links and nodes represent the structure of the deliberation and the content of the nodes represent what was discussed.

An issue base is always started by raising an issue. Once this has been done, the team members add possible solutions to the problem as position nodes. The positions are linked to the issue with a response link to indicate that they have been added in response to that issue. Argument nodes can also be added that contain arguments that either support or refute the positions. These are linked to the positions with the appropriate links, either support or against (Figure 3.5). When an issue is resolved a solution is recorded in a decision node which is linked to the issue to indicate it has been resolved.

Unlike the approaches discussed above, IBIS supports the deliberation process by recording it as it evolves. It also explicitly represents both the type of node and the links between them. These aspects improve the accessibility of information as well as making the structure of the argumentation within the deliberation clearer.
Figure 3.5. The structure of the Issue Based Information System representation

The IBIS representation has been widely used in many different areas including: software design (Potts and Bruns 1988, Lubers 1991, Ramesh and Sengupta 1995); policy discussion (Conklin and Begeman 1988); architectural design (Fischer et al. 1991, McCall et al. 1990); discussing and accessing building standard regulations (Casson and Stone 1992) and process plant design (Goodwin and Chung 1994, King et al. 1995). Although widely used, limitations of this representation have been identified and it has been developed and extended. Some of these modifications are discussed in the section below.

McCall (1979) had two major criticisms of IBIS:

issues that did not require alternatives to be explored, and hence no deliberation, could not be supported in IBIS;

the representation as it stood could not support the hierarchical nature of deliberation where one issue often relates to another.
The first criticism is not valid as any question and decision can be recorded using IBIS even if it only has one position and no arguments. It is still useful to record the fact that the issue arose and a decision was made, even if alternatives were not explored or did not exist. The second criticism is more of a problem as the solution of one problem often depends on the solution of another. This situation occurs when problems are decomposed into smaller related sub-problems.

McCall extended IBIS to a representation called Procedural Hierarchy of Issues (PHI), which he claimed differs from IBIS in two crucial ways:

- it uses a broader definition of the concept issue;
- it includes links to represent the dependencies between issues.

McCall claims that:

"in IBIS, the term issues denotes a design question that is deliberated; in PHI however, every design question counts as an issue whether deliberated or not."

(McCall 1991: p398)

As mentioned above the narrow view of the term issue is not necessarily correct as there is no reason why IBIS cannot represent design questions that are not deliberated. This first change to the IBIS representation is not in fact a change but simply an interpretation of how the term issue is used.

The second change to IBIS involves the addition of a new link. McCall claims that the similar to and replaces inter-issue links proposed by Rittle do not support a hierarchical structure of issues. A new link sub-issue of was added and is used to indicate that one issue is a sub-issue of another. This link is used if, and only if, the resolution of one issue influences the resolution of another. In the example cited in Fischer et al. (1991) this link is only used to represent the decomposition of one issue into smaller sub-issues. When one problem is decomposed into several smaller sub-problems the solution to the original problem depends on the solutions of the sub-problems. The replaces link could be used to describe the relationship between the
problem and sub-problems. However the replaces link is also used when an issue is thought to be incorrect or no longer relevant and is replaced. In the second scenario the solution of the second issue would not have a bearing on the solution of the initial issue. The sub-issue of link provides a means of making a distinction between these different situations.

The PHI representation has been used in the design systems, MIKROPLIS (McCall 1979), JANUS-ARGUMENTATION (Fischer et al. 1989) and PHIDAS (McCall et al. 1990). These systems were developed to support and capture design deliberation in an attempt to record DR. The JANUS-ARGUMENTATION was developed as a deliberation component for the JANUS system which was used in architectural design. The full system was tested in the kitchen design domain and produced mixed results. Some of the users commented that; it was difficult to access relevant information; the design environment changed during the design and that designers sometimes used knowledge or expertise that was not articulated or recorded in the system.

This work has influenced other research which has used the IBIS representation. Conklin and Begeman (1989) developed a graphical IBIS system (gIBIS) in order to help with the problem of accessing and navigating through a large body of information in an issue base. This project had three main aims:

- to support computer-mediated teamwork,
- capture design rationale,
- investigate the navigation of large hypertext information spaces.

gIBIS used the IBIS as the basic underlying representation with a few modifications. Issues could be linked to other issues with a specialised or generalised link to indicate the dependencies between issues (one of the criticisms made by McCall). Conklin and Begeman (1989) added another node, other, which could be used to represent information not supported by existing nodes and links. The most important features of gIBIS were the graphical browser and the search facility. The graphical browser provided the users with an easy to use graphical representation of the issue base which
could be viewed and modified using a mouse. The search and query facility allowed users to search the issue base for relevant information. Their findings showed that a relatively simple query tool was sufficient for searches of issue bases of moderate size and they did not extend the search mechanism to allow more complex Boolean or context searches.

Potts and Bruns (1988) also used IBIS as the underlying representation for a generic model for representing design deliberation. They present a system that represents design history as a network of artefacts and related deliberation nodes. Previous research discussed above, had not attempted to link the IBIS representation with the design artefacts.

Potts and Bruns make the distinction between two different types of design, the *results* of the design process and the *process* itself. They claim the design artefact only records the *result* of a design, not the *process*, and that both of these aspects of the design should be recorded. In their model the process (design deliberation) is represented as an issue base that is attached to the artefacts. Thus the rationale is recorded independently from the artefact but the relationship between them is explicitly recorded. The artefact nodes are attached to the alternative (position) nodes that are linked to an issue. Thus the issue from which the alternative (artefact) arose and the argument relating to that alternative are linked to the artefact (Figure 3.6).

Although this model provides a mechanism for explicitly linking the deliberation with the design artefacts it does not record issues that are not directly related to a design artefact. In many cases during a design, especially in the early conceptual stages, design issues are discussed that do not necessarily result in the production of a specific design artefact.

Lee (1991), while agreeing with Potts and Bruns that it is important to link the deliberation with the design artefacts, identifies several limitations with their model. The Potts and Bruns model has issues, alternatives and justifications but no distinction is made between supporting or opposing justifications. Lee states that a model should
be able to represent opposing as well as supporting arguments and that it should make both the goals and issues explicit. She also argues that any model of design rationale should not affect the design process itself and add to the workload of the designer. Lee developed the Decision Representation Language (DRL) to address these concerns.

![Diagram of DRL](image)

**Figure 3.6.** Potts and Bruns representation of design deliberation and design artefacts

Lee makes a distinction between DR and decision rationale. She claims that DR is a superset of decision rationale as it represents the artefacts or design space as well as the deliberation that takes place about those artefacts.

DRL also uses IBIS as the basic underlying representation and the issues and alternatives in DRL are the same as those in the Potts and Bruns model. Fourteen different types of relationships are presented including: derives, achieves, supports, denies, raises. As well as these different relationships DRL also has links for questions, procedure and viewpoint.
Using this representation the designers can express ideas, as well as the relationships between the different types of information, in a more precise manner than they would have been able to with the Potts and Bruns model. However, it can be argued that the increase in the number of link types increases the complexity of the model and makes it more difficult to use. While a representation can be made richer by adding new link types, every type that is added increases the complexity.

The SIBYL design tool implements the DRL model (Lee 1991). SIBYL has two basic parts: a user interface which allows the user to record information; and a component that allows the user to examine and query the information. In SIBYL a user first creates a decision problem which is analogous to an issue. Goals and alternatives that relate to the decision problem are then added to the system to record the criteria for the solutions and possible solutions. The alternatives represent the positions in the IBIS representation and the nodes that support or achieve goals are similar to the arguments. The goals and alternatives for a decision problem are recorded in a matrix referred to as a decision matrix. The value of the cells in the matrix represents the evaluation of the alternative with respect to the goal. The user can browse, search, or modify the matrix cells. Supporting, denying or qualifying claims that are added to the nodes result in an automatic update of the corresponding matrix cell. This matrix allows users to clearly identify to what extent the alternatives satisfy the goals. Table 3.1 shows a decision matrix for the decision problem, ‘which system is responsible for the window sub-structures?’ with two alternatives: window manager and applications, and two goals: efficient window management and provides a uniform interface.

<table>
<thead>
<tr>
<th>Alternatives</th>
<th>Goals</th>
<th>Efficient window management</th>
<th>Uniform interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Window manager</td>
<td>Low</td>
<td>High</td>
<td></td>
</tr>
<tr>
<td>Applications</td>
<td>Medium</td>
<td>Medium</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1 An example of Lees’ decision matrix
This section discussed some of the design deliberation representations that have been developed to capture DR. The premise for these approaches is that DR is expressed as dialogue between design team members as they highlight problems, identify and evaluate solutions. The following sections discuss some of the tools developed to capture other aspects of design that are components of DR.

### 3.3 Design Constraints

Constraints play an important role in design. Smithers et al. (1990) described the design process as exploration of the design space that is bound by constraints. The basic task of the designer is to create a design that meets the required functionality and specified constraints. The following are examples of different types of constraint:

- **C1** - the total cost of the design should not exceed 10,000 pounds.
- **C2** - the cost of vessel 1 should not exceed 1000 pounds.
- **C3** - the cost of any vessel should not exceed 1000 pounds.
- **C4** - all pressure vessels must be fitted with a relief valve.
- **C5** - vessels designed to contain water should not be constructed from steel.
- **C6** - oxygen and gas should not be allowed to mix.
- **C7** - the plant should not produce high levels of waste products.

There are many different schemes for classifying constraints, although the differences between them are not always clear cut. Two of the most common classifications used are: local vs. global constraints (Miki 1995, Papalambros 1993) and hard vs. soft constraints (Guan and Freidrich 1992, Konukman et al. 1995).

Local constraints apply to specific artefacts or components in the design. For example, C2 relates to the cost of one artefact. Global constraints apply to the design as a whole. For example, C1 relates to the total cost. However, the same cost constraint can be imposed on vessel 1 by both a local (C2) and a global constraint (C3).

The terms ‘hard’ and ‘soft’ constraints refer to the rigidity of the constraints. Hard constraints are those which are outside the designers’ control and cannot be modified or deleted. Examples of these are most safety constraints (C4). Soft constraints are set...
by the designer and can be removed or modified. C2 is an example of a soft constraint.

Explicitly recording constraints provides several benefits for current and future designers. For example, if future designers are aware of the constraints that have been applied, either by the designer or by regulations and standards, they are less likely to violate them when redesigning an artefact. Recorded constraints can be applied to completed or partially completed designs to check for the violations. Also, other members of the design team can see which constraints have been applied to the design. Finally, making the constraints explicit can help to bridge the gap between novice and expert designers as previously inaccessible information becomes available to the design team.

Many different techniques have been used to record and apply design constraints. These include: mathematical equations, graphs and trees, frames and rules. In the rest of this section these techniques are discussed and some tools that have been developed using these different techniques are presented.

### 3.3.1 Mathematical Representations

Mathematical representations are good for representing geometric and numerical constraints such as C1 to C3 listed above. The language of mathematics provides a formal language and structure that is complete, consistent and flexible and can be processed easily and quickly by computers. Mathematical representations of constraints have been used in several different domains including: logic circuit design (Fujita et al. 1994); engineering design (Lu and Thompson 1988, Salustri and Venter 1992, Taleb-Bendiab and Oh 1993, Debenham 1995).

Debenham (1995) presents a mathematical representation of design constraints. An example of his system using constraint C2 is given in Table 3.2.
Table 3.2  Constraint C2 represented using Debenham’s notation

<table>
<thead>
<tr>
<th>Cost of Vessell</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Vessell</td>
<td></td>
</tr>
<tr>
<td>X : Vessell</td>
<td>Y : Cost</td>
</tr>
<tr>
<td>X is Vessell and Y is Cost in Pounds</td>
<td></td>
</tr>
<tr>
<td>0 &lt; X &lt; 1000</td>
<td>0 &lt; Y</td>
</tr>
<tr>
<td>X = 155 -&gt; Y &lt;= 1000</td>
<td></td>
</tr>
</tbody>
</table>

Row A contains the name of the constraint (cost of vessel1); row B contains the name of the artefact and the attribute (vessel1 and cost); row C contains variable names (x and y); row D represents the relationship between the variables, artefact and attribute (vessel1 and the cost in pounds); row E contains information about the general constraints (x - part number - is greater than 0 and less than 1000 and y - cost - is greater than 0); row F contains the specific constraint (part number (155) implies that cost is less than 1000).

ENVIED (Environment of Integrated Engineering Design) was developed by Lu and Thompson (1988) and is an example of a system that uses mathematical representations. It provides a distributed framework for supporting engineering design. The basic representation of the system is frame based. The artefact, design plans, product information, manufacturing information and constraints are all recorded in the frames slots. Although the basic representation uses frames, the constraints themselves are recorded as lambda expressions in slots in the frames. Lambda expressions are based on a branch of mathematical logic called lambda calculus. The lambda expression contains the attributes and the constraint values that apply to that attribute for the specific design objects.
While mathematical representations provide a consistent, computationally simple and easy method of representing design constraints, many constraints are not mathematical in nature. As a result, a mathematical representation is not always appropriate. Constraints C4 to C7 are examples of some constraints that cannot be represented using a mathematical representation.

3.3.2 Graphical Representations

Graphs and tree structures have been used to record design constraints by several researchers (Serrano and Gossard 1987, Schwarz et al. 1994, Fu and Depennington 1994, Kawashima et al. 1993). Like mathematical representations they can represent numerical constraints. However, graphical representation can also represent the relationships between different design artefacts or attributes in the constraints. For example, in constraint graphs, the nodes in the graph represent some parameter, attribute or artefact and the arc between the nodes represents a constraint relationship. In directed graph representations the designer can also represent the direction of the constraint. For example, constraint C4 states, all pressure vessels must contain a relief valve, it does not necessarily follow, that all relief valves need to be fitted to a pressure vessel. Directed graphs have the ability to record such information.

Serrano and Gossard (1987) use a constraint graph (Friedman and Leondes 1969) to represent constraints in conceptual mechanical engineering design. They claim there are three basic principles that are important to any constraint management system: the ability to evaluate the design against the constraints, have minimum computational complexity and maintain consistency. Graph theory provides them with a representation which allows them to satisfy all three of these principles. However, a graph can become large and complicated, as can be seen in the following diagram (Figure 3.7) taken from Serrano and Gossard (1987), and the constraints are not readily apparent.

In Figure 3.7, the constraints f1 to f3 are represented by the arcs between the nodes, X, Y, C, G and Z. The mathematical representation of the constraints are:
\[ x^2 + y^2 - c^2 = 0 \] (f1)
\[ 3x + y < 0 \] (f2)
\[ g = F(x \, y \, z) \] (f3)

Figure 3.7. Constraint graph representation of constraints f1 to f3 above

The graph provides information about how the constraints relate to the different aspects, X, Y, C, G and Z. However, the constraints themselves cannot be understood from studying the graph structure independently.

Kawashima et al. (1993) used a tree structure to represent geometric design constraints. They propose a framework called the Constructive Constrained Framework, which is based on a tree structure called the Constraint Tree or C Tree. Each node in the tree represents a geometric entity such as surface, axes, directions etc. The nodes are connected by directed arcs which represent constraints. This is very similar to the graph theory representation and suffers from similar problems.

3.3.3 Frame Representations
Researchers have also used frame representations (Minsky 1981) to record constraints, (Klein 1993, Sriram et al. 1991). Within the frame representation constraints are represented by slots and the values recorded in the slots. Frames can be used to record numerical, spatial and temporal constraints. Figure 3.8 shows how the constraints relating to the temperature, pressure and spatial relationship of a pump can be
recorded. In practice the frame would also include other aspects of the design artefact relating to its sub components and relationships with other design artefacts.

(Pump
  max_temp(70),
  min_temp(-20),
  max_temp > min_temp,
  operating_temp(50),
  operating_temp < max_temp and > min_temp,
  max_press(40),
  operating_press(30),
  operating_press < max_press,
  has_inlet(pipe1),
  has_outlet(pipe2),
)

Figure 3.8. A frame representation of constraints applied to a pump

Klein (1993) uses a frame representation in his Design Rationale Capture System (DRCS). In this system, an artefact is represented by a module (frame) which in turn has sub-modules and attributes. The attributes have associated values and or constraints. He illustrates this with an example of a module aeroplane. This has several attributes: cost, paint, material, etc. and sub-modules body, wing etc. which can be further decomposed (see Figure 3.9). This example demonstrates how constraints can be recorded using a frame representation.

Aeroplane

<table>
<thead>
<tr>
<th>has-attribute</th>
<th>Cost</th>
<th>has-specification</th>
<th>(&lt;1,000,000 pounds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>has-attribute</td>
<td>Paint</td>
<td>has-specification</td>
<td>(or blue, red)</td>
</tr>
<tr>
<td>has-attribute</td>
<td>Turning radius</td>
<td>has-specification</td>
<td>(&lt;180 feet)</td>
</tr>
<tr>
<td>has-attribute</td>
<td>Material</td>
<td>has-value</td>
<td>(or aluminum, graphite)</td>
</tr>
</tbody>
</table>

Figure 3.9. DRCS Representation of an Aeroplane
Another system that uses frames to represent constraints is DICE (Distributed and Integrated environment for Computer-aided Engineering) (Sriram et al. 1991). The design artefacts are represented as frames which record constraints and the sub-modules in the appropriate slots. The constraints are represented as functions that are called when the constraint in the slot is checked. These functions can in turn refer to other frames and the slots within them.

Although frames have many benefits they too have limitations. They are not particularly good at representing situations where alternatives are available. For example:

all pressure vessels must have a relief valve or be connected without restriction to an item that contains a relief valve.

3.3.4 Rule Representations

Rules and logical representations provide a means of representing symbolic constraints. Production rules have been widely used in many expert systems and other knowledge based systems and hence are an obvious choice for representing design constraints. Rules have the structure: IF a set of conditions are true THEN perform a set of actions. Constraint C5 can be represented as a rule in the following way:

If Item is a vessel and the contents of the Item is water and the material of construction is steel THEN warn the designer vessels that contain water should not be constructed from steel.

Rules and logic can also represent numerical, spatial and geometric information and can represent hierarchical constraints. They also have the ability to deal with alternatives as they can have the structure If X or Y then Action. Only one alternative needs to be satisfied before the action is performed.

Waters and Ponton (1992) describe an AI system which uses rules to represent constraints. Their system uses a database for the management of constraints within a design project. They use Knowledge Craft, which is a commercial AI toolkit, to
record, manage and apply design constraints. The constraints are recorded as rules and the attributes that apply to the different design objects are recorded and checked in the condition part of the rule.

DESIGN-KIT (Stephanopoulos et al. 1987) was developed as a design aid for process engineering tasks such as the synthesis of process flowsheets, configuration of control loops and the planning and scheduling of plant operations. This system also incorporates a commercial package (IntelliCorps KEE) to represent constraints. The constraints in this system are recorded in the similar way as those recorded by Waters and Ponton.

In summary, an advantage of graphical over mathematical representations is that they provide a richer representation while at the same time maintaining computational simplicity. However, with constraint graphs it is often difficult to identify the constraint that is being represented. As mentioned above, the frame representations cannot easily support constraints that contain alternatives. They also have difficulty in representing the more symbolic and abstract constraints such as $C_S$ to $C_7$. 

Production rules are a popular representation as they can represent all the different types of constraints discussed in the preceding sections. There are many commercial packages available to represent and manage rule bases and the rules can also be used to provide an explanation to the user. The structure of the rules makes the constraints clearer, making them easier for designers to understand than some of the other representations presented in this section. Although rule representations are the most powerful representations discussed in this section, there remain some constraints, such as $C_7$, that are difficult to represent using rules. These types of constraints are more general and abstract than the other constraints presented.
3.4 Design Space

Designs are seldom the result of a purely linear process. The process starts with the initial requirements which raises a series of problems or issues to be resolved. Solutions are proposed, evaluated and then one or more are selected. These solutions are then further improved or refined to arrive at a final (or partial) solution for the design. The new designs give rise to new issues that in turn need to be resolved. This iterative exploration model of the design process has been adopted by many AI researchers in the field of design, Brown and Chandrasekaran (1989), Mittal and Araya (1992), Smithers et al. (1991) and Chung et al. (1993). Four main activities that are performed during the design process were described by Chung et al. (1993):

1. Exploration - generates several alternatives of the design.
2. Decomposition - breaks down a design into sub-parts.
3. Refinement - modifies and improves the design.
4. Integration - merges the decomposed designs.

The design alternatives are explored within a design space defined by the constraints. The information relating to alternative designs is important if the design is to be modified and/or re-used. Linking the dependencies between the alternatives also provides a record of how each alternative evolved. Banares-Alcantara (1991) discusses three different representations that can be used to show how a design evolves:

- linear representation (Figure 3.10),
- tree representation (Figure 3.11),
- network representation (Figure 3.12).

![Figure 3.10 Linear representation](image)

The linear representation is the most simple as it allows the designer to record only a distilled version of the design space (the path from the initial starting point to the final design). It is of limited use when trying to understand the complete design process. This is because it does not record alternative design solutions that were discussed and
subsequently discarded. This additional information could be invaluable when trying to modify the design.

A tree representation (Figure 3.11) is a more accurate reflection of the design process. The alternatives that have been explored and rejected are recorded as well as those accepted. However, while resolving a design issue a problem may be decomposed into smaller sub-problems that are solved independently. The resulting solutions are then integrated to achieve an overall solution. Decompositions and alternatives represent different types of information. Alternatives represent several different solutions to the same problem. Decompositions represent several different components (sub-problems) of the same problem. The links in the tree representation denote alternatives that have been explored and they do not support decomposition and integration.

Figure 3.11 Tree representation

Figure 3.12 Network representation
The network representation (Figure 3.12) includes links for both alternatives and decompositions. In a network structure, nodes that have evolved from the same parent, can be combined at a later stage allowing decomposition and integration to be represented. This feature is shown in nodes A3 to A6 in Figure 3.12 above.

All of the above representations explicitly link the artefacts within the design space thus providing a chronological link between the specification and final design. The main difference between them is in their ability to represent design alternatives and support decomposition and integration. One final aspect of the design space that has not been discussed above is the ability to propagate changes through the design alternatives. All of the representations provide explicit links between the alternatives and as a result can support the inheritance of changes. However this is often not supported in many of the systems based on these representations.

Linear representations have not been used to represent the design space because they cannot support many of the features required. In the rest of this section some of the tools that have been developed using trees and network representations are discussed.

3.4.1 Tree Representations
MacLean et al. (1991) developed the Questions Options and Criteria (QOC) representation which resembles the IBIS structure in that Questions represent issues, Options positions and Criteria the arguments and facts. MacLean et al. claim that the two systems are significantly different. First, in IBIS the issues refer to any question raised about the design; in QOC the Questions specifically relate to the design artefacts. Second, in IBIS the positions do not necessarily always represent an alternative design artefact, whereas in QOC, the Options always represent design alternatives. Within both systems the arguments and Criteria are used in similar ways.

The Options in QOC represent design alternatives within the design space. Questions representing design issues are raised and linked to the Options. The possible solutions for these Questions are linked to the Question as Options. This Option - Question - Option structure provides a link between the different alternatives within the design
space. As this structure evolves the alternatives (Options) in the design space are recorded along with the Questions that gave rise to those alternatives.

The work by Potts and Bruns (1988) and Lee (1991) also use an IBIS type representation to link the design artefacts within the design space. Their aim was to explicitly link the design deliberation with the artefacts. The models presented also support a structured recording of the design space. These models were discussed in section 3.2.4.

All of the IBIS type models suffer from the problem that the artefacts in the design space are intricately linked to the deliberation. Integrating the different aspects of DR provides a more detailed representation of the DR. However, it also restricts the recording of the design space to alternatives that are discussed during the resolution of design issues. While these models have the ability to record the path from the initial specification to the final design and the alternatives explored they do not support inheritance of changes or decomposition and integration.

Chung et al. (1993) implemented a viewpoint mechanism for recording the design space. Each point in the design space is represented by a node in a tree and is referred to as a viewpoint. The design starts with a single root node that is expanded as the design evolves. Children nodes are added to parent nodes as different alternatives are generated and explored. Each child node inherits all of the changes that have been made to the design up to that point. Through this process a hierarchical structure of the design space is built up as the design develops. The designer moves through the design space by selecting the appropriate viewpoint node. The origins of any design can be traced by following the path taken from the root node to that design.

Kusiak and Szczerbicki (1992) identify three different types of design space: requiremental, functional and physical space. They present a system that consists of a tree structure constructed from a series of and/or clause structures. This model supports the explicit recording and linking of the requiremental and functional design spaces during the early conceptual stages of design. In the first stages of the design the
requirements are addressed. The initial requirement is recursively decomposed into sub-requirements until the leaf nodes of the tree represent a single requirement. Each of the leaves in the requirement tree are then addressed and the function(s) needed to meet them are explored. As with the requirements these functions recursively decomposed into smaller sub-functions. This structure allows the designer to decompose the problem into smaller more manageable problems and provides a clear representation of the requirement each function is trying to satisfy. Alternative designs are represented by splitting the function with an or link and decomposition is represented by splitting the function using an and link. This is shown in Figure 3.13.

![Diagram of requirement and functional space representation](image)

Figure 3.13. Requiremental and functional space representation

The Kusiak and Szczerbicki representation has a number of limitations. First, only the requirements that have corresponding functions are represented. However, design requirements are not always represented by an aspect of the design functionality. An example of this could be the cost of a design or the location of a building/plant etc.
Second, although the representation allows the designer to decompose the problem into smaller sub-problems, it does not support true decomposition and integration as there is no means of integrating these solutions. Third, this representation does not support inheritance as any change to the requirements or functions is not propagated to the lower levels. Finally, the design artefacts, (physical space), are not explicitly linked to the design functions and requirements.

Gay et al. (1993) present a design environment that uses a series of databases and a version management system to record and maintain the design alternatives. Their aim is to provide a flexible yet consistent design environment that supports concurrent design teams. Within their system there are four different classes of databases: public; stable; working and private. Designs are created by the designers in their own private database and added to the working database when they are considered to be at a ‘suitable’ standard. Within the working database other designers can develop them further. When the design is considered to be ‘stable’ it is added to the stable database. From this point on, only the creator of the design can modify it; the other designers may only reference it. The public database is the final one and contains the completed design and or parts of design that will not be developed further.

When alternatives are created they are stamped with the parent details. Each alternative has a unique identification code which is used to ensure that only one version can be active at any one time within the working, stable and public databases. Alternatives can be added to the working database by any designer. Within this system, it is possible for several designers to explore different alternatives from the same design. Alternatives and other designs created by one designer can be referenced by other designers in the working and stable databases. The unique identifiers also allow any member of the design team to trace the evolution of a design. As the new versions are created from the parent node they inherit all of the changes that have been made to that design up to that point. By doing this, the representation supports the inheritance of the design changes.
All of the systems discussed above support the recording of design alternatives and a chronological record of how they evolved. The viewpoint mechanism and data base version system presented above both support the inheritance of changes made to the design. The QOC and Functional and requirement trees do not support this feature. None of the above systems support the decomposition and integration of designs.

3.4.2 Network Representations

The Edinburgh Design System (EDS) (Logan et al. 1991) uses an Assumption Based Truth Maintenance System (ATMS) (De Kleer 1986) to represent the design space. A Truth Maintenance System records the dependencies between sets of facts or data. In an ATMS a subset of these facts, called assumptions, are the foundation upon which all the information is stored in the system. Any assumption that is supported, and is not contradicted by known facts, is deemed to be valid. When facts are updated or added to the system any assumptions that are contradicted are invalid and removed from the database. The ATMS in the EDS system maintains consistency within the design space. This is an important feature and Logan et al. state that:

"The production of a large number of alternative design solutions results in a major consistency maintenance problem. If the system is to effectively support the exploration activities of the designer, the various incompatible designs must be considered in isolation."

(Logan et al. 1991: p427).

Within EDS, the design is stored as a set of module classes linked by semantic links: such as; is related to and is a part of. The design evolves by the addition and linking of these modules. As the designer adds more assumptions the ATMS checks the design (modules) for any inconsistencies and highlights conflicting designs.

EDS enables the designers to record all aspects of the evolution of a design discussed in Section 3.4.1. The facts added or amended at any node are propagated to other related nodes, thereby supporting the inheritance of design changes. If a new fact
contradicts a supporting assumption upon which the design module is based, the module is highlighted as being inconsistent.

Banares-Alcantara (1995) also uses an ATMS to represent the design in the Knowledge Based Design System (KBDS). His work highlights the problems of storing large amounts of information about the constituent components of alternative designs. He claims that a component is conceptually the same in all of the design alternatives provided it has not been explicitly modified by the designer. In KBDS the ATMS maintains the consistency of the common units or schema used in the different alternatives. This allows the system to:

- save space as only one representation of the unit is necessary;
- maintain consistency as a single representation for the common units is used;
- propagate changes throughout the design by only changing one representation.

The combination of the object oriented hierarchy of schema and the ATMS, provides the designer with a means of exploring the design space. It also records the alternatives that were explored in a consistent and efficient manner. The KBDS is the only system that supports the decomposition and integration of designs. This is achieved by providing a common representation of the streams that connect the sub-designs. The streams are represented in both diagrams and are linked to a common underlying representation. If the properties of a stream are modified in one design the common model is changed and the changes are propagated to the other sub-designs. This feature maintains the consistency of the links between the sub-designs which is crucial to the success of the integration process.

### 3.5 Design Functionality

The design functionality of an artefact is the intended function that the designer expects that item to perform. For example, a pump may be added to a pipe to raise the pressure of the contents between two vessels. The intended function, to raise the pressure, is quite straightforward. However, in some instances the intended function
of an item may not be so easy to identify especially when an item performs more than one function. The following scenario describes why a record of the functionality is important.

A large heat exchanger is positioned under a pipe to provide structural support. The primary function of the heat exchanger is to raise or lower the temperature of the liquid entering it. The secondary function which is to provide structural support, is not obvious and other designers may not be aware of this function if it is not explicitly recorded. If the heat exchanger was replaced with a smaller one or repositioned without supporting brackets being added to the pipe, the pipe may sag and break. This could result in the loss of life and money.

Information relating to the functionality of the design components is often available in other sources of DR. For example, the functionality of components may have been discussed during the design deliberation or as part of the constraints. An extract from part of the deliberation may read:

Problem - "We need to raise the pressure of the liquid in the pipe"
Solution - "Add a pump to raise the pressure of the liquid"

While it is possible to abstract the intended functionality from other sources of DR, Chandrasekaran et al. (1993) and Ganeshan et al. (1991), have argued that this information should be recorded explicitly in its own right. The heat exchanger example highlights the difficulty of identifying secondary functions and demonstrates why it is good practice to record the functionality. Ulrich and Seering (1992) recognise the attraction to designers of components having multiple and possibly unconventional functions, as this may enable them to achieve a simpler and more compact design. The growth in such practices means that it is even more important to have an explicit representation of the intended function(s).

Several different techniques have been used to represent functionality of design components including; semantic, mathematical and tree representations. Each of these
representations have been discussed above. The following section examines how these approaches have been used to record functionality.

3.5.1 Semantic representations

Chandrasekaran et al. (1993) describe a Functional Representation (FR) language that is used to record the functionality of an artefact. Within this language, the design artefacts are recorded as frames with the functions and the sub-components recorded in the appropriate slots. The overall function of the artefact is described in terms of its sub-components' functions and the relationships between them. For example, a nitric acid cooler would consist of a heat exchange chamber, a pump and some pipe-work. The function of this cooler is to lower the temperature of the nitric acid. Each of the components within the system would also have an associated function, the pump would distribute the cooling water into the chamber, the pipes would connect the components and the heat exchange chamber would transfer heat from the nitric acid to the water. This demonstrates that the functionality of the larger design is made up of the combined functionality of its sub-components.

One disadvantage with this system is that it does not allow the designer to change or add functions to a design artefact. In the case of the heat exchanger, the designer would not have been able to record the secondary function which was to support the pipe.

The KBDS (Banares-Alcantara 1995) also uses a semantic representation to record functionality. Within this system the desired function that a design or sub-design must meet are explicitly recorded. Banares-Alcantara recognises the possible multi-functions of a component and have developed a system that records associated functions. He identifies five categories of functions:

- transformation
- separation
- change in pressure
- change in temperature
- combination (mixing)
During the evaluation of the design, the system checks to ensure that all the required functions have been met. KBDS allows the designer to replace sub-components in the design with other components that perform the same function. This representation has similar limitations as FR, in that the functions associated with each component are fixed and the designer cannot add functions to them.

### 3.5.2 Mathematical representations

Ganeshan et al. (1991) present a model based on a mathematical representation that is intended to specifically reason about the design objectives. Their model has four types of entity: objectives; decisions; alternatives and operators. The functionality of the design objects is recorded in the objectives. At any stage in the design process, a design state contains a set of objectives and alternatives. Within this state there may be a set of functions that need to be met, a set of functions (objectives) that have been met and the current objective in focus. Using this representation the designer can see what functions are required, which ones have and have not been met and which one is currently being addressed. This system allows the designer to set the required functions associated with each component.

### 3.5.3 Tree representations

In Kusiak and Szczerbicki's (1992) tree representation, the requirements and functions are explicitly represented. In this system each requirement has an associated function that must be met. However, the representation is intended for modelling the conceptual stages of design. As a result, the functions recorded are not linked to the actual design artefacts which are produced at later stages in the design. Thus the functionality of any one artefact may not be accessible to the designer.

In summary there are different methods of recording the functionality of a design and its sub-components. The advantages of the systems described above are that they explicitly link the functions with the requirements. This allows the systems to identify requirements that have not been met. The disadvantage is that when the functions of a component have been defined, the designer cannot amend or add to them. This prevents them from including secondary functions.
3.6 Relationships Between Design Objects

A design normally consists of many inter-connected components which when altered, could affect other parts of the design. The difficulty is that the relationships between components are not always easy to identify. The corollary of this might be that a modification to one component may have unforeseen, and possibly undesirable consequences on another part of the design. By explicitly recording the relationships between the different components, designers will have a better understanding of how changes to one component may affect other parts of the design.

Semantic representations provide an ideal method of representing such information and are the only method that have been used. This section discusses a number of design tools which use some form of frame to implement the representation. The major difference between them lies in the slots included and the types of information recorded in these slots. The tools discussed are Design Rationale Capture System (Klein 1993), Functional Representation (Chandrasekaran et al. 1993), Semantic Model Extension (Clayton et al. 1996) and NODES (Duffy et al. 1996).

Klein's (1993) Design Rationale Capture System (DRCS) represents the relationships between the design modules and design tasks. DRCS uses a set of pre-defined relationships to produce the semantic model. Those relationships which apply to the design modules are: has-submodule; has-specialisation; has-attribute; is-of-type; has-interface and is-connected-to. The relationships which apply to the tasks are: has-priority, has-greater-priority-than, has-subtask, comes-before, is-of-type, has-plan and has-action. Figures 3.14 shows how information relating to an aeroplane design can be recorded using DRCS.

By recording the tasks as well as the modules, DRCS is able to express the relationships that exist between these two aspects of the design. If any sub-task is not, or cannot be completed, the designer can identify which design modules will be affected.
Although the links between the modules are recorded, the changes made to any sub-module are not propagated to others throughout the system. In the above example, an aeroplane has sub-modules wing and body which have the interfaces wing mount and body mount. The relationship between the wing mount and body mount are shown by an 'is-connected-to' link. However, a change to the wing sub-module will not be propagated through the wing mount to the body mount and on to the body. Hence, the consequence of the change is not shown. While this is a particular limitation of this system, it does provide a good method of representing the relationships between both sub-modules and the tasks which are required to produce them.

The Functional Representation (FR) model (Chandrasekaran et al. 1993) despite being developed primarily to record design functionality, is able to record relationships between sub-components. The following two figures demonstrate this using a nitric
acid cooler as an example. Figure 3.15 is a schematic diagram of a nitric acid cooler and Figure 3.16 is a FR representation of the same structure.

Figure 3.15. The structure of a nitric acid cooler (NAC)

Structure((Device(NAC;cooling-capacity and temperature parameters:ports, p1,p4,p5,p7))
Components: pipe1(len,diam,input,p2), pipe2(len,diam,p2,p3), pipe3(len,diam,p3,output),
Heat-exchange-chamber(dimensions,inport,outport)
Water-pump(input,output)
Function (pipe1): conduit (input,output)
Function(Heat-exchange-chamber): exchange-heat(parameters)
Function(water-pump) ..... ....
Relations: component(pipe2) contained-in component(Heat-exchange-chamber)
component(pipe1) conduit-connected(pipe2) (ports: connected ports)
component(Water-pump) conduit-connected component(Heat-exchange-chamber)
(ports: connected ports)

Figure 3.16. A FR representation of the above nitric acid cooler

The design system by Chandrasekaran et al. (1993) uses a FR model to predict changes in the design state. For example, the nitric acid will change state after passing through the cooler, i.e. its temperature will be lower. If the function of one of the sub-components was changed, or the relationships between the sub-components were
modified, the effect on the other sub-modules would be propagated through the system.

The Semantic Model Extension (SME) developed by Clayton et al. (1996) provides the designer with a means of developing semantic models of 3D CAD drawings. SME was developed in the architectural design domain and allows the designer to produce instances of the objects represented in design diagrams. Each instance records the geometry of the object, its associated function and behaviour. The behaviour represents the expected performance of the object and includes aspects such as: length, height, width, area, cost and energy flow.

Once the designer has manually produced the SME it can be used to predict how changes to an object affect the design as a whole. The model allows designers to focus on different aspects of the design such as cost, energy and space. Changes to a 3D CAD object on the screen are propagated through the model thereby demonstrating how the design aspect of interest, cost, space, etc. is affected by the change.

The SME model uses symbolic reasoning and numerical analysis to enable the designer to predict the impact of changes on any object within the design. As the changes are made graphically to 3D CAD objects, this tool integrates the design diagram with the model and makes the tool easier to use. As it stands SME is very domain specific and it may not be easily transferred to other domains. A further limitation of this model is that some aspects of the design may be affected by factors not represented as objects in the diagram. For example, the cost of a building may also be related to the geographical location. These types of relationships cannot be recorded by SME.

Duffy and colleagues (1996) developed NODES to support the early stages of the design process. The system is essentially a model of the design objects and their associated numerical relations. This model is intended to both support the designer during the design process and to support the re-use of the design.
NODES consists of three basic entities; objects, characteristics and formulae. The objects are used to represent both design components and concepts. The characteristics and formulae are sub-components of an object. The basic relationships between objects are recorded by the slots, part-of and parts. For example, an object representing a house could have a sub-part, kitchen. This would be represented as an object and could have the characteristic, area. ‘Area’ would then be described in terms of a formula based on the length of the walls in the kitchen. The walls themselves may also be recorded as objects and could contain the values height, length, thickness, etc. If the length of one of these walls changed the area of the kitchen would automatically change because the values would change in the formula used to calculate the area. This value could be passed upwards to the house object which would change its total area.

NODES provides a clear representation of the relationships between design objects. It achieves this by showing how the objects are connected as well as how the values of one object affects those of another. The parameters of an object that affects the value or state of any other object or concept, will be recorded in the associated formula. Groups of objects and their relationships can be built up into larger design units and re-used in the current or future designs.

All of the systems reviewed use semantic models to represent the relationships between design components. Each supports a different aspect of design as well as recording the relationships between the components. DRCS allows designers to represent the relationships between the design tasks as well as the components. FR language links the functionality of the components and their relationships within a single object. SME is the only system that provides an explicit link between the design diagram and the model. It allows designers to manipulate the model by changing parameters in the diagram. Finally, NODES differs from the other systems in that it provides the designer with a means of storing and re-using sets of design objects. It also records design concepts as well as objects and can propagate changes through the design.
3.7 Requirements

The requirements of a design represent the aims and goals and are essentially the driving force behind decisions or actions. The objective of the designer is to produce a design that meets the requirements. The design intent has been defined as:

"... the 'expected' effect or behaviour that the designer intended that the design object should achieve to fulfil the required function.'

Sim and Duffy (1994: p2)

Design intent and design rationale have sometimes been used synonymously but they are in fact different aspects of the design. The design intent is a combination of the objectives and functionality of a design while the rationale is the reasons for the goal or for choosing a particular method of achieving a goal. If one understand the goals the designer is trying to achieve, it is easier to understand why certain decisions are made. This section discusses some of the tools based on semantic, tree and logic representations that have been used to record the goals of a designer.

3.7.1 Semantic Representations

In Lee's (1991) DRL an IBIS type representation is used. Figure 3.17 compares how the same problem is represented in DRL and IBIS. This diagram shows how Lee uses the IBIS representation but modifies in such a way that the issue, 'who is responsible for the windows sub-culture?', while being the decision problem, is presented as if it were a goal. The possible solutions are represented as alternative nodes (position nodes in IBIS). In DRL the sub-goals are effectively arguments but are stated as an aim rather than as a benefit of selecting the alternative.

While DRL provides a semi-formal notation that allows the designer to explicitly state the goals, the decision problem in DRL and the sub-goals are represented by the same goal node. The issue or decision problem that is raised is a question that needs to be resolved and may or may not contain the goals. Therefore, it is not appropriate to represent both the decision problem and the sub-goals with the same goal node. This is a weakness in DRL. For example, a decision problem during the early stages of
design might be “where should we build the process plant?”. The goals are to locate it near a motorway and in a region that receives government development aid. The decision problem and the goals are different and should be represented by different nodes.

**Figure 3.17. A comparison of the IBIS and DRL structures.**

### 3.7.2 Tree Representation

Baxter (1992) developed a Design Maintenance System (DMS) which records goals during software development. The DMS uses a tree structure to represent the goals and sub-goals hierarchically. Each goal is decomposed into sub-goals until the sub-goal can be satisfied by some action or functional specification which is referred to as a schema. This is shown in Figure 3.18.

The DMS structure is very similar to the requiremental and functional trees that were used by Kusiak and Szczerbicki (1992). In their system the requirements are broken down into sub-requirements until each sub-requirement can be satisfied by a function. The requirements are effectively a representation of the goals in the DMS system. A major difference between the two systems is that Kusiak and Szczerbicki’s system allows the designer to explore and record alternative requirements and functions.
Both tree representations allow the designer to represent the requirements of the design. However, DMS does not support the exploration of alternatives and neither provide an explicit link from the final design components to the requirements.

### 3.7.3 Logical Representations

Clibbon and Edmonds' (1996) logic based model explicitly represents the relationships between the requirements and the design objects. Their system embodies the domain design knowledge and strategies that the designer uses to meet the requirements. Their design model is based on a multi-based logic representation with separate declarative representations for the objects and requirements. The objects and requirements are represented as clauses and are recorded in the object and requirement layers. They are linked by design knowledge and strategies which are also represented as clauses. The strategies are used to help the designer to meet the stated requirements by linking the objects to the requirements. The structure of this model is shown in Figure 3.19.

As the strategies are domain specific new clauses would have to be developed for any new domain. Another problem with this representation is that it only records the requirements that are specifically related to design objects. Often during the design process there are other requirements that need to be met that are not specifically related to any one design artefact. For example, a requirement of a plant to be located...
near a motorway, is not related to any specific object in the design and therefore is not represented in the model.

Fig 3.19. Clibbon and Edmonds' representation of requirements and design objects

3.8 Integrating Different Aspects of Design Rationale

So far this chapter has examined the types of information which make up DR and discussed the systems and models which have attempted to record this information. Table 3.3 shows the various aspects of design rationale that some of the systems
discussed incorporate. The ability to fully integrate all the various aspects of DR is still lacking in these systems.

### Table 3.3 A comparison of some tools reviewed and aspects of DR they address

<table>
<thead>
<tr>
<th>System</th>
<th>Deliberation</th>
<th>Design Space</th>
<th>Constraints</th>
<th>Functionality</th>
<th>Requirements</th>
<th>Component Relationships</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIBYL (Lee 1991)</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>QOC (MacLean et al. 1991)</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KBDS (Banares-Alcantara 1995)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>DRCS (Klein 1993)</td>
<td></td>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>Functional Trees (Kusiak &amp; Szczierbicki 1990)</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>FR (Chandrasekaran et al. 1993)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>NODES (Duffy et al. 1996)</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>PHI (McCall 1991)</td>
<td>Y</td>
<td></td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A further problem with some of these tools is that the DR is often not integrated with the design artefacts. Klein states that:

"Since the associated decision rationale capture tools are not integrated with design capture tools they face the potential for inconsistency between the rationale and design descriptions, spotty capture of design rationale and the tendency to waste time on issues that prove unimportant. Existing rationale language technology is also limited in that it typically does not capture design intent, the relationships between design decisions or their history. Finally, techniques are needed to reduce the significant burden to design engineers involved in describing design rationale".

(Klein 1996: p107)
Although the current systems fail to provide a truly integrated design system that supports DR capture many of them integrate several aspects of DR. The work by MacLean et al. (1991), McCall (1991), Lee (1991) and Potts and Bruns (1988) provided tools that integrated the design space with the design deliberation. They are all based on an IBIS type representation. The integration of the deliberation and design space was achieved by linking the positions in the deliberation to the artefacts in the design space to which they relate. However, the problem with these systems is that the deliberation that is recorded is limited to that which takes place about specific design artefacts. The design space itself is also limited in that the relationships between the different artefacts relies on a deliberation component to link them. A further problem is that in some cases the design deliberation may relate to higher level abstract topics such as the location of a process plant and will not relate to any design artefact in the design space. These representations do not support the recording of such information. Lee's model is the only one of these type of models that also integrates the requirements and relationships between the artefacts. This is achieved by extending the IBIS type representation to support many more node and link types. Unfortunately this results in a much more complex structure that confuses the difference between issue and goal nodes.

The work presented by Duffy et al. (1996), Klein (1993) and Chandrasekaran et al. (1993) all provide a means of integrating the intended function of the artefact and the relationships between them. These systems are all based on a frame representation which is well suited to recording such information. Design artefacts are represented as frames that contain slots for their intended function and a list of their sub-components. The systems developed by Klein and Duffy et al. also record constraints within the frames slots by providing a formulae for the slot or a set of acceptable values. Frames provide a good method of recording hierarchical relational information and storing an artefacts attributes. However none of these systems support the recording of design deliberation or the design space. Frame representations can be used to support these aspects but the frames used in the above systems focus on individual artefacts and their associated information.
The work by Banares-Alcantara (1995) provides a means of recording the design space, functionality, requirements and the relationships between the components. The KBDS was extended to include an IBIS component that records design deliberation (King et al. 1995, Banares-Alcantara 1997). The addition of the IBIS component provides a similar model to the one described by Goodwin and Chung (1994). However, there are significant differences between the two models. The original KBDS basically comprised of three different design spaces that are linked: the alternative space, objective space and design model space. The objective space records the objectives of a design or one of its sub-components. The alternative space records the design alternatives that are generated to meet the objectives and the model space is used to evaluate the alternatives in terms of the stated objectives. The IBIS structure provides another space, the deliberation space, that is linked to the original three design spaces. The model proposed by Goodwin and Chung (1994) uses the alternative space as a single representation of the design space. All of the related information is integrated with this space to provide a clear and consistent representation of the design space. Although the KBDS records several different types of information that is useful in recording DR, it does not provide a clear and consistent integration model. As a result the designer must access the information through four different design spaces and may not always be able to easily identify the relationships between them.

In summary, although some of the tools have integrated several aspects of DR the level of integration achieved is still limited. Many of the tools also fail to integrate the DR capture with the design artefact itself. DR capture tools should record and integrate more aspects of DR in a single environment with the design artefact itself.
3.9 Conclusions

DR is the reason why design decisions are made and it is essentially a by-product of the design process. Often only one or two aspects of the DR are recorded and these are rarely recorded in conjunction with the design artefact itself. The preceding section highlights this problem.

Another problem with many of the DR capture systems have limited, if any, management facilities. For example, gIBIS was one of the first systems to offer a simple search facility. This enabled the designer to access information with greater ease and speed. In SIBYL, Lee has included a precedent manager which helps designers identify knowledge in past decisions that might be used to solve the current problem. She states that, "it heavily depends on interactive feedback provided by the user" (1991: p122). This limits its usefulness.

The other systems reviewed do not refer to management facilities at all. The two that do are concerned only with accessing information rather than managing the DR capture process and the design project. For those who use DR capture systems, key concerns are accuracy and ease of use. If any system is not able to accurately record the process by which the design requirements are transformed to a final product, and does not give the designer access to this information, then the system may hinder rather than help the designer(s).

This review of current literature has highlighted a need for a system which is able to capture several aspects of DR, integrate it with design artefacts and address information management issues. A system which consists of these elements will overcome many of the shortcoming of current DR tools. The DR capture system developed as the basis of this thesis, has three distinct features. First, it provides designers with an environment that supports the integration of different aspects of DR. Second, it integrates this information with design artefacts and third, it has a management facility that supports the access and control of the information recorded. The development of this system as well its implementation and use, are discussed in the following chapters.
Chapter 4

An Overview of an Integrated Design Information System

DR consists of different types of information. The previous chapter discussed these and the tools used to capture the information. The review of available DR capture tools showed that many aspects of DR are captured in isolation. To provide a more complete representation of DR several aspects of design information should be recorded in an integrated framework within a single design environment. This design environment should include a Computer Aided Design (CAD) package to support traditional design practices and integrate the DR with the design artefacts. In this chapter an Integrated Design Information System (IDIS) is presented that provides such an environment.

The first section discusses in greater detail the need for an integrated framework before presenting the structure of IDIS. The representations used to record the design space, design diagrams, deliberation, constraints, functionality and the changes made to the design are described in the subsequent sections. The final section describes how IDIS manages and accesses the information recorded.
4.1 An Integrated Design Information System

The different aspects of DR are not generated in isolation; the design deliberation that occurs during the resolution of a problem, may result in the exploration of several alternative designs. The exploration of these designs expands the design space which has been explored. The new designs may provide some of the required functions and different alternatives may provide the same functions in different ways. New plant items can lead to additional constraints or, result in the generation of more problems which in turn, require more deliberation. The constraints imposed on the existing design, will also affect which alternatives can be explored and can introduce new functions that the different design components must perform. When one considers that the deliberation could be about any one of a number of aspects of the design and the constraints can affect the functions and alternatives explored it is easy to see how all this information is intricately linked.

Some of the methods used to integrate different aspects of DR were discussed in Chapter 3. The representations used by the different tools and their limitations were described. IDIS aims to provide an integrated framework that supports the design process while recording various aspects of the design that make up DR. The aspects of DR that are recorded in IDIS are: the alternatives that are explored in the design space, design diagrams, design deliberation, constraints, functionality and the changes that are made to the design. In order to integrate these different aspects a common frame representation is used. The focal point of the integration in IDIS occurs in the design space. Each aspect is represented independently and they are integrated by providing a link to the appropriate point in the design space. Every point in the design space represents an alternative, these are linked in a hierarchical structure allowing designers to navigate through the design space and view the relationships between the designs. This model allows us to integrate the various aspects of DR without restricting the type of information that can be recorded.

The models presented by MacLean et al. (1991), Lee (1991) and Potts and Bruns (1988), restricted the type of deliberation recorded as they used the link between the
deliberation and the artefacts to record the design space. The representation of the design space also relied on the links between the deliberation and the artefacts. The model presented by Banares-Alcantara (1995, 1997), has four different types of design space: alternative space, objective space, design model space and deliberation space. These design spaces are linked but no one design space provides a focal point where all the relevant information is integrated. IDIS uses a single design space representation, the alternative space. This provides a more coherent structure of the overall design and a single point of integration. Figure 4.1 shows how these different types of information are grouped together at a single point in the design space in IDIS.

The first design, design A, is generated from the initial specification and is the starting point of the exploration process. Designs B and C are alternatives that have been generated by extending design A. After evaluation design B is rejected and design C is extended to produce design D. This process of generating and evaluating alternative
designs continues until a completed design is produced that meets all of the requirements.

At every point in the design space the system records the design diagram, design deliberation, design constraints and a list of the changes that have been made to the design. The intended functionality of each individual design item is recorded in the design diagram. The node representing design C in Figure 4.1 has been expanded to highlight the information stored at that point. The points in the design space are linked in a hierarchical structure to provide a record of the design history and the alternatives that have been explored.

The information recorded by IDIS, can be used during the design process to aid the design team or can be used in the future when the design is reused or modified. In the rest of this chapter the design issues that relate to the different components of IDIS are discussed.

4.1.1 Design Space

A record of the design space is more than just the path from the initial starting point to the final design, or the collection of all alternatives explored. In IDIS a representation of the design space was required that:

- records the alternatives explored;
- keeps a record of the alternatives rejected as well as those accepted;
- provides the designers with a method of navigating through the design space;
- provides a record of the design history of each alternative.

Some of the representations used to record the design space were reviewed in the previous chapter and in light of this discussion a tree representation was chosen. In IDIS a viewpoint mechanism, similar to the one proposed by Chung et al. (1993) is used to record the design space. As alternatives are generated new nodes are created and attached to the parent node in the tree. The viewpoint mechanism records not only the areas of the design space that have been explored, but also provides a structure that describes how the designer reached each alternative. The designers can navigate through the design space by changing the current viewpoint. The complete viewpoint
structure can be displayed to enable the design space to be examined and see which areas have been explored.

The viewpoint mechanism presented by Chung et al. (1993) cannot support the decomposition and integration of designs. The limitations of tree structures in this area were discussed in the previous chapter.

The viewpoint mechanism implemented in IDIS is an extended version of the one suggested by Chung et al. (1993). This new representation partially supports decomposition and integration by adding two new links. When a new node is created in the design space, designers have the option to explore alternatives or decompose the design. Decompositions are treated differently from alternatives as the decompositions can be integrated at a later stage but alternatives cannot. The integration process cannot be performed without a consistent interface between the sub-designs. IDIS allows the designer to indicate that a design has been decomposed but it does not record and maintain a consistent interface between the decomposed designs so cannot provide an automated mechanism for complete integration. It does however, allow designers to indicate that decomposed designs have been integrated but it does not integrate the information in the different designs itself.

The viewpoint mechanism was used in preference to other representations for a number of reasons. The functional and requirement tree structure only recorded aspects of design that were specifically related to a requirement or performed a specific function. Often in design, especially in the early conceptual stages, abstract design details that do not have specific requirements are explored. These cannot be recorded using requirement and functional trees. QOC representation uses the IBIS representation and links the designs to the deliberation which is part of the objective of the project. However, in the QOC representation every issue is an issue about a design artefact and every alternative that is generated represents an alternative design. Combining the deliberation and the design space in such a way looses some of the flexibility of the IBIS representation and does not support the deliberation of abstract design problems.
An ATMS was used in the EDS (Logan et al. 1991) and KBDS (Banares-Alcantara 1995) to record the design space of alternatives. While the viewpoint mechanism is not a complete ATMS (it does not perform any resolution of conflicts or incremental updating of changes) it does have some of the same features of this representation. The incremental updating of the design changes are specifically prevented in IDIS. It is important for designers to be able to see what parent designs were used to create the alternative designs. If a parent design was modified after alternatives had been generated from it this would not be possible. When a parent design is modified the changes recorded at the alternative would not accurately represent the differences between the two designs. To maintain the consistency of the design the parent design is frozen whenever an alternative is created from it.

4.1.2 Design Diagrams
Any design system must provide designers with a means of representing design diagrams. Two types of diagrams that designers may want to represent are, detailed design diagrams and sketches. Different tools are used to represent these in IDIS as they represent different types of information.

The detailed design diagrams are recorded using AutoCAD (Autodesk Ltd.). This is a sophisticated CAD system that is widely used in many domains. The AutoCAD diagrams represent alternatives in the design space and are linked using the viewpoint mechanism. Every viewpoint is represented as an AutoCAD screen and in most cases they contain a design diagram. However, in some cases there may be no diagram related with that viewpoint when more abstract aspects of the design are explored. All alternatives are represented by an AutoCAD screen irrespective of whether they include a diagram or not. The AutoCAD menus have been modified to provide the user with access to the features offered by IDIS (all of the original AutoCAD features are still available). The advantages of using a tool such as AutoCAD are that: it is widely used and provides a fully functional CAD system; it provides a mechanism for linking DR with traditional design information and it is a familiar working environment for designers.
As well as the modified menus other additions have been made to AutoCAD. A set of pre-defined design items (pumps, heat exchangers, open vessels, closed vessels, etc.) are included as icons and are used to create the design diagrams. These items are linked to an internal IDIS representation that is used to maintain a semantic model of the diagram. The icons can be joined by either a pipe or a control line depending on the type of item and in/out ports. The physical design properties and the chemical information relating to the plant items are viewed and modified through the AutoCAD diagram. Any changes made to the design or individual items are automatically recorded in the IDIS semantic model.

The semantic model recorded in IDIS is used to reason about the relationships between the design items and check for the violation of design constraints. This model can be written to a file to provide a representation of the design that can be used by other systems. This allows other design systems to access the design information captured in IDIS. To date two systems have been tested, QUEEN and MEPPI. These are discussed in Chapter 5.

The other package incorporated in IDIS is Xfig. This is a freeware drawing package that runs under XWindows. Xfig provides designers with a means of drawing quick sketches to help clarify points or explain ideas. These drawings are distinct and different from the detailed design diagrams recorded using AutoCAD. The AutoCAD diagrams are effectively part of the product of the design process while the sketches are part of the information used during the design process. The Xfig sketches, as with the other types of design information, are recorded at the specific point in the design space to which they relate. The sketches are not specifically linked to any node in the issue base but may be referred to in the issue base nodes during the deliberation process.
4.1.3 Design Deliberation

As mentioned in Chapter 3, there are many places in the design process where deliberation occurs. The deliberation about an aspect of the design takes place at a specific point in the design space. IDIS records the deliberation at that point with the design to which it relates.

There are several techniques that can be used to represent deliberation: Rhetorical Structure Theory (Mann and Thompson 1987), Toulmin Form (Toulmin 1958), Semi-Structured Message Templates (Malone et al. 1987), and Issue Based Information Systems (Kunz and Rittle 1970). These techniques were discussed in the previous chapter. The section below, builds on this earlier discussion but concentrates on the rationale behind the choice of technique to represent the deliberation in IDIS.

During the design process designers may discuss one or more issues concerning any item or conceptual aspect of the design. However, certain decisions can be made based on experience and or accepted practices and may therefore, require no deliberation. To support and record the deliberation in IDIS a suitable representation should:

- provide a clear structure of the deliberation that has taken place;
- allow the structure to develop as the deliberation progresses;
- support the deliberation process as well as recording the deliberation;
- allow designers to discuss aspects of design not directly related to a single design item;
- allow designers to discuss more than one topic about a single design item;
- allow designers to record decisions made based on experience or accepted practices.

In IDIS, an IBIS representation is used to record design deliberation as it supports all of the requirements described above. Unlike RST and TF, IBIS supports the deliberation process and records it as it evolves. IBIS also explicitly represents both the type of node and the relationships between them. Neither RST or TF has this feature.
In IDIS an issue base refers to a collection of nodes which relate to a single design problem. The issue bases are recorded at the point in the design space where the design is stored (A, B, C or D in Figure 4.1).

### 4.1.4 Design Constraints

A further component of IDIS is the rule base which records the design constraints. Again there were a number of possible techniques that could have been used including, mathematical, graph and rule representations. The requirements for the IDIS representation was that is should be able to:

- be understood by designers;
- provide an explanation to designers;
- be able to represent both numerical and abstract symbolic constraints;
- allow designers to add constraints quickly and easily.

A rule representation was adopted in IDIS, this takes the form, IF conditions THEN action. There are two types of rules, user and system defined. The system defined rules represent the standards and codes of practice that must be met. The user defined rules are added by designers and relate specifically to the current design or some component of it. Designers cannot delete or modify system defined rules and the design must satisfy all these constraints. The system defined rules are effectively hard constraints and the user defined rules are soft constraints.

The system defined constraints are stored at the root node and apply to all the designs. The specific constraints the user adds are stored at the point in the design space where they are added. The rules support mathematical and logical concepts such as: is greater than, less than, equal to, is not etc. Symbolic associations are also available: has a, is a, is connected to, is connected upstream, is connected downstream. The constraints are checked against the semantic model maintained in IDIS. The rules do not fire automatically because designers may violate constraints during the development of a design. To avoid annoying messages, an explicit command must be issued to IDIS to check the current design. The error messages generated by any rules that have fired are presented in a file along with the identifiers of the rules that have created them.
4.1.5 Design Functionality
One of the reasons why it is important to record the intended functions of design items is because designers often use standard parts for secondary and or unconventional functions. For this reason it was important that IDIS had the ability to:

store the intended function of an item in the design;
change or add to the function of an item;
list the functions that a specific design or part of the design performs.

In IDIS the intended function of an item is recorded in the internal semantic model. Each item originally contains a default value which describes the conventional function it performs. Designers can add to this description or change it if the item is being used in an unconventional way. At any point in the design space that contains a design, IDIS can list the functions that a particular item performs. This allows designers to see quickly and easily which functions have and have not been satisfied.

The major benefit of this, over the representations discussed in the previous chapter is that designers can record and change the functions of the design items themselves. In the other representations discussed, the functionality of the design items were fixed and designers could not change nor add to them. The advantage of having fixed constraints is that the system can match these against a set of fixed requirements recorded by designers. This allows the automatic detection of requirements that have not been satisfied. One of the limitations of the representation used by IDIS is that it cannot check the stored functions against a set of stored requirements to see if they have been satisfied. The disadvantage of providing a more flexible way of representing the design functions is that the ability to reason about the functions is lost.

4.1.6 Design Changes
Although design changes do not necessarily provide an insight into the DR, they do provide valuable information that can be used by a design system during the design process. Designs are created incrementally by modifying, expanding and exploring alternatives. Hundreds or even thousands of changes may be made during the design
process and these need to be recorded as many may not be apparent to other members of the design team. In this section some methods used to record design changes are described as is the representation used in IDIS.

Winter et al. (1992) and Holley (1992) describe a history file representation used to record design changes and both their systems use a single history file. Their systems record which aspects of the design have changed, who made these changes and when. The single history file representation is a simple and useful method of capturing valuable information. However, it does have disadvantages. As there is only one file for the whole design, designers cannot have multiple versions of the same item with different values. While this single file representation is acceptable for a simple linear representation of design it cannot record alternative designs.

IDIS uses a history file to record design changes. Each viewpoint contains its own change list file which is created when a new point in the design space is explored. This file contains a list of all the changes, each entry stores: the actual changes made, by whom and when. When designers change viewpoints IDIS uses this information to reconstruct its own semantic model of the design. This change list also enables designers to view all the differences between two designs.

4.2 Recording and Accessing Design Information

A design information system is only useful if it does not hamper the design process and users can easily access the information recorded. As mentioned in Section 4.1.2 IDIS uses AutoCAD to create the detailed design diagrams and provide a focal point for the alternatives explored in the design space. The IDIS functions and information recorded are accessed via menus on the AutoCAD screen. The different types of information recorded by IDIS have already been discussed, what remains to be explained is how IDIS captures this information and how designers access it. This is discussed below.
4.2.1 Design Diagrams and Alternatives

Every design project starts with the creation of a new project. This generates a root viewpoint (represented by a blank AutoCAD screen) which is the starting point for the new design. Design diagrams are created by selecting items from an AutoCAD menu and inserting them on the screen. Items are connected by selecting the type of link required (pipe or signal line) and then clicking on the inport and the outport of the items to be connected.

As the design evolves, designers explore new alternatives which are created and linked to the existing designs. A hierarchical structure of the design develops and the resulting structure provides a record of the design space explored. When a new node in the design space is created, either an alternative or a decomposition link is added between the parent and the new design. As the new design has evolved from the parent design it inherits its design diagram. Any subsequent modifications made to the design are stored at the new viewpoint independently.

The structure of the design space is recorded automatically by IDIS so again there is no additional work required by designers. IDIS lists the viewpoints and their relationships to allow designers to view the structure of the design space. Moving through the design space is achieved by changing the current viewpoint. At any stage designers may also list the information stored at a viewpoint. This includes a list of the associated issue base nodes, Xfig sketches, parent and children viewpoints and any changes made to the design at that viewpoint.

4.2.2 Design Deliberation

Designers would normally discuss design issues in team meetings. However, if an issue arose between meetings that had to be resolved, it may be discussed informally with other team members. This deliberation may or may not be formally recorded in the design documentation. In IDIS all of the deliberation can take place through the system. It is not claimed that meetings can be abandoned as it is still important for designers to get together as a team. However, issues may be identified between meetings, solutions suggested and evaluated and then the decision made at the
meeting. This information can be subsequently recorded in the system. Alternatively the decisions can be made by the designers and recorded in IDIS during the deliberation process.

Whenever a design problem is identified a new issue is created. This issue is 'tagged' with the point in the design space where it was created to indicate that the deliberation is about some aspect of the design at that point. Members of the design team read the issue node and respond to it by creating a position (possible solution) node and linking it to the issue node. Argument nodes can be linked to the position in the same way and allow designers to raise arguments for or against positions. In addition to these three basic nodes, designers can also add comments, facts and decisions. The issue base expands as the deliberation progresses providing a record of the information added by each member of the design team. Each component of the deliberation can be clearly identified by viewing the structure of the issue base. The designers can see which positions have been added, the number of arguments added and whether they were for or against the positions etc. Any aspect of the design, or any number of topics about the design can be discussed at any point.

The structure generated provides a record of the deliberation that has taken place about the issue. As with any body of information that can grow to a considerable size, it is important that the relevant information can be accessed quickly and easily. IDIS has three features to achieve this.

Every issue base node in IDIS has a list of who has seen the node. When the *unseen nodes* option from an AutoCAD menu is selected all of the nodes that have not been seen by the current user are listed. This feature enables designers to find all unseen nodes without searching all the issue bases.

The second feature allows designers to view the structure of an issue base. This can be done from any node in an issue base. Only the nodes added in relation to the selected node are displayed. If the issue is chosen then all the nodes in the issue base will be displayed. If a position is chosen, only the position and its associated arguments, facts
and comments will be displayed. The nodes and their relationships are displayed along with the person who created the node, the date it was created and its one line summary. This allows designers to easily view the structure of all or part of an issue base.

The final feature to help access information from the issue bases is a search mechanism. Designers can search for a string contained within either a node summary or in the text of the node. This is a relatively simple search mechanism and it does not perform context or logical searches. However, it is more than adequate for the desired task which is to identify the relevant nodes where information is recorded. To help limit the search space, the type of node and the viewpoint can be specified. If the default values 'all' are used for both of these, all the nodes in all of the viewpoints will be searched.

4.2.3 Design Constraints

Design constraints are added to IDIS using a menu comprised of pull down menus and edit boxes. The pull down menus provide a list of possible connections and tests that can be performed by IDIS. The connections supported by IDIS are: and, and not, or, or not, and then. The test conditions supported by IDIS include: >, <, =, isa, connected, connected without isolation, connected upstream, etc. Designers add constraints by entering the unit name or type, the parameters that are to be tested, and the test values in the edit boxes. The relationship between them are denoted by the option selected from the pull down menu. Table 4.1. shows the structure of the constraint, “the pressure of heater2 should not exceed 24 bar”. The warning message that will be displayed if the rule succeeds is added in the final edit box after the connection then. Once completed this constraint is added to the rule file of the current viewpoint. Rules are edited using the same graphical front end and are deleted by selecting the appropriate rule from a list. Designers cannot add, delete or edit the systems rules.
The constraints that apply to any design can be displayed at any time. The files that contain all the relevant rules, (the current and parent viewpoints and the system defined rules), are combined in a single file that is displayed. The design is not checked automatically. This avoids annoying messages that can be produced due to the violation of constraints during the intermediate stages of a design. Designers must explicitly issue a check design command to test the current active design. When the design is checked all the error messages that are produced by the rules that fire are displayed in a file.

4.2.4 Design Parameters and Functionality

Every item in the diagram has various parameters associated with it that must be specified as part of the design. These parameters include: material of construction, thickness of the walls of a vessel, operating temperature and pressure, design temperature and pressure, the intended functionality, etc. These parameters are set and displayed using an edit box. When the view a units spec sheet menu is selected and an icon in the diagram is clicked with the mouse an edit box is created that displays the item's parameter values, including its intended function. Some values are set by IDIS and cannot be changed by designers. These include the connections made to the inlet and outlets, the date the item was created and by whom. When a new item is created most of these parameters are given the default value 'undefined'. There are some exceptions, the intended function value of pump for instance is set to 'raises pressure'. Designer set or modify the parameters of an item by clicking on the item in the
AutoCAD diagram and typing in the new value in the appropriate slot in the edit box. The functionality slot can contain multiple values which allows designers to assign several functions to a single design item.

The chemical information relating to each item is accessed and changed in the same way. To display and set these values the chemical info menu is selected and the appropriate item is clicked with the mouse. This displays an edit box that contains a list of values that can be set for each steam that enters and leaves the item. These values include: the content of the stream, the mass fractions of the contents, the temperature and the pressure. All of these values have the default setting 'undefined'.

When any of the parameters are changed the semantic representation that IDIS maintains is automatically updated. IDIS also records the changes automatically in the change list file of the current viewpoint. No additional work is required from the designer to record and maintain this information.

4.2.5 Design Changes
As mentioned in the previous section all of the changes that are made to the AutoCAD design diagram are automatically recorded by IDIS. Designers do not have to perform any additional action to explicitly record these changes. The changes that are recorded have several uses. IDIS uses these changes to maintain a consistent semantic representation of the AutoCAD diagram when designers change viewpoints. IDIS traces the path from the root viewpoint to the new viewpoint. All of the changes that have been recorded at the viewpoints are then performed by IDIS in order. When the new viewpoint is displayed IDIS has an accurate representation of the state of the design at that point.

There are two ways that designers can access and use the changes. First, they can simply list all of the changes that have occurred to a design at a specific viewpoint. If they perform this action the changes that have been made to each of the items in the design, along with the date the change was made and the name of the person who made it are listed. These changes are displayed in reverse order so the most recent
changes appear at the top of the list. This information is useful if one is only interested in one design and wants to find out the changes that have been made to it.

The second way designers can use this information is to identify the differences between two designs. Often the differences between two designs are not apparent and any decision made on incorrect information could have serious unforeseen consequences. For example, it is very difficult to identify changes that have been made to a parameter of the same item that is represented in two diagrams. A pressure vessel may have an operating pressure of 100 bar in one design and 120 bar in another. These changes can be identified using the list of changes that have been made to both designs. Viewing all the changes and attempting to find the differences manually is a difficult and time consuming task, even when all the changes have been explicitly recorded. IDIS provides a feature that automatically identifies all the differences between two designs. All the differences between two designs can be displayed in a single file by selecting the appropriate menu and typing in the two viewpoints of interest.

4.2.6 Project management

The above discussion assumed that all designers had the same status and privileges. This is rarely the case in a design team as there is normally a team leader who is responsible for delivering the project and for managing other team members. IDIS provides facilities that help with these tasks.

Designers can have the status either, leader, member or other in IDIS. Only a designer who has leader status can add, delete or modify other designers' privileges. The leader has the option to allow designers to: make decisions, add, link and view nodes. By setting these functions the team leader can control who has access to and can add information to the issue base as well as who makes decisions about issues. All the designers can list the team members who are working on the current project and IDIS can display their name alongside their status and the tasks they are entitled to perform.
IDIS also provides a facility for monitoring the progress of the team members as well as the design project. When an issue node is created it contains an optional slot that can be filled with the date by when the issue is to be resolved. Any member of the team can access this information about deadlines. All project deadlines, regardless of whether they have been met or are outstanding, can be displayed. IDIS can also separately display any deadlines that have been missed as well the next ones due. Once a decision node has been added to an issue it is regarded as being resolved and the deadline date no longer applies to that node. This function allows designers to keep track of which issues are outstanding and which ones should be resolved next.

Designers can also check other aspects of the design. They can monitor the progress of a design by checking the date the design project was last modified or listing the changes made between two given dates. Using these features the team leader can see how much progress has been made and on which aspects of the design. The team leader may also have the responsibility of monitoring and managing other members of the design team. IDIS provides facilities to help with this. A team leader can check when a designer last used the system and what contribution they have made to the deliberation. They can view a designers contribution to the whole design or to a particular issue. This allows the leader to see which areas the team members are working on and the contribution they have made to each aspect.

IDIS also addresses the problem of maintaining the integrity of the information that is recorded in the issue bases. The links between nodes are semantic links that represent the structure of the argumentation so it is important that the correct links are made between the correct nodes. The IBIS component is intended to support and capture the design argumentation so designers should not be allowed to corrupt or lose information by adding or deleting nodes to the issue base in an undisciplined manner. IDIS checks the designers actions and does not allow the creation of illegal nodes or links. There are two types of integrity that are identified and maintained within IDIS, semantic and temporal.
Temporal integrity is maintained by preventing designers from adding information directly to an issue base after an issue has been resolved. However, they may want to re-examine an issue after a decision has been made. To enable this to happen a follow-up link was added to the IBIS representation. This node allows designers to follow up on an issue once it has been resolved. The new information is linked to the original issue base via the new link and is recorded as a new issue base. This mechanism supports the process of re-examining issue bases but makes a clear distinction between the information added before and after the decision was made. The semantic integrity of an issue base relates to its structure. This is maintained by ensuring the correct links are selected between the nodes.

4.3 Summary

The chapter provides an overview of IDIS. This system supports the design process and records information relating to DR. The different components of the system are identified and the representations used are discussed. Some of the features used to add and access the information recorded in IDIS are also described. Finally some of the project and team management facilities are presented. In the next chapter the implementation details of the system are discussed with a focus on the data structures used to represent the different aspects and the method used to integrate them.
Chapter 5

Implementation of IDIS

The previous chapter presented an Integrated Design Information System (IDIS) and its components. The various representations used for these components were discussed. This chapter describes how these are implemented in IDIS.

The chapter begins with a brief overview of the different components of IDIS. Section 5.2 describes the underlying representation techniques that are used to implement the components. Section 5.3 describes the data structures of the various components in more detail. Section 5.4 describes the user interface and methods used to integrate the components. Section 5.5 describes the mechanism that is used to export a representation of the design to a file that can be used by other design tools. The chapter concludes with a description of the hardware and software requirements.

5.1 System Components

IDIS has 7 components. (1) An issue based information system that supports and records the design deliberation. (2) A drawing package that allows users to draw sketches and express ideas graphically. (3) AutoCAD a CAD package is used to create the detailed design diagrams and provide the user interface for the IDIS program. (4) A rule based system is used to record and check constraints. (5) A semantic model of the design diagram that is used to reason about the relationships between the design components. This model is also used by the rule based system when checking for the
violation of constraints. (6) A list of the changes that have been made to the design diagrams which are used to maintain the consistency of a design when changing viewpoints. These change lists are also used to identify the differences between design alternatives. (7) A viewpoint mechanism that provides a structure to the design space, records the design alternatives and provides a focal point for the integration of the different types of information. The rest of this chapter will concentrate on the implementation details of the various components and describe how they are integrated in IDIS.

5.2 Frame Based Representation

Many techniques can be used to represent knowledge. Rich and Knight (1991) review a range of different techniques that are commonly used and classify them into two broad groups: syntactic and semantic representations. At the extreme end of the syntactic scale, the representations are not concerned with the meaning of the knowledge represented. They often have a strict formal grammar and use simple uniform rules for manipulating the representation. These representations are generally more rigid and have a more complex structure than semantic representations but they are often easier to process computationally. Semantic representations provide a more flexible and richer method of representing knowledge. They are better at supporting meaningful relationships between objects than syntactic representations.

For IDIS, a general representation must be able to represent the relationships between design components and the information that relates to them. A frame based semantic representation was selected for the underlying representation for most of the components in IDIS. The only exception is the design constraints which use a production rule representation. Although frames can be used to represent constraints they are not used in IDIS because they do not support the more symbolic and complex constraints.
Frames support the hierarchical nature of the issue base and viewpoint mechanism. They also enable the relationships between different components to be recorded. This representation is flexible and the same frame system is used to implement the issue base, viewpoint mechanism and the design items. The general structure of a frame representation was discussed in Chapter 3.

Frames are organised into hierarchies and are linked together by a method called inheritance-specialisation. Inheritance and specialisation describe the two main features of a frame representation. Generic frames are basic building blocks that are used to represent particular types or classes of objects. Individual items are represented as instances of these classes. Instances inherit the attributes from the generic class frames and include other attributes that relate only to specific types or instances of objects.

The following example shows how a pump which is an instance of a design unit can be represented using frames. The top generic frame is `unit` and has the representation:

```plaintext
frame(unit,
    [project is "project",
     date is "date",
     name is "designers' name",
     unitName is "unit name",
     function is "function"
    ]).
```

This frame states that a unit has the attributes: project that it belongs to, date it was created, the name of the person who created it, the name of the unit and the function it performs. A pump is a type of unit and represented as an instance of a unit. It inherits the three attributes from the unit frame and includes its own specific attributes.

```plaintext
frame(pump isa unit, [has_inlets [inlets],
                      has_outlets [outlets],
                      rating is "rating",
                      max_pressure is "max pressure",
                      max_temperature is "max temperature",
                      has_parts [parts list],
                     ]).
```

82
Instances of the unit type pump are created in a similar manner and contain attributes and values that are specific to that pump. The frame system that is used in IDIS was developed by Chung (1993), and was developed and runs in Prolog.

5.3 Data Structures

As mentioned above, frames are used as the basic representation for most of the components in IDIS. The top generic frame node is the basic building block for the issue base and viewpoint mechanism and is defined as:

frame(node, [
    summary is "one line of summary",
    project is "project name",
    name is "creator name",
    date is "date created",
    groupList info [],
    seenBy info []
]).

The slots have the following meaning and uses:

**Summary**: Stores a one line summary of a description of the object. This slot contains a string and is used to provide summary information when IDIS displays an overview of an issue base or the design space to designers.

**Project**: Stores the name of the project to which the node belongs as a string. The project name is set when a new project is created. This information is used to ensure that the information is stored in the correct project directory.

**Name**: Stores the login name of the designer who created the node as a string. This information allows IDIS to keep track of who has added the information.

**Date**: Records the date when the node was created. This information is obtained from the computer on which IDIS is running. Like the project name and creator name it is recorded automatically by IDIS when a new node is created. This slot allows IDIS to keep track of when the information was added to the design.

**GroupList**: This slot contains a list of all designers who have permission to view that node. This is determined by the set of designers that have been added to the project and their level of privileges. The information is stored as a list.
SeenBy: This slot records a list of all designers who have seen the node. This slot starts off with an empty list. When designers create nodes their name is added to the list. Other designers names are added to the list as they view the node. This information is used by IDIS when designers ask to see if there is information stored in the system that they have not seen.

This generic frame node is used as a parent for the other frames: viewpoint, issue, position, argument, comment, fact and decision. These are discussed below.

5.3.1. Viewpoint Mechanism

The viewpoint mechanism is the core component of IDIS and all the other aspects of DR are recorded in relation to the viewpoints. The generic viewpoint frame used as a template for all the viewpoint instances, is a sub-class of the generic node frame. The viewpoint frame has the following structure:

frame(viewpoint isa node,
    [    
    links info [parents, children, type, issue, position, argument, fact, comment, decision, changeList, diagram], 
    parents ref [], 
    children ref [], 
    type ref [], 
    issue ref [], 
    position ref [], 
    argument ref [], 
    fact ref [], 
    comment ref [], 
    decision ref [], 
    changeList info [], 
    sketches info [], 
    diagram info []]
);

Every viewpoint has a unique name; the root viewpoint is viewpoint1, the next is viewpoint2, etc. IDIS keeps a counter variable called viewpoint that is incremented by one every time a new viewpoint is created. The slots: children, parents and type are used to record the relationships between the different viewpoints. When a new viewpoint is created the parent viewpoint is automatically placed in the parents slot of the new viewpoint and the name of the new viewpoint is added to the list of children viewpoints in the parent. The type slot records the type of viewpoint node created. This can be either an alternative, decomposition or integration node. This information
allows IDIS to identify the type of node the viewpoint represents and prevents designers from integrating alternative nodes. The structure of the design space is described by the links between the parent and children viewpoints. The history of a design is represented by the path from the root viewpoint to the design.

The issue, position, argument, fact, comment and decision slots are used to store information about the issue base nodes that relate to a viewpoint. These slots store a list of the nodes of the type denoted by the slot name. This does not provide any information about the structure of the issue base but it has several uses. This information is used to check that two nodes are from the same viewpoint before they are linked. IDIS does not allow the user to link issue base nodes across viewpoints. IDIS also uses this information when displaying issues that have been discussed at one particular viewpoint.

The changeList slot stores a list of all the changes that have been made to the design at that viewpoint. The changes are stored as items in a list in the following form:

[action, unit type, unit name, user name, date]

Each time the design is changed a list with the above structure is created and added to the changeList slot. Inserting a pump, closed vessel and then connecting them would be recorded in the following way:

```plaintext
[
  [insert,pump,pump1,[roger,1/5/95]],
  [insert,closedVessel,closedVessel1,[roger,1/5/95]],
  [connect,[pipe,pipe1],[[pump1,outlet1],[closedVessel1,inlet1]],[roger,1/5/95]]
]
```

The list of changes has two uses, it allows IDIS to retrace all of the changes that have been made by designers and it allows designers to identify the differences between two designs.

Often designers cannot easily identify the differences between two designs. This is especially true when the same attribute of a design item has different values in different designs. By comparing all the changes that have been made to two designs
from a common point in the design space, IDIS can identify how two designs differ. It is not reasonable to expect designers to check all the parameters of the items in every alternative to see what the differences are as most of the parameters will be the same in both diagrams. The ability to easily identify the difference between two designs is important as minor changes may have significant consequences. For example, a small change to a pressure relief valve may have significant safety implications if all of the members of the design team are not aware of its correct value.

Each item in the change list contains all the information that IDIS requires to redo the actions performed by designers. When designers change viewpoints IDIS finds the path from the root viewpoint to the desired viewpoint. The designer’s steps are then retraced by redoing all of the changes in the changeLists from the root viewpoint to the required current viewpoint in the correct order. When this has been completed the semantic representation of the design maintained by IDIS, will match the state of the design in the diagram at the current viewpoint.

The final two slots in the viewpoint frame are the diagram and sketches slots. The sketches slot records a list of all the sketches that have been created in that viewpoint. This slot is originally empty when a new viewpoint is created and any sketches that are created at that viewpoint are added to the list. The diagram slot simply records the name of the AutoCAD diagram that represents the viewpoint. The diagram is given the same name as the viewpoint itself, i.e. if the viewpoint is called viewpoint1 then the diagram is called viewpoint1.dwg. As there may be several different diagrams in different projects with the same name, the full path of the diagram is stored. This path could be constructed by using the viewpoint name and the name of the project, but the speed of changing viewpoints is increased by explicitly recording the complete path in the viewpoint.

When a new viewpoint is created the diagram at the parent viewpoint is frozen, preventing it from being altered, and it is copied to provide the starting point for the new viewpoint. Designers are prevented from making any changes to the parent design to maintain the consistency of the design space. If a parent viewpoint was
changed after a new viewpoint had been created from it, it would not be possible to trace the history of the new viewpoint. This is because the new viewpoint would have evolved from a different point in the design space. Designers are notified that the parent design will be frozen when they create a new viewpoint and are asked if they want to continue before any action is taken.

5.3.2 Design Diagrams and Sketches

As mentioned in Chapter 4, sketches are drawn in Xfig and detailed diagrams in AutoCAD. Xfig is an X windows based graphics package which is powerful and easy to use. The sketches created by designers are not given a default name in the same way as the viewpoints and AutoCAD diagrams. Rather, designers enter the name of the sketch when it is created and IDIS checks none already exists with that name. When the sketch has been created the file is closed and saved in the current project directory. The name of the new sketch is then added to the list of sketches for that viewpoint. Designers can list all the available sketches, view them, edit their own sketches or create new ones at any stage during the design process. This feature complements the deliberation component and provides designers with a means of expressing their ideas graphically.

AutoCAD is used to represent the detailed design diagrams. The design units within the diagrams are represented by AutoCAD blocks and IDIS frames. An AutoCAD block is a user defined icon which can be referred to as an AutoCAD object within the AutoCAD environment. A collection of the most commonly used units in a process plant are provided with the system. This includes: pump, open vessel, closed vessel, heat exchanger, pipes, signal lines, trip valve, distillation column etc. At present there are fifty two different units available. A full list of the units included in the system is given in Appendix A. When an icon is inserted in the AutoCAD diagram IDIS creates a frame representation of the same unit. All the units are created from the generic frame unit that is defined as:
The slots: project, date, and name are used to record the project, the date the item was created and the name of the designer who created it respectively. The slots unitName and function record the name of the unit and the function(s) that it performs. Although the instance of the unit provides a unique name for the unit, the unitName slot is used to record a more accurate description of the unit. For example, a heat exchanger might have the value 'shell and tube heat exchanger' in this slot. The additional features required by the different design units are provided as specific slots for each unit type. Among the additional slots that a pump frame might have are: design pressure, design temperature, operating temperature, operating pressure, rating, inlet and outlet.

The description so far has only referred to the physical properties of the design units. These are important aspects of the design but designers are also interested in the chemical information that relates to the units. In IDIS a separate frame is used to record this information. This frame has the same name as the frame used to store the design information but it is prefixed by the letters chem to indicate it relates to the chemical information. The information stored at each design unit records the chemicals that enter and exit every inlet and outlet, in what fractions and their temperature and pressure. The following frame shows how the information associated with an inlet would be represented:

frame (chemClosedVessel isa chemFrame
[...
  inlet1 is [name of inlet],
  inlet1_comp is [chemicals components in inlet 1],
  inlet1_temp is [temperature of chemicals entering inlet1],
  inlet1_press is [pressure of chemicals entering inlet1],
  inlet1_mas is [mass fraction of the chemicals entering inlet1],
  ....
]);

88
5.3.3 Issue Base

The structure of an issue base is basically a node and link type representation with the links denoting the relationships between the nodes. Each node type is represented by a specific frame which is a sub-class of the generic frame node. The issue base frames consist of: issue, position, argument, comment, fact and decision. Each of these types of nodes has its own frame representation because they can have different links to other nodes. The links that are supported in IDIS are: response, supports, against, comment, decided, followUp, combinedWith, replaced, relatedTo and copyOf. A decision node, for example, can only be linked to an issue because decisions only represent solutions to issues. The decided link must be used to link these nodes as the structure of the issue base would not be correct if any other type of link was used.

The direction of the link between the nodes is also important because it provides a coherent structure of the deliberation that has taken place. A new node is always linked to a node in an existing issue base. In some cases, two nodes of the same type may be linked, for example an issue can be decomposed and replaced by two sub-issues. Without a record of the direction of the links it would not be possible to identify which issue was the original issue from the structure of the issue base alone. As this information is important the issue base frames have two different classes of links, those from the node and those to the node. The frame issue has the structure:

```plaintext
frame(issue isa node,
    links info [deadline, replaced, relatedTo, followUp],
    reverseLinks info [responseBy, decidedBy, replacedBy, followUpBy,
        relatedToBy, commentBy],
    viewpoint is "viewpoint",
    deadline is "undefined",
    replaced ref [],
    relatedTo ref [],
    followUp ref [],
    responseBy ref [],
    decidedBy ref [],
    replacedBy ref [],
    followUpBy ref [],
    relatedToBy ref [],
    commentBy ref []).
```

89
Two lists of links are stored in each frame, *links* and *reverseLinks*. The *links* list stores the list of all the possible links an issue can make to other nodes. The *reverseLinks* stores all the possible links that other nodes can make to the issue. In IDIS the convention used to distinguish the different type of links is the *reverseLinks* have the suffix *By*. Each of the slots records either the name of the nodes that the issue has been linked to or, the nodes that have been linked to the issue. The name of the slot denotes the type of link made between the nodes.

The additional two slots are *viewpoint* and *deadline*. All the issue base frames contain the *viewpoint* slot. This records the viewpoint where the node was created and by doing this IDIS is able to record which design it relates to. The *deadline* slot is specific to the issue frame and contains an optional date by when the issue should be resolved. If a date is provided IDIS can check it against the system date to see if the deadline has been missed or what issue should be resolved next.

When a new node is created an instance of the appropriate frame is created with the name of that node. The node names are numbered sequentially as the viewpoints and plant items are and IDIS keeps track of the current number for each node type. The frames only provide a representation of the nodes in a network and their relationships. However, each of these nodes represents part of some deliberation and this information, the contents of the nodes, must also be recorded.

The content of each node is stored in free text in a file. When a new node is created a file with the same name as the frame (issue1, position3, etc.) is created in the current project directory. The name and type of node, the designer who created it, the date it was created and a one line summary that was provided when the frame was created are added to this file. This file is then displayed using the Xemacs editor and designers add any additional information relating to the node. The combination of the issue base frames and the files provides both the structure as well as the content of the deliberation that has taken place.
5.3.4. Rule base

Design constraints in IDIS are represented as production rules and are implemented in Prolog. The main advantage of using Prolog is that it has a built in backward chaining mechanism. If a Prolog clause fails when it tries to instantiate variables and satisfy the specified conditions Prolog will automatically try to reinstantiate the variables with another value to satisfy the conditions. User defined operators such as if, then, and, or are used to provide a structure for the rules. By extending this set and adding operators: connectedWithoutIsolation, connectedUpStream, connectedDownStream, isa and hasValue rules can be written in the following format:

```
rule1 : if not X connectedWithoutIsolation Y and X isa relief and Y isa pressureVessel then write('Pressure vessel 'Y' is not protected by a relief mechanism').
```

This rule states that if we have a plant unit Y (a pressure vessel) and there is not a plant unit X such that X is a relief mechanism and is connected to Y without isolation then the warning message that Y is not protected by a relief mechanism will be displayed.

The inference engine that parses the rules is written in Prolog and performs the following actions. The rules are first split into a set of conditions and an action. The list of conditions is then split into individual conditions. The conditions for the above rule would be represented in the following way:

```
((not X connectedWithoutIsolation Y),(X isa relief),(Y isa pressureVessel)).
```

Each condition is then tested individually and if they all succeed the action is performed. All the actions in the rules are warning messages that are printed to a file. This file is displayed once all the constraints have been checked. Some of the operators, such as the ones used above, are represented by Prolog clauses that test the condition they represent. The isa operator calls a predicate that tries to match the value on the right hand side with the frame type of the unit on the left hand side.

The predicate that represents connectedWithoutIsolation traces the path between the two units X and Y and stores all the units between them in a list. Each unit in this list
is then tested to see if any of the units are of type \textit{valve}. This is achieved by calling the \texttt{isa} operator described above. If none of the units are a valve then \( X \) and \( Y \) are considered to be connected without isolation and the predicate succeeds.

Each viewpoint has its own unique file, \texttt{viewpoint#.rul}, that contains a list of the constraints that apply to the design at that point. When designers check a design, IDIS finds the path from the root node to the current design and loads all the rule bases associated with each of the viewpoints in the path. Each rule is then tested using the inference engine in the manner described above.

There are several different types of rules:

- general design principles,
- codes of practice and standards,
- specific design rules.

General design principles are simple rules that designers are taught or learn through experience. These rules are accepted procedures or methods within a particular design domain and across domains. These general principles may be violated within a design if the designer has some reason for doing so. Codes of practice and standards are rules and regulations that are laid down by companies, safety organisations, governments etc. and must be met by designers. The final set rules are user defined constraints that designers chooses to add to the design. User defined rules can be added, deleted or modified by designers and are stored at every viewpoint. The general design practices and codes of practice are stored in separate files, \texttt{design_principles.rul} and \texttt{standards.rul}, in the project directory. These files are loaded automatically when any design is checked.

5.4 User Interface and Component Integration

5.4.1 User Interface

Each viewpoint in the design space represents an alternative design that has been explored and is represented by a diagram. As stated above, IDIS uses AutoCAD, which is widely used in industry, to represent the design diagrams. This means that
IDIS provides a familiar front end environment for designers. All of the standard AutoCAD features are available and IDIS provides additional features that support the recording and accessing of other aspects of design information. The IDIS features are accessed via menus on the AutoCAD screen.

There are five main menus along the top of the AutoCAD screen and one down the right hand side. The five menu items are: Icons; IDIS; Management Info; Unit Info and Rules. This is shown in Figure 5.1.

Figure 5.1. The AutoCAD screen with IDIS menus

The Icon menus contains a selection of icons of the available design units used to create the detailed design diagrams. The IDIS menu contains five sub-menus: Nodes; Viewpoints; Sketches; File and Quit. The Nodes sub menu contains all the commands used to create, link, view and edit the issue base nodes. The commands under the Viewpoint sub-menu are used to create new viewpoints, change viewpoints and display a viewpoint's information. The sketches designers use to describe concepts or
explain ideas, are created, viewed and edited via the Sketches sub-menu. The Files sub-menu contains the general project and file handling commands such as loading a project, creating a new project, deleting a project and saving a project. The Quit sub-menu quits the IDIS program. The Management Info menu has all the commands used to manage a project and design team and has ten sub-menus. The functions they perform were discussed in Chapter 4. The Password option allows any designer to change their status to leader if they have the correct password.

The Unit Info menu provides designers with a means of accessing and changing the information relating to plant units in the design. View a units spec sheet and chemical info options allow designers to view and change a plant unit's parameters and chemical information. The remaining two sub-menus are used to export a semantic representation of the current design diagram for other design tools.

The menu on the right hand side of the AutoCAD screen contains the commands: New Unit, Connect, Renumber, Move, Remove, Comment, Redraw, Zoom and Print. The New Unit option allows designers to insert a new unit by selecting a unit from a list. Connect allows the designer to connect units. Renumber and move allows designers to renumber or move a unit. Remove is used to delete a unit from the diagram. Comment is used to add a comment to the design. Redraw, Zoom and Print allow designers to refresh the diagram, zoom in or out or print the screen. The AutoCAD drawing system is based on the original work of Schelkin (1992) who developed a method of linking predefined AutoCAD blocks to Prolog frames. This work was extended and allows IDIS to maintain an internal semantic representation of the AutoCAD diagram.

IDIS displays information by writing it to a file and opening it using the Xemacs editor. A new Xemacs window is created for every file displayed allowing designers to view several different pieces of information at the same time. All the features are accessed via menus and any information required is added via dialogue boxes or using the Xemacs editor. The Xemacs editor is itself a menu driven application.
5.4.2 Integrating the Components

This section describes the method by which the different components are integrated. The majority of IDIS has been developed in Prolog and most components are represented using a common frame system. AutoLisp code is used to provide an interface between AutoCAD and the IDIS Prolog program. The core component of the system is the viewpoint mechanism which records the design diagrams and the structure of the design space. As all the other information is stored in relation to the design space the component integration will be discussed in terms of how they are integrated with the viewpoint mechanism.

IDIS communicates with AutoCAD through two named UNIX pipes created when the system is started. AutoCAD reads from one of the named pipes, acadpipe and Prolog reads from the other, prologpipe. IDIS writes AutoLisp commands to the acadpipe to pass information to AutoCAD. AutoCAD writes Prolog predicates to the prologpipe to pass information to IDIS. The information recorded in IDIS can only be updated through AutoCAD. When a command is sent to IDIS, it performs the required action and then either displays information or passes it back to AutoCAD. When designers change viewpoints IDIS updates its representation of the current state of the design and then sends the command to AutoCAD to load the appropriate viewpoint diagram.

Each viewpoint is represented by a frame and an AutoCAD diagram. The diagram associated with a viewpoint has the same name as the frame and the path of the diagram file is explicitly recorded in the frame. This links the viewpoint to the AutoCAD design. The hierarchical viewpoint structure provides a record of the design space. The combination of diagrams and frames records the diagrams at the correct points in the design space.

The different aspects of the design are primarily linked through the viewpoint frame structure. The viewpoint is the focal point of the design and the other aspects that relate to the design are associated with it either through slots in the frames or by naming convention. The changes made to the design at a specific viewpoint are stored as a list in the viewpoint frame. These changes describe the differences between the
parent and the current designs. All the issue base nodes created in a viewpoint are also recorded in the viewpoint frame. The issue base frames themselves store the structure of the deliberation. The actual deliberation is recorded in text files that have the same name as the issue base frames. The design constraints specific to a viewpoint, are recorded in a file with the same name as the viewpoint thus linking the constraints to the appropriate design.

5.5 Exporting Design Information

IDIS can export a representation of the design diagram to a file that can be read by other design systems. To date IDIS only supports two systems, QUEEN (Chung 1993) and MEPPI (Pemberton 1993).

QUEEN (QUalitative Effect ENgine) allows designers to examine the effects of qualitative changes, (Leitch 1990), to a design. A sign directed graph representation, (Iri et al. 1979), is used in QUEEN to model the propagation effects of process variables in a plant. QUEEN allows designers to ask questions such as, what happens if we raise the pressure in this pipe. Most design systems cannot cope with concepts such as 'raise the pressure' and require specific numeric values to perform their calculations. QUEEN uses the same underlying frame representation as IDIS and has the ability to import a frame description of a design from a file. IDIS provides this input by writing the semantic model of the current design out to a file as a set of frames.

MEPPI was developed to help identify the source of waste materials produced during a process. With the aid of this system it is possible to identify the sources during the design process and possibly reduce or eliminate them. MEPPI requires information about both the structure of the design and the chemical content of plant items. As with QUEEN, it can import a frame representation of a design. IDIS produces a file for MEPPI by writing the chemical frame representation of the current design to a file.
By providing a facility to export the frame representation of the design diagram represented in IDIS, the information recorded in the system can be used by other design systems. This is important as no design package will be able to perform all the features required by designers.

5.6 Hardware and Software Requirements

IDIS uses several packages but the main two are the CAD package AutoCAD and the CProlog system. IDIS was developed using these two packages and they are essential for the system to run. Other software used includes, the Xemacs editor and Xfig a drawing package. Any other UNIX editor or drawing package can be used. Xemacs and Xfig are the default packages, both are easy to use and are public domain so have the advantage of being free.

A computer running the UNIX operating system is required to run IDIS, although IDIS can be run on a PC over the network using an Xterminal emulator. A full specification of the hardware and software requirements for IDIS and AutoCAD are listed in Appendix B. The user manual for IDIS is included in Appendix C.

5.7 Summary

This chapter presented the underlying frame and production rule representations used in IDIS and describes the implementation details for the various components. The techniques used to integrate the different components and the ability to export information to other systems were also described. The next chapter presents an example of how the system can be used.
Chapter 6

Using IDIS

Chapter 4 presented an overview of IDIS (Integrated Design Information System) and the implementation details were discussed in Chapter 5. This chapter describes how the system is used. This is illustrated by using some of the content of a case study which reconstructed a final year design project using IDIS.

A group of students were given a design project which they had to complete over a period of six weeks. A member of staff and an industrial supervisor, with experience of design, supervised the project. The students were asked to produce a design for an ethyl acetate plant and to design several of the larger plant items in detail. This case study is presented in more detail along with some of the lessons learned in the next chapter. This chapter uses some of the issues raised and diagrams created to demonstrate how the students may have used IDIS during the project. The first section describes the creation of a new project and the addition of new users to the system. The following two sections describe how the design team identified and discussed important issues and how they explored the design space based on the possible solutions. Sections 6.4 and 6.5 describe how the team add design diagrams as well as design constraints and check for violations of these constraints. The project and information management features provided by IDIS are described throughout the chapter. Screen shots are provided to illustrate the steps the users go through. In this project the users will be referred to as Bill, Sue and John.
6.1 Starting a New Project

The first task is to create a new project and add information about the design team members. IDIS is loaded by typing 'IDIS' at the command prompt. The user is presented with an AutoCAD screen that contains a single menu with several options: use AutoCAD without IDIS, Load an existing project, Create a new project or Quit. In this example Bill chooses to create a new project. When this option is selected IDIS asks for a project name. If a project with that name does not already exist a new directory is created in the IDIS project directories and it is set as the current working directory. An instance of the frame ProjectInfo is created and its slots are set accordingly. This frame stores the current number of issue base nodes (initially set to zero), the path where the project information is stored, the current viewpoint (initially set to viewpoint1) and the design team members who can access the design (initially set to the user who created the project). A UserInfo frame is also created for the user who created the design project.

Bill is given project leader status as he created the new project and this allows him to add new users to the project and to set their status. A new AutoCAD screen (viewpoint1) and the frame which represents this viewpoint is created. The current screen then changes to viewpoint1 which is a blank AutoCAD screen containing all the menus used to access IDIS. There are three people in the design team and Bill adds Sue and John to the design project as team members. To create these new users Bill adds the user names, their status in the project (leader, member or other) and sets their privileges. Sue and John are added to the design team as members and are allowed to view, create and link new nodes to the issue base. They are not allowed to create decisions as Bill decides he should be the person who makes the final decisions about the issues. Figure 6.1 shows Bill adding Sue as a project team member.
6.2 Creating a New Issue Base

The first stage of any design starts with the initial specification given to the design team as this outlines the aims as well as some of the constraints of the design. In this project the initial specification given to Bill is:

A market survey has identified a niche in the market for ethyl acetate. This is used in the production of inks, adhesives and lacquers. It is an effective solvent for many resins and used extensively as a cellulose nitrate solvent in the manufacture of leathers, inks, cements, photographic films and linoleum. The aim of the project is to design a plant that produces 15,000 tonnes of ethyl acetate (98% purity) p.a.

There are several different chemical processes that can be used to produce ethyl acetate, and before any design can be created, the design team has to decide which process route to use. Bill identifies this as the first problem which needs to be
resolved. He creates issue 1 to record the specification and the question about which process route to use. Figure 6.2 shows how Bill adds this issue to the design project.

Before Bill quits the session he adds two process routes as possible solutions which can be used to produce ethyl acetate. He also adds arguments for these positions. To link one node to another Bill selects the type of link he wants between the two nodes from the AutoCAD menu. IDIS then prompts him for the name of the nodes to be linked and indicates the type of nodes it expects. For example, with a response link it prompts from position and to issue. If the type of nodes added were incorrect IDIS would not link them and would return the error message: 'only positions and issues can be linked with a response link'. This ensures that the structure of the issue base that evolves is correct and meaningful. Figure 6.3 shows one of the positions and how he links it to issue 1.
Acid catalysed liquid phase esterification of ethanol and acetic acid. We could use this method to produce ethyl acetate by acid catalysed liquid phase esterification of ethanol and acetic acid. The ester is produced by the substitution of ionisable hydrogen in the carboxylic acid by an organic radical produced from the alcohol. The reaction does not proceed to completion but instead approaches an equilibrium.

IDIS lists one unseen issue, two positions and three arguments. Each of these nodes are displayed with their one line summary, the date and the name of the person who
created them (Figure 6.4). These nodes, listed in isolation, do not tell Sue very much about the structure of the issue base and the summary only gives her an idea of the content of the node.

Sue decides to view the full content of issue1 along with the structure of the issue base created. She chooses the issue base structure option from the IDIS Issue menu and is prompted for the name of the issue she wants to see. She enters the name issue1. IDIS displays the summary information of issue1 and all related nodes. Sue also chooses the view node option from the IDIS issue menu and enters the name issue1. This displays the full contents of the node issue1. This provides a more detailed picture of the information that has been recorded and shows how the information is related. The

Figure 6.4. Sue views all the nodes she has not seen
information that Sue is currently viewing is shown in Figure 6.5. After reading the positions and arguments added by Bill, Sue adds her own positions and arguments. The new nodes are created and linked to the issue base using the same commands used by Bill.

Figure 6.5. Sue views the contents of issue 1 and the structure of the issue base

Before leaving the project Sue decides to check who else is working on the project. She selects the list user info from the management info menu which lists all team members and their status (Figure 6.6). This shows Sue that there are three users Bill, John and herself. Bill has project leader status and she and John have been defined as team members. This means that only Bill can add, delete or change the properties of users and ask to see the contribution each has made to the design. Sue then saves the changes that she has made and quits the project.
When Bill returns to the project he checks the current state of the issue base by looking at the positions and arguments which have been added by other team members. He chooses the issue base structure option from the IDIS Issue menu as Sue did and adds the name issue1. The current structure of this issue base is displayed in Figure 6.7.

He reads the positions added by Sue, by viewing the full content of the nodes and adds further positions and arguments. He also notes that John has not added any information to the issue base. Bill is interested to see if John has seen the issue but has no information to add or whether he has not yet loaded the project. As project leader, Bill can check the last time a user accessed the project and view the contributions made by any one designer to the design as a whole or to a specific issue.
To do this Bill selects the option *users contribution* from the *management info* menu and enters John's name in the user name slot and the value *all* for nodes in the edit box presented. This lists any contribution that John has made to any of the nodes in the whole project and displays the date when he last worked on the project. IDIS checks the files and returns the answer *none* for contribution and *never* for the date he last worked on the project (Figure 6.8). The default setting for when a user last used the project is *never*. It is replaced by the current date every time a user loads the project to keep a record of the last time the user worked on it.

John subsequently gets involved in the design and the deliberation about what process route to take. The issue base evolves in the manner described above with Bill, Sue and
John adding positions, arguments, facts and comments. When all the discussion has taken place Bill decides that two possible process routes should be explored further: condensation of acetaldehyde by the Tischenko reaction; and acid catalysed liquid phase esterification of ethanol and acetic acid (positions 1 and 2). The decision is recorded by Bill and he raises another issue, issue2: “Should we use a batch or continuous process?” The deliberation for this issue takes place as it did for issue 1.

![Diagram](image)

Figure 6.8. Bill views John’s contribution to the project to date

### 6.3 Exploring The Design Space

Issue2 is resolved and the team decides to use a continuous process. At this point they decide to continue the design by exploring the two different process routes selected as possible solutions to issue1 (positions 1 and 2). Bill creates two new viewpoints, viewpoint2 and viewpoint3 to represent these. These are added as children to
viewpoint1. The new viewpoints are created by selecting the *new viewpoint* option under the *IDIS / Viewpoint* menu. Bill has the choice of selecting *alternative, decomposition* or *integration*. He wants to create two new alternative designs so he chooses the *alternative* option. IDIS then asks Bill for the name of the parent viewpoint, Bill enters viewpoint1. IDIS informs Bill that any diagram in viewpoint1 will be locked if a child viewpoint is created from it. He is given the option to continue or cancel the command at this point. He continues with the process and creates a new viewpoint. The new viewpoint is created and IDIS asks for a one line summary to describe it. Once this information is added the representation of the design space is updated. This process must be performed for each new viewpoint created. Figure 6.9 shows Bill creating the new viewpoint for the acid catalysed liquid phase esterification of ethanol and acetic acid route.

![Figure 6.9. Bill adds a new viewpoint to the design space.](image-url)
A viewpoint is locked when it is used to create a new viewpoint to ensure the consistency of the design space is maintained. The parent diagram is copied to the child viewpoint when it is created and any changes made to the diagram are recorded at the new viewpoint. If the users are allowed to change the original diagram the design would be inconsistent because the parent design would not be the one the children designs were created from. Up to this stage, no diagrams have been created and the two new viewpoints are represented by blank AutoCAD screens. This discussion now concentrates on viewpoint2, which explores the acid catalysed liquid phase esterification of ethanol and acetic acid route, as this is the option adopted by the design team.

The design team creates a schematic diagram of the process. This is achieved by using boxes to represent the different stages of the process: feed stock; reaction; purification; recovery; and waste processing. The diagram created for the acid catalysed liquid phase esterification of ethanol and acetic acid is displayed in Figure 6.10. The process by which the design diagrams are created is discussed in section 6.4.

On completion of the diagram the design team returns to the task of discussing the design issues to be addressed before the design can evolve any further. Three issues are discussed at this viewpoint. The most important of these is ‘what type of separation process to use?’.

Bill sets deadlines for the resolution of the issues when he enters them because the team has spent longer discussing the process routes in viewpoint1 than he would have preferred. The deadlines help to focus the discussion and give the team some ideas about the time scale they have to work to. While the deliberation of these issues takes place Bill monitors the input of Sue and John and checks that the deadlines for the issues have not been missed. Using the management features of IDIS, Bill checks which issue deadline is due next and ensures that Sue and John spend more time resolving this issue than other less pressing ones.
Figure 6.10. A schematic diagram of the acid catalysed liquid phase esterification of ethanol and acetic acid process.

The main issue that affects the structure of the design is the type of separation process used. Two types of separation are identified: liquid extraction and distillation. As with the process routes the design team decides to explore both these options. The other two issues discussed at this viewpoint are also resolved and a single solution for each is identified. Two new viewpoints are created from viewpoint2 to represent the two alternative separation processes: viewpoint4 represents distillation and viewpoint5 liquid extraction. This time when the new viewpoints are created they contain the completed schematic process diagrams from their parent viewpoint. The two new viewpoints, viewpoint4 and 5 inherit the diagram from viewpoint2. As with the process routes, the remainder of this discussion concentrates on the distillation option as this was the option adopted by the design team. At this point it is identical to viewpoint2, Figure 6.10, as no changes have been made to the diagram yet.
Viewpoint4 represents the new design being developed using distillation as a means of separation. The distillation method of separation involves feeding a liquid into a distillation column where it is heated by steam. The different components of the liquid separate at different temperatures and can then be removed at different points in the column. In some cases the different components have to be separated again as several different components may be removed at any one stage. Eventually by passing the outputs from one column to another, the output from the reaction process is separated into all of its required constituent parts.

One of the major decisions the design team needs to make at this stage is how many distillation columns are needed and what their functions will be. Issue6 raises the question, "How many distillation columns are needed?". Laboratory experiments and model simulations indicate that four distillation columns are required. The schematic diagram is changed accordingly to represent the new design with four different distillation columns (Figure 6.11).

Figure 6.11. Viewpoint 4 after it had been modified
The columns labelled A, B, C and D are used to:

A : split the output from the reactor into ethyl acetate containing impurities and water with unreacted ethanol;

B : take the ethyl acetate and impurities from A and split it into two streams, water containing light impurities and ethyl acetate containing heavy impurities;

C : take the ethyl acetate and heavy impurities from B and split it into ethyl acetate and heavy impurities;

D : take water and impurities from A and B and split it into waste water and recovered ethanol which is fed back into the feedstream.

Issue 6 is the only issue discussed at this viewpoint. By this stage the design team has made all the major decisions needed before the detailed design starts. The next stage involves the detailed design of the distillation columns and the related plant items. Up to now only schematic diagrams have been produced but the design team go on to produce a flow sheet diagram.

Sue has been away on holiday and returns to the design project after a break of a week. To find out what new issues have been discussed she lists all the issue base nodes that she has not seen. This was discussed in Section 6.1. However, this does not tell her anything about the changes that have been made to the design diagrams. She sees that the items she created before she left have been connected but does not know if any other changes have been made to them. To find out how the diagram had changed she chooses the show changes option under the IDIS viewpoint menu. IDIS displays a file which lists all the changes made to the diagram at viewpoint4 along with the date they were made and the name of the person who made the changes. This information shows Sue that Bill has connected the items but that no other changes have been made to the design. Figure 6.12. shows the output displayed by IDIS. The changes made are displayed in reverse order with the most recent changes at the top of the file.
Listing all the changes that have been made to the diagram in viewpoint4 tells Sue what modifications have been made to this diagram. However, she wants to know what the differences are between the diagrams in viewpoint4 and viewpoint5. Changes such as the addition or deletion of plant items may be identified by viewing the two AutoCAD diagrams. Changes in the parameters of a common item in both diagrams are not readily apparent by simply viewing the two diagrams. She can list all the changes for both diagrams as she has just done for viewpoint4 and look for differences between common items. This would be a difficult and time-consuming process as there can easily be over one hundred changes many of which may not be relevant. She selects the view differences option from the AutoCAD menu and is prompted for the names of two viewpoints. She enters viewpoint4 and viewpoint5 and IDIS uses the change list information to identify how these two designs differ. All of the differences between the two designs are displayed in a single file. This feature allows Sue to identify the differences between two designs quickly and easily.
6.4 Creating Detailed Design Diagrams

Once the diagram and deliberation in viewpoint4 are completed Bill creates a new viewpoint where the team start to represent the flow sheet diagram. Viewpoint6 is created as a child of viewpoint4 and represents the flowsheet diagram. Bill deletes the original schematic diagram by selecting the delete item command from the AutoCAD menu and clicking on the items he wants to delete. In this case, he removes all the items. New items are added by selecting the appropriate item from one of two AutoCAD menus. One menu provides a list of all units available and items are selected by clicking on the unit name. The other menu provides a list of items along with the icons used to represent them, these items are selected by clicking on the icon. In both cases Bill is presented with an icon that represents the plant item and he places it in the required location on the AutoCAD diagram.

After adding several items to the design, Bill wants to start connecting them. The items are connected by selecting the type of link required, either a pipe or a signal line and then clicking on the outlet and inlet of the two items to be connected. A line is automatically drawn between the selected inlet and outlet to represent the connection. The type of connection selected between the inlet and outlet is checked by IDIS to ensure that the correct type of connection is made. A pipe can only be connected between an inlet and outlet and a signal line can only be connected between a signal line in and signal line out. Each connection must contain an inlet and outlet port: two inlet or two outlet ports cannot be connected. If Bill violates these rules the connection will not be made and he will receive a warning message. Figure 6.13 shows the diagram that was completed in viewpoint6.

All the issues that have been discussed so far, have been high level issues which have related to the whole design. Once the designers start creating the detailed design diagrams, issues are raised that relate to the specific plant units. Bill, Sue and John need to make decisions about the many different physical design parameters that must be set for each of the design items. For example, a pressure vessel will have a design temperature and pressure, operating temperature and pressure, material of construction, thickness of the vessel walls, its volume etc.
One of the issues being discussed relates to the specifications for the four different distillation columns. John raises an issue asking what the specifications for the distillation columns should be. Sue points out that they are in fact discussing four different issues as the specification for each column is a separate issue. Four new issues replace the original issue. To replace an issue with other issues, Sue creates the new issues as normal and then links them to the original issue with a replaced link. This indicates that these issues replace the original issue. Bill adds a decision node to the original issue stating that the issue has been split into four separate issues. These issues are resolved using ASPEN (Aspen Technology Inc.), a process design and simulation package commonly used in the process industry. The ASPEN output files are attached to the positions to support the specifications selected. Once the team has the required information they set the parameters for the distillation columns (Figure 6.14).
Figure 6.14. Sue sets the parameters for one of the distillation columns

Sue selects the **unit specification** option from the **unit** menu and is asked to click on the item she wishes to edit. She clicks on the distillation column icon and an edit box which contains a list of the parameters that can be set is displayed (Figure 6.14). Some of the information about the unit is set automatically by IDIS and cannot be altered. This includes the date it was created, what project it was created in, who created it and the connections made to and from it. Sue changes the parameters and closes the edit box. When the edit box is closed any parameters that have been changed are recorded by IDIS. IDIS then stores the changes made, who made them and the date. The semantic model of the design held by IDIS is also updated with the changes. For some parameters the design team discuss the options before setting the values for the units. This is done using the IBIS component as they had done previously.

Sue also wants to record information about the chemicals entering and leaving the distillation column. The chemical information about the different inlet and outlet streams of each plant item is recorded in a separate chemical information frame in
IDIS. Sue selects the chemical information option from the AutoCAD unit information menu and clicks on the distillation column she is working on. An edit box similar to the previous one is presented. This edit box contains slots for each inlet and outlet stream where Sue can record the chemicals and the proportions in which they are present.

John wants to review the deliberation that took place about distillation column B, as he is going to undertake the detailed design of that column. He uses the management functions to search for all nodes where column B was discussed. Selecting the search option under management info menu he is presented with an edit box that asks him to enter a keyword and select a viewpoint. He enters Column B and clicks the OK button. IDIS searches all the issue bases in that viewpoint and returns a list of all the nodes where the term 'Column B' appeared. John reads all this information before he produces the detailed design of Column B.

6.5 Adding and Checking Design Constraints

Throughout the design of the plant the designers have to meet certain design constraints. Some design constraints are set in the initial specification, the plant must produce 15,000 tonnes of ethyl acetate a year and the purity must be 98%. Other constraints which apply are associated with legislation regarding specific types of design or plant items. The design team also add their own constraints to the design. To record this information IDIS has a frame that represents the project and stores general information. This includes: the tonnage and the purity of the output; cost; location and the size of the plant. This feature allows the designers to add constraints about project as a whole.

While John is designing heater2 he does so according to a specification sufficient to meet the requirements of the current design. If the design is changed, i.e. the process modified which increases the temperature or pressure, the heater would need to be re-designed. John wants to add a constraint which states that if the inlet pressure of
heater2 is greater than 24 Bar, the user is to be warned the heater has exceeded its design pressure limit. The *add rule* option under the *rules* menu is used to add constraints to the design. John selects this option and is presented with the edit box shown in Figure 6.15.

![Figure 6.15](image)

Figure 6.15. John loads the rule template in order to create a new rule.

The constraints are recorded as IF ... THEN ... rules and have the structure ‘rule name : if condition1 and/or condition2 then action’. The action is always a warning message which is displayed if the conditions of the rule are satisfied. Both the conditions that can be tested and the logical connections which can be made between the conditions are provided in a pre-defined list in the edit box. A list of the available connections are shown in Table 6.1.

When creating the new rule John adds the parameters he wants to check into the blank spaces in the edit box as well as the values they are to be tested against. The logical connections and test conditions for these parameters are selected from pull down
menus. Once all the conditions of the rule are added he selects the word ‘then’ from the menu to indicate the antecedents of the rule have been completed. Finally, he adds the warning message to be displayed if the rule fires. The completed rule is shown in Figure 6.15.

Table 6.1 Possible conditions and connection available in IDIS

<table>
<thead>
<tr>
<th>Conditions</th>
<th>Connections</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;</td>
<td>and</td>
</tr>
<tr>
<td>&gt;=</td>
<td>and not</td>
</tr>
<tr>
<td>&lt;</td>
<td>or</td>
</tr>
<tr>
<td>&lt;=</td>
<td>or not</td>
</tr>
<tr>
<td>=</td>
<td>then</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td></td>
</tr>
<tr>
<td>connected</td>
<td></td>
</tr>
<tr>
<td>directly_connected</td>
<td></td>
</tr>
<tr>
<td>connected_upstream</td>
<td></td>
</tr>
<tr>
<td>connected_downstream</td>
<td></td>
</tr>
<tr>
<td>connected_without_isolation</td>
<td></td>
</tr>
<tr>
<td>isa</td>
<td></td>
</tr>
</tbody>
</table>

As the rule applies to a specific plant item in viewpoint6, it is stored at that viewpoint. All the constraints added to the design in this viewpoint are stored in the file viewpoint6.rls. The design is not automatically checked for violation of the constraints at this point rather, IDIS must be explicitly issued the command check design to check the current design.

Bill wants to check the design to see if any constraints have been violated when he was working on the project. On selecting the check design option from the rules menu IDIS loads the general design practice rules and all the rules which apply to the current design. It then checks the design for violation of any of these rules and displays any error messages. Bill views the error messages and decides he wants to view all the constraints that have been added to viewpoint6. This allows him to see the full content of the constraints that have produced the error messages. Figure 6.16. shows Bill viewing the resulting messages and the list of the constraints checked.
Figure 6.16. Bill views the constraints that have been recorded and the results of the check on the design.

A considerable amount of effort has been invested in the project and Bill is interested in reviewing the contributions of each team member. He uses the users contribution option under the management info menu to list the contributions of both Sue and John. From the results he sees that Sue has created more issues than John but that John has contributed more than Sue during the more detailed design phase. This facility is useful in providing information Bill may need when writing the project reports.

6.6 Exporting the Design Information to Other Packages

IDIS also allows users to write information out to a file which can be read by other applications. As mentioned in earlier chapters, it currently supports QUEEN (Chung 1993) and MEPPPI (Pemberton 1993). By selecting the create prolog file or create
meppi file options under the unit menu an output file is produced containing a frame representation of the design at the current viewpoint. The prolog file contains a representation of the plant units and their relationships and can be used by QUEEN to provide a qualitative analysis of the design. The MEPPi file contains the information recorded in the chemical frames and provides a record of the plant units, their relationships and the chemicals present in the different plant units. This information is used by MEPPi to check where by-products and undesirable chemicals are created and can help the designer to remove or reduce these during the early stages of design.

6.7 Summary

This chapter provides a description of how IDIS is intended to be used. This was achieved by reconstructing a design project. Some of the scenarios described are real while others were created to demonstrate how designers may use IDIS. The final structure of the design space with the issues and constraints is shown in Figure 6.17.

Figure 6.17. The final structure of the design space in the Ethyl Acetate Project
This case study demonstrates how IDIS can effectively integrate the various types of design information in a single environment. In this example, the various aspects of DR recorded were linked with the design diagrams. The management features described throughout the chapter allow designers to manage both the design information as well as the project.

In the following chapter the case studies conducted during the development of IDIS are described. The lessons learned from these studies and the subsequent modifications to the system are also discussed.
Chapter 7

Evaluation of IDIS

This chapter presents four case studies which have been conducted in order to evaluate IDIS during its development. Each case study logically follows on from the preceding one and tests a different aspect of the system. The first case study tested the representation used to support and capture design deliberation. The second, tested the system's ability to retrospectively represent information currently recorded in a design project. The third case study was used to test the system's ability to record design information off line as it is generated during a design project. The final case study tested IDIS's ability to support the design process in an industrial setting and capture information on line as it is generated. This case study also tested the system's ability to support multiple users. Each case study is presented separately below. A full listing of the issue base nodes for the first two case studies are presented in Appendices D and E. The content of the final two case studies contains commercially sensitive material so cannot be included.

7.1 Issue Base Case Study

In this initial case study the issue base component of the system was used independently to explore and record the alternatives of a design issue which arose during the modification of a process plant. This example was developed with the aid of a chemical engineer who developed the positions and arguments related to the issue. This case study highlighted some limitations of the representation and possible
improvements are discussed. The modifications consist of new links which are necessary for representing real engineering problems. This case study emphasises the need for changes to be carried out in a disciplined way. The issue was:

A chemical company wants to build a new plant in the USA. There is an existing plant for the same tonnage and product in the UK which uses both recycled cooling water (RCW) at 21 °C and chilled water at 5 °C. There are heavy demands on the cooling. The atmospheric ambient and wet bulb temperatures are higher at the proposed USA site by 5 °C and 8 °C respectively on average. Are any modifications needed to the plant items and/or cooling supply systems?

Five possible solutions were initially identified for this issue and these were added to the issue base as position nodes. Arguments for and against these positions were also added and linked to the positions to create the issue base shown in Figure 7.1.

![Figure 7.1. The initial issue base](image)

<table>
<thead>
<tr>
<th>Key to the links</th>
<th>Summary of the major nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response Link</td>
<td>ISSUE 1 — Is the cooling adequate?</td>
</tr>
<tr>
<td>Supporting link</td>
<td>POS 1 — Transfer more duties to chilled water</td>
</tr>
<tr>
<td>Against link</td>
<td>POS 2 — Redesign plant items where practicable</td>
</tr>
<tr>
<td>POS # Position</td>
<td>POS 3 — Redesign the cooling tower to obtain a lower RCW temp.</td>
</tr>
<tr>
<td>A # Argument</td>
<td>POS 4 — Redesign the cooling supply system and add parallel plant items.</td>
</tr>
<tr>
<td></td>
<td>POS 5 — Reschedule the plant operations</td>
</tr>
</tbody>
</table>
At this point a further investigation of position 3 (re-design the cooling tower to obtain a lower RCW temperature) established that this position in itself would not be sufficient to overcome the cooling problem. Even a re-sized cooling tower could not lower the RCW to the required temperature because the atmospheric temperature would be too high. However, it was realised that this would be a beneficial change to make as it would reduce the number of changes needed as some plant processes may not need to be redesigned if the temperature of the RCW is lowered. Position 3 only provides a partial solution to the problem and the representation as it stood could not support a position that represents a partial solution. Position 3 could be combined with position 2 to provide a complete solution to the issue.

The representation was extended and provided a new link, a combinedWith link, that allowed the user to combine position 3 with position 2. The response link from position 3 to issue 1 was removed and a combinedWith link added between positions 2 and 3 (Figure 7.2).

This issue base was further expanded with two more positions (positions 6 and 7) and their corresponding arguments added. Position 3 was also combined with position 1 as lowering the RCW temperature would reduce the number of duties that would need to be switched to chilled water. It was noted at this point that position 3 could be combined with every other position and would be a beneficial change to make no matter what other changes were made. Instead of combining position 3 to all the other positions it was decided that it should be adopted as part of the solution. Another issue was created (issue 2) that stated the problem highlighted in issue 1 but also added the
cooling tower will be re-sized to reduce the RCW as much as possible. This also asked what other changes would need to be made. This new issue replaced the original so it was linked to it via a replaced link. All of the existing positions were still valid solutions for the new issue and to indicate this all the positions were linked to issue2. This resulted in the issue base represented in Figure 7.3.

There is however a problem with this issue base. It occurs because position3 is combinedWith positions 1 and 2 which are linked to issue2. This is incorrect because position3 has already been included as a partial solution to the problem in issue2. A change to an issue base should never violate its integrity. One solution may be to remove the combinedWith link from position3 to positions 1 and 2 (Figure 7.4). However, the fact that position3 had originally been combined with positions 1 and 2 for issue1 would have been lost. Position3 would also be a 'free floating' position not attached to any node and hence not attached to the issue base. It is important that not to lose the information relating to the initial issue. When a change to an issue base causes a loss of integrity, or the structure of the issue base becomes incorrect, we call this a violation of temporal integrity of the issue base. The corollary is that by conserving the temporal integrity of an issue base it is possible to examine the issue base structure to see how it evolved.
A solution would have been to duplicate positions 1 and 2, and give them new names and then link them to the new issue using response links. Although this is a possible solution it is not satisfactory because the content of the new positions is identical to the original positions 1 and 2 and this information should be made explicit. The problem was solved by creating a new type of link, a copyOf link. Copies of positions 1 and 2 were made and renamed position1a and position2a. These new nodes were linked to issue2 with response links and linked to positions 1 and 2 with a copyOf link to represent that they were a copy of these nodes. The response links from positions 1 and 2 to issue2 were removed as these violated the integrity of the issue base. This solution preserves the information and temporal integrity and represents the situation correctly for both issues (Figure 7.5). Positions 4 to 7 had not originally been combined with position3 so they could be linked directly to both issues.
Another example of the violation of temporal integrity occurs when additional information is added to an issue base after the issue has been resolved. Designers often want to re-examine an issue when possible solutions that were not explored become apparent. If a new position was added directly to such an issue base it would violate the temporal integrity (i.e. it is not clear that this new position had been considered after the original decision was made). To allow designers to re-examine an issue base and add new nodes once they have made a decision, a new link was created, the followUp link. When an issue base is re-opened the user adds more information by creating a new issue node stating the problem (e.g. Not all the possible solutions were considered when the original issue was resolved. Is there possibly a better solution?). This issue is then linked to the original issue via a followUp link and is explored the
same as any other issue (Figure 7.6). This solution preserves the temporal integrity of the original issue base, captures the new information and makes it explicit that it was added at a later date.

![Diagram](image)

**Figure 7.6. The use of the FollowUp link**

To ensure that designers do not accidentally violate the temporal integrity in this way IDIS does not allow them to add information directly to an issue base when the issue has a decision node linked to it. If they attempt to do this they are notified that a follow up link should be used to add more information to the issue base.

As well as new links that preserve the temporal integrity, another link has been added to support the way designers work. Designers often decompose a design problem into smaller problems, solve these independently and then integrate the solutions. They may also want to replace an issue with another issue as in the above example. IDIS supports this by providing a *replaced* link. If designers think that one issue consists of two or more issues, they can create new issue nodes and link them to the original node with a *replaced* link. These new issues can then be addressed independently and the solutions combined at a later date. The same link can be used to replace two or more issues by a single issue if it is decided that they are in fact addressing the same problem. This *replaced* link is also used by designers if they want to replace a decision by a new decision after re-examining an issue base as described above.

Issue bases cannot be changed in an arbitrary way. The links between the nodes are semantic links and represent the structure of the argumentation so it is important that the correct links are made between the correct nodes. The IBIS component is intended to support and capture the design deliberation so designers should not be allowed to corrupt or lose this information by adding or deleting nodes and links to the issue base.
in an undisciplined manner. IDIS checks designers’ actions and does not allow the creation of the illegal links described above.

The engineer found this tool useful. By using the semi-formal representation it helped him to identify some positions that he might not have considered. The IBIS representation also helped to make the structure of the argumentation clearer. The engineer found it useful to have this information recorded.

This case study tested the suitability of the IBIS representation for recording the issues discussed during the design process and helped identify limitations. The representation was extended to provide solutions for the limitations identified. After the success of this case study the next logical step was to test the system’s ability to represent the design information that is currently recorded in a design project.

7.2 Reconstruction Case Study

In the second case study IDIS was used to reconstruct a completed design project. The aim was to test the ability of IDIS to record design information that is currently produced in a design project. It was also hoped to identify some of the limitations with existing design documentation procedures and to show how a system such as IDIS can help overcome some of these limitations.

In this case study, a group of final year students conducted a project to design a process plant over a six week period. It was supervised by a member of staff and an industrial supervisor who had design experience. The design of a process plant goes through several stages, many of which overlap. However, they can be split into six main categories:

1. conceptual design,
2. production of a flowsheet and mass/energy balance,
3. detailed design of the plant equipment,
4. the operating procedure of the plant,
5. a hazard and operability study of the plant items (HAZOP),
6. commissioning.
Due to time constraints the students do not go through this full procedure. They work up to stage 3 producing a process and instrumentation diagram (P&ID) and a detailed design of several plant items. Limited operating procedures are produced for some sections of the plant. A HAZOP study, (Crowl and Louvar 1990), is then conducted on one of the plant items.

The project chosen for reconstruction was an award winning final year design project. The industrial supervisor helped clarify and identify missing information in the design documentation and to represent this information along with the designs in IDIS. This project was discussed in Chapter 6 when some of the material was used to illustrate how IDIS can be used. The specification given was:

A market survey has identified a niche in the market for ethyl acetate. This is used in the production of inks, adhesives and lacquers. It is an effective solvent for many resins and used extensively as a cellulose nitrate solvent in the manufacture of leathers, inks, cements, photographic films and linoleum. The aim of the project is to design a plant that produces 15,000 tonnes of ethyl acetate (98% purity) p.a.

When the students were given this initial design specification, they were asked to identify all feasible chemical routes for the process and to highlight the advantages and disadvantages of each. There was no formal methodology or tool available to help with this task.

The IBIS representation used by IDIS to support and record the design deliberation, is ideally suited to recording such information. This was the only part of the design where the students explicitly recorded the alternatives explored and the reasons why they had been accepted or rejected. During this stage the students identified eight different routes and produced twenty one different arguments for and against these routes. The information was recorded easily using the IBIS component and was more readily accessible in this format than it had been in the final project report.

Although the design project was well documented, the reasons for exploring, adopting and rejecting options were not explicitly recorded in most cases. The industrial
supervisor helped to identify the alternatives that may have been explored and provided the reasons for their choices. On several occasions the rationale behind the students’ decisions was not clear and the supervisor consulted other designs and text books in order to provide an explanation. This highlighted some of the problems with existing documentation as even an expert in the area, who had been involved in the project, could not provide an immediate explanation for some of the design choices.

Given this situation, what chance would a future designer who may want to modify or re-use the design have to understand the design fully.

This case study reconstructed the design up to the flowsheet diagram stage and recorded sixteen issues and twenty three positions in six different viewpoints. Five design diagrams were also represented in the system and some of the parameters of the plant items were recorded by the system. During some stages of the design, the parameters used were not discussed, but were calculated using computer programs such as ASPEN (Aspen Technology Inc.), a commonly used flow sheet package. These computing packages were used to calculate: the size of the column, the number of plates required, etc. Although no discussion took place regarding these design parameters it is still important to record the source of such design decisions. The output from ASPEN was recorded in IDIS as fact nodes which were linked to positions and provided justification for the design parameters used.

This case study demonstrated how IDIS could represent the different types of design information currently recorded. It also highlighted the limitations with existing documentation procedures as not every design decision was easily understood, despite the fact that the documentation was of a very high standard. IDIS makes it easier to identify gaps in this information and provides a coherent framework for capturing different types of information. By using such a system designers can easily record the missing information and access as and when necessary. Unlike the first case study, both the AutoCAD and the design deliberation components were used during the design. This demonstrated how the deliberation relating to the design is integrated with the design diagrams. All the designs diagrams that were created during this project were represented in AutoCAD.
One criticism of the IDIS representation was that it was not able to record decisions where no deliberation had taken place, McCall (1991). It is important to record the information used to make a decision even if no alternatives were explored and no deliberation took place. Without this information designers will not be able to understand the rationale for the decisions. IDIS can record this information by including only one position, its supporting facts, arguments and the decision. Within this case study, this situation arose when the students were discussing the specifications for the distillation columns. Only one specification was generated for each of the columns and no deliberation took place. The students used ASPEN, a process simulation tool, to generate the specifications for the distillation columns. The specification was added as a single position node to the issue. The output files produced by ASPEN were attached to the position as a fact nodes. This feature allows designers to record decisions and the rationale upon which they are based even if no deliberation took place.

This example also highlights how information from other design packages can be represented in IDIS. The fact nodes which can be linked to a position can contain any type of information. Output files from other packages and input parameters can simply be attached to the appropriate position. This may arise when only one position is explored as in the above example or there may be several positions with support from different packages or the same package with different output files. The above example demonstrates that IDIS can record both issues that have been resolved without exploring alternatives and information from other design packages.

The first two case studies were reconstructed retrospectively and the next logical step was to test the system’s ability to capture design information as it is generated.
7.3 Off-Line Case Study

In the third case study IDIS was used to record design information during a design project as it was produced. A research project investigating novel designs of new gas domestic appliance was used. This project was conducted by a team of three researchers at the British Gas, Gas Research Centre. Project meetings were attended and notes taken. These notes, minutes from meetings and other documents relating to the design, were entered into the system at regular intervals. One of the members of the design team assisted in recording the information in IDIS.

A considerable amount of information was recorded in the system. Twenty three issues, 65 positions and 86 arguments were recorded. The designers also produced 10 sketches and 6 detailed design diagrams. Two of the most promising techniques for the appliance were investigated further and issues relating to the type of material, positioning problems as well as aspects of control were discussed. Design diagrams were produced for the two most promising techniques and prototypes were built and tested.

A limitation of IDIS was identified early on in this study. The documentation that described some of the burners contained sketches demonstrating how they worked. As IDIS stood, it could only represent detailed design diagrams using AutoCAD. These sketches were not part of the detailed diagrams but were in fact part of the deliberation process. IDIS needed some means of representing such sketches as they are often used by designers. Xfig, a public domain graphics package, was interfaced with IDIS and used to represent the sketches produced by designers in some of the documents. This enable the sketches to be viewed along with the issue or position that they refer to.

Output from IDIS provided feedback to the design team at subsequent meetings. This output mainly consisted of a diagrammatic representation of the issues recorded in the system with their positions and arguments.
At several points during the project, when the information was being recorded in IDIS, it was found that the documentation was incomplete and information had not been recorded. In some cases the rationale for decisions which had been made, were not recorded in the project documentation. When these gaps were identified the members of the design team provided the rationale for the decision which was then added to the system. In most cases the designers remembered why they had made particular decisions and were surprised that they had not been recorded. In some cases the arguments for and against certain options were also missing. Again the designers could explain the advantages and disadvantages of the options when asked. By recording the information in a semi-formal structure in IDIS, it was easier to identify missing information. With existing design documentation techniques it is often not easy to identify gaps in information when it is represented in a textual form and spread throughout several documents. The missing information identified by using IDIS may only have come to light when a designer wanted to re-use or modify the design in the future. By then it may have been too late to collect this information.

The design team found the feedback from IDIS useful. When the information was represented in an issue base structure, the options that had been explored and the arguments for and against them could be clearly identified. For example one of the techniques that had been selected had only one argument for it and four against it. The designers were unaware of this until it was presented in the IBIS representation. The advantage of this particular option was one of the most important factors for the new appliance. The disadvantages included: controllability; the material was brittle; and it may have deteriorated with long term use. Although the designers had discussed this position and identified its advantages and disadvantages they were still surprised to see how biased the arguments were against the position.

This example highlights the fact that some features are more important than others and an option with a critical feature for success may be chosen even though there are many arguments against it. It is not just the number of arguments that are important in choosing a solution, but rather the content of the arguments as well as the features that the particular option provides.
The designers found that by recording the information in IDIS they had a clearer picture of what had been discussed and why particular options had been accepted or rejected. As the information recorded in IDIS came from several different sources, the designers found it easier to access all the information and to see the various relationships. Integrating the different sources of information in a semi structured representation also helped to identify missing information.

This case study identified a limitation of IDIS: it could not represent sketches often used by designers. To overcome this, a graphics package was integrated with IDIS.

Up to this point IDIS had only been used by a limited number of people: the author; those who helped to reconstruct the information in the first two case studies; and the designer who recorded the information in the third case study. The next stage of testing involved IDIS being used in an industrial setting with multiple users.

7.4 On-Line Case Study With Multiple Users

In the fourth case study IDIS was used to support the development of a new methodology and risk assessment engineering system. This project was conducted at the British Gas, Gas Research Centre by two researchers employed there and an MSc student on placement. The project lasted for a period of three months. IDIS was installed at the Gas Research Centre and the three members of the team used the system to discuss various aspects of the project. Although IDIS supports the collaborative nature of design by allowing several designers to work on a single project, it does not support concurrent access to projects. As a result only one designer can work on a project at any one time. This did not cause problems for this case study as the members of the team were working on several projects and did not require continual concurrent access to the project through IDIS. The author was not involved in the project although he was on hand from time to time to answer any queries that arose and to make any necessary modifications to IDIS.
The main aims of the project were:

- identify the different risks that may be present in the system they were working on;
- develop a methodology for assessing the risk of a given system;
- develop a computer system to assess the risk of a system based on this methodology.

The team decided that all three members would have leader status and would be able to perform all the functions including making decisions relating to the issues. During the project, the team used the issue base component to raise and discuss the questions relating to the project and they also used the graphics package to draw sketches which related to the deliberation. The AutoCAD component of the system was not used as the team were not producing detailed design diagrams. The users raised and discussed issues through the system and also had regular project meetings. At these meetings the deliberation that had taken place using IDIS was often discussed and sometimes used to resolve issues. Any information generated was subsequently added to the system. In some cases the issues were raised, discussed and resolved solely using IDIS.

The feedback from the users was encouraging. They commented that the issue management features were especially useful. The main ones used were, listing unseen nodes and checking when the issue base had last been changed. Features such as checking a user's contribution, were not used and did not appear to be needed for this project. However, they did see this feature being useful, especially in a larger project which involved more people and had tighter deadlines. The users found the structure of the information in the issue base to be clear and this helped them to see what had been discussed as well as the advantages and disadvantages of the different options. They commented that IDIS was easy to use once they became familiar with the system. They appreciated the fact that they could communicate through the system as not all the members of the team were always available.

IDIS was modified during the project to resolve some of the problems identified while the users were using the system. A common error that arose when the users first started IDIS was that they would create new nodes but forget to link these to the issue base. They commented that it would be useful to have some sort of prompt to remind
them to link the nodes. IDIS was changed and it now checks the issue base for any unlinked nodes when the user creates a new node. If any unconnected nodes are found the user is informed and are asked if they wish to proceed and create a new node or link the existing one first.

A further problem was that initially the users were unsure of which way the links should go. The rule is that new nodes should always be linked to an existing node in the issue base. However, some of the users found they were not sure of which names to type into the slots in the create link dialogue box. The original default names were from_node and to_node. Again IDIS was changed slightly to provide more meaningful default names in the dialogue boxes when a user links two nodes. When creating a response link from a position to an issue for example the system now displays from_position and to_issue instead of from_node and to_node. The other dialogue boxes for other types of links were changed accordingly.

All the users said that they would prefer the output to display the name of the person who created the node and the date it was created. This was in addition to the summary and relationships with other nodes. IDIS was modified and this information is also displayed when the users view the structure of an issue base.

The users also asked for some sort of key word search. Listing unseen nodes was useful but the users sometimes wanted to look back through the issues that had been discussed to see if anyone had mentioned a particular type of risk. There was no easy way to do this apart from viewing all the nodes and their summaries. A feature was added to the system that allowed users to search all the issue bases for a given key word. A user simply types in the word they are interested in and IDIS returns a list of all the nodes where that word appears.

When this design project was completed, the MSc student had to write his dissertation based on the project. He found that IDIS helped him to do this as the information he needed was recorded in the system in an accessible format. The other members of the
team also commented that it was useful for them to have the information recorded online if they wanted to re-examine the project at a later date.

7.5 Summary

The four different case studies presented above helped identify some of the limitations and to highlight the useful features of IDIS. The feedback from all the users involved in the trials was encouraging. They all thought it was important to record the rationale behind design decisions and found the representation used by IDIS easy to understand and use.

IDIS was modified both during and after each trial to overcome limitations identified. The concept of the temporal integrity of an issue base was an important aspect of the first case study. This aspect of Issue Base Information Systems had not been previously addressed by other researchers. The second case study identified some of the limitations with existing design documentation procedures and demonstrated how IDIS could provide an integrated framework for overcoming some of these limitations. The third case study showed how IDIS could be used on an industrial project. This project very quickly demonstrated the importance of recording sketches and this feature was added. This case study also showed how information from various different sources could be recorded in one system. The final case study was a live test of IDIS in an industrial setting with multiple users. Minor modifications were made to IDIS during this study to make the system easier to use and information easier to access.

More case studies should be conducted to test other aspects of IDIS. For example, the performance of the rule base component in an industrial setting should be fully tested. Aspects of particular importance are:

- how easy this component is to use;
- how useful the information is to the designers;
- how appropriate are the different files for recording different types of constraint.
The next stage in the development of IDIS is to test it over a longer period of time and with a larger team of designers who would stretch the project management facilities. Being able to monitor how well IDIS performs when designers have tight deadlines to meet and have little time available to invest in learning a new system is an exciting prospect. The findings from the initial tests are encouraging and suggest that IDIS could be of use to designers as it is able to capture several aspects of DR, integrates this information with the design artefacts and enables designers to easily access the information they need.
Chapter 8

Conclusions

Design rationale is the reason a course of action is taken during the design of an artefact. It is not an entity in its own right but consists of various aspects of information which goes into producing a design. As much of this information comes from disparate sources it is often difficult to capture DR within a single design environment. The capture of the DR should be unobtrusive and should not produce any additional work for designers. It must also be able to support the collaborative nature of design and any tool should be compatible with existing CAD packages.

Previous work in the area concentrated on recording the design deliberation which is one of the richest sources of DR. More recent work has addressed the problem of recording other aspects of DR such as the alternatives explored, modifications, functionality of the components and the requirements of the design. However the tools developed to support these various aspects of DR have provided only limited support in integrating them. To provide a more complete representation, DR capture tools should record and integrate these different aspects in a single environment. The problem of developing such a DR capture tool was addressed in this thesis.

An Integrated Design Information System (IDIS) was developed to bridge the gap between existing tools and the needs of designers. IDIS provides a framework which supports the design process and allows designers to record the DR unobtrusively. The different aspects of DR captured are: deliberation, alternatives explored,
modifications, constraints, functionality and design diagrams and sketches. IDIS is integrated with a popular CAD tool, AutoCAD. This enables it to record the design diagrams and provides support for traditional design tasks. The features contained in IDIS are accessed through menus in the AutoCAD screen.

8.1 Contributions of IDIS

8.1.1 Integration
IDIS is not unique in the fact that it records DR but it is unique in the way this information is recorded. The model of integration used by IDIS provides a novel way of recording the various aspects of DR in a single environment. Different representations are used to record different aspects of the design. An Issue Based Information System (mIS) is used to represent the design deliberation. A viewpoint mechanism is used to record the design space and the alternatives explored. The design constraints are represented by production rules and modifications are recorded in design specific lists. The functionality of the design items recorded can be modified or extended to record unconventional uses. This information is integrated with the design diagrams and sketches in a CAD tool that supports traditional design tasks. All of the aspects of DR that relate to a design alternative are recorded with the design diagram at the relevant point in the design space. The ability to integrate these aspects of DR in a single environment with the artefact, while at the same time supporting traditional design tasks, is unique to IDIS.

8.1.2 Maintaining the Integrity of an Issue Base
IDIS addresses the problem of maintaining the integrity of the deliberation recorded using the IBIS representation. Although IBIS has been used extensively since it was developed in the 1960's, no researcher has addressed the problem of maintaining the integrity of the information that it records. During the development of IDIS a case study was used to evaluate the development of an issue base. This case study demonstrated that it is possible to loose or corrupt information if it was not added in a
disciplined manner. Two types of integrity were identified, semantic integrity and temporal integrity.

Semantic integrity relates to the understanding of the information recorded in the issue base. The structure of the issue base will only provide an accurate record of the deliberation if the correct nodes are used to record the information and, the relationships between the nodes are described with the correct links. The semantic integrity is maintained in all the tools that use the IBIS representation.

Temporal integrity relates to the order in which information is added to the issue base and how this affects the information stored. Designers may wish to re-examine an issue that has been resolved to explore positions not previously identified. New information should not be added to an issue base in an undisciplined manner as it will not be clear what information was available when the decision was made. By conserving the temporal integrity of an issue base it is possible to examine the issue base structure to see how it has evolved. The IBIS representation used in IDIS was extended with the addition of a followup link to achieve this. When an issue base is re-examined after a decision has been made designers create a new issue and link it to the original issue with a followup link. This allows designers to explore new positions while maintaining the temporal integrity as the order in which the information was added is clearly represented in the structure of the issue base.

Maintaining the temporal integrity of an issue base has not been addressed by other systems. If the deliberation is intended to provide an understanding of decisions, as it is in DR, then the integrity of the information upon which the decision is based must be maintained. IDIS is unique in its ability to maintain both the semantic and temporal integrity.

8.1.3 Identifying the Differences Between Designs

IDIS provides designers with a feature that allows them to identify the differences between two designs. Recording the design space and the modifications made to a design is not in itself unique. However, IDIS uses this information in a unique way to
provide designers with a novel feature. The differences between two alternative designs cannot always be easily identified by viewing the designs. Slight changes to a design can have significant and unforeseen repercussions. For example, re-sizing the pressure relief of a pressure vessel without modifying the related inlet and outlet pressures could have significant consequences in terms of safety. In the worst case scenario, the pressure vessel could explode resulting the loss of life and serious damage to the plant. By keeping track of the modifications made to designs and the relationships between the designs IDIS is able to display how they differ. None of the existing DR capture tools have the ability to provide such information for designers.

8.1.4 Managing Design Information and the Design Project

Finally, IDIS provides support for managing the design project and managing information about the design. Designers must be able to find new and relevant information quickly and easily if it is to provide support for the design process. IDIS aids a project leader in: controlling access to the information; identifying the contribution of each team member; and focusing the team on particular issues. Search facilities are provided to support access to the information recorded in the system. A report can be generated on the deliberation that has taken place during any given time period, providing an insight into the work of the design team. The project management features also allow project leaders to monitor the progress of the project against deadlines. All these management features are unique to IDIS.

8.2 Limitations of IDIS

IDIS supports the design process and provides a novel way of capturing various aspects of DR in a conventional CAD environment. It goes some way to bridging the gap between existing DR capture tools and the needs of designers. This section discusses some of the limitations and possible improvements that could be made to IDIS.

One of the major drawbacks of IDIS is its inability to fully support the process of decomposition and integration. Although it allows designers to indicate a design has
been decomposed, it does not provide any mechanism for merging decomposed designs. This is a common design practice, especially for large projects, and the system should be extended to support such processes. This can be achieved by recording and maintaining the consistency of the interface between decomposed designs.

Another feature that is lacking in IDIS is the ability to record the design requirements. Some of the tools reviewed in Chapter 3 offer this facility. Many of these tools link the intended functionality of the components to the requirements. However, in doing so they lose the ability to record additional or unconventional functions of design components. IDIS should be extended to provide a means of explicitly recording design requirements. These requirements should be linked to the intended functionality of the components. This additional feature should not result in the loss of the ability to record additional or unconventional functions of design components.

There are currently fifty two different units available as AutoCAD icons within IDIS. Most of these are process plant components such as: open and closed vessels, distillation columns, pumps, heat exchangers, storage vessels, etc. New icons and their corresponding IDIS and chemical information frames must be added manually to the system. This is not ideal as designers have to modify the code to extend the set of units available. The process of generating the underlying frames and linking them to an icon should be automatic. Designers could then create a new plant unit by simply creating the icon, providing it with a name and supplying the relevant information to IDIS. IDIS's inability to do this is another limitation of the existing system.

The issue base component of IDIS uses a purely textual representation. The addition of a graphical representation similar to gIBIS would provide a more user friendly front end for recording and accessing design deliberation. Although the designers found the information recorded in IDIS useful they commented that a graphical representation would help clarify the structure of the information recorded.
During the deliberation process conflicts may arise between designers. The problem of identifying and resolving conflicts has been addressed in areas such as software specifications (Easterbrook and Nuseibeh 1996) and legal reasoning systems (Dewitz et al. 1994). IDIS could be extended to include a conflict resolution feature that would help designers to identify and resolve conflicts. In addition, options generated could be evaluated and the most appropriate ones identified by the system. This would allow IDIS to provide decision support features as well as capturing DR.

Finally, although IDIS supports the collaborative nature of the design process it does not support concurrent access to projects. Only one designer can access a project at any one time. This could be problematic if several designers were working on a project full time and all required access to it. IDIS should be extended to provide features that support concurrent access to projects while maintaining the consistency of the information recorded.

8.3 Future Work in the Field of Design Rationale Capture and Use

The current trend in DR research is moving towards integrating different sources of information within a single environment. This is set to increase as new commercial products provide support for such practices. QuestMap (Corporate Memory systems) is an IBIS based general discussion tool that has recently been developed. Lotus Notes (Lotus Notes Corp.) provides a discussion database and information storage facility that supports the integration different types of information. These systems provide an environment that supports collaborative work and provides the users with a means of accessing: documents, databases, diagrams and deliberation as well as information on the internet. Commercial systems such as these will provide the foundation for the future design systems and will support the collaborative exploration and access of large amounts of information. However simply recording more information in a single environment will not necessarily provide additional benefits. To be of any use, designers must have support for accessing, managing and reusing this information. Future research in DR should address these problems.
The case studies presented in the DR capture literature demonstrated how DR can be recorded and accessed during the lifetime of the design. This is only one of the uses of DR. What is lacking is a long term study that captures the DR during the design process and evaluates its usefulness when the design is modified or reused. This type of study will need to be performed over a considerable period of time as a design may not be modified and reused for several years. The time scales of such a project and the fact that DR capture is still relatively new has meant that such a project has not been undertaken. It is argued that a long term study and an evaluation of the usefulness of DR during redesign is required and is an important aspect of DR research that must be addressed.

The topics discussed above have concentrated on the use of DR during redesign. This is one of the major and most useful aspects of DR and one that designers can identify with most easily. However an explicit representation of the DR may have other uses that should be investigated.

The DR should provide a better understanding of the design process. It can be used to identify areas of possible future research in design support tools. In order to use DR in this way future researchers should analyse the information recorded by DR capture tools and improve current models of the design process. Common routine practices may be identified which may result in the development of more proactive design support tools.

The DR recorded provides an understanding of the designers reasoning processes. As mentioned above this will prove an insight in to the design process itself. This information can be used for educational purposes. A record of DR provides an explicit record of the processes and information that the designer has used during the design process. This information can be used to develop intelligent tutoring systems to train students or to help bridge the gap between novice and expert designers.
It was argued at the beginning of this thesis that the DR is a by-product of the design process. It is rarely recorded in the final design documentation which is often seen as the product of the design process. DR can be used to help generate the design documentation. The understanding of the design provided by the DR should help improve the quality of the information that is recorded. During one of the case studies presented in this thesis an MSc student used the information recorded in IDIS to help write the final report.

There are still many unresolved areas in the field of DR research. The main areas of focus for future research are: an evaluation of the benefits of DR during design modifications; how to access and manage the information recorded; identify different uses for DR as described above.
References

ASPEN, Aspen Technology Inc., Cambridge, MA, U.S.A.

AutoCAD, AutoCAD12, Autodesk Ltd. Cross Lane, Gilford, Surrey, U.K.


149


Intergraph, Intergraph Electronics, Delta Business Park, Great Western Way, Swindon, Witshire, U.K.


QuestMap, Corporate Memory systems, 11824 Jollyville Road, Austin, Texas, U.S.A.


Appendices
Appendix A

A List of the Units in IDIS
All of the plant item frames are created as instances of the parent frame unit. The naming convention used for the items is: itemType number of inlets i number of outlets o. Where an item has a single inlet and outlet the i and o are omitted. E.g. a closed vessel with one inlet and one outlet would be named, closedVessel, a closed vessel with 2 inlets and one outlet would be named, closedVessel2i1o.

The frames for unit, pipe, signal line, closed vessel and a centrifugal pump are presented in full. These are followed by a list of all the available items.

/* **********************************************************
/* UNIT IS THE BASE FRAME, ALL OTHER FRAMES ARE CREATED AS INSTANCES
OF THIS FRAME */

frame(unit,[project is 'project_name',
    date is 'date_created',
    name is 'creators_name',
    unitName is 'undefined',
    function ref [ ],
]).

frame(pipe isa unit, /*PIPE*/
    [ slots ref [project,date,name,unitName,function,design_press,design_temp,diameter,
        material,contents,inlet1,outlet1],
        design_press is 'undefined',
        design_temp is 'undefined',
        diameter is 'undefined',
        material is 'undefined',
        contents is 'undefined',
        inlet1 ref 'undefined',
        outlet1 ref 'undefined',
        variable_slots ref [unitName,function,design_press,design_temp,diameter,material,contents] ]).

frame(sigline isa unit, /*SIGNAL LINE*/
    [ slots ref [project,date,name,unitName,function,sigin1,sigout1],
        sigin1 is 'undefined',
        sigout1 is 'undefined' ]).
frame(closedVessel isa unit, /*CLOSED VESSEL*/
  [    
    slots ref[project, date, name, unitName, function, design_press, design_temp, capacity, material, 
    wall, inlet1, outlet1],
    design_press is 'undefined',
    design_temp is 'undefined',
    capacity is 'undefined',
    material is 'undefined',
    wall is 'undefined',
    inlet1 is 'undefined',
    outlet1 is 'undefined',
    variable_slots ref [unitName, function, design_press, design_temp, capacity, material, wall]  
  ]).

frame(centrifugalPump isa unit, /*CENTRIFUGAL PUMP*/
  [    
    slots ref[project, date, name, unitName, function, design_press, design_temp, flow_rate, 
    rating, inlet1, outlet1],
    design_press is 'undefined',
    design_temp is 'undefined',
    flow_rate is 'undefined',
    rating is 'undefined',
    inlet1 is 'undefined',
    outlet1 is 'undefined',
    variable_slots ref [unitName, function, design_press, design_temp, flow_rate, rating]  
  ]).

/* VALVE FRAMES */

valve
controlValve
tripValve
ventValve
slamshut

/* PIPES AND ASSOCIATED ITEMS */

pipe
t_join
divider
header
dummyHead
dummyTail

/* SIGNAL LINES AND ASSOCIATED ITEMS */

sigline
sensor
controller
sigSplitter
sigHeader
tripSwitch

A 2
/* VESSELS*/
closedVessel CLOSED VESSEL
closedVessel2o CLOSED VESSEL (1 INLET, 2 OUTLETS)
openVessel OPEN VESSEL
openVessel2i OPEN VESSEL (2 INLETS, 1 OUTLET)
decant2o DECAN TOR (1 INLET, 2 OUTLETS)
decant3o DECAN TOR (1 INLET, 3 OUTLETS)
feed FEEDSTOCK (1 OUTLET ONLY)
feed3ilo FEEDSTOCK (1 INLET, 1 OUTLET)
storage STORAGE TANK (1 INLET ONLY)

/* PRESSURE UNITS */
pressure_switch PRESSURE SWITCH
press_ctrl PRESSURE CONTROL SWITCH
press_safe PRESSURE SAFETY SWITCH
relief RELIEF MECHANISM

/* DISTILLATION COLUMNS */
dis2i2o DISTILLATION COLUMN (2 INLETS, 2 OUTLETS)
dis2i3o DISTILLATION COLUMN (2 INLETS, 3 OUTLETS)
dis3i2o DISTILLATION COLUMN (3 INLETS, 2 OUTLETS)
dis3i3o DISTILLATION COLUMN (3 INLETS, 3 OUTLETS)
dis4i3o DISTILLATION COLUMN (4 INLETS, 3 OUTLETS)

/* GENERAL UNITS */
centrifugalPump CENTRIFUGAL PUMP
heater HEATER
reboiler REBOILER
reboil3ilo REBOILER (3 INLETS, 1 OUTLET)
cooler COOLER
cooler3ilo COOLER (1 INLET, 1 OUTLET)
vaporiser VAPORISER
scrubber SCRUBBER
filter FILTER
diffuser DIFFUSER
reducer REDUCER
regulator REGULATOR
fan ELECTRIC FAN
burner GAS BURNER
selector SELECTOR
condl CONDENSOR (LEFT)
condr CONDENSOR (RIGHT)
reflux REFLUX

/* BOXES AND NON SPECIFIC UNITS*/
box BOX (1 INLET, 1 OUTLET)
box3o BOX (1 INLET, 3 OUTLETS)
box2i2o BOX (2 INLETS, 2 OUTLETS)
box2o BOX (1 INLET, 2 OUTLETS)
Appendix B

Software and Hardware Requirements of IDIS
Software

IDIS was developed under AutoCAD12 and requires version 12 or higher of AutoCAD. The program was developed in the programming language CProlog and requires a CProlog compiler. These two packages are essential for the program to run.

Other software that is Xemacs a UNIX editor and Xfig a drawing package. Any other UNIX editor or drawing package can be used, Xemacs and Xfig are the default packages. If these packages are replaces the calls to Xemacs and Xfig in the idis.pl file must be replaced with the names of the new package. IDIS runs under UNIX and uses the following UNIX commands: ls; grep; rm; rmdir; mknod (create a named pipe); date; whoami; pwd; cd; mkdir; chmod; sed; >; >>; and &.

Hardware

IDIS runs under the UNIX operating system. It can be run either on the host machine or over the network by an Xterminal. The program itself does not have stringent hardware requirements although AutoCAD does has several essential hardware requirements. The main requirements are:

- A monochrome or colour monitor;
- A computer running the Sun OS operating system version 4.1.1 or higher;
- OpenWindows version 3.0;
- 32 MB of RAM;
- 30 MB of fixed-disk storage in the directory where AutoCAD is installed;
- A swap partition twice as large as the amount of RAM on the machine;
- A digitising tablet or mouse.

For more details see Chapter 1 (System Requirements) in: AutoCAD Interface, Installation, and Performance Guide - SUN Sparc.
Appendix C

IDIS User Manual
Index

1. What is IDIS
   1.1 The issue base
   1.2 Sketches
   1.3 Design diagrams
   1.4 Design alternatives
2. Hardware and software requirements
   2.1 Software
   2.2 Hardware
3. Using IDIS
   3.1 Getting started with IDIS
   3.2 Loading an existing project
   3.3 Managing project information
4. Recording design deliberation
   4.1 Starting a new issue
   4.2 Creating and linking nodes
   4.3 Viewing an issue base
      4.3.1 Unseen nodes
      4.3.2 Listing nodes
   4.3.3 Viewing a node and its links
   4.4 Making a decision and following up an issue
5. Design diagrams and sketches
   5.1 Adding and removing units
   5.2 Connecting units
   5.3 Viewing and changing a units parameters
   5.4 A units chemical information
6. Exploring alternative designs
   6.1 Creating a new viewpoint
   6.2 Changing viewpoints
   6.3 Viewing a viewpoints information
      6.3.1 Listing viewpoints
      6.3.2 View a viewpoints changes
      6.3.3 View a viewpoints information
7. Interfacing with other systems
   7.1 QUEEN
   7.2 Meppi
8. Future features of IDIS
   8.1 Recording Constraints
   8.2 Recording Functionality
This document provides a brief overview of an integrated design system called IDIS. The manual provides a description of the system, the benefits of using it and how to get started. The software and hardware requirements and the interfacing capabilities with other systems are also outlined. This system was developed as part of a PhD project at Loughborough University and the research was sponsored by British Gas by a British Gas research scholarship. The PhD. was supervised by Dr Paul Chung. If you have any questions about the software please contact:

Roger Goodwin  
British Gas R&T  
Electronics and Computing  
Gas Research Centre  
Loughborough  
Leicestershire  
LE11 3QU  
U.K.

Phone (01509) 282247  
E Mail Roger.Goodwin@bggrc.co.uk
1. What is IDIS

IDIS is an integrated design information system. The aim of this tool is to provide an integrated environment in which designers can work while at the same time recording various aspects of design that are not traditionally captured. IDIS records the design deliberation, sketches, the alternative designs that are explored, the detailed design diagrams and all the changes that have been made to the design. This information can be used to:

- Justify design decisions.
- Restore interrupted designs.
- Provide information for the final design documents.
- Prevent accidents occurring by making design assumptions explicit.
- Assist in subsequent modifications to existing designs.
- Assist in the design of future similar artefacts.
- Help to provide better models of the design process.

The main components of the system are an issue based information system which records the design deliberation. A drawing package that allows the users to draw quick sketches when they want to describe concepts, ideas, etc. A viewpoint mechanism that combines all the components, allows the designers to move through the design space and also records the alternative designs that are explored. AutoCAD a CAD package is used as the front end to the system and is used to create the detailed design diagrams. A list of changes is also recorded that allows us to identify what design parameters have been changed who changed them and when. These changes are also used by the system to restore interrupted designs and to load the correct state when moving through the design space. The rest of this section provides a high level description of these components and the following sections describe them in more detail.

1.1 The issue base

An Issue Base Information System (IBIS) representation is used to support and record the deliberation that takes place during a design project. An IBIS representation is basically a network of nodes connected with semantic links. Each node represents part of the deliberation and its relationship to the other nodes is denoted by the type of link between the nodes.

An issue base is always started by creating an issue which highlights some decision that has to be made or some problem that needs to be solved. Once this issue is created positions are added that propose possible solutions. Arguments that either support or attack the positions can also be added. These nodes are linked to form a network structure that represents all the discussion that takes place during the resolution of an issue (see figure 1).
The type of link used is important because it is these links that record the structure of the argument. If the correct link is not used the issue base will not correctly represent the discussion that took place. E.g. a position is added as a possible solution in response to an issue so it is linked to the issue with a responseTo link. An argument node can linked to a position with a supports link or against link because it may either support or refute a possible solution. A full list of nodes and links and a discussion about other features related to the issue base can be found in section 4.

1.2 Sketches

The system has the ability to call up a simple drawing package that allows designers to draw quick sketches which they often want to do to demonstrate a point or a concept. These sketches should not be confused with the detailed design diagrams. The drawing package used by this system is the UNIX based Xfig.

1.3 Design diagrams

AutoCAD is used to record the detailed design diagrams. A selection of plant units are available as predefined symbols which can be selected by clicking on the appropriate unit name on the menu at the right hand side of the AutoCAD screen. The system also creates its own representation of these units and updates this when any of the units parameters are changed. These features are discussed in more detail in section 5.
1.4 Design alternatives

During a design project many alternative designs may be explored, evaluated, then accepted and refined or rejected. These alternatives are recorded by the system so the design team can see clearly what designs have been explored and which ones have been accepted, which ones rejected and why. These alternatives are represented in the system as a single point in the design space which we refer to as a viewpoint. Designers can move through the design space by changing viewpoints. A single viewpoint brings together all the relevant information for that point in the design. This includes the deliberation about that design (the issue base), the changes made to the design at that point and the detailed design (the AutoCAD diagram).

2. Software and Hardware Requirements

2.1 Software

IDIS uses several packages, the main two are the CAD package AutoCAD and the programming language Cprolog. These two packages are essential for the program to run. Other software that is used includes Xemacs a UNIX editor and Xfig a drawing package. Any other UNIX editor or drawing package can be used, Xemacs and Xfig are the default packages and are both public domain and have the advantage of being free. The object oriented code that is used as the main representation in the system was developed by Dr Paul Chung.

2.2 Hardware

A Sun-4 or SPARCstation is required to run the program. The program can be run over the network by an Xterminal. There are Xterm emulators for PC's so it is possible to run the program using a PC which is connected to the network running an Xterm emulator.

AutoCAD has several essential requirements. The main requirements are:

- A Sun-4 or SPARCstation with a monochrome or colour monitor.
- Sun OS operating system version 4.1.1 or higher
- OpenWindows version 3.0
- 32 MB of RAM
- 30 MB of fixed-disk storage in the directory where AutoCAD is installed
- A swap partition twice as large as the amount of RAM on the machine
- A digitising tablet or mouse

For more details on these see Chapter 1 (System Requirements) in the AutoCAD Interface, Installation, and Performance Guide - SUN Sparc.
3. Using IDIS

3.1 Getting started with IDIS

To run IDIS type idis from the command line i.e.

```
machine% idis <return>
```

An AutoCAD screen will appear with a single menu item on the left hand side with the title Start IDIS. This menu has 6 options they are:

1. List projects  
   Lists all the available IDIS projects.

2. Load a project  
   Allows you to load an existing project.

3. Start a new IDIS project  
   Allows you to start a new project.

4. Delete a project  
   Deletes an existing project.

5. Use AutoCAD without IDIS  
   Allows you to use AutoCAD on its own.

6. Quit  
   Quits the program.

When starting the program initially you will probably want to create a new project. To do this you should select the third item on the menu, Start a new IDIS project. The system will ask you for a project name and a one line summary for the first viewpoint. If a project does not exist with the same name the system will create a new project and you will be added as a user with leader status for that project.

The AutoCAD screen will change and you will be placed at the initial viewpoint for that project. This screen will have 4 menu items on the top left of the screen and a list on the right hand side under the heading AutoCAD. The 4 menu items are, Icons, IDIS, Management Info and Unit Info.

- Icons contains a selection of icons of the available plant units.
- IDIS contains all the commands you will need to create, link, view and edit nodes, create new viewpoints and change viewpoints, load, save and delete projects, create and edit sketches, etc.
- The Management Info has all the commands that are used to provide information about a project. These include listing the users for a project, adding and deleting users, looking at the deadlines for the issues, finding out if there are any nodes you have not seen, finding out when the issue base was last changed, etc. The last menu Unit Info provides you with a means of viewing a units specification sheet and changing the units parameters, adding information about the chemicals in the unit and it also allows you to create files that may be used by other programs. These are discussed in more detail in later sections.
3.2 Loading an existing project

To load an existing project you can either type `idis project name <return>` at the prompt or you can load the program and choose the Load a project option from the menu. You must be a valid user for the project you wish to load. When a project is loaded you will not start at the initial viewpoint but will start at the viewpoint where the project was saved.

3.3 Managing project information

There are several features that are necessary for the management of a project and the people in the project. These options can be found under the menu item Management Info which has the following commands:

1. List deadlines  Lists all the deadlines for the issues which have not been resolved.
2. Next deadline  Finds the next deadline that has to be met.
3. Last modified  Tells you when the issue base was last modified and what was added.
4. Created since  Tells you what nodes have been created since the given date.
5. List users  Lists all the users in the project and displays their status.
6. Users contribution  Lists the contributions that the user has made in this project.
    *  Lets you change the users status.
8. Add user  Lets you add a new user.
9. Delete user  Lets you delete a user.
10. Password  This option updates you automatically to leader status if you know the correct password.

Items marked with a * can only be performed by the project leader. It is possible to have more than one leader for a project.
4. Recording Design Deliberation

An IBIS representation is used to support and record the design deliberation (see section 3). All of the commands discussed in the following section can be found under the menu item IDIS-Nodes.

4.1 Starting a new issue

When you want to start a discussion you will need to create a new issue. This is achieved by selecting the **Create a new node** option. This will give you a list of all the available node types. After selecting **Issue** from this list you will be asked to provide a one line summary for the issue and a deadline (day/month/year) before which the issue should be resolved. After providing the summary and deadline for the issue you will be given a file that already contains some information about the node, the date, type of node, who created it and the summary. You can now type in a full description of the problem. When you have finished you exit the editor and a new issue has been created.

4.2 Creating and linking nodes

A similar procedure to the one described above is used to create other nodes the only difference being that you are not asked for a deadline for any other type of node. The types of node you can create are, issue, position, argument, comment, fact and decision. When you have created a node you will have to link it to the appropriate node to say how it relates to the existing discussion. This is achieved by selecting the **Link two nodes** option. This will display a list of all the possible link types. It is important that the correct link is used or the issue base will not correctly represent the structure of the discussion. The system will not allow you to make incorrect links.

Here is a list of all the possible links and the type of nodes that can be linked by them. The direction of the link is important.

<table>
<thead>
<tr>
<th>Link from</th>
<th>Link type</th>
<th>Link to</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>responseTo</td>
<td>position</td>
<td>Position is a response to the issue</td>
</tr>
<tr>
<td>argument</td>
<td>supports</td>
<td>position</td>
<td>Argument supports the position</td>
</tr>
<tr>
<td>fact</td>
<td>supports</td>
<td>argument</td>
<td>Fact supports the argument</td>
</tr>
<tr>
<td>argument</td>
<td>against</td>
<td>position</td>
<td>Argument is against the position</td>
</tr>
<tr>
<td>comment</td>
<td>comment</td>
<td>any node</td>
<td>Provides a comment on a node</td>
</tr>
<tr>
<td>decision</td>
<td>decided</td>
<td>issue</td>
<td>The decision for the issue</td>
</tr>
<tr>
<td>issue</td>
<td>followUp</td>
<td>issue</td>
<td>An issue has been followed up after a decision has been made</td>
</tr>
<tr>
<td>position</td>
<td>combinedWith</td>
<td>position</td>
<td>Positions provide a combined solution</td>
</tr>
<tr>
<td>issue</td>
<td>replaced</td>
<td>issue</td>
<td>An issue is replaced by several or several are replaced by one</td>
</tr>
<tr>
<td>position</td>
<td>copyOf</td>
<td>position</td>
<td>Position is a copyOf an existing position</td>
</tr>
</tbody>
</table>
4.3 **Accessing the information an issue base**

There are several different types of information that you may want to access when working with an issue base. The following sections deal with accessing the information stored in the nodes and the structure of the issue base. Some of the project management commands also deal with aspects of the issue base, e.g. Listing deadlines, finding out when the issue base was last modified, etc. (see section 3.3).

4.3.1 **Unseen nodes**

The system can tell you if new nodes have been added to the issue base by other team members since the last time you looked at the project. To get a list of the nodes that have been added and who added them you select the **List unseen nodes** option.

4.3.2 **Listing nodes**

You can also list all the nodes in an issue base. When you choose the option **List nodes** you will be asked to type in the type of node you wish to view, all issues, positions, etc., and which viewpoints nodes you wish to view. The default is all for both nodes and viewpoints and this will display all the nodes that have been created for that project. This option does not display the relationship between the nodes.

4.3.3 **Viewing a node and its links**

In all of the above cases only the summary of a node is displayed, to view all the information about a node you should choose the **View node** option and then enter the name of the node you wish to view, issue1, position2, etc. This will display the full file and allow you to read all the information relating to that node. The links that have been made to a node are also important because they provide a representation of the discussion that has taken place to date. To view this information you should choose the **View a nodes links** option. Again you will be asked to provide that name of the node you are interested in. The system will then display all the nodes that are linked to that node. The type of link and the summary of each node is displayed and each new link type is indented. The links made **FROM** a node are not highlighted only **TO** the node are displayed.

4.4 **Making a decision and following up an issue**

Making a decision about an issue changes the nature of an issue base. Once a decision node has been linked to an issue no other nodes can be linked to any part of that issue base. This is prevented because it is not clear to the user that the new information was not available when the original decision was made. The temporal integrity of the issue base would have been violated. To resolve this problem we have a special
followUp link. To add information to an issue base after it has been 'closed' you need to start a new issue and link this issue to the existing issue with a followUp link. The new issue can then be explored in the normal manner. By using this link we are making it explicit that additional information has been added. If another decision is made it can be added to the new issue.

5. Design diagrams and sketches

5.1 Adding and removing units

Xfig a shareware UNIX based drawing package is used by IDIS to provide a rough sketching facility for the designers. This is accessed though the sketches menu. This allows the user to create, edit, list, and delete sketches.

5.2 Adding and removing units

AutoCAD is used to create the detailed design diagrams. The system has a set of predefined units that are represented by an appropriate symbol. IDIS also keeps a frame representation of the unit. To add a unit to a diagram you select the New unit menu at the right hand side of the AutoCAD screen. Under this menu there are several other menus with a range of plant units. when you select a unit you will be given a symbol to represent that unit and can place it on the diagram where you choose. There is also a selection of icons on the left hand side of the AutoCAD window under the menu Icons. You can select the plant unit by its symbol and place it on the diagram. To remove a unit from the diagram you select the Remove menu item on the right hand side of the screen and you will then be asked to select a unit to delete. Click on a unit in the diagram and it will be removed.

5.3 Connecting units

There are two type of connection that can be made between units, pipes and signal lines. To connect two units choose the Connect menu on the right hand side of the screen. This gives you the option of choosing a pipe or a signal line. Once you have made your choice you will be asked to select a the starting point for the pipe or signal line. Click on an outlet from a unit on the diagram and then you will be asked where it is to go to, click on an inlet of the required unit on the diagram. The procedure is the same for both signal lines and pipes.

5.4 Viewing and changing a units parameters

Each unit has a set of parameters that can be set to represent its design specifications. These specification sheets are in no way complete and are only provided as an example of the capabilities of the system. Only the basic parameters are available in
the system at the current time. These include, the design temperature, design pressure, material of construction, flow rate, etc. To view or set these parameters select the **View a units spec sheet** option from the **Unit Info** menu at the top of the screen. You will be asked to select a unit. Click on a unit and the specification sheet for that unit will be displayed. Certain parameters are set by the system and cannot be changed, these include the project name, who created the unit and the inlets and outlets. To change the values of the unit simply click on the slot and enter the new value. The default value for all slots is 'undefined'.

### 5.5 A units chemical information

The system also allows you to store information about the chemicals that will be present in the unit. To view or change this information select the **Chemical Info** option under the same **Unit Info** menu. Again you will be asked to select a unit. The chemical information will be displayed and you will be able to update this by entering the new value in the appropriate slot. This information is used when creating a file for the Meppi program (see section 7).

### 6. Exploring Design Alternatives

All of the commands referred to in this section can be found under the **IDIS-Viewpoint** menu at the top of the AutoCAD screen.

#### 6.1 Creating a new viewpoint

**NOTE:** When you create a new viewpoint the parent viewpoint that you use will be locked and you will not be able to add or delete any units from the diagram.

When you want to explore a design in several ways you can create new viewpoints. This will create new viewpoints that have the same diagram as the parent viewpoint. The parent viewpoint will be 'locked' and you can then explore alternative designs by changing the children viewpoints. To create a new viewpoint you select the **Create a new viewpoint** option. You will be asked what viewpoint to use as the parent viewpoint, and then you will be asked to provide a one line summary for the viewpoint. The parent viewpoint you give does not have to be the current viewpoint and when you create a viewpoint you are not moved to that viewpoint until you change to it.

When a new viewpoint is created the user has the option to select an alternative, decomposition or integration link. IDIS does not support true integration and does not automatically integrate diagrams. The option simply allows the designer to keep track of the types of link that have been made and to indicate where a design is an alternative or a decomposition.
6.2 Changing viewpoints

You can move through the design space by changing viewpoints. This is achieved by selecting the **Change viewpoint** option. You are then asked to enter the name of the viewpoint. That viewpoint then becomes the current diagram.

6.3 Viewing a viewpoints information

There are several different types of information that you can obtain about viewpoints.

6.3.1 Listing viewpoints

To list all the available viewpoints in a project you select the **List viewpoints** option. This will display all the available viewpoints with their summaries, parent viewpoint and any children viewpoints.

6.3.2 View a viewpoints changes

The **View a viewpoints changes** option will display all the changes made to the viewpoint. These include any units added, deleted and any changes that have been made to a units specification sheet.

6.3.3 View a viewpoints information

Selecting **View current viewpoint** will display the current viewpoint you are on and the viewpoints summary. You can also get a more detailed summary of any viewpoint by selecting the **View a viewpoints Info** option. This will provide the viewpoint, its summary, parent and children and all the nodes that are available in that viewpoint, i.e. all the issues, positions, arguments, etc.

7. Interfacing With Other Systems

The system has the ability to create input files for other programs that can be used to aid designers in the process industry. The two other systems that this program can interface with are QUEEN and Meppi.

7.1 QUEEN

QUEEN stands for QUalitative Effects ENgine. This program performs a qualitative analysis of the effect of deviations in a continuous process plant. This analysis is performed by determining how deviations propagate using a sign directed graph of the plant. IDIS can provide a semantic representation of a plant, a file containing a frame
representation of the plant units, which QUEEN uses as input. To generate a QUEEN file you select the Create a prolog file under the Unit Info menu. This will generate a representation of the current viewpoint and store it in a file viewpoint#.pl.

7.2 Meppi

Meppi stands for Minimising Environmental Problems in Process Industries. The aim of this program is to provide an advisory system which will assist engineers in process design and modification to minimise the production of waste. The Meppi program uses the chemical information about the units in the diagram to perform this task. To produce a Meppi file you select the Create a Meppi file option under the Unit Info menu. You will be asked for the process name, the level of analysis and a value for Meppi Ext variable. This will then create a file viewpoint#.mp file that contains all the information for the Meppi program. Meppi is an ongoing project at the Hong Kong University of Science and Technology (HKUST). For more details about this project you should contact David Pemberton at HKUST.

8. Future features of IDIS

8.1 Recording constraints

IDIS is currently being developed to support the recording of design constraints. This feature will allow designers to explicitly record design constraints that apply to the designs and check the designs for the violation of these constraints.

8.2 Recording functionality

Another feature that is currently being added is the ability to record the intended function of design items. By allowing designers to add their own functions to items they will be able to represent secondary functions.
Appendix D

Issue Bases From First Case Study
It is proposed that we build a new batch plant for the production of plastic granules in the USA. There are heavy demands for cooling. There is a design for an existing plant of the same tonnage (2000 tpa) in the UK which uses both recycled cooling water (RCW) at 21°C and chilled water at 5°C. Atmospheric ambient and wet bulb temperatures are higher at the proposed USA site by 5°C and 8°C respectively on average. The existing design of cooling tower would deliver an RCW supply temperature which is likely to be above, perhaps well above, 29°C.

Are any modifications needed to the plant items and/or the cooling supply systems?

We have decided that we will adopt position 3 as a partial solution. If we resize the cooling tower it will alleviate some of the problem although we will still need to make some changes, however these changes may be on a smaller scale than they may have been if we did not resize the cooling tower.

What other changes should we make to the plant?

We should redesign the cooling tower to lower the RCW temperature. This will not solve the complete problem but it will make it easier for us to solve the problem. We should have to make fewer changes or less drastic changes.

Any duty can be transferred to chilled water without redesign but items would be larger than necessary.
We can redesign any item in which the duty is to cool the process fluid to a temperature which exceeds the RCW temperature by 5°C.

Some items may be oversized in the UK design and adequate in the USA. Any other items would have to be transferred to chilled water.

The lowest practicable temperature is judged to be 29°C.

Even if the cooling tower is redesigned the plant items will still need to be resized but to a lesser extent.

The cooling system is redesigned to increase the RCW available to supply an increased number of plant items in parallel.

In addition the temperature of the RCW may be reduced.

May be able to cool with a higher temp over a longer time.
30/04/93
cgrg
Type of File: position
Name of File: position7
Summary: Switch RCW to chilled water sooner.

In batch cooling duties RCW could often be followed by chilled water to complete the cooling process. In the USA where the RCW is higher the RCW could be switched to chilled water sooner. Thus some of the duty is transferred to chilled water.

30/04/93
cgrg
Type of File: position
Name of File: position8
Summary: Transfer more duties to chilled water. Any duty can be transferred to chilled water without redesign but items would be larger than necessary.

30/04/93
cgrg
Type of File: position
Name of File: position9
Summary: copy of position 1

09/04/93
cgrg
Type of File: argument
Name of File: argument1
Summary: Chilled water costs more than RCW.

Increases running and capital costs.

09/04/93
cgrg
Type of File: argument
Name of File: argument10
Summary: The running costs are increased significantly.

Due to the addition of new plant items the running costs of the plant will rise significantly. This is caused by the increased cost of the maintenance, management and power supply costs.

09/04/93
cgrg
Type of File: argument
Name of File: argument11
Summary: Plant availability is improved.

As there are more plant items performing the same function the loss of one of these items will have less of an effect on the output of the plant. This improves the availability of the plant.
The plant will operate exactly the same as the original. As the RCW and chilled water will be at the same temperature as used in the UK plant. No changes will have to be made to the plant items and they will all operate the same as they do in the UK plant.

supports pos 7

against pos 7

No redesign of plant items necessary.

No change of cooling supply system necessary.

Larger plant items.

The larger plant items may necessitate revising the layout.
By using both the redesign of the cooling supply system and the resizing of plant items the most optimum capital cost can be obtained.

There is no resizing or redesign as only additional items are added.

With the addition of more items to the plant the plant will grow significantly in size and the plant layout may need to be changed.

If the RCW can be cooled to the UK RCW temperature or below then this position is valid on its own.
Appendix E

Issue Bases From Second Case Study
Issue Base Nodes: Viewpoint1

15/10/93
cgrg
Type of File: issue
Name of File: issue1
Summary: What is the best route to produce ethyl acetate.

A market survey has identified a niche in the market for ethyl acetate. This is used in the production of inks, adhesives and lacquers. As well as the protective coating applications it is an effective solvent for many resins and used extensively as a cellulose nitrate solvent in the manufacture of leathers, inks, cements, photographic films and linoleum.

The aim of the project is to design a plant to produce 15 000 tonnes per annum of 99.8% w/w ethyl acetate. We need to decide what chemical route to use.

15/10/93
cgrg
Type of File: position
Name of File: position1
Summary: Acid catalysed liquid phase esterification of ethanol and acetic acid.

The ester is produced by the substitution of ionisable hydrogen in the carboxylic acid by an organic radical produced from the alcohol. The reaction does not proceed to completion but instead approaches an equilibrium.

15/10/93
cgrg
Type of File: position
Name of File: position2
Summary: Condensation of Acetaldehyde by the Tischenko Reaction.

The ester is produced directly by the condensation of acetaldehyde in the presence of aluminium ethylate. Acetaldehyde is usually passed at 0-5 °C through a mixture of aluminium fillings and traces of AICls in ethanol and ethyl acetate. Refrigeration is used to remove the reaction heat. The reactor effluent is passed to an evaporator where the ethyl acetate is removed from the remaining catalyst before being purified by distillation.

15/10/93
cgrg
Type of File: position
Name of File: position3
Summary: Alcoholysis.

An alcohol is reacted with an ester to form a new ester. This process is a special case of catalysed esterification, with strong acids used as catalysts as in conventional esterification, but more common catalysts are sodium alkoxides. These must be used in an anhydrous system otherwise they hydrolyse the esters.
Carboxylic anhydrides are used as acylating agents and are more reactive than the corresponding carboxylic acids in the reaction with an alcohol to form an ester, but less reactive than acyl halides.

High yields of esters can be obtained by acylation of alcohols with carboxylic acid halides.

A metal salt is heated with an alkyl halide to form an ester. It is a slow reaction except at temperatures over 100 °C and is normally carried out in an autoclave to avoid any loss of volatiles. When the reaction is complete the vessel is cooled and water is added. The ester is separated, washed free of salt, and dried or rectified.

Alcohols and nitriles react directly to produce an ester. The nitrile is saponified and then esterified as an acid to form the ester. The ammonia formed must be removed from the equilibrium. This is done using an excess of a strong mineral acid e.g. Sulphuric acid, which also acts as a catalyst. A larger quantity of acid catalyst, higher reaction temperatures and longer reaction times are needed than for simple esterification.

All Ketene is converted into the product using a suitable catalyst.
Type of File: argument
Name of File: argument1
Summary: We have no patent problems.

The company has the patent rights to this process.

Type of File: argument
Name of File: argument2
Summary: It is known and used technology.

This is known and used technology and the company has 20 - 30 years experience of operating this process.

Type of File: argument
Name of File: argument3
Summary: We have easy access to the raw materials.

The raw materials required are readily available. Both Ethanol and Acetic Acid are produced on site.

Type of File: argument
Name of File: argument4
Summary: This is a cleaner process.

There are fewer by products as this is a cleaner process. As a result the plant will need less equipment to deal with by products.

Type of File: argument
Name of File: argument5
Summary: This is unknown technology.

The company has no operating experience of this process.

Type of File: argument
Name of File: argument6
Summary: More hazardous chemicals are required.

Aluminium filings and aluminium chloride are potentially more difficult to handle.
A refrigeration system is required. This increases both the capital and running cost of the plant.

This is a fast reaction so the same amount of ethyl acetate can be produced in less time.

This process requires an anhydrous system which causes handling problems and also increases the cost of the plant.

As water is not produced as a by-product we do not need to design and install equipment that removes it from the product.

The raw materials required by this process are not readily available on site so there will be an additional cost of buying the raw materials or transporting them if they are available at another plant.

This process produces high yields of the product.
Halides are very corrosive. There may be additional costs to the plant as Halides are very corrosive so the equipment must be designed to withstand this or must be protected.

The reaction is slow.

This is not a clean reaction. The cost and size of the plant is increased because more separation stages are needed to remove the waste product and we also have problems with disposing them.

It produces volatile materials. This is an increase risk to the plant and workers and may increase the cost of the plant by increasing the safety measures needed.

There are two stages in this process. This process has two stages so it will take longer and may be more expensive.

Ammonia is produced as a by-product. Ammonia is a by product of this reaction and its disposal will cause problems.
18/10/93
cgrg
Type of File: argument
Name of File: argument19
Summary: More catalyst is required.

As more Catalyst is required the cost will increase.

18/10/93
cgrg
Type of File: argument
Name of File: argument20
Summary: Has a high reaction temperature.

As the reaction temperature is high the reactor will cost more to build and there are also safety problems.

18/10/93
cgrg
Type of File: argument
Name of File: argument21
Summary: Ketene is very reactive.

Ketene is very reactive it is difficult to handle and is dangerous.

19/10/93
cgrg
Type of File: decision
Name of File: decision1
Summary: Explore two possible routes

There are two possible routes that we may feasibly use and should explore. They are:

Acid catalysed liquid phase esterification of ethanol and acetic acid and Condensation of Acetaldehyde by the Tischenko Reaction.

Of these two the first is probably the best because we have no problems with patents, raw materials and energy requirements, however they are both possible routes.

19/10/93
cgrg
Type of File: issue
Name of File: issue2
Summary: Should we use a batch or continuous process

We can either use a batch or continuous process to produce ethyl acetate.

Which is the best type of process to use?
Issue Base Nodes: Viewpoint2

19/10/93
cgrg
Type of File: issue
Name of File: issue3
Summary: What temperature and pressure should we operate under

Given that we are using an acid catalysed liquid phase esterification process to produce the ethyl acetate we need to decide on a temperature and pressure for the process reaction. What is the best pressure and temperature to operate under?

22/10/93
cgrg
Type of File: position
Name of File: position11
Summary: use 90 oC and 1.1 bar

The best conditions to operate under are 90 oC and 1.1 bar. These figures are based on lab experiments.
22/10/93
cgrg
Type of File: fact
Name of File: fact1
Summary: best temperature and pressure is 90°C and 1.1 bar

Lab experiments found that the best results are obtained under a temperature of 90°C and a pressure of 1.1 bar.

This file can contain all the lab experiments calculations and results etc.

22/10/93
cgrg
Type of File: decision
Name of File: decision3
Summary: Operate under 90°C and 1.1 bar

The best temperature and pressure to use is 90°C and 1.1 bar.

These figure were obtained by performing experiments in the pilot plant and the lab

20/10/93
cgrg
Type of File: issue
Name of File: issue4
Summary: Should we use a catalyst?

Is a catalyst needed during the reaction and if so what catalyst should we use?

20/10/93
cgrg
Type of File: position
Name of File: position12
Summary: Use Sulphuric Acid

Sulphuric acid is the best catalyst to use for this reaction

20/10/93
cgrg
Type of File: argument
Name of File: argument23
Summary: It is a tried and trusted method.

Sulphuric acid has been used as a catalyst for this reaction in the past and has proved to be a good catalyst. As a result of using this method in the past we also have operating experience using this method.
Experimental results from the lab have shown that sulphuric acid is the best catalyst to use.

Sulphuric acid is the most economical catalyst to use for the process.

Lab experiments have shown that a catalyst should be used and sulphuric acid is the best catalyst. This is also the most economical catalyst to use and we have operating experience of using it with the process.

During the process we need to separate the final product from the by-products. We will also want to recover raw materials from the by-products.

What type of separation should we use?

Use distillation

Use liquid extraction
23/10/93
cgrg
Type of File: argument
Name of File: argument26
Summary: It is known technology
This is tried and tested technology

23/10/93
cgrg
Type of File: argument
Name of File: argument27
Summary: There are validated calculation methods available

23/10/93
cgrg
Type of File: argument
Name of File: argument28
Summary: Steam is readily available
The steam that is needed for this process is readily available in the existing plant.

24/10/93
cgrg
Type of File: argument
Name of File: argument29
Summary: This process requires less energy.
The energy requirements of this process are lower than those of the distillation process.

24/10/93
cgrg
Type of File: argument
Name of File: argument30
Summary: A suitable liquid is not readily available.
A suitable liquid is not readily available so lab experiments are needed to find a suitable liquid.

24/10/93
cgrg
Type of File: argument
Name of File: argument31
Summary: A solid recovery stage is needed.
If we use liquid extraction a solid recovery stage is needed and this increases the size and cost of the plant.
We will use distillation because we know the technology, we can perform the calculations needed and the steam needed for energy is readily available.

**Issue Base Nodes: Viewpoint4**

Given that we are using distillation as a separation process how many distillation columns do we need?

4 distillation columns are needed.

Column A will be used in the reaction stage and will receive ethyl acetate, water, a small amount of unreacted ethanol and trace acetic acid from the reactor vessel. This will be passed up through the column and then condensed. The condensate will have a composition close to the ternary azeotrope and on cooling will split into 2 phases. The water rich phase is passed onto column D, part of the ethyl acetate rich layer is passed back to column A as reflux and the rest is passed onto column B.

Column B is used to remove water and light organic impurities from the input stream. The heated feed is passed up column B and then condensed. The condensate is passed back into the column as reflux. Light impurities consisting mainly of ethyl formate are allowed to accumulate at the top of column B and are periodically purged. At the base of the column the tail stream is reheated where it vaporises. The remainder of the tails containing the heavy impurities are passed onto the final purification stage in column C.

Column C removes heavy impurities (such as acetic acid carried over from Col A) and produces 99.9% ww ethyl acetate. The feed is passed up column C and condensed. Some of the condensate is returned to the column as reflux and some is cooled and stored as the final product. The tails of column C contains acetic acid, heavy impurities present in the feedstock and products from the side reactions. Part of this is heated and the rest is purged to the heavy impurity storage tanks.
Column D has a water rich input stream containing 5-10% wt ethanol and some ethyl acetate. This column recovers the ethanol and passes pure ethanol back to the feedstock stream.

27/10/93
cgrg
Type of File: fact
Name of File: fact2
Summary: Lab experiments and simulation model results

Using results from lab experiments and model simulations it was decided that 4 distillation columns are needed.

This file can contain all the lab results, calculations and output files from any packages used.

27/10/93
cgrg
Type of File: decision
Name of File: decision6
Summary: Use 4 distillation columns

Our experimental results show that we need 4 distillation columns: A, B, C, and D.

Issue Base Nodes: Viewpoint6

1/11/93
cgrg
Type of File: issue
Name of File: issue7
Summary: What are the specifications for the distillation columns?

Given that we need 4 distillation columns what are the specifications of these columns.

2/11/93
cgrg
Type of File: comment
Name of File: comment1
Summary: There are 4 issues here.

This is in fact 4 issues because each column should be considered on its own.

We should have separate specifications for column A, B, C, and D.

2/11/93
cgrg
Type of File: decision
Name of File: decision7
Summary: Each column should be considered on its own.

There are four distillation columns and each one has a specification
Each distillation column will be discussed separately.

2/11/93
cgrg
Type of File: issue
Name of File: issue8
Summary: What is the specification for column A

What is the specification for column A.

10/11/93
Type of File: position
Name of File: position16
Summary: Specification for column A

| Pressure Spec | Stage 1          | 100,000.0 N/SQM |
|              | Stage 15         | 110,000.0 N/SQM |
| Temperature EST | Stage 1          | 340.00 K       |
|                | Stage 15         | 340.00 k       |
| Top stage Temp | 350.085 K        |
| Bot stage Temp | 363.345 K        |
| Top stage Liquid Flow | 0.036527 Kmol/Sec |
| Bot stage Liquid Flow | 0.036657 Kmol/Sec |
| Top Stage Vapour Flow | 0.11074 Kmol/Sec |
| Bot Stage Vapour Flow | 0.11146 Kmol/Sec |
| No of Plates | 25               |
| Total Height | 15.5 M           |
| Diameter | 0.8 M            |
| No of Ideal stages | 15               |
| Reflux Ratio | 0                |
| Condenser Duty | 0.0              |
| Reboiler Duty | 0.0              |
| Input Streams | 2                |

Output Streams

| Stream 5 (Col A Vapour Feed) |
| Stream 9 (Col A Reflux)      |
| 2                             |
| Stream 6 (Col A Liquid Tails)|
| Stream 7 (Col A Vapour Tops) |

10/11/93
Type of File: fact
Name of File: fact3
Summary: Aspen output for column A

| Pressure Spec | Stage 1          | 100,000.0 N/SQM |
|              | Stage 15         | 110,000.0 N/SQM |
| Temperature EST | Stage 1          | 340.00 K       |
|                | Stage 15         | 340.00 k       |
| Top stage Temp | 350.085 K        |
| Bot stage Temp | 363.345 K        |
| Top stage Liquid Flow | 0.036527 Kmol/Sec |
| Bot stage Liquid Flow | 0.036657 Kmol/Sec |
| Top Stage Vapour Flow | 0.11074 Kmol/Sec |
| Bot Stage Vapour Flow | 0.11146 Kmol/Sec |

E 13
| No of Plates | 25 |
| Total Height | 15.5 M |
| Diameter | 0.8 M |
| No of Ideal stages | 15 |
| Reflux Ratio | 0 0 |
| Condenser Duty | 0.0 |
| Reboiler Duty | 0.0 |
| Input Streams | 2 |

**Output Streams**

2/11/93
cgrg
Type of File: issue
Name of File: issue9
Summary: What is the specification for column B

What is the specification for column B

10/11/93
Type of File: position
Name of File: position17
Summary: Specification for column B

| Pressure Spec | Stage 1 | 100,000.0 N/SQM |
| Top stage Temp | 341.584 K |
| Bot stage Temp | 356.584 K |
| Top stage Liquid Flow | 0.125000 KMol/Sec |
| Bot stage Liquid Flow | 0.031212 KMol/Sec |
| Top Stage Vapour Flow | 0.0 KMol/Sec |
| Bot Stage Vapour Flow | 0.31989 KMol/Sec |
| No of Plates | 33 |
| Total Height | 15.5 M |
| Diameter | 1.4 M |
| No of Ideal stages | 22 |
| Reflux Ratio | 1,000.000 |
| Condenser Top Duty | 877.0 |
| Condenser Side Duty | 1394.0 |
| Reboiler Duty | 2051.0 |
| Input Streams | 4 |

**Output Streams**

Stream 10 (Col B Feed)
Stream 18 (Col B Reflux)
Stream 22 (Col B Side Return)
Stream 25 (Col B Vapour Return)

3
Stream 16 (Col B Vapour Tops)
Stream 20 (Col B Vapour Sides)
Stream 24 (Col B Tails)
10/11/93
Type of File: fact
Name of File: fact4
Summary: Aspen output for column B

**Pressure Spec**

<table>
<thead>
<tr>
<th>Stage</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1</td>
<td>100,000.0 N/SQM</td>
</tr>
<tr>
<td>Stage 22</td>
<td>120,000.0 N/SQM</td>
</tr>
</tbody>
</table>

**Top stage Temp**

- 341.584 K

**Bot stage Temp**

- 356.584 K

**Top stage Liquid Flow**

- 0.125000 KMol/Sec

**Bot stage Liquid Flow**

- 0.031212 KMol/Sec

**Top Stage Vapour Flow**

- 0.0 KMol/Sec

**Bot Stage Vapour Flow**

- 0.31989 KMol/Sec

**No of Plates**

- 33

**Total Height**

- 15.5 M

**Diameter**

- 1.4 M

**No of Ideal stages**

- 22

**Reflux Ratio**

- 1.000.000

**Condenser Top Duty**

- 877.0

**Condenser Side Duty**

- 1394.0

**Reboiler Duty**

- 2051.0

**Input Streams**

- Stream 10 (Col B Feed)
- Stream 18 (Col B Reflux)
- Stream 22 (Col B Side Return)
- Stream 25 (Col B Vapour Return)

**Output Streams**

- Stream 16 (Col B Vapour Tops)
- Stream 20 (Col B Vapour Sides)
- Stream 24 (Col B Tails)

2/11/93
cgrg
Type of File: issue
Name of File: issue10
Summary: What is the specification for column C

What is the specification for column C

10/11/93
Type of File: position
Name of File: position18
Summary: Specification for column C

**Pressure Spec**

<table>
<thead>
<tr>
<th>Stage</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1</td>
<td>100,000.0 N/SQM</td>
</tr>
<tr>
<td>Stage 15</td>
<td>120,000.0 N/SQM</td>
</tr>
</tbody>
</table>

**Top stage Temp**

- 341.584 K

**Bot stage Temp**

- 356.584 K

**Top stage Liquid Flow**

- 0.125000 KMol/Sec

**Bot stage Liquid Flow**

- 0.031212 KMol/Sec

**Top Stage Vapour Flow**

- 0.0 KMol/Sec

**Bot Stage Vapour Flow**

- 0.31989 KMol/Sec

**No of Plates**

- 23

**Total Height**

- 14.6 M

**Diameter**

- 1.0 M

**No of Ideal stages**

- 15

**Reflux Ratio**

- 5
Condenser Duty: 877.0
Reboiler Duty: 2051.0

Input Streams:
Stream 26 (Col C Feed)
Stream 37 (Col C Reflux)
Stream 40 (Col C Vapour Return)
Stage 3

Output Streams:
Stream 35 (Col C Vapour Top)
Stream 39 (Col C Tails)

---

### Specifications for Column D

<table>
<thead>
<tr>
<th>Stage</th>
<th>Pressure Spec</th>
<th>Top stage Temp</th>
<th>Bot stage Temp</th>
<th>Top stage Liquid Flow</th>
<th>Bot stage Liquid Flow</th>
<th>Top Stage Vapour Flow</th>
<th>Bot Stage Vapour Flow</th>
<th>No of Plates</th>
<th>Diameter</th>
<th>No of Ideal stages</th>
<th>Reflux Ratio</th>
<th>Condenser Duty</th>
<th>Reboiler Duty</th>
<th>Input Streams</th>
<th>Output Streams</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100,000.0 N/SQM</td>
<td>341.584 K</td>
<td>356.584 K</td>
<td>0.125000 KMol/Sec</td>
<td>0.031212 KMol/Sec</td>
<td>0.31989 KMol/Sec</td>
<td>23</td>
<td>1.0 M</td>
<td>1.0 M</td>
<td>15</td>
<td>5</td>
<td>877.0</td>
<td>2051.0</td>
<td>Stream 26 (Col C Feed)</td>
<td>Stream 35 (Col C Vapour Top)</td>
</tr>
<tr>
<td>5</td>
<td>120,000.0 N/SQM</td>
<td>341.584 K</td>
<td>356.584 K</td>
<td>0.125000 KMol/Sec</td>
<td>0.031212 KMol/Sec</td>
<td>0.31989 KMol/Sec</td>
<td>23</td>
<td>1.0 M</td>
<td>1.0 M</td>
<td>15</td>
<td>5</td>
<td>877.0</td>
<td>2051.0</td>
<td>Stream 26 (Col C Feed)</td>
<td>Stream 35 (Col C Vapour Top)</td>
</tr>
</tbody>
</table>

---

**Summary:** What is the specification for column D

What is the specification for column D

---

**Summary:** Specification for column D

---

<table>
<thead>
<tr>
<th>Stage</th>
<th>Pressure Spec</th>
<th>Top stage Temp</th>
<th>Bot stage Temp</th>
<th>Top stage Liquid Flow</th>
<th>Bot stage Liquid Flow</th>
<th>Top Stage Vapour Flow</th>
<th>Bot Stage Vapour Flow</th>
<th>No of Plates</th>
<th>Diameter</th>
<th>No of Ideal stages</th>
<th>Reflux Ratio</th>
<th>Condenser Duty</th>
<th>Reboiler Duty</th>
<th>Input Streams</th>
<th>Output Streams</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100,000.0 N/SQM</td>
<td>345.053 K</td>
<td>372.829 K</td>
<td>0.026627 KMol/Sec</td>
<td>E 16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Stream 37 (Col C Vapour Return)</td>
</tr>
<tr>
<td>10</td>
<td>100,000.0 N/SQM</td>
<td>345.053 K</td>
<td>372.829 K</td>
<td>0.026627 KMol/Sec</td>
<td>E 16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Stream 39 (Col C Tails)</td>
</tr>
</tbody>
</table>

---

**Summary:** What is the specification for column D

---

**Summary:** Specification for column D

---

**Summary:** What is the specification for column D
Bot stage Liquid Flow: 0.31406 KMol/Sec
Top Stage Vapour Flow: 0.0 KMol/Sec
Bot Stage Vapour Flow: 0.076740 KMol/Sec
No of Plates: 15
Total Height: 10.9 M
Diameter: 0.5 M
No of Ideal stages: 10
Reflux Ratio: 1.5000
Condenser Duty: -1,730,280
Reboiler Duty: 0.0
Input Streams: 3

Output Streams:
Stream 47 (Col D Feed)
Stream 50 (Col D Reflux)
Stream 52 (Steam Inject)
Stream 48 (Col D Vapour Tops)
Stream 53 (Water Purge Stream)

10/11/93
Type of File: fact
Name of File: fact6
Summary: Aspen output for column D

<table>
<thead>
<tr>
<th>Pressure Spec</th>
<th>Stage 1</th>
<th>100,000.0 N/SQM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 10</td>
<td>100,000.0 N/SQM</td>
<td></td>
</tr>
<tr>
<td>Top stage Temp</td>
<td>345.053 K</td>
<td></td>
</tr>
<tr>
<td>Bot stage Temp</td>
<td>372.829 K</td>
<td></td>
</tr>
<tr>
<td>Top stage Liquid Flow</td>
<td>0.026627 KMol/Sec</td>
<td></td>
</tr>
<tr>
<td>Bot stage Liquid Flow</td>
<td>0.31406 KMol/Sec</td>
<td></td>
</tr>
<tr>
<td>Top Stage Vapour Flow</td>
<td>0.0 KMol/Sec</td>
<td></td>
</tr>
<tr>
<td>Bot Stage Vapour Flow</td>
<td>0.076740 KMol/Sec</td>
<td></td>
</tr>
<tr>
<td>No of Plates</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Total Height</td>
<td>10.9 M</td>
<td></td>
</tr>
<tr>
<td>Diameter</td>
<td>0.5 M</td>
<td></td>
</tr>
<tr>
<td>No of Ideal stages</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Reflux Ratio</td>
<td>1.5000</td>
<td></td>
</tr>
<tr>
<td>Condenser Duty</td>
<td>-1,730,280</td>
<td></td>
</tr>
<tr>
<td>Reboiler Duty</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>Input Streams</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Output Streams</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stream 47 (Col D Feed)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stream 50 (Col D Reflux)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stream 52 (Steam Inject)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stream 48 (Col D Vapour Tops)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stream 53 (Water Purge Stream)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

15/11/94
cgrg
Type of File: issue
Name of File: issue12
Summary: What are the heating requirements for column A

Column A is the first distillation column in the production.
It takes feed from (3) Ethanol at 75 oC and 1.00 BAR and (4) Acetic Acid at 15 oC and 1.00 BAR.
What sort of heating requirements does this column have?

We will have to consider the column feed, the reflux and any other inputs to the column. We should think about how many and what types of heaters are required.

---

One reboiler is required for column A to heat the Ethanol (3) and Acetic Acid (4) streams and to reheat the column A liquid tails (6). Column A reflux (9) does not require any additional heating.

---

A reboiler is required for the reaction process so a reboiler is required for column A.

---

Column B takes its feed from decant A (10) at 20 oC and 1.00 BAR. Column B will require several heaters.

One reboiler is required for the column to reheat the column tails (24). The feed from decant A (10) has a temperature of 20 oC and so heating is required to raise the temperature of this feed.
The side return of column B (22) also needs a heat exchanger to increase the temperature of this feed before it enters the column.

16/11/94
cgrg
Type of File: argument
Name of File: argument33
Summary: The feeders for column B contain organic products

The feeds are organic so cannot be mixed directly with steam to be heated. We don't want to contaminate the product/feeds with water.

Several heaters are required because there are 3 feeders for this column and they all need to be heated.

15/11/94
cgrg
Type of File: issue
Name of File: issue14
Summary: What are the heating requirements for column C

Column C takes its feed from column B tails (26) at 84 oC and 1.20 BAR.

What sort of heating requirements does this column have?

We will have to consider the column feed, the reflux and any other inputs to the column. We should think about how many and what types of heaters are required.

16/11/94
cgrg
Type of File: position
Name of File: position22
Summary: One heater is required for column C

There are three feeds to column C, column B tails (26) column C vapour returns (40) and column C reflux (37).

Only the vapour returns require reheating.

16/11/94
cgrg
Type of File: argument
Name of File: argument34
Summary: The feed contains organic products

The feeds are organic so cannot be mixed directly with steam to be heated. We don't want to contaminate the product/feeds with water.
16/11/94
cgrg
Type of File: argument
Name of File: argument35
Summary: The reflux for column C does not require reheating

The temperature of the reflux (37) is high enough (75 oC) so no heater is required for this stream.

15/11/94
cgrg
Type of File: issue
Name of File: issue15
Summary: What are the heating requirements for column D

Column D takes its feed from col A (11) and col B (23) (joined to produce 47). The pressure and temperature of this feed is 22 oC and 1.00 BAR.

What sort of heating requirements does this column have?

We will have to consider the column feed, the reflux and any other inputs to the column. We should think about how many and what types of heaters are required.

17/11/94
cgrg
Type of File: position
Name of File: position23
Summary: Direct steam can be used to heat column D

We can use low pressure direct steam to heat this column because there is no problem with mixing water into the products.

17/11/94
cgrg
Type of File: argument
Name of File: argument36
Summary: The feed contains mainly water and very little organic compounds

The feed (47) contains over 75 % water so we can use direct steam to heat the column. There is no need to use a heat exchanger and keep the products and the water separate.

18/11/94
cgrg
Type of File: argument
Name of File: argument37
Summary: The reflux does not require reheating

The reflux (50) does not require heating because it leaves the reflux vessel D at 75 oC. No heater is required for this column.