A method for computer-aided hazard identification of process plants

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

Additional Information:

- A Doctoral Thesis. Submitted in partial fulfillment of the requirements for the award of Doctor of Philosophy of Loughborough University.

Metadata Record: [https://dspace.lboro.ac.uk/2134/7279](https://dspace.lboro.ac.uk/2134/7279)

Publisher: © J.C. Parmar

Please cite the published version.
This item is held in Loughborough University’s Institutional Repository (https://dspace.lboro.ac.uk/) and was harvested from the British Library’s EThOS service (http://www.ethos.bl.uk/). It is made available under the following Creative Commons Licence conditions.

For the full text of this licence, please go to: http://creativecommons.org/licenses/by-nc-nd/2.5/

by

Jayesh C. Parmar

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

September 1986

© J.C. Parmar 1986
This thesis is dedicated to my parents

for providing me with opportunities

that they themselves were denied

"Imagination is more important than knowledge"

Albert Einstein

1879 - 1955
ACKNOWLEDGEMENTS

I would like to thank my supervisor, Professor F. P. Lees for valuable guidance and discussion throughout the course of this work.

I would like to thank my colleagues in the Department of Chemical Engineering and the staff of the Computer Centre for their help and encouragement.

This work was jointly funded by the Science and Engineering Research Council and Technica Ltd.
CONTENTS

1 Hazard Identification

1.1 Introduction 1
1.2 The HAZOP Process
   1.2.1 The HAZOP Team 8
   1.2.2 The HAZOP Study Procedure 10
      1.2.2.1 Valid Guideword and Physical Variable Combination 11
      1.2.2.2 The Decomposition of the Design for Study 14
      1.2.2.3 A HAZOP Session 14
      1.2.2.4 Recording and Reporting a HAZOP Study 18
      1.2.2.5 Computer Aids for HAZOP Studies 21
1.3 Literature on HAZOP Studies 21
   1.3.1 HAZOP Study on the Feed Section of a Proposed Olefin Dimerisation Plant 21
   1.3.2 HAZOP Study of a Reactor Section 22
   1.3.3 HAZOP Study of a Gas Drying Unit 23
   1.3.4 HAZOP Study of a Liquid Propane Transfer and Vaporisation Unit 23
   1.3.5 HAZOP Study of Ethylene Oxide Feed System to Batch Reactors 24
   1.3.6 HAZOP Study on an Ethylene Oxide Reactor 25
   1.3.7 HAZOP and its use in the Elimination of Potential Process Hazards 25
   1.3.8 HAZOP Study on a Flammable Liquid Storage Tank 26
## Artificial Intelligence

2.0 Background and History of Artificial Intelligence

2.1 The Architecture of Expert Systems
   2.1.1 The Features of Expert Systems
   2.1.2 Components of Expert Systems
   2.1.3 Languages
      2.1.3.1 LISP
      2.1.3.2 PROLOG
      2.1.3.3 POPLOG
      2.1.3.4 POP-11
   2.1.4 Expert System Shells

2.2 Knowledge Representation Techniques
   2.2.1 Semantic Nets
   2.2.2 Frame Systems
   2.2.3 Production Systems

2.3 Construction of an Expert System

## HAZOP Studies

3.0 The Expertise of HAZOP Studies

3.1 The Different Types of Knowledge
   3.1.1 Knowledge of the Procedures of HAZOP Studies
   3.1.2 Knowledge of Process Equipment
   3.1.3 Knowledge of Process Material
   3.1.4 Knowledge of the Particular System under Consideration
   3.1.5 Knowledge of Hazards
   3.1.6 Knowledge of Probabilities
   3.1.7 Knowledge of Costs

3.2 The Interaction of Different Types of Knowledge

3.3 The HAZOP Checklist

3.4 HAZOP Pruning
3.4.1 Rationalisation by Process Line 67
3.4.2 Rationalisation of Process Units 68

4 Computer Aid for Hazard Identification 69

4.0 Overview 69
4.1 The Different Approaches to Developing a Computer Aid for Hazard Identification 70
  4.1.1 A-Frame System 72
  4.1.2 A Production System 73
  4.1.3 Semantic Networks 74
4.2 Selection of an Approach for a Computer Aid for Hazard Identification 74
4.3 Knowledge Representation Problem 75
  4.3.1 Knowledge Embodied in "unit_model_maker" 79
  4.3.2 Knowledge Embodied in Process Unit Model Library 82
  4.3.3 Knowledge Embodied in "configurator" and the Configuration File 85
  4.3.4 Knowledge Embodied in Process Materials Properties Models 87
  4.3.5 Knowledge Embodied in "identifier" 90
  4.3.6 Knowledge Embodied in "compacter" 97
4.4 Overview of Computer Method 98
4.5 Expert System Features 98

5 Modelling 101

5.1 Modelling of Process Units 101
  5.1.1 Propagation Equations 102
  5.1.2 Event Statements 105
    5.1.2.1 Initial Event Statements 105
    5.1.2.2 Terminal Event Statements 108
  5.1.3 Process Lines and Process Vessels 108
  5.1.4 The Generic Pipe Model 110
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.3.2</td>
<td>Running the &quot;configurator&quot; Program</td>
<td>185</td>
</tr>
<tr>
<td>8.4</td>
<td>Performing a Computer Hazard Identification; Running the &quot;identifier&quot; Program</td>
<td>187</td>
</tr>
<tr>
<td>8.5</td>
<td>Producing a HAZOP-like Document; Running the &quot;compacter&quot; Program</td>
<td>188</td>
</tr>
<tr>
<td>9</td>
<td>References</td>
<td>189</td>
</tr>
<tr>
<td>10</td>
<td>Appendices</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>Listing of &quot;unit_model_maker&quot; Program</td>
<td>193</td>
</tr>
<tr>
<td>B</td>
<td>Listing of &quot;material_modeller&quot; Program</td>
<td>206</td>
</tr>
<tr>
<td>C</td>
<td>Listing of &quot;configurator&quot; Program</td>
<td>209</td>
</tr>
<tr>
<td>D</td>
<td>Listing of &quot;identifier&quot; Program</td>
<td>226</td>
</tr>
<tr>
<td>E</td>
<td>Listing of &quot;compacter&quot; Program</td>
<td>250</td>
</tr>
<tr>
<td>F</td>
<td>Configuration File for Water Separator Example</td>
<td>269</td>
</tr>
<tr>
<td>G</td>
<td>Unit and Process Material Models for Water Separator Example</td>
<td>273</td>
</tr>
<tr>
<td>H</td>
<td>&quot;compacter&quot; Output for Water Separator Example</td>
<td>296</td>
</tr>
<tr>
<td>I</td>
<td>Configuration File for Propane Pipeline Example</td>
<td>322</td>
</tr>
<tr>
<td>J</td>
<td>Unit and Process Material Models for Propane Pipeline Example</td>
<td>327</td>
</tr>
<tr>
<td>K</td>
<td>&quot;compacter&quot; Output for Propane Pipeline Example</td>
<td>358</td>
</tr>
</tbody>
</table>
# INDEX TO FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Sequence of Examination During a HAZOP Study</td>
<td>16</td>
</tr>
<tr>
<td>1.2</td>
<td>HAZOP Worksheet</td>
<td>19</td>
</tr>
<tr>
<td>2.1</td>
<td>Anatomy of Ideal Expert System</td>
<td>34</td>
</tr>
<tr>
<td>2.2</td>
<td>Semantic Network 1</td>
<td>45</td>
</tr>
<tr>
<td>2.3</td>
<td>Semantic Network 2</td>
<td>45</td>
</tr>
<tr>
<td>2.4</td>
<td>Semantic Network 3</td>
<td>45</td>
</tr>
<tr>
<td>3.1</td>
<td>Simple Flow Scheme</td>
<td>58</td>
</tr>
<tr>
<td>4.1</td>
<td>HAZOP Study Knowledge Flow</td>
<td>76</td>
</tr>
<tr>
<td>4.2</td>
<td>Program Schematic Depicting Flow of Knowledge</td>
<td>78</td>
</tr>
<tr>
<td>4.3</td>
<td>Tank</td>
<td>80</td>
</tr>
<tr>
<td>4.4</td>
<td>Centrifugal Pump</td>
<td>81</td>
</tr>
<tr>
<td>4.5</td>
<td>Flow Scheme - Tank to Two Sinks</td>
<td>86</td>
</tr>
<tr>
<td>4.6</td>
<td>Schematic of Reactor Unit</td>
<td>90</td>
</tr>
<tr>
<td>5.1</td>
<td>Section of Process Line</td>
<td>103</td>
</tr>
<tr>
<td>5.2</td>
<td>Pump</td>
<td>106</td>
</tr>
<tr>
<td>5.3</td>
<td>Pipe</td>
<td>107</td>
</tr>
<tr>
<td>5.4</td>
<td>Vessel</td>
<td>109</td>
</tr>
<tr>
<td>6.1</td>
<td>Plant System for Water Separator Example</td>
<td>117</td>
</tr>
<tr>
<td>6.2</td>
<td>Configuration Diagram for Water Separator Example</td>
<td>118</td>
</tr>
<tr>
<td>6.3</td>
<td>Interaction of Causes and Consequences in a HAZOP Study</td>
<td>135</td>
</tr>
<tr>
<td>6.4</td>
<td>Plant System for Propane Pipeline Example</td>
<td>141</td>
</tr>
<tr>
<td>6.5</td>
<td>Configuration Diagram for Propane Pipeline Example</td>
<td>142</td>
</tr>
<tr>
<td>8.1</td>
<td>Open Isolation Valve</td>
<td>170</td>
</tr>
<tr>
<td>8.2</td>
<td>Binary Header</td>
<td>172</td>
</tr>
<tr>
<td>8.3</td>
<td>Simple Closed Tank</td>
<td>174</td>
</tr>
<tr>
<td>8.4</td>
<td>Centrifugal Pump</td>
<td>175</td>
</tr>
<tr>
<td>8.5</td>
<td>Steam Kettle</td>
<td>178</td>
</tr>
</tbody>
</table>
# INDEX TO TABLES

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Some Potential Hazards</td>
<td>4</td>
</tr>
<tr>
<td>1.2 Information Required on Hazardous Materials</td>
<td>6</td>
</tr>
<tr>
<td>1.3 HAZOP Guidewords</td>
<td>12</td>
</tr>
<tr>
<td>1.4 Physical Variables Considered in a HAZOP Study</td>
<td>13</td>
</tr>
<tr>
<td>1.5 Valid Combinations of Guidewords and Physical Variables</td>
<td>15</td>
</tr>
<tr>
<td>3.1 HAZOP Checklist</td>
<td>63</td>
</tr>
<tr>
<td>6.1 Configuration Links for Water Separator Example</td>
<td>120</td>
</tr>
<tr>
<td>6.2 Library Models used for Water Separator Example</td>
<td>121</td>
</tr>
<tr>
<td>6.3 Units and Models of the Water Separator Example</td>
<td>122</td>
</tr>
<tr>
<td>6.4 Comparison of Results for Water Separator Example</td>
<td>123</td>
</tr>
<tr>
<td>6.5 Configuration Links for Propane Pipeline Example</td>
<td>143</td>
</tr>
<tr>
<td>6.6 Library Models used for Propane Pipeline Example</td>
<td>145</td>
</tr>
<tr>
<td>6.7 Units and Models of the Propane Pipeline Example</td>
<td>146</td>
</tr>
<tr>
<td>6.8 Comparison of Results for Propane Pipeline Example</td>
<td>149</td>
</tr>
</tbody>
</table>
1 Hazard Identification

1.1 Introduction

The identification of hazards on process plant is the most crucial stage in Loss Prevention. Modern chemical plants are increasingly complex and, for reasons of efficiency and economy, may need to be run closer to known hazard conditions. Kletz [1] says

"The traditional method of identifying hazards - in use from the dawn of technology until the present day - was to build the plant and see what happens - "every dog is allowed one bite". Until it bites someone we can say that we did not know it would. This is not a bad method when the size of the incident is limited but is no longer satisfactory now that we keep dogs which may kill many people in one bite".

Thus, the identification of hazards has become more critical.

Once a hazard has been identified, steps can be taken to either contain the hazard or eliminate it altogether but this can only be done after the hazard has been identified. The process of identification of hazards is no simple matter, however. Loss Prevention tends, increasingly, to rely on management systems and tools. Lees [2] gives a complete account of the management systems and tools currently available. Lees also indicates the techniques that are most appropriate to the different stages of the design process. These management systems and tools will lay out the procedures, rules and guidelines to follow in order to aid the identification of specific hazards. The major management tools are
1) Safety audits. These subject the activity under consideration to the closest possible scrutiny. The objective of this close examination being to minimize the frequency of loss making incidents and the severity of the loss caused by an accident. The audit appraises the quality of the safety and hazard control efforts of the operation. Wells [3] breaks down a safety audit into the following constituent stages.

- Identify possible loss producing situations
- Assess potential losses associated with these risks
- Select measures to minimize losses
- Implement these measures within the organisation
- Monitor the change

ii) Checklists. These are used as a check on the design to ensure that nothing has been neglected. They are a very essential part of hazard identification and a primary means of passing on and documenting experience. The documenting of evidence is a critical step in the process of Loss Prevention. Kletz [4] says

"If the incident is a trivial one, people forget after a couple of years and the accident is repeated. If the accident is a serious one, the people concerned do not forget but after 10 years or so they have moved on, their successors do not know what happened in the past, and the accident occurs again. "Organisations have no memory". Only people have memories and they move."

There are checklists for a wide variety of
objectives. Wells [3] gives checklists for potential hazards (Table 1.1), information required on hazardous materials (Table 1.2), pre-commissioning operations and a full sample safety checklist for use in plant design.

iii) Hazard Indices. These are a rough guide to the hazards intrinsic in plant and not, strictly speaking, a tool for hazard identification. Indices can, however, indicate or be used to narrow down the area of specific hazards. The process of using hazard indices is one of the summation of the base hazards for each of all the unit operations of the plant and for each of all the process materials present on the plant. The most widely used of these indices is the Dow Chemical Company's Fire and Explosion Index (the Dow Index).

iv) Hazard and Operability Studies (HAZOP). Developed by a team at ICI [5], these are a family of techniques used to review a design. HAZOPs are carried out by a multi-disciplinary team using an approach similar to that of a checklist. HAZOP studies are essentially an application of the technique of critical examination, an early account of which is given by Elliot and Owen [6].

Hazard identification has, traditionally, been conducted on a rather "ad hoc" basis. Design plans for new plant were circulated to the various departments and sections concerned. These sections focused primarily on the aspects that each party or department was most knowledgeable. There was limited consideration of the design as whole, dynamic, working unit. Now, this approach sufficed for designs where ...
Table 1.1 Some Potential Hazards

| Energy source                  | Process chemicals, fuels, nuclear reactors, generators, batteries  |
|                               | Source of ignition, radio frequency energy sources, activators, radiation sources |
|                               | Rotating machinery, prime movers, pulverisers, grinders, conveyors, belts, cranes |
|                               | Pressure containers, moving objects, falling objects |

| Release of material            | Spillage, leakage, vented material |
|                               | Exposure effects, toxicity, burns, bruises, biological effects |
|                               | Flammability, reactivity, explosiveness, corrosivity and fire-promoting properties of chemicals |
|                               | Wetted surfaces, reduced visibility, falls, noise, damage |
|                               | Dust formation, mist formation, spray |

| Fire hazard                   | Fire, fire spread, fireballs, radiation |
|                               | Explosion, secondary explosion, domino effects |
|                               | Noise, smoke, toxic fumes, exposure effects |
|                               | Collapse, falling objects, fragmentation |
Table 1.1 (contd.)

<table>
<thead>
<tr>
<th>Process state</th>
<th>Environmental effects</th>
</tr>
</thead>
<tbody>
<tr>
<td>High/low/changing temperature and pressure</td>
<td>Effect of plant on surroundings, drainage, pollution, transport, wind and light change, source of ignition/vibration/noise/radio interference/fire spread/explosion</td>
</tr>
<tr>
<td>Stress concentrations, stress reversals, vibration, noise</td>
<td>Effect of surroundings on plant</td>
</tr>
<tr>
<td>Structural damage or failure, falling objects, collapse</td>
<td>Climate, sun, wind, rain, snow, ice, grit, contaminants, humidity, ambient conditions</td>
</tr>
<tr>
<td>Electrical shock and thermal effects, inadvertent activation, power source failure</td>
<td>Acts of God, earthquake, arson, flood, typhoon</td>
</tr>
<tr>
<td>Radiation, internal failure, overheated vessel</td>
<td>Site layout factors, groups of people, transport features, space limitations, geology, geography</td>
</tr>
<tr>
<td>Failure of equipment/utility supply/flame/instrument/component</td>
<td>Security</td>
</tr>
</tbody>
</table>
Table 1.2 Information Required on Hazardous Materials

Name of chemical; other names

Formula

Physical state; purity, appearance, odour, detectable concentration

Physical characteristics

Corrosivity

Flammability information

Reactivity information

Toxicity information

Biological properties

Exposure effects

Radiation information
i) the process was simple and well-established  
ii) no new technology was involved  
iii) no new chemistry was involved

However, closer consideration reveals that very few designs actually fall into one of the above categories. All new projects embody some element of change but in the chemical industry the degree of change from one plant to the next is often considerable. It is important to recognise that the body of established experience expressed in codes etc. is limited by the extent of existing knowledge and can only be relevant to the extent to which it is possible to apply it to the new design. In recent years it has become increasingly clear that although these codes of practice are indispensable it is extremely important to supplement them with an imaginative anticipation of hazards in new projects and technology. One such method of examination is the HAZOP study.

1.2 The HAZOP Process

HAZOP studies are now well established as a means of identifying hazards on process plants. Accounts of the basic method have been given by Lawley [5], Kletz [1], Gibson [7] and others [8-12]. There is available a guide [13] and a number of specific studies have been published [14-22]. The HAZOP study process is, in essence, an abbreviated form of "critical examination" based on the principle that a problem can only occur when there is a deviation from intent or from what is normally to be expected. The procedure is, thus, to go through the design systematically in the general direction of flow and consider every conceivable deviation and all the possible causes and consequences of that deviation. Deviations are considered in all the relevant process variables in every single unit in the design.

The study can vary considerably in scope and depth.
depending on

i) the cost and time resources available compared to the possible risk or expense due to downtime of the plant. If the risks posed by the plant are great, then this provides the justification for a thorough study. Similarly, if the cost of downtime is high then expenditure in the form of an extensive operability study is justified. Note that HAZOP studies are also valuable in the commissioning of process plant.

ii) the detail available on the new design. The design may be at any particular stage between a conceptual word model to a detailed piping and instrumentation diagram (P&ID).

1.2.1 The HAZOP Team

HAZOPs are carried out by a multi-disciplinary team of specialists, directed by an experienced group leader. The study team is chosen with care in order to provide experience and detailed knowledge of the design. The choice of team members is critical because the HAZOP process will not make up for any deficiencies or gaps in the knowledge. Thus, the team has to be large enough to provide the knowledge of the design commensurate with the scope and depth of the study. However, the team also has to be small enough to be efficient.

Lawley [14], Fawcett [23], Fitt [24] and Mitchell [25] provide the basis for the qualifications required of the team members and the team leader and secretary. It is desirable, generally, to have one member for each department concerned. This member should be senior enough to make on-the-spot decisions if called upon to do so. Ideally, there will be members of the HAZOP team from other than the project design team. These will provide fresh input that is not, in any way,
influenced by the members of the project team. Other specialist design engineers may be called in to attend particular HAZOP study sessions when requested to do so. This is because design intent may not be perfectly clear from the P&IDs alone and the specialists may be required to provide extra information. For a major new project, the study team would also include representatives from the production, technical and engineering departments together with an instrument engineer to advise on the instrumentation and to deal with any control problems. If new chemistry is involved then the study team should also include a member of the research and development department. A typical study team would comprise the following.

HAZOP leader
HAZOP secretary
Process Engineer (usually a chemical engineer)
Mechanical Engineer
Production Operations Manager or Supervisor
Instrument Engineer
Other specialists (civil engineer, electrical engineer, chemist etc.)

The HAZOP study leader should possess the following qualities
- a technical background
- design and operations experience
- a thorough understanding of HAZOP techniques
- experience as a member and secretary of HAZOP studies
- the ability to plan the study session sequence and duration from prior examination of the P&IDs
- the ability to direct progress through the study schedule whilst stimulating creative thinking by the team members and ensuring the thoroughness of the study.
The Chemical Industries Association [13] says of the study leader

"The study leader has a role to play throughout a study. He should help whoever has commissioned the study to define its scope. He may help in the selection and training of the team. He will advise on the assembly of the necessary data and may help convert this into a suitable form. However, his most obvious role emerges during the examination sessions where he guides the systematic questioning and he must be thoroughly trained for this job. It is not desirable that he should be responsible for making a major technical contribution. If possible he should not have been closely associated with the subject of the study as there is a danger of developing blind spots and failing to use the technique objectively. But he should have sufficient technical knowledge to be able to understand and control the team discussions."

It is usually desirable to have a supporting member of the team to make a note of the hazards as they are detected. This person is known as the study secretary. The study secretary will also take responsibility for ensuring that the HAZOP session runs smoothly. This will entail the collection of all the relevant diagrams and information prior to the start of the HAZOP session.

1.2.2 The HAZOP Study Procedure

Basically, a Hazard and Operability study takes a description of plant and subjects that description to a critical examination. The description may be in the form of a word model or a flow sheet but more often than not consists of a number of P&IDs. The level of detail available in the description will determine the extent of the results of the
study and thus also the time that the study merits. The critical examination consists of questioning every part of the process design to discover how deviations from the intention of the design can occur and deciding whether these deviations can give rise to hazards.

The study is conducted by the team leader systematically asking questions of the team based on a series of guidewords. These guidewords, with explanatory notes, are listed in Table 1.3. The guidewords are used to ensure that the questions, which are posed to test the integrity of the design, will explore every conceivable way in which the design could deviate from the design intention. The guidewords are combined with a set of physical variables (listed in Table 1.4) to provide the questions to be posed of the design. The process of questioning the design consists of taking each guideword and physical variable combination and applying it to the design, one section at a time. Each guideword and variable combination is considered to determine how, if at all, it could be caused and what, if any, the consequences of it would be. Now, the causes of a particular deviation may be deemed to be unrealistic and improbable and in this case the derived consequences are rejected as not meaningful. Similarly, there may be a deviation that has rather trivial or no consequences. In this case the deviation is again rejected. On the other hand, there may be a deviation which has both valid and probable causes as well as potentially hazardous consequences. Here, the deviation together with its causes and consequences is noted for remedial action.

1.2.2.1 Valid Guideword and Physical Variable Combinations

Not all the permutations of the combination of guidewords with the physical variables will be valid. There are some 'specials', mainly for flow. For instance, the combination of the guideword "NONE" with the physical variable "TEMPERATURE"
Table 1.3 HAZOP Guidewords

<table>
<thead>
<tr>
<th>Guideword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE / NO</td>
<td>No flow when there should be</td>
</tr>
<tr>
<td>LESS</td>
<td>Lower flow, temperature, viscosity, pressure etc. than the design intention.</td>
</tr>
<tr>
<td>MORE</td>
<td>More of flow, temperature, viscosity, pressure etc. than the design intention.</td>
</tr>
<tr>
<td>REVERSE</td>
<td>Flow or pressure gradient in the opposite direction to the design intention.</td>
</tr>
<tr>
<td>PART OF</td>
<td>Change in composition of the process stream.</td>
</tr>
<tr>
<td>MORE THAN</td>
<td>Impurities or an extra phase present in the stream.</td>
</tr>
<tr>
<td>OTHER THAN</td>
<td>Consideration of start-up, shut-down, maintenance etc.</td>
</tr>
</tbody>
</table>
Table 1.4 Physical Variables Considered in a HAZOP Study

- Flow
- Pressure
- Temperature
- Concentration
- Level
- Viscosity
will yield the nonsensical deviation "TEMPERATURE NONE". Thus, the combination of the guidewords with the physical variables has to be selective. A list of all the valid combinations is shown in Table 1.5.

1.2.2.2 The Decomposition of the Design for Study

The complete and whole design needs to be broken down into sections for study. The recommended sectioning for a HAZOP study is to consider the design on a "line-by-line" and a "vessel-by-vessel" basis. This decomposition by pipe is natural and logical for a chemical plant. In effect this specifies the size of the section to which the deviations of 1.2.2.1 are applied. The sections are either a single process line, which can contain any number of process units, or a single process vessel. A process line is defined as a number of linked units bounded at either end by process vessels. The order of consideration of these lines and vessels is in the general direction of flow. The Chemical Industries Association [13] gives a flowsheet (Figure 1.1) detailing the sequence of examination during a HAZOP study.

1.2.2.3 A HAZOP Session

The examination sessions are strictly structured with the HAZOP leader controlling the discussion. Dealing with each vessel in turn, the team is asked to explain its function and the explicit intention of each of the process lines connected to the vessel. Deviations cannot sensibly be considered until the function and operation of the equipment is precisely understood. The study leader applies the HAZOP guidewords in turn. The function of the team is not only to provide the answers to the study leader's questions but also to be creative in their thinking. The study leader can inspire this creative thinking by the way he questions the team and sparks off the discussion about a particular deviation.
Table 1.5 Valid Combinations of Guidewords and Physical Variables

<table>
<thead>
<tr>
<th>Physical Variables</th>
<th>More</th>
<th>Less</th>
<th>No</th>
<th>Reverse</th>
</tr>
</thead>
<tbody>
<tr>
<td>More Flow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Less Flow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No Flow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reverse Flow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>More Pressure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Less Pressure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reverse Pressure Gradient</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>More Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Less Temperature</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>More Concentration</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Less Concentration</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>More Viscosity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Less Viscosity</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>More Level</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Less Level</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>More Than</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other Than</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Select a vessel
Explain the general intention of the vessel and its lines
Select a line
Explain the intention of the line
Apply the first guide words
Develop a meaningful deviation
Examine possible causes
Examine consequences
Detect hazards
Make suitable record
Repeat 6–10 for all meaningful deviations derived from first guide words
Repeat 5–11 for all the guide words
Mark line as having been examined
Repeat 3–13 for each line
Select an auxiliary e.g. heating system
Explain the intention of the auxiliary
Repeat 5–12 for each auxiliary
Mark auxiliary as having been examined
Repeat 15–18 for all auxiliaries
Explain intention of vessel
Repeat 5–12
Mark vessel as completed
Repeat 1–22 for all vessels on flow sheet
Mark flow sheet as completed
Repeat 1–24 for all flow sheets

Figure 1.1
Sequence of Examination During a HAZOP Study
When a hazard is detected, it is usually dealt with in one of the following ways.

i) If the solution for a particular hazard is straightforward and local (that is, without ramification for other process plant or streams) then a decision can be taken and the design amended accordingly. It is to be stressed that instant design is to be avoided. Only when the team are absolutely certain that the proposed solution is straightforward and local should the design be amended.

ii) If the suggested solution for a particular hazard is not local (i.e. it may have effects in other process lines or vessels) then that solution is usually further probed to see exactly what the effects elsewhere would be. The proposed solution to the hazard may, in some rare instances, be shelved until the later consideration of another process line or vessel. In this case a detailed note is made to redirect the study team's attention at that stage.

iii) If a question is noted for further evaluation or for further information a note is also made of the person nominated to follow it up.

For the benefit of the secretary as well as the team, the study leader needs to sum up the discussion on each guideword before moving on to the next one. The secretary uses this summing up to check the record of the discussion (see 1.2.2.4). Once a section of pipeline or a vessel or utility has been fully examined, the study leader must mark his copy of the P&IDs to indicate this. This rather simple step will ensure comprehensive coverage of even very complex P&IDs.

The duration of the study sessions is important. Very
long study sessions are likely to lead to the falling off of concentration and of alertness of the team. This may result in hazards going undetected simply because the study team was too tired to have visualised the hazard. The study sessions are very taxing on the team members and so the Chemical Industries Association [13] states that study sessions should be limited to 3 hours and preferably take place in the morning. The rest of the day can be usefully spent by the study team in following up that mornings work. The study leader and secretary usually spend the afternoon recording that mornings work and preparing for the following morning.

A detailed study of the features of and activities in a HAZOP study have been described by Roach and Lees [26], who sat in on one and tape recorded it.

1.2.2.4 Recording and Reporting a HAZOP Study

The main recording effort of a study is the filling in and completion of the HAZOP worksheets. This task is normally performed by the study secretary. A standard format for a HAZOP worksheet is shown in Figure 1.2.

There are two modes of recording.

1) Full recording. Every single deviation is noted regardless of whether or not it has any probable causes and whether or not any hazardous condition arises as a result of that deviation. This is obviously very time consuming and produces a bulky report in which the important information is lost. The one advantage of this form of reporting is that there is a record of the thoroughness of the study.

ii) Recording of only the points requiring action. Only the deviations that are deemed to have probable causes and hazardous effects are noted. This is the
<table>
<thead>
<tr>
<th>GUIDE WORD</th>
<th>DEVIATION</th>
<th>POSSIBLE CAUSES</th>
<th>CONSEQUENCES</th>
<th>ACTION REQUIRED</th>
</tr>
</thead>
</table>

**Figure 1.2**
HAZOP Worksheet
preferred form of recording as it cuts down considerably on the time spent on recording. As only the points requiring attention are noted, the report is concise and there is no effort required to separate the noteworthy from the inconsequential.

The report of a HAZOP study will typically comprise the following sections.

- Summary
- Conclusions
- Scope of work
- Hazop recommendations and queries with the appropriate responses and the comments of the HAZOP study team
- Methodology
- HAZOP study worksheets
- Database list
- Summary of the modes of operation of the plant
- Design changes made as a result of the HAZOP study

It must be emphasised that the above list represents the contents of a very full and complete report. To produce such a report will require significant effort and expenditure after the HAZOP study itself has been completed. It may be decided by project management to dispense with some of the above sections and thus conserve resources.

ICI have developed a computer recording system for HAZOP studies. The function of this system is primarily one of recording although it also serves to prompt the HAZOP team through a full list of the process deviations to be considered. This reduces the burden on the secretary considerably.

There has also recently been a great deal of interest in the follow up work and the documentation of that work after a
HAZOP study. It is rather difficult to keep track of the modifications and the responsibility for those modifications made as a result of the study. There are various procedures that could be followed but the easiest seem to be the processing of a list of records of all the points raised by the HAZOP study. This processing is best done by list or word processing software.

1.2.2.5 Computer Aids for HAZOP Studies

HAZOP studies are themselves a relatively recent feature and thus computer aids for this task are not in great evidence. ICI's computer recording system has already been mentioned in the previous section. Jones and Lihou [27] have developed a computer aid for operability studies (CAFOS). CAFOS aids the transformation of HAZOP studies in the form of cause and symptom equations into pictorial fault trees and provides facilities for the calculation of probabilities in the fault trees. CAFOS is not a computer aid for HAZOP studies as developed here.

1.3 Literature on HAZOP Studies

HAZOP studies have been documented in a number of papers by various authors. Following are some of the more notable accounts.

1.3.1 HAZOP Study on the Feed Section of a Proposed Olefin Dimerisation Plant

One of the team at ICI who pioneered the work on HAZOP studies, Lawley [5] introduces the HAZOP study and details an operability study performed on the feed section of an olefin dimerisation plant. The process involves the transfer of the olefin from storage to a buffer/settling tank where the water impurity is settled out. The olefin passes from the
buffer/settling tank through a feed/product heat exchanger to a dimerisation reactor.

The results of the HAZOP study are not notable in themselves but they do illustrate the principle and also the usefulness of HAZOP studies. For instance, the HAZOP study identifies one of the main problems as poor separation of water in the settler. The causes that are listed are "more flow" and increased levels of the water impurity. The consequence listed is that poor separation of the water would lead to an undesirable amount of water passing through to the reactor section. Lawley also provides much valuable background on the planning and execution of HAZOP studies. Much of this work has already been referred to in this chapter.

1.3.2 HAZOP Study of a Reactor Section

Austin and Jeffries [16] perform a HAZOP study on the reactor section of a process design. This study is unique in that the plant in question operates in a semi-batch manner. Each of the three reactors is considered together with each of the periods and modes in the operating cycle. This application of the HAZOP study procedure illustrates its flexibility. The semi-batch operation requires the amendment of the table of valid guideword and physical variable combinations (see Table 1.5) to encompass the different process operations. The process operations used in addition to Table 1.5 are listed below.

- Dehydrogenate
- Oxidise
- Purge
- Reduce
- Cycle time
- Preheat
- Stand-by
The HAZOP study reveals mainly the problems of leakage; either reverse flow past leaking valves or leakage to the environment. Reverse flow is countered by the addition of non-return valves at the appropriate points. Leakage to the environment is dealt with mainly by the provision of gas detectors and alarms.

1.3.3 HAZOP Study of a Gas Drying Unit

Rushford [15], also of ICI, describes the concept and procedures of the HAZOP study and details part of an actual case study. The study concerns the small section of plant dealing with the gaseous product from a cracker unit. The gas is heated and passed through a suction catchpot before being fed to a compression train. Some 21 points concerning the design were raised by the study team for just this small section of plant. The quality of the study is indicated by the fact that the follow-up of the action report recommended implementation of 16 of the 21 points listed. One especially salient point concerned the redundancy of a whole vessel. This was one of the 16 points implemented and the complete deletion of this vessel resulted in a significant saving in capital cost.

1.3.4 HAZOP Study of a Liquid Propane Transfer and Vaporisation Unit

Lawley and Mitchell [25] originally presented this case study as part of a HAZOP study workshop and this work has subsequently been quoted by Kletz [21]. The workshop was intended to simulate the procedure of an actual HAZOP study. A full P&ID and detailed information on the process is included. This is, of course, done to provide the expert knowledge that the HAZOP study team would normally input. The process plant under study can be roughly broken down into two sections:
propane transfer from main storage to a buffer tank via a 10 mile pipeline and propane feed system to vaporiser. The study raises a number of points in each of the two sections. The basis of most of these points is that the material being handled is propane. A brief list of the main hazardous consequences is as follows.

- Leaks
- Loss of suction to and the subsequent overheating of pumps
- Flashing of the propane causing low temperatures and possible backflow

In addition to the above, there is also consideration of the problems that may arise on start-up or shut-down or due to online maintenance.

1.3.5 HAZOP Study of Ethylene Oxide Feed System to Batch Reactors

In this work, Lawley [14] outlines the procedures of a HAZOP study. The process plant under consideration is the ethylene oxide feed supply to a group of batch reactors. A summary of the results of the HAZOP study is included. As in 1.3.4 the fact that the handling of the process material is intrinsically hazardous provides the basis for most of the consequence scenarios listed. The points requiring action can be briefly classified as follows.

- Leaks
- Loss of suction to and the subsequent overheating of pumps
- Polymerisation and blockages due to the oxide polymer
- Overtemperature / Overpressure due to increased heat of reaction
- Overfilling
1.3.6 HAZOP Study on an Ethylene Oxide Reactor

The work of Piccinini and Levy [20] illustrates the use of a HAZOP study to render a 30,000 t/yr ethylene oxide plant "reasonably safe". They give a detailed account of that part of the HAZOP study which refers to the cooling system of the reactor. This system has been selected as an example of a particularly critical installation. Ethylene is epoxidised to ethylene oxide with pure oxygen on a silver base catalyst. The main undesirable reaction is the complete combustion of ethylene to carbon dioxide and water. The heat evolution of the complete combustion is some eleven times that of the epoxidation. An increase in reaction temperature leads both to a loss of selectivity and the possibility of a reaction runaway. Thus, temperature control of the reaction is of the greatest importance. The cooling system uses an evaporating liquid to maintain a gradient of less than 4 degrees C. in the medium between the inlet and outlet of the reactor shell. There are a number of control loops that are crucial to the safe operation of this cooling system and, not surprisingly, most of the points raised by the HAZOP study are related to these control loops. The integrity of the control loops has been increased by employing redundant components. Control loops have also been added to alarm or control deviations that had originally been ignored or overlooked.

1.3.7 HAZOP and its use in the Elimination of Potential Process Hazards

Kletz [21] gives the general background to HAZOP studies and details the study performed by Lawley and mentioned in 1.3.4 above. Kletz also illustrates the fact that HAZOP studies carried out on seemingly simple systems can be useful. The case of a HAZOP study on the feed system to a distillation column is mentioned. A pump is used to feed liquid from a storage tank to a distillation column. A non-return valve on
the pump outlet and a kickback line with a restriction plate protects the pump against reverse and no flow situations.

Twelve points worthy of mention resulted from the HAZOP study of this system.

Kletz also mentions the use of quantitative methods during a HAZOP study. The need for a second independent trip to protect a compressor is assessed by a rough calculation of the probability of failure of the original trip configuration.

1.3.8 HAZOP Study on a Flammable Liquid Storage Tank

Ozog [22] goes through the techniques of hazard assessment and gives a detailed example of the use of a HAZOP study. The HAZOP study technique is applied to a flammable reagent storage tank. The tank operates under a nitrogen blanket and has associated with it a pump to deliver liquid from the tank to the process. The main problem areas highlighted by the HAZOP study are listed below.

- Overpressure of the tank due to failure or inadequacy of relief or the isolation of the vessel from the relief valve
- Possible implosion of the tank due to failure of pressure control or loss of the nitrogen blanket through maloperation of valves or failure of control valves

Ozog also introduces a system of classifying the points raised by a HAZOP study. After the HAZOP session all notes on the worksheets are identified as category A, which signifies a hazardous event, or B, which signifies a potential hazardous event, depending on resolution of unanswered question regarding operation or design. Once these questions are answered from data on file or by communication with the designer, all B notes are reclassified as As or Cs (non-hazardous events).
2 Artificial Intelligence

2.0 Background and History of Artificial Intelligence

This chapter deals with Artificial Intelligence (AI) and Expert Systems technology and the history of this technology.

Before any attempt is made to describe artificial intelligence we must define or describe intelligence. What is intelligence? With a concept as complex and as nebulous as intelligence it is probably impossible to find a definition that is both complete and exact. Thus, intelligence is best described by enumeration i.e. by listing its characteristic features. The features of intelligence may be summarised briefly by the following list:

- judgement
- comprehension
- reasoning
- concept formation
- response selection
- adaptation
- creativity

This list, by no means complete, illustrates the complexity of intelligence. A truly intelligent being will display all the characteristics listed. Attempts at describing intelligence are vital to the building of artificial intelligence. Thus, artificial intelligence is the conferring of intelligent behaviour on to a machine. Expert systems are defined as particular embodiments of artificial intelligence.

Man's fascination with intelligent machines has stretched back through history. Here, a distinction must be made between the early automatons which were essentially elaborate moving devices and the early so-called calculating machines. The
Early automatons cannot really be referred to as intelligent machines. They were 'programmed' with the use of machinery to perform one set of movements. Mechanical calculators have been in existence for a long time. The early forms of these were the different types of abacus found in ancient China. These mechanical tools, though useful, could in no way be considered to be computers; they had no means of storing and executing an internal set of instructions. The first industrial revolution was essentially one of power, its production and its use. The second industrial revolution was of information and control, information technology.

Charles Babbage, born in 1792, is often hailed as the father of modern computing. Babbage developed two machines: the "Difference Engine" and the "Analytical Engine". These machines demonstrated, in mechanical terms, the components essential to any computer system. These components are

- input of numbers and program instructions
- storage of numbers and program instructions
- arithmetic unit to perform the calculations
- control unit to control task performance under the direction of the stored program
- output of the results of the computation

The next milestone was the introduction, in 1892, by William Burroughs, of the first commercially available adding machine. Following the invention of thermionic valves in the early 1940s it proved possible to use electronic components in digital computing circuits. This led to the development of the digital computer.

Second generation computers, based on the transistor rather than the glass valve were developed in the 1950s. Third generation technology, in the early 1970s, involved the silicon chip containing thousands of transistor elements on
one chip. Denser circuit integration and new languages was termed the fourth generation. In the 1980s, we are seeing the start of the fifth generation of computers in which artificial intelligence is to play the major role.

We have just had a brief history of computers; the hardware. However, the history of artificial intelligence involves both hardware and software. Software advances have almost always, albeit loosely, been linked with hardware advances. As more computer power, speed and memory have become available computer software has become more ambitious. In 1937, Turing published a paper on 'computable numbers' in which the concept of the 'universal Turing machine' was launched. In this, he proposed that a machine could carry out any mathematical procedure, providing the machine was supplied with an adequate instruction table (the equivalent of today's computer program). In 1947, Turing produced a paper entitled "Computing Machinery and Intelligence". This has since proved to have been remarkably prophetic. In the paper, Turing details the 'imitation game', known today as the Turing test. In the game, an interrogator is separated from the machine under interrogation, and communication is only possible using a teletype. If the interrogator cannot tell, through the interrogation, whether the communication is with another human or a machine then the machine may be regarded as intelligent.

In the 1950s AI researchers tried to build intelligent machinery by imitating the structure of the brain. This technique, referred to as a neural net, depended on a richly interconnected system of simulated neurons. It was felt, at the time, that this system could be subject to a training program and eventually end up an intelligent machine. These attempts were doomed to failure; the human brain contains 10 billion neurons, each of which is considerably more complicated than modelled by the neural net.
The following decade led to the notion of heuristic search. The concept of heuristic search came from the belief that human thinking was accomplished by simple symbol-manipulating tasks such as comparing, searching, modifying a symbol and so on. These symbol manipulating tasks were easily modelled on the computers of the day. Problem solving was thought to be a search through a space of potential solutions, guided by heuristic rules which helped direct the search to its destination. Heuristics, or rules of thumb, are necessary in a great many applications in order to reduce the size of the problem search space. Heuristics are not logic but rules that have been developed as a result of experience. The technique is perhaps better described as one where a problem is broken repeatedly into pieces until a subproblem is reached that is small enough to be solved directly i.e. a depth first search (see 2.1.2 below). Heuristic search was probably best known for its applications to computer chess playing.

In the very early 1970s a team led by Edward Feigenbaum at Stanford University began to refine the techniques of heuristic search. The team narrowed their focus. Instead of trying to find a few very powerful general problem solving heuristics they concentrated on the heuristics for a very specific problem - the interpretation of mass spectogram data. This resulted in the first expert system, DENDRAL. DENDRAL was an unprecedented success. It was clear that the solution was to concentrate on a small clearly defined problem and the know-how, the useful tricks and rules of thumb associated with that problem.

MYCIN, produced by Shortliffe was DENDRAL's immediate successor. MYCIN is a computer system which diagnoses bacterial infections of the blood and prescribes drug therapy. MYCIN has spawned a whole series of medical-diagnostic 'clones', one of which is PUFF, a lung function diagnostic tool. DENDRAL and MYCIN introduced several new features which
have since become the hallmarks of the expert system. These features will be elaborated in the next section.

The current trend in AI research has been towards machine learning. Hayes-Roth et. al. [28] define knowledge as facts, beliefs and heuristics. Facts are propositions that are generally held to be true. Beliefs are a particular person's confidence in a proposition that is uncertain. Heuristics are rules of thumb for performing the particular task. An example of a fact is "Today is Monday", whereas a belief may be whether or not it will rain. An example of a heuristic for forecasting the weather is "Red sky at night, shepherd's delight - Red sky in the morning, sailor's warning". Until recently, machine learning has been concentrated on the automatic collection of more facts and beliefs. In the last few years, however, the focus has shifted to include the heuristics so that machine learning now embraces the development of an expert system that improves and extends its own body of heuristic rules, automatically. Douglas Lenat [29-30] of Stanford University has created such a machine learning system, EURISKO.

2.1 The Architecture of Expert Systems

2.1.1 Features of Expert Systems

DENDRAL and MYCIN, the first expert systems, introduced several features new to computer programs. These features are

1) The knowledge consists of hundreds of rules such as the following MYCIN rule quoted by Forsyth [31].

31
IF (1) the infection is primary-bacteremia, and
(2) the site of the culture is a sterile site, and
(3) the suspected portal of entry of the organism is
the gastro-intestinal tract
THEN there is suggestive evidence (0.7) that the identity
of the organism is bacteroides.

ii) The rules are probabilistic. There is uncertainty in the
evidence and the rules. The evidence may be incomplete or
incorrect and the rules may not be very precise. DENDRAL
and MYCIN deal with this probabilistic information with
the use of certainty factors. This allows the formulation
of the most correct conclusions in spite of the
uncertainty in the evidence.

iii) DENDRAL and MYCIN can explain their reasoning processes.
They can explain why they asked a particular question or
how they reached a particular conclusion.

DENDRAL is the prime example of how the software tradition of

DATA + ALGORITHM = PROGRAM

has been changed to

KNOWLEDGE + INFERENCE = EXPERT SYSTEM

Forsyth [31] lists the distinctive features of expert
systems as

1) An expert system is limited to a specific domain of
   expertise
2) It can reason with uncertain data
3) It can explain its train of reasoning in a comprehensible
   way
4) Facts and inference mechanism are clearly separated
   (Knowledge is not hard-coded into the deductive procedures)
5) It is designed to grow incrementally
6) It is typically rule-based
7) It delivers advice as its output - not tables of figures, nor pretty video screens but sound advice
8) It makes money (This is a performance requirement)

2.1.2 Components of an Expert System

Hayes-Roth et. al. [28] give the anatomy of the ideal expert system as shown in Figure 2.1. They describe the ideal system and its components as follows:

"The ideal expert system contains a language processor for problem-oriented communications between the user and the expert system; a 'blackboard' for recording intermediate results; a knowledge base comprising facts as well as heuristic planning and problem-solving rules; an interpreter that applies these rules; a scheduler to control the order of rule processing; a consistency enforcer that adjusts previous conclusions when new data (or knowledge) alter their bases of support; and a justifier that rationalizes and explains the systems behaviour".

Few, if any, expert systems will contain all of the above components but all expert systems will contain many of them.

The list given in the paragraph above can be consolidated into four essential items. These are

i) The knowledge base
ii) The inference engine
iii) The knowledge acquisition facility
iv) The explanation facility
Figure 2.1
Anatomy of Ideal Expert System
These items are discussed in turn.

i) The Knowledge Base

The knowledge base will contain facts and rules. Facts are short-term information on specific topics. This information is likely to change as the consultation with the expert system proceeds. Rules are longer term information on how to handle the facts and how to generate new facts or hypotheses from the existing ones.

ii) The Inference Engine

This is the "how to do it" knowledge or the solution search strategy of the problem. Broadly speaking, there are two overall inference strategies: forward chaining and backward chaining. Basically, forward chaining is the reasoning from the data to the hypotheses whilst backward chaining involves selecting a hypothesis and then attempting to prove or disprove it with the facts. Engineering analogies of forward and backward chaining are event trees (bottom-up) and fault trees (top-down), respectively. The selection of a search strategy will depend on the particular application. It is conceivable even that a mix of both forward and backward chaining may be the best solution for a particular problem.

Given an orientation, forward or backward, there are also several different systematic orders in which the nodes of the solution space may be considered. There are three main types of systematic orders.

1) Depth first. This is a process where successive nodes in the solution space are searched prior to those nodes at the same level.
2) Breadth first. This is a process where all the nodes at the
current level are searched prior to the search of any successive nodes.

3) Heuristic search. Here, information specific to a particular domain is used to guide the search process to a satisfactory solution. This guiding the search process can also be regarded as the selection of rules on the basis of probability i.e. high probability rules are tried first.

As with overall search strategies the selection of a systematic order of search will depend on the particular application.

One requirement of the inference engine may be, depending on the problem, an ability to deal with uncertainty. There are a number of measures of uncertainty; fuzzy logic, Bayesian logic, certainty factors and so on. The choice of measure of uncertainty will, again, be dependent on the problem.

iii) The Knowledge Acquisition Facility

An expert system must be able to learn in order to develop. It is inconceivable that an expert system could be developed that did not need to be refined as a result of experience. Thus, a means by which the expert system can add facts or rules to the knowledge base is essential. Lenat's EURISKO is the forerunner of such machine learning systems.

A mention must be made here of Quinlan's Interactive Dichotomizer 3 (ID3) algorithm as an example of a learning mechanism. When presented with a set of training examples the program operates as follows:

1) Select a subset of the training examples
2) Repeatedly sub-divide the training examples according to the variable with the greatest discriminatory power. Each subset as defined by the most discriminatory variable is
sub-divided again by the next most discriminatory variable and so on. The process of sub-division stops when a subset contains only one class of data. This sub-division gives a decision tree for a new rule.

3) Scan the entire set of training examples to find exceptions to the latest rule.

4) If there are exceptions insert some of the exceptions into the subset of (1) and repeat from (2). If there are no exceptions stop and display the rule.

There are also a number of other learning mechanisms that are either data driven, model driven or heuristic driven.

iv) The Explanation Facility

Michie [32] has warned about the dire consequences of expert systems that do not operate within the "human cognitive window". The explanation facility is essential.

The explanation facility is the user interface. The user should be able to enquire, at any time, why the system has made a particular deduction or asked a particular question. Classic expert systems such as MYCIN have had enormous care exercised over the user interface. Forsyth [31] says "a reasoning method that cannot be explained to a person is unsatisfactory, even if it performs better than a human expert".

2.1.3 Languages

The requirements of a general purpose programming language with which to build expert systems are listed below.

i) Readability
ii) Manageability
iii) Speed and memory
iv) Sufficiently high level to speed up the development process and enable extensions to be added easily

The criteria listed above could be applied to the selection of a language for any computational problem. The fourth item of the list, however, has special significance for the expert systems application of a language. There are now many high-level or fifth generation languages that claim to best fit the list above. Some of the more popular of these languages are described in turn.

2.1.3.1 LISP

The language, List Processing (LISP), originated from research establishments in the U.S.A. in the 1960s. LISP has therefore been around for many years and as a consequence there are several LISP dialects. The different dialects differ primarily in the features offered as standard. The dialects can be divided into two main sub-groups; variants of MACLISP or of INTERLISP. LISP has been used in the implementation of a number of AI products and expert systems. The more notable products in LISP are MYCIN and its offshoots. LISP has developed strongly in the U.S.A. where it is currently the favoured language for AI applications.

There are two different types of machines that run LISP. The first type is the ordinary computer that has a front-end interpreter/compiler that enables programming in LISP. The second type are a special class of machines called symbol manipulating machines. In the latter category the machines usually run LISP and nothing else. The difference between the two categories is that the specialist machines offer extra speed, memory and special features but cost a great deal more.

LISP goes through a standard cycle whenever an entry is made at the terminal. It READS what has been typed, EVALUATES
it and then PRINTS the result. This is universally referred to as the READ-EVAL-PRINT loop. This is not dissimilar to the conventional languages such as FORTRAN or BASIC but is rather different from the PROLOG process described in the next section. Where LISP differs from the conventional languages is the manipulation of symbols. This is a design feature of LISP whereas FORTRAN or BASIC specialise in handling numbers.

2.1.3.2 PROLOG

The language, Programming in Logic (PROLOG), had its origins in Europe in the early 1970s. Clocksin and Mellish [33], of Edinburgh have established a standard PROLOG. PROLOG is now being used by many programmers for applications of symbolic computation and AI. Prolog is, in Europe, the major AI language.

PROLOG is a powerful alternative to LISP as an approach to symbol manipulation, because it is also particularly useful for knowledge representation. PROLOG differs from languages like BASIC and C in that it is declarative rather than procedural. This means that a PROLOG program consists of items of information and rules rather than a series of instructions. Unlike LISP's READ-EVAL-PRINT loops, PROLOG programs consist of sets of clauses, where each clause is either a fact about the given information or a rule about the how the solution may be inferred from the given facts. These basic units of PROLOG are also referred to as HORN clauses which can be either assertions such as

Jack is Jill's brother

or implications such as

IF x is male AND y is male AND x and y have the same parents THEN x and y are brothers

39
The use of these implications and rules, declarative programming, can serve the development of a knowledge base built up in PROLOG. Thus, PROLOG is naturally suited to problems of knowledge representation. This advantage of PROLOG has resulted in its gathering impetus as a language for expert systems applications. Coelho et al. (34) have demonstrated PROLOG solutions to an extremely wide range of knowledge representation problems. There are now available a considerable number of different versions, both interpreted and compiled, of PROLOG.

2.1.3.3 POPLOG

POPLOG is a multi-language system that was initially developed in the Cognitive Studies Programme at Sussex University. POPLOG provides integrated facilities for developing and mixing programs in different languages. The core language of POPLOG is POP-11 (described in the next section) but it also contains incremental compilers for PROLOG, LISP and for other utilities. It is also possible to link programs dynamically in languages such as PASCAL, FORTRAN and C.

2.1.3.4 POP-11

POP-11 is derived from POP-2, a language originally invented at Edinburgh University for research in AI. POP-11 is fully interactive and like LISP and PROLOG does not require separation of data and program, allowing the user to mix factual assertions with program instructions. The syntax and structure of POP-11 is similar to that of the more sophisticated versions of LISP. The main drawback of POP-11 when compared to PROLOG is that POP-11 is not suited to declarative programming, the main strength of PROLOG.
2.1.4 Expert System Shells

There are a number of expert system shells on the market at the moment. Shells are essentially expert system building tools. They contain all the components of the expert system bar the domain specific knowledge base. The best known shell is EMYCIN (or Empty-MYCIN) which is MYCIN without the bacterial infections knowledge of MYCIN. The obvious advantage of using an expert system shell is that there is less to implement. There is a ready-to-run system that only needs the domain specific knowledge added. Shells also ease the task of adding this domain specific knowledge. The disadvantage of shells is the balance between generality and completeness that is struck by the designer of the shell. The designer of an expert system shell will naturally want to build a general purpose tool that will be applicable to a wide range of problems. However, this is done at the expense of efficiency and the number and importance of the features of the shell. Thus, the choice of shell for a particular application will be determined largely by the nature of the application.

Some of the more significant commercially available shells are discussed below.

1) EMYCIN

EMYCIN is the domain independent version of MYCIN. EMYCIN has been used in the production of PUFF, a lung disfunction diagnostic tool. EMYCIN needs the domain specific information supplied in the form of production rules (see 2.1.1 for an example of a MYCIN production rule). The basic control strategy employed by MYCIN is backward-chaining. EMYCIN is also able to handle uncertainties in the form of certainty factors.
2) KAS

PROSPECTOR is an expert system developed for geological diagnosis during mineral exploration. KAS (Knowledge Acquisition System) is the PROSPECTOR program without the domain specific knowledge. KAS allows the use of both forward and backward chaining and also uses a heuristic evaluation of functions to choose the most promising rules. This combination of search techniques has been employed at the expense of limiting the flexibility of the database.

3) HEARSAY-III

The main difference between HEARSAY-III and EMYCIN and KAS is that HEARSAY-III is able to represent and manipulate competing solutions. HEARSAY-III can also reason about partial solutions. The inference engine for HEARSAY-III is extremely complex - it needs to trigger, manage and execute each rule and also needs to schedule in a sensible manner the many different rules that may be vying for attention.

4) AGE

AGE differs from the above tools in that its objective is to provide an environment in which the user specifies the knowledge representation and processing methods. There is a selection of such methods offered by the shell.

5) MICRO-EXPERT

Micro-Expert is a shell based on the PROSPECTOR model. It is therefore a well-tried tool. Micro-Expert is a product of ISI Ltd.
6) SAVOIR

Savoir is a Micro-Expert type shell and also comes from ISI Ltd. Savoir has been used by ICI to develop an expert system, COUNSELLOR, for advising arable farmers on fungal disease control.

The list given above is intended only to be illustrative and not exhaustive. There are new shells launched every month and thus it would be impossible to do a complete review. The major points about expert system shells are that there is a great variety available and that the choice of shell will be determined by its compatibility with the application in question.

2.2 Knowledge Representation Techniques

This section deals with the techniques available for representing the knowledge associated with a problem. This is often the most critical part of obtaining a solution. Designing a good representation is often the key to turning difficult problems into simple ones. There are many forms of knowledge representation that we could choose: propositional logic, predicate logic, semantic networks, analogue representations, frames, semantic triples, production rules, to mention but a few. Three of the above list have found popularity with AI researchers and are described in greater detail in the following sections.

It is advantageous to consider the types of knowledge required by an expert system. Simons [35] describes the various types as
knowledge of objects (knowing that ...)
knowledge of actions and events
knowledge about performance (knowing how to ...)
meta-knowledge (knowing the limitations of our knowledge)

It is important to note that whatever representational scheme is chosen, its suitability for all the types of knowledge related to the problem must be checked.

2.2.1 Semantic Nets

Semantic nets or networks are really acyclic graphs containing nodes and arcs. Nodes represent objects, concepts or situations and the arcs represent the relations between the nodes.

Let us consider an example. The fact that Fred is a boy could be represented by the semantic network of Figure 2.2. Note that the nodes "Fred" and "boy" are connected by the relationship arc "is-a". If we wanted to express the fact that "boy" is an instance of "human", then we could connect the nodes "boy" and "human" with the arc "instance-of". The human attribute "walks on two legs" can be ascribed to "human" with the arc "has-attribute". This network is shown in Figure 2.3. Now, what happens when we have another boy, John, to be added to the knowledge base? We define the node "John" and link it to "boy", using the arc "is-a". The attributes that Fred possessed by virtue of being a boy are now possessed by John as well (Figure 2.4). This illustrates the ease with which the network can be expanded.

One disadvantage of semantic nets is that for complicated systems the networks become unmanageable. Another disadvantage is illustrated by referring back to our last example. What if John was a baby? We may not want the attribute "walks-on-two-legs" ascribed to John but we may want to express the fact
Figure 2.2
Semantic Network 1

Figure 2.3
Semantic Network 2

Figure 2.4
Semantic Network 3
that John is a boy. This would require some restructuring of the network. In a large network such restructuring may prove impossible.

In summary, semantic nets are powerful representational tools (as evidenced by their use in PROSPECTOR) but they do have limitations that preclude their use for certain applications.

2.2.2 Frame Systems

Minsky [36] suggested that a useful way to organise a knowledge base was to break it into modular chunks, called frames. A frame is a data structure for representing a situation. Attached to each frame are several kinds of information including information on which frame to call up next. Minsky pointed out that the human mind operates along lines much like a frame system. This is best demonstrated with the use of an example.

Consider the thought processes of a retailer as he conducts a sale. His knowledge or frame of "conduct a sale" may call up the frames "payment option", "ring up sale" and so on. Now, "payment option" may, if triggered, call up frames on how to deal with a particular kind of credit card, for instance. Thus, the thinking of the human mind can be seen to be fairly easily modelled by a frame system.

As with semantic nets, a frame system comprises a network of nodes and relationships. There is one critical difference, however. Frame systems refer also to the processes involved in performing tasks. This is part of the knowledge base. Frames will often also carry default assignments. These are assumed true unless contradicted. Referring back to the previous example, the retailer may have the "cash" frame as the default to the "payment option" frame. This would be true unless the
customer proffered a credit card.

The advantages of frame systems are their flexibility and the ease with which human knowledge can be modelled.

2.2.3 Production Systems

Much problem solving knowledge can be packaged into small forms called productions. Production systems are often said to model closely human problem solving processes. Production systems consist of a set of conditional statements (rules), a collection of given and derived facts (database) and an interpreter or control strategy that schedules the invocation of the rules. Each rule is regarded essentially as a "pattern-action" or a "situation-action" pair where the pattern or situation is a specified combination of facts.

The rules comprising the knowledge base can have many different formats. The most common format is the IF-a condition-THEN-an action. The THEN section need not be an action; it may be an inference or an assertion. The IF section may contain more than one condition that needs to be satisfied.

The classic example of a production rule system is the much quoted animal identification program. The problem is to discriminate between different zoo animals. The 'expert system' should be able, after querying the user, to identify the particular animal. The production rules to discriminate between the example set of four animals are

1. IF the animal has hair THEN it is a mammal
2. IF the animal gives milk THEN it is a mammal
3. IF the animal is a mammal AND it eats meat THEN it is a carnivore
4. IF the animal is a mammal AND it has pointed teeth AND it
has claws AND its eyes point forward THEN it is a carnivore  

5 IF the animal is a mammal AND it has hoofs THEN it is an ungulate  

6 IF the animal is a carnivore AND it has a tawny colour AND it has dark spots THEN it is a cheetah  

7 IF the animal is a carnivore AND it has a tawny colour AND it has black stripes THEN it is a tiger  

8 IF the animal is ungulate AND it has a long neck and long legs AND it has dark spots THEN it is a giraffe  

9 IF the animal is ungulate AND it has a white colour AND it has black stripes THEN it is a zebra  

Note that there is no need, in this example of four animals, to include the fact that all the animals are mammals or that both the carnivores are tawny coloured. This information is redundant provided that no further animals are to be added to the database. However, there is no requirement insisting that the information in the productions be minimal. It is better to focus on the individual production and include all the facts that seem relevant. Note also that in the example we have two rules for identifying a mammal. This is desirable because it increases the possibility that we may be able to identify an animal as a mammal. Although not illustrated in the example, it is possible to have production rules with multiple conclusions.

Now, if an animal with a tawny colour and dark spots is observed, rules 6 and 8 are suggested or 'triggered'. If the animal is further observed to be hairy and to be eating meat, rules 1 and 3 'fire' establishing the animal as first a mammal and then a carnivore. This fact leads rule 6 to 'fire' showing the animal to be a cheetah. This is forward chaining. The investigation could have been conducted the other way round. For instance, rule 1 would have resulted in the user being asked if the animal had hair (in order to establish the animal...
as a mammal). This is backward chaining, i.e. assuming a hypothesis and then attempting to prove it.

In the event of many matching productions requiring attention, there needs to be a way of deciding which one to choose. This is termed conflict resolution. There are a great many number of ways in which to determine which of many viable productions to act on next. Listed below are four of the more popular methods.

i) Use the matching production that is first in the list
ii) Use the matching production with the longest list of constraints, i.e. the most discriminatory production
iii) Use the matching production that was most recently used
iv) Use the matching production that contains the greatest number of situations with predefined priority

The advantages of production systems are

i) They provide a good model of basic human problem solving
ii) They allow incremental growth through the addition of individual productions
iii) They are easy to update because productions are easily added or removed
iv) They are a very flexible form of knowledge representation
v) It is possible to include heuristic information

The disadvantages of production systems are

i) The program execution is inherently inefficient
ii) In large systems, overlaps may develop between different productions
2.3 Construction of an Expert System

There are two parallel approaches to the construction of an expert system. The two approaches, identical for most stages, differ in that one involves the use of an expert system shell while the other involves building the expert system from scratch. The stages of constructing an expert system are

i) Identifying the requirements of the expert system

ii) Acquisition of the knowledge pertaining to the problem

iii) Selection of either an expert system shell or a knowledge representation technique

iv) Formalising of knowledge

v) 'Feeding' of knowledge to system

vi) Evaluation of expert system

vii) Improvement of expert system

Each of these stages is discussed in more detail below.

1) Identifying the requirements of the system

The first hurdle is to identify a likely application for an expert system. The pertinence of expert systems technology to the problem must be assessed. The scope and extent of the expertise of the problem must be defined. The domain of the expert system must be specified. Let us say we are to develop an expert system to identify zoo animals. What should our expert system be able to do? What range of animals should the system know about? Should the system be able to differentiate between a leopard and a cheetah or an African and an Indian elephant? Is identifying the particular animal all that is likely to be required of the system? Should the system also be able to identify all animals that require a particular habitat? Questions like these will need to be asked in order to define the requirements and domain of the system.

50
ii) Acquisition of the knowledge pertaining to the problem

The problem-solving knowledge of the human expert needs to be collected in a sensible form. This knowledge will include the concepts, relations and information flow characteristics of the problem. All strategies, subtasks, constraints, etc. of the problem solving activity will also need to be specified.

Knowledge acquisition is sometimes performed by a specialist knowledge engineer who elicits the knowledge from the expert. This process of elicitation is accomplished by the knowledge engineer interviewing and interrogating the expert.

iii) Selection of either an expert system shell or a knowledge representation technique

Having decided on the requirements of the system and gathered together all of the knowledge related to the problem, the next step is to decide on an implementation. The advantages and disadvantages of expert system shells have already been discussed (see 2.1.4). If no expert system shell is deemed appropriate, then a technique to represent the knowledge must be decided on (see 2.2). It is also important to decide what explanation facilities are required.

Note that considerations of cost and hardware also have a part to play. It may be that a more expensive shell may be more suitable but that the extra financial outlay is not justified. It is possible that the most suitable shell may not be suited to the hardware that is available.
iv) **Formalising of knowledge**

In the light of the decision of (iii) above, the knowledge of the problem must be converted into a form amenable to the system. This is quite important because subtle parts of the knowledge may need careful formalising. The process of formalisation must also not distort the original bit of knowledge.

v) **'Feeding' of knowledge to the system**

This involves the transfer of knowledge to the expert system. Each bit of the knowledge should be checked on entry to ensure the formalisation process has been correct and exact.

vi) **Evaluation of expert system**

Once the knowledge of the expert system is complete it can be tested. The expert system may require some fine tuning. The evaluation may turn up inconsistencies or deficiencies in the knowledge. Probabilistic information may require alteration in the light of the conclusions to test cases.

vii) **Improvement of expert system**

As the system is exposed to real situations, it may be found that the domain needs to be increased to cover a particular area. Additional heuristic information may become obvious. Additional rules or facts may become known. These additional items of knowledge will need to be assimilated by the expert system, preferably automatically.
3 HAZOP Studies

3.0 The Expertise of HAZOP Studies

The techniques involved in a HAZOP study have already been extensively described in Chapter 1. This chapter will examine the expertise or the knowledge required to perform a HAZOP study and the skills required to manipulate this knowledge.

The results of a HAZOP study are only as good as the knowledge that the HAZOP study team put into it. The fact that the team is formed of members of different disciplines guards against gaps or deficiencies in the knowledge of the individual members.

3.1 The Different Types of Knowledge

There are seven types of knowledge required to perform a HAZOP study. These are:

i) Knowledge of the procedures of HAZOP studies
ii) Knowledge of process equipment
iii) Knowledge of process material
iv) Knowledge of the particular system under consideration
v) Knowledge of hazards
vi) Knowledge of probabilities
vii) Knowledge of costs

3.1.1 Knowledge of the Procedures of HAZOP Studies

This knowledge may be termed "knowledge of control techniques" i.e. how to go about the solving the problem. Included in this category are the following:-
1) Order of examination
2) Meaning and use of the guide words
3) Presentation of the results of the HAZOP study

This knowledge has already been detailed, in some depth, in Chapter 1.

3.1.2 Knowledge of Process Equipment

Process equipment can be divided into two broad classes.

i) Common or everyday items
ii) Uncommon or unusual items

The knowledge of the more common items of equipment will have been built up as a result of the experience of the team members with those items of equipment. Included in this class of common items are:

- piping; different grades and materials
- pumps; different types and brand names
- compressors; different types and brand names
- valves; different functions and types
- non-return valves; different types
- control loops and valves; different functions and types
- instruments and sensors; different functions, types and brands

The above list consists of fairly basic units. The action or function of these units is simple and therefore the malfunction of the units is correspondingly simple. For instance, consider the simplest unit, a section of pipe. The function of a pipe is to carry flow at the expense of a relatively small pressure drop. The malfunction is thus that the pipe is not able to carry the flow or only able to carry the flow at a high pressure drop. The causes of this
malfunction may be that the pipe is fractured or blocked, and so on. Although the information may not seem to be as simply structured as that in the example above, the team will have built up, as a result of their experience, models of these basic units in their minds.

Equipment that the HAZOP team is not closely acquainted with is usually fairly specialist or else incorporates new or unfamiliar technology. Particular designs of the following items will fall into this category:

- Tanks, vessels and associated ancillary equipment
- Heat exchangers
- Distillation columns
- Reaction vessels
- Settlers
- Scrubbers

The members of the HAZOP study team will combine their knowledge of any similar unit with details on the particular unit. These details will be gleaned from the P&IDs, the written design intention, mass and heat balances etc. The study team will thus extend their knowledge to cover the particular application.

The interrogation, during a HAZOP study, of common or basic units and the uncommon units is essentially the same. Each time a unit is considered for causes and consequences of a particular set of guidewords, the role model of the unit is, almost unconsciously, consulted. This consultation consists of questions asked of the models. Examples of such queries are shown below.

"What fault could result in this set of guidewords?"
"What could go wrong with the unit that could result in this set of guidewords?"
"What would be the consequences of this set of guidewords on the unit?"

For the uncommon units, closer attention may need to be focused on the interaction of the unit with the process deviation under consideration.

Knowledge of process equipment is also assumed to include knowledge of the materials of construction. Included in this class will be, for example, the knowledge of the deleterious effects of chlorides on stainless steels.

Knowledge of process equipment is defined, in this instance, as including knowledge of intended operating procedure and knowledge of design intent. The importance of these categories of knowledge is illustrated by the examples below.

i) There is no point in considering the provision of proper isolation of and manholes for entry into a vessel if it is not intended for the vessel to be entered.

ii) It is important to know the basis for sizing of the pressure relief for a process vessel.

3.1.3 Knowledge of Process Material

This knowledge is critical. There is no point in considering fire as one of the consequences of a major leak if the process material is water. An illustrative list of the important properties of process materials has already been given in Table 1.2, Chapter 1. It is to be expected that the physical properties of the process material will have been taken into account when designing the process. It is the effect of process deviations on the material that could lead to hazards. For example, a material may be prone to
polymerisation when static. Thus, no flow could result in polymerisation. The list of Table 1.2 should be supplemented with the items relating to the effects of process deviations on the process material.

3.1.4 Knowledge of the Particular System under Consideration

This knowledge can be passed on to the HAZOP study team in one or more of the following ways.

i) written description  
ii) flow diagram  
iii) mass and heat balances  
iv) logic diagrams (for control and trip systems and for emergency procedures)

The modes of transfer of knowledge given above differ markedly in manner and procedure. Thus, the different modes differ also as to their suitability for the transfer of the various types of knowledge or information.

Written descriptions are useful especially for the transfer of knowledge relating to, among others, the following issues.

- operating procedures  
- sequencing operations  
- material specification  
- design intention  
- philosophy of design  

On the other hand, a written description would not be particularly suited to conveying information about the flow scheme.
The flow diagram is by far the best method of transferring the knowledge relating to the configuration of the process units. Consider the diagram in Figure 3.1. The flow scheme is simple and yet to transfer the same knowledge using a written description would be impracticable. The situation would be considerably worsened by the addition of the instrumentation information that a P&ID would normally contain.

Figure 3.1 Simple Flow Scheme

Mass and heat balances and logic diagrams are obviously fairly specialised and are designed to cope with a specific type of information.

3.1.5 Knowledge of Hazards

This knowledge is actually knowledge of what constitutes a hazard. This type of knowledge is essential to the evaluation of the consequences of any process fault. For instance, consider the heated storage of crude oil. In the absence of recirculation and the presence of water there is the possibility of boilover or slopover. This knowledge is gained through experience, either direct or indirect. Included in this class of knowledge is the expertise that allows the, almost probabilistic, sorting of consequences into probable and improbable hazards. For instance, fire protection against direct flame impingement may be considered necessary with certain light or pressurised hydrocarbons.
Knowledge of hazards can, if so desired, be considered as slightly more specialist knowledge of the three previous types of knowledge i.e.

i) Knowledge of process equipment
ii) Knowledge of process material
iii) Knowledge of the particular system under consideration

3.1.6 Knowledge of Probabilities

This type of knowledge is needed to determine the likelihood of the realization of any hazard potential. Knowledge of probabilities includes statistical knowledge of failure rates, knowledge of the factors influencing the failures (e.g. corrosive process fluids) and the knowledge of how to manipulate individual item failure rates in order to obtain a system failure rate. An example of the use of this knowledge is given below.

Consider a vessel protected against overpressure by a trip. Let us assume the demand rate on the trip is 1/year and the failure to danger rate of the trip is 0.5/year. If the trip is tested monthly the hazard rate is calculated to be once in 48 years. Now, if this is felt to be too high, the test interval may be reduced or some or all of the components of the trip system may be duplicated.

3.1.7 Knowledge of Costs

Cost is a major factor in all decisions about process design and safety in process design. Alternative solutions to a particular problem are usually compared on the basis of cost. Decisions on loss prevention are also subject to such comparisons. Expenditure on containment and prevention of loss
must, in the majority of cases, be justified on the basis of return. Even human life is quite often compared in this way. The "value of a life" criterion is used to compare marginal reductions in the rate of fatal accidents.

Consider again the example of 3.1.6. If the pressure vessel were sited in an extremely remote location where the hazard potential to humans was negligible and the cost of a duplicated trip system outweighed the cost of replacing the vessel then a simple trip system should be chosen.

3.2 The Interaction of Different Types of Knowledge

The seven classes of knowledge listed above each have a central and reasonably well-defined core. The edges of the various classes are, however, rather blurred and there is a considerable degree of overlap between the different classes. For instance, the knowledge of hazards can be readily broken up into specialist knowledge of process equipment, process material, and the particular process configuration. An example of this is the case of boilover or slopover cited in 3.1.5 which can be readily broken down as follows.

Knowledge of process material:
i) Crude oil has water impurity
ii) Water and oil are immiscible
iii) Water is more dense than oil.
iv) Water has a lower boiling point than oil.

Knowledge of process equipment:
i) This crude oil tank is heated by a high pressure steam coil at the base of the tank.
ii) There is a recirculation loop for the oil in the tank.

Knowledge of the particular system:
i) The recirculation loop is inoperative.
The different sets of knowledge above could lead to the following line of reasoning.

Recirculation inoperative AND crude oil has water impurity AND water is denser THUS water settles out.

water settles out AND water has a lower boiling point than oil AND high pressure steam coil at base of the tank THUS temperature of the water raised to boiling point resulting in boilover.

The above example illustrates how a lot of the specialist knowledge of hazards can be broken down. What has been neglected in the above example is the vital role of common-sense knowledge. This has been used at a number of points. e.g.

1) immiscibility and a density difference lead to "settling out"
2) boiling produces large amounts of vapour
3) steam at "high pressure" will be at a temperature that is higher than the boiling point of water.

The decomposition process as shown above is much simpler with the benefit of hindsight and this is because of the importance of seemingly minor bits of common-sense knowledge.

3.3 The HAZOP Checklist

Most HAZOP study leaders employ an "aide-memoire" to ensure that they spark off the thinking of the study team in all the possible directions. These aids are very similar to a
checklist in intention, namely that they attempt to ensure that nothing is omitted. The main difference is that the checklists mentioned in 1.1 are usually meant to be exhaustive whereas HAZOP "checklists" are an open-ended reminder of points to watch out for.

The author has compiled one of these HAZOP checklists by noting the salient points of a number of real-life HAZOP studies. This checklist is shown in Table 3.1. Sitting in on a HAZOP study is the only way to build up expertise of this sort because the documentation of a HAZOP study will not be a complete record of all the discussion. When considering the checklist of Table 3.1 the following points are to be noted.

i) For all the categories apart from OTHER-THAN the intention is to provide a basic list of how the deviation may arise. The consequences of the deviation are not detailed. This is because of the fact that consideration of the consequences is not readily amenable to a checklist form. It is difficult to create a checklist where DEVIATION leads to CONSEQUENCE IF there are ADDITIONAL FACTORS.

ii) The category OTHER-THAN is a guide to the points of note relating to specific items. For instance, under the category of "relief" there is the item "location of discharge". This is intended to mean that where relief is employed for the protection of the process equipment care must be taken over the route of the discharge. If the discharge is to atmosphere, is it pointing away from spots where operators may be affected, and so on.

The American Institute of Chemical Engineers [37] has referred to just such a checklist type approach for HAZOP studies. This particular checklist is, unlike that of Table 3.1, broken down into sections by process unit and process deviation. Separate parts of the checklist refer to pumps,
Table 3.1 HAZOP Checklist

HIGH FLOW
More than 1 pump, pump racing, control loop failure, increased suction pressure, other routes, exchanger tube leaks, greater density.

LOW FLOW
Pump failure or loss of pump efficiency, partial blockage, vessel failures, valve leaks, fouling of delivery, low suction pressure, heat exchanger leaks, control loop failure.

NO FLOW
Wrong routing, blockage, slip plate, faulty non-return valve, pipe burst, control loop or pump or vessel failure.

REVERSE FLOW
(As for no flow), emergency venting, possibility of 2-way flow.

HIGH PRESSURE
Pressure surge, hammer, inadequacy of relief, leakage from high pressure connection, temperature rise, too high a rate of pressuring up.

LOW PRESSURE
Generation of vacuum, pumping/draining out too rapidly, condensation of gases, gases dissolving in liquid, blocked in pump or compressor suction.

HIGH TEMPERATURE
Fouled cooler tubes, cooling water failure, higher rate of reaction, failed exchanger tubes.
Table 3.1 (contd.)

LOW TEMPERATURE
Winter conditions, loss of pressure, loss of heating, failed exchanger tubes.

HIGH VISCOSITY
Change of material/material specification, temperature change.

LOW VISCOSITY
(As for high viscosity)

(PART-OF)
COMPOSITION CHANGE
Passing isolations, leaking exchanger tubes, wrong or off-specification feed, phase change.

(MORE-THAN)
CONTAMINATION
Passing isolations, leaking exchanger tubes, off-specification feed, air ingress, shut-down and start-up conditions.

(OTHER-THAN)
RELIEF
Location of discharge, fire relief.

INSTRUMENTATION
Control loops, pressure, temperature and flow measurement, trips and testing of trips, alarms, set-points.

SAMPLING
Type of sampling, cooling of samples, response time.

CORROSION/EROSION
Material specification, fluid velocities, baffles, pipe-guides, hangers and anchors, external attack.
Table 3.1 (contd.)

SERVICE FAILURE
cooling water, instrument air, steam, nitrogen, power to
machines, control circuits and computers.

MAINTENANCE
Isolation, drainage, purging, cleaning, drying, slip-plating,
access, catalyst change, start-up, foundations and supports.

STATIC
Earthing, low conducting liquids, insulated conductors, fine
strainers, relaxation times, pouring and sieving of dusts.

SPARES
One-off items, catalyst charge, by-passes, stand-by items.

SAFETY
Lagging, fire-fighting, TLVs of process materials, safety
showers, noise and noise protection.
compressors, reactors, furnaces and so on. Consider an excerpt of the checklist shown below. This deals with overtemperature in fixed bed single phase and stirred tank reactors.

"Overtemperature (Often with accompanying overpressure)

* Excessive preheat

* Exothermic reaction
  - Quench failure or loss of external cooling
  - Excess or deficiency of one reactant?
  - Can loss of agitation in cooled stirred reactor lead to excessive temperature/pressure?
  - Could loss of agitation in heated jacketed reactor lead to localised overheating at liquid surface and subsequent runaway?
  - Local hot spot due to partial bed obstruction?
  - Excessive point or surface temperature lead to thermal decomposition or runaway?
  - Delayed onset of batch reaction while continuing reactant addition?
  - Would leakage into reactor of coolant from jacket or internal coil react exothermically?
  - Could backflow of a reactant through a depressuring system lead to or exacerbate a runaway?
  - Excessive preheat drives reaction further?
  - Appropriate normal control/emergency instrumentation provided?
  - Provision for on-line test of emergency cut-off/dump/isolation?

* Regeneration (in place)
  - Maximum regeneration temperature provided for?
  - Too much burn medium i.e. too high concentration?

* (Sufficient TI/THA coverage in beds?)"
It is to be noted that, as in the checklist of Table 3.1, the above checklist is open-ended and utilises a multiplicity of forms. Some of the points deal with the possible causes of overtemperature whilst others deal with the provision of adequate instrumentation and protective devices.

3.4 HAZOP Pruning

This chapter has so far dealt with the knowledge required to successfully complete a HAZOP study. There has, however, yet been no mention of the knowledge required to collate the relevant results of the HAZOP study into a concise form. This "pruning" of the results takes place not at the end of the study but whilst the study is in progress.

One half of the pruning process is, rather obviously, to make a distinction between those points that are noteworthy and those that are not. This is accomplished by utilising knowledge of hazards and knowledge of probabilities to highlight the not improbable hazards.

The second half of the pruning process is the actual trimming of the results. There are two basic types of trimming or rationalisation.

i) Rationalisation by process line

ii) Rationalisation of process units

3.4.1 Rationalisation by Process Line

This process, in essence, is the collecting together of all the causes and effects of process deviations on the sections of pipe between the process units in one process line. This eliminates the duplication that would result if each of the sections of pipe were considered individually.
This may seem an extremely basic point but is actually a very salient one. The process line is thus studied as a collection of units connected together. This is important because in a HAZOP study the objects of study are either process vessels or process lines.

There is one major problem in this type of rationalisation and that is the absence of differentiation between upstream and downstream. Consider the case of a pump with suction and discharge isolation valves. On searching for the causes of high pressure we find that this may be the result of having the discharge valve shut. One consequence of this would be the rupture of the discharge pipe were it not able to withstand the shut-in pressure of the pump. This situation does not arise at the suction end of the pump. Rationalisation by line discards these distinctions in order to present concise output.

3.4.2 Rationalisation of Process Units

This type of rationalisation is the lumping together of the same type of process units within the same process line. A typical process line may contain, say, 20 isolation valves. In searching for the causes of no flow it would be rather tedious to consider "isolation valve closed" at each of the 20 valves. Thus, it is expedient to recognise that there are 20 such process units in the line and the fault "isolation valve closed" could occur at any one of these units.

This type of rationalisation is very similar to that of 3.4.1 and exhibits the same problem with directional information being discarded. Fortunately, this is not critical.
This chapter deals with the method adopted to tackle the problem of hazard identification. Here and hereafter hazard identification refers to the techniques of hazard identification as used in a HAZOP study. Chapter 2 has detailed the different forms that expert systems can take and Chapter 3 has detailed the knowledge requirement of a computer aid in the field of HAZOP studies. It is clear that expert systems technology, rather than conventional programming, is the area of promise in the development of a computer aid for hazard identification.

The process of conducting a hazard identification using this computer method is, briefly, as follows:

i) The "model_maker" program is used to develop a library of models; one for each of the different process units.

ii) The "material_modeller" program is used to develop a library of models of the different process materials.

iii) The "configurator" program is used to create a datafile containing all the configuration information pertaining to the plant under study.

iv) The "identifier" program is then run. This will call up the configuration datafile and the relevant models from the libraries of unit and material models in order to produce the raw results of the hazard identification.

v) The "compacter" program is then used to process the raw results of the "identifier" program to produce a hazard identification table similar to that produced by a
conventional HAZOP study.

Each of the above stages is described in greater detail in section 4.3.

4.1 The Different Approaches to Developing a Computer Aid for Hazard Identification

There are many different ways in which to approach the problem of how to perform the hazard identification using a computer method. These different approaches will all have their respective advantages and disadvantages. The suitability of any one approach can be determined by assessing the particular problems that hazard identification presents. Identifying hazards is an application of expert systems technology quite unlike those discussed in Chapter 2. Some of the peculiarities of this application are listed below.

i) The nature of the goal put to the computer aid is "Identify the hazards on this plant"! This is unlike the sort of goal that most expert systems are set. The conventional form of goal put to an expert system consists of one fairly well defined core. The goal put to a hazard identification program is again well defined but consists of a hierarchy of sub-tasks that are to be performed in a sequence controlled by the inference engine. The main point is that these are sub-tasks and not sub-goals as in a conventional expert system.

ii) The knowledge required to identify the hazards on a plant is not easily divisible into a problem specific knowledge base and a separate control technique/knowledge (inference engine) applicable to a range of tasks in addition to performing the hazard identification. The generality of the control technique has already been referred to (in Chapter 2) as one of the characteristics
of expert systems. An example of this is the expert system "PUFF" which is the combination of "E-mycin", the "Mycin" shell, with a different set of rules. Although this property of generality of the control technique is not a characteristic of all expert systems, it still remains a desirable characteristic - one that is a distinctive feature of many commercial expert systems. Separating the knowledge from the reasoning in hazard identification is a rather complicated task. The reasoning part of the problem constitutes a large proportion of the total knowledge required. Thus, it is only to be expected that the inference engine for hazard identification is inapplicable to any other problem.

iii) Chapter 2 says an expert system should be able to deal with uncertainties in data. This means that an expert system should be able to cope with a "DON'T KNOW" answer to a question. This problem does not manifest itself in a computer aid for hazard identification. The knowledge is either present or assumed to be irrelevant. For instance, if no rules match the search for causes of low temperature at a centrifugal pump then it is assumed that there are no causes of low temperature at the pump.

iv) The majority of conventional expert systems are designed to be run interactively, with explanations when required and the possibility of supplying additional information on the satisfaction of each goal. Performing a hazard identification is a problem on a much grander scale. Although one can envisage an interactive hazard identification session, consider the following points

a) If the quality of the modelling of the process units and process materials is good, there would be no need for the expert system to query the user because the user would not be able to provide any new information.

b) An explanation of the search is not really necessary
c) The results of the hazard identification are most often studied on the completion of the study. There is little need to examine each stage of the search.

v) The search for causes and effects is not one that develops through a great number of levels. The connections between the levels are also relatively obvious. For example, searching for causes of no flow at a centrifugal pump results in the development of "pump fails". This cause of no flow when further developed leads to, say, "impeller failure". The connection between these various levels of the search is immediately apparent.

The peculiarities listed above are some indication of the unusual nature of the application of expert systems techniques to a computer method for hazard identification. Some of the more appropriate approaches to developing such a computer method are listed in in the following sections with their inherent advantages and disadvantages.

4.1.1 A Frame System

It is not difficult to imagine an expert system for hazard identification that is based on knowledge frames. The advantages are that such a system would mimic more closely the thought processes of a human expert. The disadvantage of this approach is that it is speculative. A great deal of information would have to be processed into different frames. There are also a great many ways to structure a frame system such as this. One of the peculiarities of hazard identification listed in 4.1 was that the control knowledge of the problem represented a large proportion of the total knowledge required. One of the disadvantages of frame systems
is the difficulty of representing control information. Thus, a system based purely on frames is considered to be not ideally suited to the task of hazard identification.

4.1.2 A Production System

This would be the purists' approach to the problem. An expert shell would be fed with the rules, or productions, gleaned from a set of conventionally performed HAZOP studies. These productions would then be applied to subsequent hazard identification sessions. The main problem with this approach is that breaking down the results of conventional HAZOP studies and formulating that knowledge into productions is an enormous task. Consider the cases of isolation valves either side of a pump. Possible productions for this system might be as follows.

IF isolation valve shut
AND isolation valve downstream of pump
THEN overheating of pump
AND pressure in line upto isolation valve taken up to end-of-curve pressure of pump.

IF isolation valve shut
AND isolation valve upstream of pump
THEN possible cavitation of process fluid in pump

As the above example illustrates, productions for every eventuality would result in a huge number of rules. Also, the nature of hazard identification is such that predetermination of the extent of the knowledge base is not really feasible. There are infinite variations in the situations proposed by a HAZOP study, so is it really possible to get production rules to cover every eventuality? The answer is that ultimately, yes, it should be possible. The difficulties that can be foreseen are that the development of such a system would take
many, many man-years of effort and that the knowledge base would contain so many rules that control of the rules would pose grave problems.

Production systems have a great many strengths relevant to hazard identification but a system based purely on production rules would be difficult to formulate and utilise efficiently.

4.1.3 Semantic Networks

It may be possible to use a semantic network for hazard identification, but the size of the network so developed would be immense. Controlling and managing the knowledge contained within the network would be impossible. The use of a semantic network is not really viable.

4.2 Selection of an Approach for a Computer Aid for Hazard Identification

It has been illustrated, in 4.1, that the hazard identification problem is unusual. It has also been argued, again in 4.1, that the conventional approaches to developing expert systems are not particularly suited to this task. The solution must be to use the best attributes of the different conventional approaches and, somehow, combine them.

The biggest advantage of production systems is the ease with which control knowledge can be represented. Control knowledge in the case of hazard identification is the knowledge of how to identify the hazards given knowledge of the plant configuration, the process units and the process materials. The major advantage of frame systems, on the other hand, is the ability to encapsulate a frame of knowledge about one particular object. Thus, it is plain to see that frames could be used to represent the knowledge on each of the
process units, on the plant configuration and the process materials.

The solution thus seems to be to employ a production system that knows how to perform the hazard identification calling up and referring to frames that know about the different process units, process materials and the plant configuration. This is the approach that has been taken in this work.

4.3 Knowledge Representation Problem

After having decided, in section 4.2, on a technique to adopt, the next stage is to develop a strategy to implement the technique.

Let us examine the demarcation of the various types and forms of knowledge. Consider, first, the frames that know about the different process units and process materials. Most of these frames will not be peculiar to the study of just one particular plant. These frames will be used in different combinations for different studies. Thus, it would seem that the creation of a library of such frames to be used as and when necessary is the logical next step. Next, consider the frame that contains the knowledge of the process plant configuration. This configuration frame is, stating the obvious, unique and refers only to the plant configuration of the one particular study.

The control knowledge, or knowledge of how to perform the hazard identification is, of course, generic to all the studies. This control knowledge calls up the various frames already referred to, in order to produce the results of the hazard identification. This flow of knowledge, for a HAZOP study, is represented by the schematic diagram of Figure 4.1. The boxes in the diagram denote the different types of knowledge. The "Control Knowledge" box denotes the program
Figure 4.1
HAZOP Study Knowledge Flow
that contains the control knowledge. In this computer method, this program is termed "identifier".

There is one major problem in the creation of the frames that are used by the control knowledge. The user would have to ensure that the knowledge contained in the frames obeys all the rules of grammar and syntax demanded by the control knowledge. It is in order to reduce the tedium of following these rules of grammar and syntax that programs were developed to take in knowledge written in a simple way and "translate" this into knowledge written in the desired form. There are three of these programs.

- "unit_model_maker" to help create the frames containing the knowledge of the process units
- "material_modeller" to help create the frames containing the knowledge of the process materials
- "configurator" to help create the frame containing the knowledge of the process plant configuration under study

There is also a program to convert the results of the hazard identification as produced by the "identifier" program into a form closer to that of a conventional HAZOP document. This program, termed "compacter", has the function of translating between the "identifier" and the end user. The interaction of all these programs and the frames of knowledge is best illustrated by the schematic diagram of Figure 4.2. The arrows represent the flow of knowledge. The computer programs are indicated in italics.

Although the main function of the programs "unit_model_maker", "material_modeller", "configurator" and "compacter" is that of translation they do perform other important tasks. These tasks are described fully in the following sections.
Figure 4.2
Program Schematic Depicting Flow of Knowledge
4.3.1 Knowledge Embodied in "unit model maker"

The "unit_model_maker" program is one that is essentially a translator. The program encompasses the following items of knowledge.

i) There are three basic types of information on a process unit:
   a) propagation equations
   b) initial event (fault condition) statements
   c) terminal event (effect) statements

ii) The procedure for translating each of the three types of information in (i) above into rules acceptable to the "identifier" program.

The function of the "unit_model_maker" program is thus to query the user for the different types of information and then to translate them into rules. These rules are written to an ASCII file to be called up by "identifier".

The first of the three different types of information is the propagation equation. This describes the propagation of process deviations through the unit. For example, one would expect that the deviations in level in a tank would be of the same sign, positive or negative, as the deviations in flow into the tank but be of the opposite sign to the deviation in flow out of the tank. This point is better demonstrated by using the diagram below.
Note that the points 1, 2 and 3 in the diagram are hereafter referred to as ports 1, 2 and 3.

We would expect the following:

- Flow high at port 1 gives level high at port 3
- Flow high at port 2 gives level low at port 3
- Flow low at port 1 gives level low at port 3
- Flow low at port 2 gives level high at port 3

The four transitional statements above could be re-written as one propagation equation.

\[ l_3 = f(q_1, -q_2) \]

The above statement is read as "deviations in level at port 3 are (a function) of the same sign as deviations in flow at port 1 and of the opposite sign to deviations in flow at port 2". This example will have demonstrated the simplicity of using propagation equations to represent the propagation of process deviations through process units.
The two other types of information are initial event statements and terminal event statements. These two categories of information are identical in their structure and will be jointly referred to as event statements. The difference between an initial event statement and a terminal event statement is that the former describes how a fault in the unit initiates a process deviation and the latter describes how a process deviation terminates in a fault at the unit. This point is explained in rather greater detail in Chapter 5. Consider, as an example, the modelling of a centrifugal pump as shown below.

![Figure 4.4 Centrifugal Pump](image)

In the modelling of this pump we may wish to express the following information.

i) no flow will have as causes the failure of the pump motor or the pump impeller (an initial event statement)

ii) low suction pressure will result in cavitation (a terminal event statement)

The two statements above can most easily and clearly be expressed in the following manner.
motor failure, impeller failure > q2 NONE
pl L0 > cavitation

The one problem of using propagation equations and event statements as in the examples above is their unintelligibility to the control knowledge, the "identifier" program. There are two ways to overcome this. The first is to build an equation parser into the control knowledge. The second is to perform the parsing, or translation, in the "unit_model_maker" program. This second alternative is preferable for the following reasons.

i) The logical alternative is to translate the equations and statements into rules as soon as they are input by the user.

ii) (i) also means that errors in inputting the equations and statements are spotted at the earliest possible stage in the process.

iii) The translation routine is rather easier to program into "unit_model_maker" than into "identifier".

Thus, it has proved expedient to build the equation parser into "unit_model_maker".

In summary, this section has detailed the function of the "unit_model_maker" program and has demonstrated the convenience of using propagation equations and event statements in the modelling of process units. Chapter 5 discusses, in greater detail, the modelling of process units.

4.3.2 Knowledge Embodied in Process Unit Model Library

The process unit model library is the term for the collection of the ASCII files created by the "unit_model_maker". Each file represents the sum total of the knowledge on that particular unit.
It is prudent, at this stage, to note the advantages of using such a library. These advantages are listed below.

1) Knowledge on the process units, the models, is retained indefinitely.

ii) The models are called up only as needed by the "identifier" program. The alternative would be to place all the information into one file and have it all called up. This is obviously a ludicrous waste of computer memory and time. Note that this would also vastly increase the search space for "identifier" resulting in greater use of computer time.

iii) The process of addition of new models or amendment of existing models is simple.

The models in the library contain, as stated in 4.3.1, the rules in a form that is comprehensible to the control knowledge. An illustration of this form is given by considering the examples of 4.3.1. The correct form of the propagation equations given for the tank of 4.3.1 as viewed by "identifier" are as follows.

```
rule(tank,p,[1,3],[[q,1]],[[q,2]]).
```

The syntax of the rule conforms to that for PROLOG, the language that "identifier" is programmed in. It is easy, by inspection, to ascribe the various elements of the PROLOG rule to the propagation equation. Given below is the relation between the propagation equation and the elements of the PROLOG rule.

- tank - the name of the process unit model
- [1,3] - a two element list giving the resultant process variable.
[\([q,1]\)] - a list containing two element lists of all the deviations that will give rise to deviations of the same sign in the resultant process variable.

\([\([q,2]\)]\) - a list containing two element lists of all the deviations that will give rise to deviations of the opposite sign in the resultant process variable.

The two event statements of 4.3.1 are as follows.

motor failure, impeller failure
pl LO > cavitation

These statements when translated are converted into the PROLOG rules shown below.

\(\text{rule(pump, functional, } \left[ \text{['motor failure', }'','']\right], \left[\text{['impeller failure', }'','']\right], \left[\text{[q,2,none]}\right]}).\)

\(\text{rule(pump, effect, } \left[\text{[p,l,low]}\right], \left[\text{['cavitation', }'','']\right]).\)

Again, by inspection, it is easy to determine the function of each of the elements of the PROLOG text rules. Consider the second of the two PROLOG text rules shown above.

pump - the name of the process unit model

effect - the category that this particular rule falls into

\([\text{[p,l,low]}]\) - a list containing three element lists giving the causes

\([\text{['cavitation', }'','']\) - a list containing three element lists giving the consequences
4.3.3 Knowledge Embodied in "configurator" and the Configuration File

Basically, the configuration file contains all the configuration information pertinent to the current study whilst the "configurator" program allows the user to create the configuration file.

The program "configurator" first queries the user for the names assigned to the process units of the configuration and the process unit models assigned to those units. For each unit, the existence of the model is verified. The model will not be accepted by the "configurator" program unless it exists in the model library. Once the names and models of the process units have been input, the program asks for information on the connections between those units. This information is input in the form of links, one at a time, between the inlet and outlet ports. For instance, the fact that the outlet of a pump (port 2) is connected to the inlet of a non-return valve (port 1) is expressed by the link below.

pump 2  non-return valve 1

After all this information is input, "configurator" divides the units into process lines and process vessels. The purpose of this division is to replicate the manual HAZOP study which proceeds on a line-by-line basis. The foundation of the division into process lines and process vessels is the definition of a process line. For the purposes of this computer method the definition of a process line is as follows. A process line is a number of in-line units connected together and bound at either end by a vessel. Vessels are defined, in this computer method, as units that either contain an inventory or have other than just one inlet and one outlet. Thus, a divider is considered to be a process vessel on the basis of its two outlet ports. This is illustrated by Figure
4.5. According to a manual HAZOP study this diagram would be considered to have one process line; the one from the tank to the sinks. For this computer method the diagram is said to be composed of three process lines. These are listed below.

i) from the tank to the t-piece (divider)
ii) from the t-piece (divider) to sink 1
iii) from the t-piece (divider) to sink 2

![Figure 4.5 Flow Scheme - Tank to Two Sinks](image)

The "configurator" program calls up files containing information on the process unit models in the library and uses this to convert the keyed-in information into process lines. This simplifies and reduces, considerably, the information that the user needs to provide to "configurator".

"configurator" also queries the user for the process material carried within each process line and for information relating to the materials of construction. The materials of construction information is treated as part of the generic pipe unit (see 4.3.5 (v)). Information on the different impurities referenced in the configuration is also requested. This just requires the user to type in the impurities that may occur in the configuration.
Summarising, the "configurator" program embodies the knowledge required to perform the following functions.

i) Query the user for and accept information on the process units and the links between those units.

ii) Convert the information gleaned in (i) above into knowledge of the process lines in the configuration.

iii) Query the user for knowledge on the process material carried in the process lines.

iv) Query the user for the consequences of process deviations on the materials of construction.

v) Translate the knowledge of (i), (ii), (iii) and (iv) above into a form amenable to "identifier" and write this to a configuration file.

4.3.4 Knowledge Embodied in Process Materials Properties Models

Each of the process materials properties models contains all the knowledge relevant to that particular process fluid. The advantages of using a library of models as opposed to just a single file containing the knowledge of all process fluids are analogous to the advantages concerning a library of process unit models. These have already been expounded on in 4.3.2.

The importance of process material models is easily demonstrated. Let us say the "identifier" program is searching for the consequences of a leak of the material. It would be possible for "identifier" to have a prepared list of the consequences of a leak; say, fire hazard, explosion hazard etc. Now, this would be fine if the process material under consideration is propylene. But, say the process material was water. It would be totally wrong to flag the consequences of a leak of water as a fire or explosion hazard. The problem of differentiating between the two cases is obviously best
overcome by the use of models of the process materials.

Each of the material models contains the information on the consequences, on the material, of each of the process deviations. Thus, for example, the following items of knowledge would be contained in the process material models.

- low temperature on water may lead to freezing
- a leak of gasoline may lead to fire
- low pressure on liquid propane may lead to the flashing off of the propane
- blockage in a line containing olefin may be the result of polymerisation
- blockage in a line containing sea-water may be caused by entrained sea-weed

It is important to note that the items above form two different types of rules. The first three examples are those of the effects of a process deviation on the process material. The last two examples are those of the properties of the material which may result in one of a predefined set of conditions. This predefined set of conditions is described later in this section. The above examples demonstrate that process material models do perform an essential role.

The process material models need, as do the process unit models and the configuration file, to be in a form legible to "identifier" and in a form acceptable to PROLOG, the language that "identifier" is coded in. As before, to save the user having to follow unfamiliar rules of syntax and grammar, a FORTRAN program, "material_modeller" has been developed. This will prompt the user for the definition of the relevant properties of the material and output these, in the correct form, to an ASCII file. This file is then called up by "identifier".

88
The "material_modeller" program contains the following items of knowledge.

i) Knowledge of a set of keywords. This is the predefined set of conditions referred to above (see also 8.2.1). An illustrative list of keywords is given below:

- blockage
- cavitation
- flashing
- boiling
- freezing
- vaporisation

These keywords are used by "identifier" in the following way. When searching for causes of a particular deviation each cause is checked to see if it belongs to the set of keywords. If the cause does belong to the set then the material model is checked for causes of the keyword. Note that the list of keywords can be changed readily by adapting "identifier" and "material_modeller".

ii) Knowledge of the process deviations that are studied by "identifier". This knowledge is used to prompt the user for the effects of those deviations on the process material.

iii) Knowledge of how to convert the information input by the user into a form suitable for "identifier".

Modelling of process materials is discussed in greater detail in Chapter 5.
4.3.5 Knowledge Embodied in "identifier"

The program "identifier" represents the control knowledge of the hazard identification problem. This is the "how to do it" knowledge of the problem. In theory, "identifier" should contain simply the knowledge associated with the process of identifying the hazards. In practice, "identifier" also contains bits of knowledge that enable it to perform the identification in a rather more efficient manner. These additional bits of knowledge could, thus, be termed heuristics or rules of thumb allied to the control knowledge for the hazard identification. Note that some of these heuristics are actually concerned with modifying the approach of the conventional HAZOP study procedure for use in a computer method. The significant items of knowledge embodied in "identifier" are listed below.

1) The process of performing the hazard identification. Conventional HAZOP studies adopt the approach where first, a vessel is selected, then all the lines off the selected vessel are studied and then the selected vessel is studied. Now a human team may drop this systematic approach if they happen on one particularly hazardous or notable event. Consider the diagram below.

![Figure 4.6 Schematic of Reactor Unit](image_url)
If, in a conventional HAZOP study, the team were examining the glycol line to the reactant preheater then under NO FLOW the team would cite as possible consequences the following :- reactant stream temperature low, poor conversion in reactor, high concentration of reactant in product stream. Notice that the study team have jumped from line to vessel to line in their consideration of the consequences. This is because the team have spotted something important and followed it through. After the documentation of this set of consequences the team revert to the original point of study, the glycol line to the preheater. On the other hand, a computer program cannot, realistically, be expected to mimic this behaviour. The procedure followed by the computer method is to save the consequences at the point at which they propagate from the current scope of study to another line or vessel. Referring back to the above example, this logical approach would proceed as follows.

Considering the glycol line to the preheater:
no flow has consequences no flow through heat exchanger and low temperature of reactant in heat exchanger

Considering the heat exchanger:
low temperature out of reactant stream has cause no flow in glycol line

Considering the reactant line between the heat exchanger and the reactor:
low temperature has cause low temperature out of heat exchanger
low temperature has consequence low temperature in reactor
Considering the reactor:
low temperature has caused low temperature in inlet to reactor
low temperature has consequences poor conversion and high concentration of reactant

Thus, "identifier" has a facility that allows it to save the consequences at the ends of process lines and the outlets of process vessels. These saved effects are called up when the corresponding line or vessel is being studied. This process of saving and recalling the consequences is further described in (vi) below. In summary, this item of knowledge allows "identifier" to break down and systematically study the lines and vessels of a process plant configuration.

ii) The knowledge of what files or models to call up. This applies to both process material models and to process unit models. Strictly speaking, this knowledge is actually transferred to "identifier" in the configuration file. The user is prompted by "identifier" for the title of the current study. "identifier" then calls up the configuration file for that study. The configuration file is then consulted for the lists of models, process unit and process material, to call up.

iii) Knowledge of process deviations. This exists in the form of a list of the process deviations that "identifier" is to use in the HAZOP study. The process deviations are divided into two types: those that are applied to just process vessels and those that are applied to process lines as well as process vessels. In the first category are the deviations concerned with level; "level high" and "level low". In the second are the deviations dealing with flow, pressure and temperature. "identifier" actually distinguishes between these two
types of deviation in their application. Clearly, "high level" cannot be applied to a process line. The full list of process deviations that "identifier" goes through is given below.

<table>
<thead>
<tr>
<th>Vessel process deviations</th>
<th>Vessel and line process deviations</th>
</tr>
</thead>
<tbody>
<tr>
<td>level high</td>
<td>flow none</td>
</tr>
<tr>
<td>level low</td>
<td>flow low</td>
</tr>
<tr>
<td>pressure high</td>
<td>flow high</td>
</tr>
<tr>
<td>pressure low</td>
<td>pressure high</td>
</tr>
<tr>
<td></td>
<td>pressure low</td>
</tr>
<tr>
<td></td>
<td>temperature high</td>
</tr>
<tr>
<td></td>
<td>temperature low</td>
</tr>
<tr>
<td></td>
<td>escape</td>
</tr>
</tbody>
</table>

The above list contains an immediately noticeable item - one that is not present in the conventional lists of process deviations. "Escape" is not a process deviation as such but is used in this computer method to overcome the problem of modelling leaks. Leaks can have two effects. One is to reduce flow in the containment and the other is to contaminate the low pressure side, usually the atmosphere. Leaks that are large enough to cause "flow none" or "flow low" can be dealt with by the appropriate process deviation. However, the handling of the contamination effects of leaks is not straightforward with the standard set of process deviations. Here, there is a distinction between contamination of another process fluid and contamination of the atmosphere. The former should be built into the equipment model for any equipment where there is potential for such contact - heat exchangers are the obvious example. For the latter, a new deviation, "escape", is the solution. The causes of "escape" are,
thus, the leak faults in the equipment models. The consequences for the deviation "escape" are on the process material itself. For instance, if the material is flammable, the consequences of escape would be "possibility of fire".

iv) How to search for causes and consequences. This knowledge is that required to search through the rules on a particular process unit for causes and consequences of a process deviation. This includes the knowledge of the structure of the rules in the models of the process units and the process materials. Note that there are differences in the way that different process deviations are handled. This point has already been touched upon in the previous section but it is profitably reiterated here. The deviations in level can only sensibly be applied to vessels and thus the application of the level deviations is so restricted. Considering the deviation "escape" it is readily apparent that although the causes of "escape" are to be found in the equipment models, the consequences will be found in the appropriate process material model. Thus, the search for the consequences of "escape" is restricted to the process material model.

The search for the causes or consequences develops through as many levels as necessary. For instance, in the search for causes of "low flow" at a pump we may go through the following stages.

"pump efficiency loss" leads to "flow low"
"pump blockage, impeller damage, motor damage" leads to "pump efficiency loss"
v) The use of a generic pipe unit. This counters the problem of how to deal with the causes and consequences that are general to the process line as a whole. For example, a cause of low temperature in a process line may be cold weather. However, this is a cause of low temperature also at every other unit in the process line. There is no point in flagging this cause at every unit and swamping the output with needless duplication. A generic unit, the pipe unit, takes care of causes and consequences that are common to the process line. The unit model library contains a generic pipe unit. Note that "identifier" automatically consults the pipe unit for every single process line studied (see (vi) below).

vi) The search along a process line. This section deals with the procedure that is followed when searching a selected process line. The source vessel is first considered for causes and consequences of the process deviation. The source vessel is also checked for any effects of deviations in previously studied process lines that may have been saved at the source vessel. Next, the search down the process line itself. This commences with the search on the generic pipe unit. This will take care of any causes and consequences common to the whole process line (see (v) above) and to the materials of construction model (see 4.3.3). Then, the units in the process line are studied, in turn, all the way down to the drain vessel. At the drain vessel, the vessel is first searched for causes and consequences of the deviation, then the propagation rules at the vessel are used to save those effects of the process deviation that are themselves process deviations. These saved effects will be called up by a later search. The final element of the search on a process line is a check on whether or not the process deviation will have any consequences on the process material contained within the line (see
(viii) below).

(vii) Keywords for causes. There is a set of keywords that is used to model the process material (see 4.3.4 and 8.2.1). If, during the search for causes of a particular process deviation at a unit, a cause is matched to one of the keywords then the process material model for that line is consulted. This consultation consists of a check for any properties of the process material that may give rise to that particular keyword.

(viii) Searching for the consequences of the process deviation on the process material. Part of the search on the process line (see (vii) above) is a check on the possible effects of the process deviation on the process material. The process material model (see 4.3.4) is consulted for the existence of rules that identify the effects of the process deviation on the material.

(ix) Output of the results of the search. There are three output options available. These are:

a) Output to the terminal for the immediate perusal of the user
b) Output to a file to be used by "compacter"
c) Combination of (a) and (b) above

Options (b) and (c) result in the creation of a file that is used by "compacter" to produce a HAZOP-like document (see 4.3.6). Options (a) and (c) allow the results of each stage of the search to be printed out directly after each stage is completed. The output to the terminal is intended only as a tool for development and debugging. Option (b) is the usual choice.
4.3.6 Knowledge Embodied in "compacter"

The program "compacter" converts a file produced by "identifier" into a document closely resembling a HAZOP worksheet. This conversion to a HAZOP-type document involves considerable pruning and manipulation of the raw results produced by "identifier". These pruning operations are more extensively described below.

i) Deleting items that appear more than once. Due to the nature of the search strategy employed by "identifier", it is quite often the case that the same entry is repeated under a list of causes or consequences of a process deviation. This is obviously undesirable and "compacter" checks the list of causes and consequences of each process deviation and deletes the duplicated items in each list.

ii) Sorting each list of causes or consequences so that units that are of the same type or model have only one entry in the list. This point is clarified as follows. A process line may, along its length, have a number of units that are exactly the same. The best illustration of this is the fact that a process line may have a great many isolation valves built into it. There is no point in, say, flagging the cause of the deviation "no flow" as "isolation valve shut" at every single one of the valves. It is better to collect together all the valves and print them out with the one entry "isolation valve shut". This is done by "compacter" not only for "no flow" and isolation valves but also for the other sets of variable deviations and the different types of process units.
4.4 Overview of the Computer Method

This computer method is actually a suite of programs. Consider again the diagram of Figure 4.2. There are two model generation programs: "unit_model_maker" and "material_modeller". The first receives the unit model data from the user and generates a set of rules for the hazard identification program "identifier". This set of rules is stored in a library of such models. The second performs a similar task, but for the process material. The program "configurator" receives from the user the plant configuration data, including the specification of the unit models and process material models. Data on the materials of construction is also entered into "configurator".

"configurator" produces as its output a configuration file for use by the hazard identification program "identifier". "identifier" calls up the unit and process material models from the libraries and generates as output, lists of variable deviations and associated causes and consequences. This hazard identification data is used by the program "compacter" to produce the hazard identification table.

"identifier" is written in PROLOG and the other programs in FORTRAN 77. The programs have been developed and run on a Honeywell Multics DPS8.

Listings of all the programs can be found in the appendices.

4.5 Expert System Features

Even though this chapter has concentrated on an expert systems type approach to producing an aid to hazard identification, this computer method has not been developed
explicitly as an expert system. Following expert systems philosophy has resulted in this computer method displaying some of the features of an expert system, albeit of a special type. Chapter 3 describes an expert system as comprising, typically, the following :-

i) Knowledge base
ii) Inference engine
iii) Knowledge acquisition facility
iv) Explanation facility

These features are discussed in turn.

i) Knowledge base : Yes, there is a knowledge base although it exists in a rather unusual form. The knowledge base consists of the model library and the plant configuration.

ii) Inference engine : Yes, an inference engine does exist. This is actually the "identifier" program which contains the rules and procedures for the manipulation of the knowledge base.

iii) Knowledge acquisition facility : This computer method does not have a direct way of increasing its knowledge about the problem. New heuristics for the manipulation of the knowledge base cannot be "acquired" by the program itself. However, there does exist a facility for expanding or renewing the knowledge base, the library of unit models. New models can be added to the library and old models may be changed. Thus, there is a knowledge acquisition facility, of sorts.

iv) Explanation facility : There is no overt explanation facility. Once the hazard identification table is produced, the reasons for the listings of causes and
consequences are usually fairly clear. The search for causes and consequences of the process deviation is a simple one; one that does not employ a great depth of search. The process variable deviation is also the explanatory link between the tables of causes and consequences. Thus, it is highly improbable that the results of the search would need further explanation or justification especially if, as in this case, the intended audience is knowledgeable about the domain.
5 Modelling

This chapter deals specifically with the philosophy of the modelling of all the different components required by this computer method. The actual practicalities of modelling, using the different modelling programs, are described in the User Manual, Chapter 8. There are four different types of models. These are process units, process materials, materials of construction and plant configuration.

5.1 Modelling of Process Units

Process units are divided into two classes, in-line units and vessels. For the purposes of this computer method in-line units are defined as having only one inlet and one outlet and have negligible inventory. All the units that cannot be classified as in-line are termed process vessels. The peculiarities of process vessels will be discussed in a later section of this chapter (5.1.3).

The model of a process unit describes the propagation of process variable deviations through the unit and the initiation and termination of these deviations at the unit. The transmission of process deviations through a healthy unit is described by the use of propagation equations. The initiation and termination of process deviations at the unit are described by a set of event statements.

The modelling of fault propagation involves the representation of the propagation of input process variable deviations and also of the initiation and termination of these deviations. Generally, fault initiation involves the relation

Initial fault : Process variable deviation

fault propagation the relation
Process variable deviation : Process variable deviation

and fault termination the relation

Process variable deviation : Terminal fault

The initial and terminal faults are respectively the cause and consequence events.

Process units may have one or more inlets and one or more outlets. These inlets and outlets, or ports, need to be uniquely identified. Thus, each port is assigned a number. In-line units normally have port 1 assigned to the inlet and port 2 assigned to the outlet. The numbering of ports for process vessels is discussed in 5.1.3.

5.1.1 Propagation Equations

Propagation equations are functional equations that describe the relation between an output process variable of a unit and the other output and inlet process variables. This relationship is described, not in terms of exact mathematical equations but just positive and negative deviations. For example if \( z \) is an output variable and if \( x \) and \( y \) are the two input variables which affect \( z \) such that an increase in \( z \) is caused by an increase in \( x \) or a decrease in \( y \) (or vice versa), the functional equation is

\[
z = f(x, -y)
\]

The above equation has been simplified by the omission of the port numbers. If the unit was an in-line unit with port 1 as its inlet and port 2 as its outlet then we would have the propagation equation
Propagation equations for in-line units are used in this computer method to describe the propagation only in the direction of flow. The reason for this is that when searching for causes and consequences of process deviations the process line is treated as a whole. It does not matter whether the unit where the fault was initiated, unit \( x \), is upstream or downstream of unit \( y \). The link between the causes and consequences listed for a particular process variable deviation is the variable deviation itself. In general, most of the causes result in most but not all of the consequences and most of the consequences result from most but not all of the causes. This point is clarified with the aid of an example. Consider the section of process line shown in Figure 5.1.

![Figure 5.1 Section of Process Line](image)

The search for causes and consequences of no flow along the line will result in the following lists.
Causes

valve directed shut
pump stopped
orifice blocked
pipe fracture

Consequences

pump overheats

Now, clearly, if no flow was due to "pump stopped" we would not have the consequence "pump overheats". If the line fracture was upstream of the pump, then we may have the consequence "pump overheats" but not if the line fracture was downstream of the pump. The link between the causes and the consequences is the deviation itself. There are no direct links between items in the two lists. The fault, "valve directed shut", is simply one cause of no flow and "pump overheats" is simply one consequence of no flow. Thus, there is no need to propagate upstream or downstream the process deviations resulting from each individual fault. However, the propagation equations in the direction of flow do perform two useful functions. The first is that they indicate whether or not the unit is open to the process variable. Consider the case of a process line that contains a closed isolation valve. Here, no flow is the NORMAL state. The unit is not open to flow or any other process deviation. (In fact, the only faults relate to flow leaking through the valve). Thus, the model for a closed isolation valve will not contain any propagation equations. This will indicate that the valve will not transmit flow or any other process variable. The second function of the propagation equations is to indicate the effect on the outlet variables of deviations in the inlet variables. This can be useful when an in-line unit is such that an outlet variable is dependent on some other process variable. For example, consider a gas compressor where the flow and pressure energy of the gas is increased by the work done on it by the compressor. In such a unit, it may be that at high flows, this process becomes more inefficient resulting in more of the
energy going to raise the temperature in the gas. This relationship could be expressed as

\[ t_2 = f(t_1, q_1) \]

i.e. an increase in temperature out will result from an increase in either the temperature in or from an increase in flow in.

The propagation of faults from one process line to another is treated rather differently. The bridge between two process lines is the process vessel. Here, the propagation of faults is extremely important. The special points relating to process vessels will be dealt with in 5.1.3.

5.1.2 Event Statements

Event statements describe the initiation of a fault in an unhealthy unit or a termination of a fault in a unit which is thereby rendered unhealthy. Thus, there are two types of event statements, initial event statements and terminal event statements.

5.1.2.1 Initial Event Statements

An initial event statement is used to model the initiation, by a fault, of a set of variable deviations and takes the form

Initial fault : Process variable deviation

The completeness of the identification of causes depends on the completeness of the set of initial events. In configuring a model it is helpful to refer to the characteristics of a process unit listed below:
A process unit has a particular function. For example, it is the function of the pump, shown in Figure 5.2, to increase the pressure and thus cause flow in a process line.

One function fault in such a unit is that a motor fault results in no flow and low pressure. This fault can be represented by the initial event statement

\[
\text{motor fault} \Rightarrow q2 \text{ NONE}, p2 \text{ LO}
\]

This is equivalent to the statement that if a motor fault occurs then \( q2 \) NONE and \( p2 \) LO may result.

A process unit also has a hydraulic function. For example, it is the function of the pipe, shown in Figure 5.3, to transmit the fluid flowing through it.
Figure 5.3 Pipe

Thus, one hydraulic fault in such a unit is that blockage causes both the inlet and outlet flows to be zero. This is represented by the initial event statement

blockage>q1 NONE, q2 NONE

Another function of a process unit is containment. For example, it is the function of the pipe shown in Figure 5.3 to contain the fluid flowing in it. Thus, one containment fault is that a leak causes the inlet flow to be high and the outlet flow to be low. This is represented by the initial event statement

leak>q1 HI, q2 LO

There may be a transfer of heat between a process unit and the environment. For example, it is the function of a pipe to transmit the fluid flowing through it without change of temperature. Thus, one environment fault is that cold weather causes the outlet temperature to fall. This is represented by the initial event statement

cold weather>t2 LO
5.1.2.2 Terminal Event Statements

Information on how a fault terminates at a process unit is provided by the terminal event statements. A terminal event is used to represent the termination of a variable deviation in an undesired event, often a hazard, and takes the form

Variable deviation : Terminal event

Most terminal event statements relate to a serious excursion of a process variable such as pressure, temperature or level. A typical terminal event statement is

13 HI>overfilling

A terminal event statement may be made conditional on some property of the process materials in the unit or of the materials of construction of the unit. Thus, the terminal event statement

t2 LO>undertemperature

and hence brittle fracture may be made conditional on the material of construction.

Some events, notably leak and blockage, can be both initial and terminal events.

5.1.3 Process Lines and Process Vessels

This section deals with the differences in the way that process lines and process vessels are handled.

The first point of difference between in-line units and process vessels is the assignation of numbers to the ports, the inlets and outlets. Process vessels in this computer
method are by definition process units that have other than just one inlet and one outlet or process units that have a significant inventory. Thus, for example, a heat exchanger is considered a vessel and so is a binary header. Unlike the case of in-line units, there is no convention for the numbering of ports of vessels. The creator of a model will assign port numbers, starting from 1, as is deemed logical. It is important to note that where it is intended to refer to process variables inside a vessel, a port number be assigned to the inside. Of course, this is not an inlet or outlet, but an internal port is essential when propagation equations and event statements are to refer to say the level or pressure inside a vessel. For example, consider the vessel of Figure 5.4.

![Figure 5.4 Vessel](image)

It may be concluded that an increase in the level in the vessel will be due to either an increase in the flow in or a decrease in the flow out. The propagation equation for this relationship is thus

\[ l_3 = f(l_1, -l_2) \]

We may also wish to flag the terminal event "loss of level" as a result of low level in the vessel. This would be done by the terminal event statement
Internal ports are thus of critical importance in the modelling of a process vessel.

Propagation equations for in-line units have already been discussed in 5.1.1. For process vessels, propagation equations have a special significance. They are the means by which deviations in one process line are transmitted to another process line. Thus, a fault may be initiated in one process line, propagated to another process line through a vessel and terminated in a unit in the second process line.

There is only one difference between the formulation of event statements for process vessels and for in-line units. This is that for process vessels, event statements may refer to internal ports and to the process variables pressure and level.

5.1.4 The Generic Pipe Model

A process line as defined in a conventional HAZOP contains minor units such as handvalves and major units such as pumpsets. Since each unit will have in its cause list the common faults "leak (major)" and "blockage" a mechanistic modelling of the units and generation of the cause lists will lead to an excessive proliferation of these faults. The fault "leak (major)" is one that will lead to the consequence "flow low" or "flow none". Clearly, we can have "leak (major)" or "blockage" at any unit. The excessive proliferation of these faults is overcome by the use of the generic pipe model. The common faults are not entered in the models of the minor units and are entered instead in the generic pipe model. The common faults are, however, retained in the models of the major units.
At this stage, it is necessary to distinguish between "leak (major)" and "leak". The former has as consequence a deviation in flow whereas the latter is simply a cause of the deviation "escape". It is felt that unlike "leak (major)", leak should be entered as a cause of escape in every process unit. This is so that all the units are considered for all the different sorts of leak, e.g. a bonnet leak at a valve or a seal leak at a pump.

Consultation of the generic pipe unit is automatically performed by the computer method. There is no need for the user to specify the inclusion of the pipe model in a process line. The computer method will include a pipe model in every single one of the process lines in the generation of causes and consequences.

5.2 Modelling of Process Materials

A process material model is created for each process material. This model defines the characteristics of the material in terms of its

i) Properties
ii) Susceptibilities

A property is defined in this context as a characteristic which may lead unconditionally to a consequence without having to be activated by a process variable deviation. For example, a fluid may contain gunk which results in the unconditional consequence blockage. A susceptibility is defined as a characteristic which may lead to a consequence but only if it is activated by a process variable deviation. For example, a fluid may be susceptible, if the temperature is high, to polymerisation which thus results in the conditional consequence blockage.
When a generic fault such as blockage is encountered in the initial events in a unit model it is entered in the cause list. Unless, however, there are specific realisations, the fault has little credibility. A search is therefore made for credible specific realisations.

It is convenient to enter specific realisations of faults in the consequence list rather than the cause list. This is primarily because, as just described, some of the realisations are conditional on process variable deviations. For example, polymerisation can be both a cause of blockage and a consequence of high temperature. If polymerisation can occur unconditionally then of course it is correct that it is entered as a property, a cause of blockage. If, on the other hand, polymerisation can only occur if the temperature is high then polymerisation is not a property but a susceptibility, a consequence of high temperature.

Process materials also have one other type of characteristic. This is the set of attributes relevant to the escapes from containment of the process material. These attributes, the noxious results of escape, are treated as susceptibilities of the process deviation escape (The process deviation escape was introduced in Chapter 4 as a special feature of this computer method). For example, a certain hydrocarbon may be susceptible to fire or explosion on escape. Thus the deviation escape will have as consequence, a flammability hazard.

For a process material, two searches are made in each process line. The first search is for any property of the material which results unconditionally in a specific realisation. These realisations are entered in the cause list. The second is for any combination of susceptibility and process variable deviation which results, due to fulfilment of
the deviation condition, in a specific realisation. Any such realisations are entered in the consequence list.

Information on the process material carried within each process line is entered as part of the configuration data.

5.3 Modelling of Materials of Construction

A materials of construction model is created for each configuration. This model defines the characteristics of the material in terms of its susceptibilities. A susceptibility is defined, as before, as a characteristic which may lead to a consequence but only if it is activated by a process variable deviation. For example, a material of construction may be susceptible to leak by brittle fracture if the temperature is low. Thus, a search is made in each process line as part of the search on the generic pipe unit. This search is for any variable deviation which activates a susceptibility and results in a specific realisation. Any such realisations found are entered in the consequence list.

Information on the materials of construction is entered as part of the configuration data.

5.4 Modelling of the Process Plant Configuration

Strictly speaking, the data file of the plant configuration is not a model. However, it does deserve a mention here because the process of system decomposition is in a sense modelling.

System decomposition may be done at various levels of detail. These different levels may be illustrated by reference to a control loop. In a coarse decomposition the control loop is treated as a single item. At a finer level of detail, the loop may be decomposed into constituent parts such as sensor,
controller and control valve. A yet finer decomposition may involve considering the elements of the sensor such as the orifice plate and isolating valves in a flowmeter.

In this computer method, the level of decomposition is at the coarser end of the spectrum. For example, a control loop, including its bypass, is treated as a single unit.

The modelling of the plant configuration involves the following:

i) Decomposition of the system. This will include the selection of the appropriate unit models to represent the units of the configuration.

ii) Selection of process materials. This is simply the input of information regarding the process materials that are carried in the different process lines.

iii) Selection of materials of construction. Again, this is simply the input of the model of the materials of construction.

iv) Input of the impurities referenced. This is the keying in of the different impurities that will be referred to in the configuration.
6 Evaluation of Computer Method

This chapter seeks to evaluate the performance of this computer method by running test cases through it. The process is to take HAZOP examples from the literature, run them through the computer method and then compare the results so produced with those published.

The literature cases chosen to be used in this comparison had to satisfy a number of criteria. These were

1) The processes involved had to be continuous. The method is restricted to continuous processes (see Chapter 7). Thus, all examples of batch or semi-batch operation were excluded.

2) The examples had to be manageable. Examples involving very large configurations would have required greater resources in planning, time and computing. This would have detracted from the emphasis on development that was the main aim of this work.

3) It would be advantageous to be able to perform comparisons on examples that were widely known.

Two of the examples cited in Chapter 1 fitted the above criteria. These were

1) HAZOP study on the feed section of a proposed olefin dimerisation plant (see 1.3.1). The main process unit in this plant is a vessel to separate the water from the olefin feed to a reactor. Hereafter this example is referred to as the water separator example.

2) HAZOP study of a liquid propane transfer and vaporisation unit (see 1.3.4). This plant has the function of...
transferring large quantities of propane to a buffer tank. Hereafter this example is referred to as the propane pipeline example.

It is important to note here that the published records of these examples were intended purely for tutorial purposes. The records are not a complete account of the HAZOP study. They do not indicate what the HAZOP team actually discussed, what items they discarded as irrelevant or trivial, and so on.

6.1 Water Separator Example

The plant system is shown in Figure 6.1. The process description is

"An alkane/alkene fraction containing small amounts of suspended water is pumped continuously from bulk intermediate storage via a half-mile pipeline section into a buffer/settling tank where residual water is settled out prior to passing via a feed/product heat exchanger to the reactor section. The water which has an adverse effect on the dimerisation reaction, is run off manually from the settling tank at intervals. Residence time in the reaction section must be held within closely defined limits to ensure adequate conversion of the alkene and to avoid excessive formation of polymer"

6.1.1 Plant Configuration

The plant system is given in the diagram of Figure 6.1. This is converted to the configuration diagram shown in Figure 6.2. The individual units are joined by links and the vessels (including sources and sinks) by process lines. The links and line numbers are shown plain and circled respectively. The links of the configuration are converted into tabular form for input to the "configurator" program. This is shown in Table
Figure 6.1
Plant System for Water Separator Example
Figure 6.2
Configuration Diagram for Water Separator Example
6.1.

The unit models are shown in Tables 6.2 and 6.3. Table 6.2 lists the complete set of library models used, many of which appear several times. Table 6.3 identifies the unit models used for particular units in the configuration.

All of the above information is entered into the "configurator" program. The program divides up the configuration into process lines and vessels and assigns the line numbers as indicated in Figure 6.2. After the input of a materials of construction model and the list of impurities, the program produces the configuration file. This configuration file "Water_sep.conf" is shown in its entirety in Appendix F. The unit and process material models referenced in this example are given in Appendix G.

6.1.2 Hazard Identification

The complete hazard identification table for the system is given in Appendix H. Table 6.4 gives a comparison of the results in Appendix H with those obtained by the industrial HAZOP team.

There is reasonably good agreement between the two sets of results. There are, however, several points to be noted. These points are listed in the order in which they are demonstrated by the comparison.

1) The causes and consequences generated by the computer method are restricted to the line under consideration and the vessels connected to it. The program does not look forward to consequences beyond the vessels. Thus, examination of "No flow" in Line 1 does not generate the consequence that polymerisation may occur in the heat
Table 6.1 Configuration Links for Water Separator Example

<table>
<thead>
<tr>
<th>Link No.</th>
<th>UPSTREAM</th>
<th>DOWNSTREAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>source1</td>
<td>pump2</td>
</tr>
<tr>
<td>2</td>
<td>pump2</td>
<td>nrv3</td>
</tr>
<tr>
<td>3</td>
<td>nrv3</td>
<td>meter4</td>
</tr>
<tr>
<td>4</td>
<td>meter4</td>
<td>pipeline5</td>
</tr>
<tr>
<td>5</td>
<td>pipeline5</td>
<td>cntrl_vl6</td>
</tr>
<tr>
<td>6</td>
<td>cntrl_vl6</td>
<td>valve7</td>
</tr>
<tr>
<td>7</td>
<td>valve7</td>
<td>settler8</td>
</tr>
<tr>
<td>8</td>
<td>settler8</td>
<td>pump9</td>
</tr>
<tr>
<td>9</td>
<td>pump9</td>
<td>divider10</td>
</tr>
<tr>
<td>10</td>
<td>divider10</td>
<td>orifice11</td>
</tr>
<tr>
<td>11</td>
<td>orifice11</td>
<td>valve12</td>
</tr>
<tr>
<td>12</td>
<td>valve12</td>
<td>settler8</td>
</tr>
<tr>
<td>13</td>
<td>divider10</td>
<td>cntrl_vl13</td>
</tr>
<tr>
<td>14</td>
<td>cntrl_vl13</td>
<td>exchang14</td>
</tr>
<tr>
<td>15</td>
<td>exchang14</td>
<td>sink15</td>
</tr>
<tr>
<td>16</td>
<td>source14</td>
<td>exchang14</td>
</tr>
<tr>
<td>17</td>
<td>exchang14</td>
<td>sink15</td>
</tr>
<tr>
<td>18</td>
<td>settler8</td>
<td>valve18</td>
</tr>
<tr>
<td>19</td>
<td>valve18</td>
<td>sink19</td>
</tr>
</tbody>
</table>
Table 6.2 Library Models used for Water Separator Example

hyd_source
pumpset_a
nrv
fq_c_bypas
pipeline
cv_c_bypas
valve
settler
divider
orifice
ht_exchg_a
sink
source
Table 6.3 Units and Models of the Water Separator Example

<table>
<thead>
<tr>
<th>UNIT</th>
<th>MODEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>source1</td>
<td>hyd_source</td>
</tr>
<tr>
<td>pump2</td>
<td>pumpset_a</td>
</tr>
<tr>
<td>nrv3</td>
<td>nrv</td>
</tr>
<tr>
<td>meter4</td>
<td>fq_c_bypas</td>
</tr>
<tr>
<td>pipeline5</td>
<td>pipeline</td>
</tr>
<tr>
<td>cntrl_v16</td>
<td>cv_c_bypas</td>
</tr>
<tr>
<td>valve7</td>
<td>valve</td>
</tr>
<tr>
<td>settler8</td>
<td>settler</td>
</tr>
<tr>
<td>pump9</td>
<td>pumpset_a</td>
</tr>
<tr>
<td>divider10</td>
<td>divider</td>
</tr>
<tr>
<td>orifice11</td>
<td>orifice</td>
</tr>
<tr>
<td>valve12</td>
<td>valve</td>
</tr>
<tr>
<td>cntrl_v113</td>
<td>cv_c_bypas</td>
</tr>
<tr>
<td>exchang14</td>
<td>ht_exchg_a</td>
</tr>
<tr>
<td>sink15</td>
<td>sink</td>
</tr>
<tr>
<td>source16</td>
<td>source</td>
</tr>
<tr>
<td>sink17</td>
<td>sink</td>
</tr>
<tr>
<td>valve18</td>
<td>valve</td>
</tr>
<tr>
<td>sink19</td>
<td>sink</td>
</tr>
</tbody>
</table>
Table 6.4

Comparison of Results for

Water Separator Example
<table>
<thead>
<tr>
<th>PROCESS DEVIATION</th>
<th>ORIGINAL HAZOP CAUSES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</th>
<th>ORIGINAL HAZOP CONSEQUENCES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO FLOW</td>
<td>No hydrocarbon at intermediate storage source - q 1 none source empty</td>
<td>Loss of feed to reaction section and reduced output. Polymer formed in heat</td>
<td>pump - pump fluid overheats</td>
<td>setler - q 1 none</td>
</tr>
<tr>
<td>PROCESS</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------</td>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>MORE FLOW</td>
<td>LCV fails open OR LCV bypass open in error</td>
<td>source - q 1 high</td>
<td>Settling tank overfills meter - inaccurate</td>
<td>material loss adjacent to public highway - inaccurate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>meter - bypass fails bypass directed open</td>
<td>Incomplete separation of water phase in tank, leading to problems in reaction section</td>
<td>material loss adjacent to public highway - inaccurate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>control - loop fails open sensor fails open controller fails open setpoint moved open bypass fails bypass directed open</td>
<td>meter - inaccurate</td>
<td>material loss adjacent to public highway - inaccurate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>valve - valve partly closed</td>
<td>settling-q 1 high</td>
<td>1 5 low</td>
</tr>
<tr>
<td>LESS FLOW</td>
<td>Leaking flange or valved stub not blanked and leaking</td>
<td>source - q 1 low blockage leak (major)</td>
<td>Material loss adjacent to public highway - inaccurate</td>
<td>settling-q 1 low outflow(s) low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pump - loss of NPSH rotation fault impeller fault cavitation low b. pt. material gassy material delivery part shut blockage leak (major)</td>
<td></td>
<td>1 5 low</td>
</tr>
<tr>
<td>PROCESS</td>
<td>ORIGINAL HAZARD</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZARD</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------</td>
<td>---------------------------------------------</td>
<td>-----------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>MORE PRESSURE</td>
<td>Isolation valve closed in error or LCV closes, with J1 pump running</td>
<td>Source = p 1 high</td>
<td>Transfer line subjected pipe to full pump delivery</td>
<td>Leak (major)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe = blockage heat to blocked in fluid</td>
<td>or surge pressure pipe rupture</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>solar heat fire</td>
<td>Line fracture or flange leak</td>
<td>Settler = p 1 high overpressure leak (major)</td>
</tr>
<tr>
<td></td>
<td>Thermal expansion in an isolated valued section due to fire or strong sunlight.</td>
<td>pump = delivery shut in blockage heat to blocked in fluid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>solar heat fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipeline-blockage hammer/surge heat to blocked in fluid</td>
<td>solar heat fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>control loop fails closed</td>
<td>sensor fails closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>valve</td>
<td>controller fails closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control valve fails closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>set point moved closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>isolation valve closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>valve</td>
<td>valve directed closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LESS PRESSURE</td>
<td>source = p 1 low source empty</td>
<td>Settler = p 1 low</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipe = leak (major)</td>
<td>olefin - liquid flashes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MORE TEMPERATURE STORAGE TEMPERATURE</td>
<td>High intermediate storage temperature</td>
<td>source = t 1 high</td>
<td>Higher pressure in transfer line and settling tank</td>
<td>olefin - pressure high blockage polymerisation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe = external heat source hot weather fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pump = pump fluid overheats</td>
<td>settler = t 1 high t 5 high</td>
<td></td>
</tr>
<tr>
<td>PROCESS DEVIATION</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------</td>
<td>--------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>LESS TEMPERATURE</td>
<td>Winter conditions</td>
<td>source -t 1 low</td>
<td>Water sump and drain line freeze up</td>
<td>pipe -leak (major) brittle fracture</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe -external cold source cold weather lagging loss</td>
<td></td>
<td>settler-t 1 low t 5 low drain/sump freeze up</td>
</tr>
<tr>
<td>ESCAPE</td>
<td>N/A</td>
<td>pipe -leak</td>
<td>N/A</td>
<td>olefin -flammable release</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pump -leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>nrv -leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>meter -leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipeline-leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control -leak valve</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>valve - leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>settler - leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HIGH WATER CONCA IN STREAM</td>
<td>High water level in intermediate storage tank</td>
<td>source -impurity a in source</td>
<td>Water sump fills up more quickly. Increased chance of water phase passing to reaction section.</td>
<td>setter-poor separation of water water sump overfills</td>
</tr>
<tr>
<td>HIGH CONCA OF LOWER ALKANES OR ALKENES IN STREAM</td>
<td>Disturbance on distillation columns upstream of intermediate storage</td>
<td>source -impurity b in source</td>
<td>High system pressure pipe -pressure high</td>
<td>setter-pressure high</td>
</tr>
<tr>
<td>ORGANIC ACIDS PRESENT</td>
<td>Disturbance on distillation columns upstream of intermediate storage</td>
<td>source -impurity c in source</td>
<td>Increased rate of corrosion of tank base, sump and drain line.</td>
<td>pipe -corrosion setter-corrosion</td>
</tr>
<tr>
<td>MAINTENANCE</td>
<td>Equipment failure, flange leak, etc.</td>
<td></td>
<td>Line cannot be completely drained or purged</td>
<td></td>
</tr>
<tr>
<td>PROCESS DEVIATION</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------</td>
<td>-------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td><strong>NO FLOW</strong></td>
<td>No hydrocarbon in settling tank</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>J2 pump fails (motor fault, loss of drive, pipe impeller corroded away or detached, etc.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Main line blockage, delivery line isolation valve closed in error or FCV fails shut.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Heat exchanger blockage on reactor feed side.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Line fracture</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>LINE 2</strong></td>
<td>settler - 1 low.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>p 5 low.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>q 2 none.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>blockage leak (major).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>loss of NPSH rotation fault.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>impeller fault cavitition.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>low b.p.t. material gassy material.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>delivery shut in.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Line section subject to standby fail on demand full pump delivery.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>no change to standby pressure.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>maloperation of valves blockage leak (major).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>LINE 4</strong></td>
<td>divider - q 1 none.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>q 2 none.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>blockage leak (major).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipe - blockage leak (major).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>control - loop fails closed sensor fails closed controller fails closed control valve fails closed set-point moved closed isolation valve closed.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**LINE 2**
- settler - 1.5 high. 
- pump - pump fluid overheats.
- divider - q 1 none. 
- q 2 none. 
- q 3 none.

**LINE 4**
- divider - q 1 none.
- q 2 none.
- exchange - q 1 none.
- q 2 none.
<table>
<thead>
<tr>
<th>PROCESS DEVIATION</th>
<th>ORIGINAL HAZOP CAUSES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</th>
<th>ORIGINAL HAZOP CONSEQUENCES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>REVERSE FLOW</td>
<td>On-line J2 pump fails</td>
<td>Sudden backflow of hot material from high pressure section into settling tank via J2 pump kickback line or via J2 pump if its NRV has also failed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MORE FLOW</td>
<td>Reactor feed FCV fails open or towards open, or FCV bypass opened in error</td>
<td>Incomplete conversion of olefin in reaction section (due to shorter residence time) leading to lower efficiencies and to problems later in the process divider - q 1 high</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Line 2 setler - 1 high p 5 high q 2 high</td>
<td></td>
<td>Line 2 settle - 1 high p 5 high q 2 high</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Line 4 divider - q 1 high q 2 high blockage-other leg</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>control - loop fails open sensor fails open controller fails open control valve fails open set point moved open bypass fails bypass directed open</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LESS FLOW</td>
<td>Flange leak, leak at valved stub or at J2 pump seals</td>
<td>Loss of material into plant area and possible fire Reactor product contaminated with reactor feed, leading</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Line 2 settle - 1 low p 5 low q 2 low</td>
<td></td>
<td>Line 2 settle - 1 high p 5 high q 2 low q 3 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Holed heat exchanger pipe - blockage leak (major)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS DEVIATION</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------</td>
<td>-------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>Less Flow (contd)</td>
<td></td>
<td>pump - loss of NPSH</td>
<td>to reduced chemical</td>
<td>Line 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>rotation fault</td>
<td>efficiencies and to</td>
<td>exchanger - q 1 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>impeller fault</td>
<td>problems later in the</td>
<td>q 2 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cavitation</td>
<td>process</td>
<td>t 2 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>low b.p.t. material</td>
<td></td>
<td>t 4 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>gassy material</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>delivery part shut</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>leak (major)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Line 4 divider - q 1 low</td>
<td>q 1 low</td>
<td></td>
<td>Line 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>q 2 low</td>
<td></td>
<td>exchanger - q 1 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>blockage</td>
<td></td>
<td>q 2 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>leak (major)</td>
<td></td>
<td>t 2 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>leak (major)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control - loop fails part</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>valve - closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>sensor fails part</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>controller fails part</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control valve fails</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>part closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>set point moved part</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>isolation valve part</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MORE PRESSURE Line or heat exchanger</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>blockage, closure of</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>isolation valve or feed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FCV, high reactor product</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>temperature or local plan fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MORE PRESSURE Line 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>settler - p 5 high</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>p 2 high</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>heat to blocked in</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>fluid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>solar heat fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Line fracture, major</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>spillage lost output,</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>possibility of major</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>plant fire</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>pipe rupture</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>flange leak</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - leak (major)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>divider - p 1 high</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>p 2 high</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>p 3 high</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pump - delivery shut in</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>heat to blocked in</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>fluid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>solar heat fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------</td>
<td>--------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>PRESSURE OR TEMPERATURE (contd)</td>
<td></td>
<td>Line 4 divider - p 1 high</td>
<td></td>
<td>Line 4 pipe - leak (major)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>p 2 high</td>
<td></td>
<td>pipe rupture</td>
</tr>
<tr>
<td></td>
<td></td>
<td>blockage</td>
<td></td>
<td>flange leak</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - blockage</td>
<td></td>
<td>exchanger - p 1 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>head to blocked in fluid</td>
<td></td>
<td>p 2 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>solar heat</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control - loop fails closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>sensor fails closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>controller fails closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control valve fails closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>set point moved</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>isolation valve closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>exchanger - tube fouling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MORE TEMPERATURE</td>
<td></td>
<td>Line 2 settler - t 3 high</td>
<td></td>
<td>Line 2 olefin - pressure high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>t 2 high</td>
<td></td>
<td>blockage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - external heat</td>
<td></td>
<td>polymerisation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>source</td>
<td></td>
<td>divider - t 1 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hot weather</td>
<td></td>
<td>t 2 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fire</td>
<td></td>
<td>t 3 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pump - pump fluid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>overheats</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Line 4 divider - t 1 high</td>
<td></td>
<td>Line 4 olefin - pressure high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>t 2 high</td>
<td></td>
<td>blockage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - external heat</td>
<td></td>
<td>polymerisation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>source</td>
<td></td>
<td>exchanger - t 1 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>hot weather</td>
<td></td>
<td>t 2 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fire</td>
<td></td>
<td>t 4 high</td>
</tr>
<tr>
<td>PROCESS DEVIATION</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------</td>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>LESS PRESSURE</td>
<td>Deterioration in J2 pumping efficiency</td>
<td>Line 2&lt;br&gt;settle - p 5 low&lt;br&gt;p 2 low</td>
<td>Reduced feed rate to reaction and lower output</td>
<td>Line 2&lt;br&gt;olefin - liquid flashes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>line 4&lt;br&gt;divider - p 1 low&lt;br&gt;p 2 low</td>
<td></td>
<td>line 4&lt;br&gt;olefin - liquid flashes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - leak (major)</td>
<td></td>
<td>exchanger-p 1 low&lt;br&gt;p 2 low</td>
</tr>
<tr>
<td>LESS TEMPERATURE</td>
<td>Fouling of heat exchanger leading to reduced heat transfer</td>
<td>Line 2&lt;br&gt;settle - t 5 low&lt;br&gt;t 2 low</td>
<td>Increased heat load on reactor pre-heater&lt;br&gt;and increased cooling requirement on reactor product after cooler</td>
<td>Line 2&lt;br&gt;olefin - leak (major)&lt;br&gt;brittle fracture</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - external cold source&lt;br&gt;cold weather lagging loss</td>
<td></td>
<td>line 4&lt;br&gt;pipe - leak (major)&lt;br&gt;brittle fracture</td>
</tr>
<tr>
<td></td>
<td></td>
<td>line 4&lt;br&gt;divider - t 1 low&lt;br&gt;t 2 low</td>
<td></td>
<td>exchanger-t 1 low&lt;br&gt;t 2 low&lt;br&gt;t 4 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - external cold source&lt;br&gt;cold weather lagging loss</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESCAPE</td>
<td>N/A</td>
<td>Line 2&lt;br&gt;settler - leak</td>
<td></td>
<td>Line 2&lt;br&gt;olefin - flammable release</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pump - leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>divider - leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Line 4&lt;br&gt;divider - leak</td>
<td></td>
<td>Line 4&lt;br&gt;olefin - flammable release</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control - leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>exchanger - leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS</td>
<td>ORIGINAL HAZOP DEVIATION</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------</td>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>HIGH WATER</td>
<td>Incomplete separation of suspended water in buffer/ settling tank</td>
<td>Reduced conversion and efficiency on reaction section</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONCn IN</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>REACTOR FEED</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HIGH CONCn OF LOWER ALKANES OR ALKENES IN REACTOR FEED</td>
<td>Upset on distillation columns upstream of intermediate storage</td>
<td>Higher system pressure settler-pressure high pipe pressure high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>POLYMER FORMATION IN HEAT EXCHANGER</td>
<td>Low feed rate to reaction section</td>
<td>Increased fouling of heat exchanger and reduced reaction efficiencies and output</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAINTENANCE OR INSPECTION</td>
<td>Fault or leak on buffer/settling tank or on associated pipework/Statutory tank inspection</td>
<td>Tank cannot be inspected or maintained unless all connecting lines are positively isolated</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
exchanger. This consequence is identified but not until examination of "No flow" in Line 4. Another example of this is that the item regarding loss of feed to reaction section appears in the output of the computer method as a loss of outflow from the settler.

2) In the case of the results of the industrial HAZOP team there is interaction between the causes and consequences. The process that the HAZOP team follow is to identify a cause of the variable deviation and then to identify all the consequences that result. With the computer method, the link between causes and consequences is the deviation itself. This does not mean that the computer method is any less comprehensive. All the causes and consequences are picked up but are listed under the most appropriate deviation. Consider the example of "No flow" in Line 1. The industrial HAZOP team's results, with the interaction between the causes and consequences, can be summarised as shown in the diagram of Figure 6.3. The arrows indicate which consequences stem from which causes.

3) Consideration of "No flow" in Line 1, the diagram of Figure 6.3 and the output of the computer method leads one to conclude initially that although the list of causes is comprehensive, the list of consequences lacks the item referring to the material discharge onto the highway. However, this item is actually picked up under the deviation escape. Escape lists as one cause leak at the pipe and as a consequence the point that the material is olefin and that it has the noxious property of flammability.

4) Interaction of the process materials models is illustrated by the list of causes of "No flow" in Line 1. One of the causes listed for the pump is "cavitation". This is one of the keywords referred to in 4.3.4. The computer method, on
Figure 6.3
Interaction of Causes and Consequences in a HAZOP Study

135
encountering this keyword searches the model of the process material for causes of "cavitation". This search results in "low b.p.t. material" and "gassy material".

5) The lists of causes generated by the computer method are more exhaustive than the lists generated by the industrial HAZOP team. This may be due to a number of reasons. It may be that the industrial HAZOP team will have discounted certain items on the basis of improbability or redundancy. It may also be that certain items will have been considered inappropriate. Consider "No flow" in Line 1, the line from the intermediate storage to the buffer tank. An example of the discounting by probability may be the computer method cause at the pump "standby fail on demand". The pumps may be reliable enough and tested often enough to make this very improbable. An example of discounting by redundancy are the causes "control valve - isolation valve closed" and "valves - valve directed shut". The first item refers to the isolation valves either side of the control valve and the second item refers to the hand valves in the line. These two items are condensed to just "isolation valve closed" by the industrial HAZOP team. Discounting on the basis of inappropriateness is shown by the computer method cause "pump - loss of NPSH". This may have been rejected by the HAZOP team on the basis that NPSH is irrelevant to this problem.

Regardless of the criticism afforded by the paragraph above, it is felt that the computer method is correct in flagging all possible causes. It is obviously better for example to list all the different modes of pump failure rather than to just flag "pump fails". One is looking for credible specific realisations of "pump fails", especially those connected with process fluid properties etc. This point applies also to the case of the isolation valves. It
is advisable to inform the user that there are isolation valves either side of the control valve which may cause no flow. Specific faults with their locations are better than a statement such as "isolation valve closed" which may lead to some valves being overlooked.

6) Concerning the deviation "More temperature" in Line 1, the industrial HAZOP team have indicated the consequence "higher pressure in transfer line and settling tank". This consequence comes from the properties of the process material; a higher temperature would lead to a higher vapour pressure. The process model for the olefin indicates that higher temperature will lead to a higher pressure, and thus this consequence of high temperature is listed for the olefin.

7) The industrial HAZOP team have listed, as a consequence of "Less temperature", "water sump and drain line freeze-up". The computer method, in its search at the settler, as part of the search on Line 1 (the settler is the drain vessel for Line 1), picks up this consequence.

8) Considering the list of consequences, it is readily apparent that the computer method is more rigorous than the industrial HAZOP team when listing the consequences of the deviation on the process material and the material of construction. For instance, according to the computer method "More temperature" in Line 1 has as a consequence the warning that the olefin is liable to polymerisation. The industrial HAZOP team do not flag this as a consequence. However, the computer method is not wrong because the only cause of high temperature the industrial HAZOP team have listed is that the source temperature is high. Maybe, the HAZOP team also recognised that the polymerisation process needs time and therefore discarded polymerisation. The process material model for olefin
indicates the fact that the olefin may polymerise if the temperature is deviated high.

9) The deviation escape is one special feature of the computer method. Escape allows the highlighting of the noxious characteristics of the process material, the characteristics that would be realised if the process material were to escape from containment. In the case of olefin, flammability is the noxious characteristic.

Note that the causes of escape are leaks at all the different units in the process line. It would be possible to specify the different sorts of leak at the different sorts of unit e.g. bonnet leaks at valves, seal leaks at pumps. This has not been done here because distinguishing between the different leaks is not critical in this example.

10) The points relating to the high concentrations of water and lower alkanes/alkenes and the presence of organic acids have been treated by the computer method as deviations high in the impurities water, lower alkanes/alkenes and organic acids. These three materials are referred to as impurities a, b and c respectively. The causes and consequences of these high impurity levels compare well with those of the industrial HAZOP team.

11) The computer method does not have a facility to deal with the item regarding maintenance. Thus, the fact that Line 1 cannot be completely drained or purged is not picked up (see Chapter 7).

12) The computer method is not able to deal with the deviation reverse flow (see Chapter 7).
13) More flow in the line from the settling tank to the heat exchanger (Lines 2 and 4 in the computer method) demonstrates clearly the use of propagation equations. The heat exchanger has as consequences of more flow the deviations more flow out, lower temperature out and lower temperature out of the other stream. These points were not indicated by the industrial HAZOP team.

14) The line from the buffer/settling tank to the feed/product heat exchanger is considered by the computer method as Lines 2 and 4. This illustrates another point. The computer method will invariably divide the configuration into more process lines than the industrial HAZOP team. The basis of sectioning into process lines is essentially the same for both but is more rigorously followed by the computer method. The industrial HAZOP team do not consider a divider as the end of a process line whereas the computer method does. This difference is unavoidable if the computer method is to proceed with the hazard identification in a logical manner.

Another example of this point is given by considering the sump drain line out of the settling tank. The industrial HAZOP team consider this as part of the settling tank but the computer method sees it as Line 8, leading from settler8 to sink19.

15) There are a number of points that seem to be omitted by the computer method but are actually listed elsewhere. An example of this occurs in the list of causes of "Less temperature" for the line from the buffer/settling tank to the heat exchanger. One cause listed by the industrial HAZOP team is fouling of the heat exchanger. The computer method does not have this listed for that section of line. However, the cause "tube fouling" is listed for "temperature low" for Line 5, the line out of the heat
exchanger. Clearly, Line 5 is the more logical point to flag the cause because that is where the deviation occurs.

6.2 Propane Pipeline Example

The plant system is shown in Figure 6.4. Refrigerated liquid propane is pumped from an atmospheric pressure storage tank via the tube side of a heater, where the temperature of the propane is raised to about +15 deg.C, through a 10-mile pipeline to a buffer tank at the consumer plant. The shell side of the propane heater contains glycol which is circulated by a group of pumps. The glycol passing to the propane heater is heated by LP steam via a heater situated just upstream of the propane heater. The temperature of the propane export is controlled by means of a 3-way valve in the glycol circulation system which permits bypassing of the propane heater as required.

6.2.1 Plant Configuration

The plant system given in the diagram in Figure 6.4 is converted to the configuration diagram shown in Figure 6.5. The links and line numbers are shown, as before, plain and circled respectively. The tabular form of the links of the configuration diagram is shown in Table 6.5.

The unit models used in the configuration are shown in Tables 6.6 and 6.7. Table 6.6 lists the complete set of library models used, many of which appear several times. Table 6.7 identifies the unit-models used for particular units in the configuration.

All of the above information is entered into the "configurator" program along with the information on the materials of construction and impurities. The "configurator" program then produces the configuration file which, for this
<table>
<thead>
<tr>
<th>Link No.</th>
<th>UPSTREAM</th>
<th>DOWNSTREAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>tank1</td>
<td>valve2</td>
</tr>
<tr>
<td>2</td>
<td>valve2</td>
<td>pump3</td>
</tr>
<tr>
<td>3</td>
<td>pump3</td>
<td>nrv4</td>
</tr>
<tr>
<td>4</td>
<td>nrv4</td>
<td>divider5</td>
</tr>
<tr>
<td>5</td>
<td>divider5</td>
<td>cntrl_v16</td>
</tr>
<tr>
<td>6</td>
<td>cntrl_v16</td>
<td>tank1</td>
</tr>
<tr>
<td>7</td>
<td>divider5</td>
<td>valve7</td>
</tr>
<tr>
<td>8</td>
<td>valve7</td>
<td>exchang8</td>
</tr>
<tr>
<td>9</td>
<td>exchang8</td>
<td>nrv9</td>
</tr>
<tr>
<td>10</td>
<td>nrv9</td>
<td>trip10</td>
</tr>
<tr>
<td>11</td>
<td>trip10</td>
<td>blk_bld11</td>
</tr>
<tr>
<td>12</td>
<td>blk_bld11</td>
<td>pipeline12</td>
</tr>
<tr>
<td>13</td>
<td>pipeline12</td>
<td>blk_bld13</td>
</tr>
<tr>
<td>14</td>
<td>blk_bld13</td>
<td>cntrl_v114</td>
</tr>
<tr>
<td>15</td>
<td>cntrl_v114</td>
<td>trip15</td>
</tr>
<tr>
<td>16</td>
<td>trip15</td>
<td>nrv16</td>
</tr>
<tr>
<td>17</td>
<td>nrv16</td>
<td>valve17</td>
</tr>
<tr>
<td>18</td>
<td>valve17</td>
<td>tank18</td>
</tr>
<tr>
<td>19</td>
<td>tank18</td>
<td>valve19</td>
</tr>
<tr>
<td>20</td>
<td>valve19</td>
<td>sink20</td>
</tr>
<tr>
<td>21</td>
<td>source21</td>
<td>cntrl_v122</td>
</tr>
<tr>
<td>22</td>
<td>cntrl_v122</td>
<td>exchang23</td>
</tr>
<tr>
<td>23</td>
<td>exchang23</td>
<td>drum24</td>
</tr>
<tr>
<td>24</td>
<td>drum24</td>
<td>cntrl_v125</td>
</tr>
<tr>
<td>25</td>
<td>cntrl_v125</td>
<td>sink26</td>
</tr>
<tr>
<td>26</td>
<td>tank27</td>
<td>pump28</td>
</tr>
<tr>
<td>27</td>
<td>pump28</td>
<td>nrv29</td>
</tr>
<tr>
<td>28</td>
<td>nrv29</td>
<td>exchang23</td>
</tr>
<tr>
<td>29</td>
<td>exchang23</td>
<td>valve30</td>
</tr>
<tr>
<td>30</td>
<td>valve30</td>
<td>tw_valve31</td>
</tr>
</tbody>
</table>
Table 6.5 (contd.)

<table>
<thead>
<tr>
<th>Link No.</th>
<th>UNIT</th>
<th>PORT</th>
<th>UPSTREAM</th>
<th>UNIT</th>
<th>PORT</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>tw_valve31</td>
<td>2</td>
<td>valve32</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>valve32</td>
<td>2</td>
<td>exchang8</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>exchang8</td>
<td>4</td>
<td>mixer34</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>tw_valve31</td>
<td>3</td>
<td>valve33</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>valve33</td>
<td>2</td>
<td>mixer34</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>mixer34</td>
<td>3</td>
<td>tank27</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Table 6.6 Library Models used for Propane Pipeline Example

storg_tnk
valve
pumpset_b
nrv
divider
cv_c_bypas
ht_exchg_b
ht_exchg_c
trip_valve
d_blck_bld
pipeline
buffer_tnk
sink
source
conds_drum
surge_tnk
thr_way_vl
mixer
Table 6.7 Units and Models of the Propane Pipeline Example

<table>
<thead>
<tr>
<th>UNIT</th>
<th>MODEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>tank1</td>
<td>storg_tnk</td>
</tr>
<tr>
<td>valve2</td>
<td>valve</td>
</tr>
<tr>
<td>pump3</td>
<td>pumpset_b</td>
</tr>
<tr>
<td>nrv4</td>
<td>nrv</td>
</tr>
<tr>
<td>divider5</td>
<td>divider</td>
</tr>
<tr>
<td>cntrl_v16</td>
<td>cv_c_bypas</td>
</tr>
<tr>
<td>valve7</td>
<td>valve</td>
</tr>
<tr>
<td>exchang8</td>
<td>ht_exchg_b</td>
</tr>
<tr>
<td>nrv9</td>
<td>nrv</td>
</tr>
<tr>
<td>trip10</td>
<td>trip_valve</td>
</tr>
<tr>
<td>blk_bld11</td>
<td>d_blk_bld</td>
</tr>
<tr>
<td>pipeline12</td>
<td>pipeline</td>
</tr>
<tr>
<td>blk_bld13</td>
<td>d_blk_bld</td>
</tr>
<tr>
<td>cntrl_v114</td>
<td>cv_c_bypas</td>
</tr>
<tr>
<td>trip15</td>
<td>trip_valve</td>
</tr>
<tr>
<td>nrv16</td>
<td>nrv</td>
</tr>
<tr>
<td>valve17</td>
<td>valve</td>
</tr>
<tr>
<td>tank18</td>
<td>buffer_tnk</td>
</tr>
<tr>
<td>valve19</td>
<td>valve</td>
</tr>
<tr>
<td>sink20</td>
<td>sink</td>
</tr>
<tr>
<td>source21</td>
<td>source</td>
</tr>
<tr>
<td>cntrl_v122</td>
<td>cv_c_bypas</td>
</tr>
<tr>
<td>exchang23</td>
<td>ht_exchg_c</td>
</tr>
<tr>
<td>drum24</td>
<td>conds_drum</td>
</tr>
<tr>
<td>cntrl_v125</td>
<td>cv_c_bypas</td>
</tr>
<tr>
<td>sink26</td>
<td>sink</td>
</tr>
<tr>
<td>tank27</td>
<td>surge_tnk</td>
</tr>
<tr>
<td>pump28</td>
<td>pumpset_b</td>
</tr>
<tr>
<td>nrv29</td>
<td>nrv</td>
</tr>
<tr>
<td>valve30</td>
<td>valve</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>tw_valve31</td>
<td>thr_way_v1</td>
</tr>
<tr>
<td>valve32</td>
<td>valve</td>
</tr>
<tr>
<td>valve33</td>
<td>valve</td>
</tr>
<tr>
<td>mixer34</td>
<td>mixer</td>
</tr>
</tbody>
</table>
example, is given in Appendix I. The unit and process material models referenced in this example are given in Appendix J.

6.2.2 Hazard Identification

The complete hazard identification table for the system is given in Appendix K. Table 6.8 gives a comparison of the results in Appendix K with those obtained by the industrial HAZOP team.

As with the comparison of 6.1.2 there is good agreement between the two sets of results. The points made in 6.1.2 will, of course, apply to the results here although the specific examples will be different. The following further points arise from this set of results.

1) The industrial HAZOP team flag as a cause of no flow, "New remote isolation valve closes spuriously". This was not modelled in the computer method because the isolation valve was added to the design by the HAZOP team to counter against "Major pump seal failure".

2) The point relating to an increase in sulphur content is not picked up. Unlike the industrial HAZOP team, the computer method is not able to look beyond the boundaries of the configuration. The method can cope with the effects of impurities within the system, but not outside it, since the 'outside' is not modelled.

3) The point relating to the contamination by nitrogen could have been inserted into either the pipe or pipeline model. However, it is felt that it would be inappropriate to do so. This is a complex fault, involving two different modes of operation of the plant: pressuring up and normal day-to-day operation.
Table 6.8

Comparison of Results for

Propane Pipeline Example
<table>
<thead>
<tr>
<th>PROCESS DEVIATION</th>
<th>ORIGINAL HAZOP CAUSES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</th>
<th>ORIGINAL HAZOP CONSEQUENCES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO FLOW</td>
<td>Level lost in main propane storage tank</td>
<td>Line 1</td>
<td>Tank1 - 1 3 none, q 2 none</td>
<td>On-line export pump(s) over-heats due to loss of suction, leading to pump seal leak and possible fire</td>
</tr>
<tr>
<td></td>
<td>Export pump(s) lost in service (due to site power failure, pump faults or pump trip)</td>
<td>pipe blockage, leak (major)</td>
<td>Propane export to consumer divider q 1 none, q 2 none, q 3 none</td>
<td></td>
</tr>
<tr>
<td></td>
<td>valve directed closed</td>
<td>pump loss of NPSh rotation fault, impeller fault, cavitation pressure low, temperature high, delivery shut in no change to standby standby fail on demand maloperation of valves no reset of delivery valve after lift</td>
<td>delay in liquid upstream of export NRV flashes back to tank. (But see also &quot;reverse flow&quot; for other consequences when pipeline backflow protection fails to operate on demand.)</td>
<td></td>
</tr>
<tr>
<td>Major pump seal failure</td>
<td>blockage, leak (major)</td>
<td>Major leakage in pump compound leading to a flammable cloud that would almost certainly ignite</td>
<td></td>
<td></td>
</tr>
<tr>
<td>urv valve stuck closed</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New remote suction isolation valve closes spuriously</td>
<td>Line 3</td>
<td>divider q 1 none, q 2 none, blockage leak (major)</td>
<td></td>
<td>exchanger q 1 none</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS DEVIATION</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------</td>
<td>------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>NO FLOW (contd)</td>
<td>Pump delivery RV lifts and fails to reset when only one pump is on line</td>
<td>Line 3 (contd) pipe - blockage leak (major) valve - valve directed closed</td>
<td>Full pump delivery rate passing back to tank via failed RV</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pump kickback FCV remains shut on demand due to control loop or valve faults</td>
<td>Line 4 exchanger-q 1 none q 2 none pipe - blockage leak (major)</td>
<td>Export system pressure rises to pump closed head delivery pressure. On-line pump(s) overheat, leading to pump seal leak and possible fire</td>
<td>Line 4 tank 18 - q 1 none 1 3 none outflow(s) none</td>
</tr>
<tr>
<td></td>
<td>Propane export low temperature trip valve closes spuriously or on demand Gross failure of pipeline due to (i) defects arising in the manufacture of pipe sections, during transit or during installation, (ii) impact damage by vehicular traffic, (iii) damage due to climatic factors, or (iv) deterioration in service e.g. corrosion (Failure due to high or low metal temperatures, or high pressure is covered later).</td>
<td>nrv's - valve stuck closed trip - spurious closure blockage double block and bleed's - isolation valve closed leak (major) drain valve open pipeline - blockage leak (major) control valve - loop fails closed sensor fails closed controller fails closed control valve fails closed set point moved closed isolation valve closed valve - valve directed closed</td>
<td>Large release of flashing liquid in public areas with high chance of ignition and risk to life from radiation burns or deflagration. The normal pipeline inventory is 160 t; and further quantities could be discharged to atmosphere due to continuing export and/or backflow from the buffer tank if the tank NRV fails to operate.</td>
<td></td>
</tr>
<tr>
<td>PROCESS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>REVERSE FLOW</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Export delivery pressure lost for whatever reason, coincident with failure of the export NRW and low-temperature trip protection to operate on demand</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>MORE FLOW</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure or misdirection of the buffer tank level control system, or LCV bypass too far open in error</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ORIGINAL HAZOP CAUSES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</th>
<th>ORIGINAL HAZOP CONSEQUENCES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liquid in the pipeline will flash back to the main storage tank, leading to low pipeline temperatures and the chance of brittle fracture if the pressure falls below about 2 bar gauge.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Line 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tank 1 - 13 high</td>
</tr>
<tr>
<td>q 2 high</td>
</tr>
</tbody>
</table>

| Buffer tank overpressured to export pump shut-off head with high chance of rupture if no remedial action taken by operator on rising level, and if tank high level trip and tank RV both fail to cope. Flare header at risk from brittle fracture when vessel overfills and RV operates. |

<table>
<thead>
<tr>
<th>Line 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tank 1 - 13 low</td>
</tr>
<tr>
<td>q 1 high</td>
</tr>
<tr>
<td>q 2 high</td>
</tr>
</tbody>
</table>

| Divider - q 1 high |
| q 2 high |

| Blockage in other leg |

<table>
<thead>
<tr>
<th>Line 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Divider - q 1 high</td>
</tr>
<tr>
<td>q 2 high</td>
</tr>
</tbody>
</table>

| Exchanger - q 1 high |
| q 2 high |

| Flare |

<table>
<thead>
<tr>
<th>Line 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exchanger - q 1 high</td>
</tr>
<tr>
<td>q 2 high</td>
</tr>
</tbody>
</table>

| Control valve fails open |
| Set point moved open |
| Bypass fails directed open |

<table>
<thead>
<tr>
<th>Line 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tank 18 - q 1 high</td>
</tr>
<tr>
<td>13 high</td>
</tr>
</tbody>
</table>

<p>| Tank overfilled |
| Liquid enters vent |</p>
<table>
<thead>
<tr>
<th>PROCESS DEVIATION</th>
<th>ORIGINAL HAZOP CAUSES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</th>
<th>ORIGINAL HAZOP CONSEQUENCES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>MORE PRESSURE</td>
<td>Sudden closure of export or buffer tank trip valves, or buffer tank LCV.</td>
<td>Line 1 - Tank 1 1 3 high 1 2 high 3 high</td>
<td>Failure of pipeline equipment or fittings due to surge pressure or hammer.</td>
<td>Line 1 - Pipe 1 leak (major) pipe rupture flange leak</td>
</tr>
<tr>
<td></td>
<td>Thermal expansion of pipeline inventory when valved-in</td>
<td>pipe - blockage heat to blocked in fluid solar heat fire</td>
<td>Failure of line, joints or valve glands.</td>
<td>divider - p 1 high p 2 high p 3 high</td>
</tr>
<tr>
<td></td>
<td>valve - valve directed closed</td>
<td>pump - delivery shut in blockage heat to blocked in fluid solar heat fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Line 3 divider - p 1 high p 2 high blockage</td>
<td>pipe - blockage heat to blocked in fluid solar heat fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>exchanger - p 1 high p 2 high</td>
<td>valve - valve directed closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>exchanger - tube fouling</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>PROCESS DEVIATION</th>
<th>ORIGINAL HAZOP CAUSES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</th>
<th>ORIGINAL HAZOP CONSEQUENCES</th>
<th>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pressure MORE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(contd)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Line 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>exchanger</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>p 1 high</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>p 2 high</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipe</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>heat to blocked</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>in fluid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>solar heat</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>trip</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>valves</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>double block and bleed's</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>isolation valve</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipeline</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>hammer/surge</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>heat to blocked</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>in fluid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>solar heat</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>control valve</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>loop fails</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>sensor fails</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>controller fails</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control valve</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>fails</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>set point moved</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>isolation valve</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>valve</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>valve directed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Line 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipe</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>leak (major)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe rupture</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>flange leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>tank 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>p 1 high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------</td>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>TEMPERATURE</td>
<td>Fire adjacent to propane transfer route (export area, pipeline or consumer area).</td>
<td>Line 1&lt;br&gt;tank 1 - t 3 high&lt;br&gt;t 2 high&lt;br&gt;pipe - external heat source&lt;br&gt;hot weather fire&lt;br&gt;pump - pump fluid overheats</td>
<td>High metal temperatures leading to softening and loss of containment at normal operating pressure</td>
<td>Line 1&lt;br&gt;propane - vapour formation&lt;br&gt;divider - t 1 high&lt;br&gt;t 2 high&lt;br&gt;t 3 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Line 3&lt;br&gt;divider - t 1 high&lt;br&gt;t 2 high&lt;br&gt;pipe - external heat source&lt;br&gt;hot weather fire</td>
<td></td>
<td>Line 3&lt;br&gt;propane - vapour formation&lt;br&gt;exchanger - t 1 high&lt;br&gt;t 2 high&lt;br&gt;t 4 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Line 4&lt;br&gt;exchanger - t 1 high&lt;br&gt;t 3 high&lt;br&gt;t 2 high&lt;br&gt;q 3 high&lt;br&gt;q 1 low&lt;br&gt;pipe - external heat source&lt;br&gt;hot weather fire</td>
<td></td>
<td>Line 4&lt;br&gt;propane - vapour formation&lt;br&gt;tank 18 - t 1 high</td>
</tr>
<tr>
<td>FLOW</td>
<td>Second export pump not commissioned when consumer requirements exceed single pump rate.</td>
<td>Line 1&lt;br&gt;tank 1 - 1 3 low&lt;br&gt;q 2 low&lt;br&gt;pipe - blockage leak (major)&lt;br&gt;valve - valve partly closed</td>
<td>Loss of vaporizer feed if no action taken, but several hours normally available</td>
<td>Line 1&lt;br&gt;1 3 high&lt;br&gt;divider - q 1 low&lt;br&gt;q 2 low&lt;br&gt;q 3 low&lt;br&gt;Propane escape to atmosphere and possible fire</td>
</tr>
<tr>
<td>PROCESS DEVIATION</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------</td>
<td>--------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>LESS FLOW</td>
<td>Tube leak on propane</td>
<td>Line 1 (contd) -</td>
<td>High pressure, and possibly</td>
<td>Line 3 exchanger - q 1 low</td>
</tr>
<tr>
<td>(contd)</td>
<td>pump (contd) -</td>
<td>delivery part shut</td>
<td>sub-zero temperatures</td>
<td>q 2 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>only one on-line when</td>
<td>developed on glycol</td>
<td>t 2 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>two required blockage</td>
<td>circulation system</td>
<td>t 4 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td>leak (major)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>nrv</td>
<td>- valve fails</td>
<td>Line 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>part closed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line 2</td>
<td>divider - q 1 low</td>
<td>q 2 low</td>
<td>q 1 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>q 2 low</td>
<td>blockage</td>
<td>q 2 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>leak (major)</td>
<td>t 2 high</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td>- blockage</td>
<td>q 2 low</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- leak (major)</td>
<td>tube fouling</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>tube leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td>valve</td>
<td>- valve partly</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>closed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line 4</td>
<td>exchanger - q 1 low</td>
<td>q 2 low</td>
<td>q 1 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>q 2 low</td>
<td>tube fouling</td>
<td>q 2 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>tube leak</td>
<td>t 2 high</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(major)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td>- blockage</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- leak (major)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>nrv's</td>
<td>- valve fails</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>part closed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>trip valves</td>
<td>- blockage</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>double block and</td>
<td>- isolation valve</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bleed's</td>
<td>- part closed</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- leak (major)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- drain valve open</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>pipeline</td>
<td>- blockage</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>- leak (major)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS DEVIATION</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------</td>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>LESS FLOW (contd)</td>
<td>Line 4 (contd)</td>
<td>control - loop fails part</td>
<td>Liquid propane for pipeline</td>
<td>Liquid flashes as tank</td>
</tr>
<tr>
<td></td>
<td></td>
<td>valve - loop fails part</td>
<td>flashes into buffer tank as</td>
<td>propane - liquid flashes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td>tank pressure falls leading</td>
<td>p 1 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sensor fails part</td>
<td>to low pressure</td>
<td>p 2 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>part closed</td>
<td>pipeline temperatures and to</td>
<td>p 2 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>controller fails part</td>
<td>the divider pressure</td>
<td>p 3 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td>part closed</td>
<td>falls below about 2 bar</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control valve part</td>
<td>gauge</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>fails part</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>set point moved</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>part closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>isolation valve part</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>part closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>valve - valve partly</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LESS PRESSURE</td>
<td>Buffer tank EV lifts</td>
<td>Line 1</td>
<td>Line 1</td>
<td></td>
</tr>
<tr>
<td>and fails to reseat</td>
<td>tank 1</td>
<td>1 3 low</td>
<td>propane</td>
<td></td>
</tr>
<tr>
<td>or tank HIC opens</td>
<td></td>
<td>p 3 low</td>
<td>- liquid flashes</td>
<td></td>
</tr>
<tr>
<td>spuriously, when</td>
<td></td>
<td>p 2 low</td>
<td>as tank pressure falls</td>
<td></td>
</tr>
<tr>
<td>pipeline route is</td>
<td></td>
<td></td>
<td>leading to low pressure</td>
<td></td>
</tr>
<tr>
<td>open i.e. in service</td>
<td>pipe</td>
<td></td>
<td>temperatures</td>
<td></td>
</tr>
<tr>
<td>Buffer tank blown</td>
<td>Line 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>down manually via</td>
<td>divider</td>
<td>p 1 low</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HIC (e.g. for a</td>
<td></td>
<td>p 2 low</td>
<td></td>
<td></td>
</tr>
<tr>
<td>planned shutdown or</td>
<td>pipe</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>in an emergency)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>without first</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>isolating the pipe-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>line.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Line 4</td>
<td>exchanger</td>
<td>p 1 low</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>p 2 low</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>tube leak</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(major)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>double block and bleed's -</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>leak (major)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>drain valve open</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------</td>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>LESS TEMPERATURE</td>
<td>Inadequate heating of Line 1 exported propane due to faults on heating system (e.g. inadequate glycol circulation rate via propane heater, glycol temperature at inlet of propane heater too low, fouling of propane heater)</td>
<td>tank 1 - t 3 low t 2 low</td>
<td>Chance of brittle fracture of pipeline if propane export low temperature trip fails to prevent liquid propane entering pipeline at below -15°C</td>
<td>Line 1 - pipe - leak (major) brittle fracture</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe - external cold source cold weather lagging loss</td>
<td></td>
<td>divider - t 1 low t 2 low t 3 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------</td>
<td>-------------------------------------------</td>
<td>---------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>ESCAPE</td>
<td>N/A</td>
<td>Line 1</td>
<td>N/A</td>
<td>Line 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>tank 1</td>
<td></td>
<td>propane,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>- fire hazard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe</td>
<td></td>
<td>explosion hazard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>valve</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pump</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>nrv</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>divider</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Line 3</td>
<td></td>
<td>propane,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>divider</td>
<td></td>
<td>- fire hazard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe</td>
<td></td>
<td>explosion hazard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>valve</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>exchanger</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Line 4</td>
<td></td>
<td>propane,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>exchanger</td>
<td></td>
<td>- fire hazard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipe</td>
<td></td>
<td>explosion hazard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nrv's</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>trip</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>valves</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>double block and bleed's</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>pipeline</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>control</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>valve</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>tank 18</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CHANGE IN COMPOSITION</td>
<td>High sulphur propane</td>
<td>Serious corrosion problems in</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(sulphur content)</td>
<td>off-loaded from ship</td>
<td>combustion chambers of user units</td>
<td></td>
<td>supplied from the site fuel gas</td>
</tr>
<tr>
<td></td>
<td>into main storage</td>
<td>system, but cracking furnaces not</td>
<td></td>
<td>adversely affected by sulphur-rich propane gas</td>
</tr>
<tr>
<td></td>
<td>tank (occasional</td>
<td>problem; normally only few ppm sulphur</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROCESS Deviation</td>
<td>ORIGINAL HAZOP CAUSES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CAUSES</td>
<td>ORIGINAL HAZOP CONSEQUENCES</td>
<td>COMPUTER-AIDED HAZARD IDENTIFICATION CONSEQUENCES</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------</td>
<td>---------------------------------------------</td>
<td>-----------------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Contamination (Nitrogen)</td>
<td>Nitrogen used for initial pressurisation at start-up not properly purged during commissioning.</td>
<td>Buffer tank pressure higher than normal and dissolved nitrogen present in feed to vaporiser</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contamination (Propane in Glycol System)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start-up Errors</td>
<td>Initial pressurisation of system with nitrogen not carried out properly, or propane heating system not properly commissioned, before export started.</td>
<td>Brittle fracture of pipeline possible if liquid propane enters pipeline at below -15°C or when pipeline pressure is below 2 bar gauge</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Start-up Requirements</td>
<td>Gearbox lube oil low pressure trips on propane export pumps are not fitted with a time delay</td>
<td>Pumps cannot be started</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shutdown Errors</td>
<td>Pipeline pressure inadequate at critical stages during shutdown and deburdening of the pipeline</td>
<td>Brittle fracture of pipeline possible if pressure falls below 2 bar gauge when significant amounts of liquid propane are present.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Routine Testing</td>
<td>Faults on automatic trip protection, NRV's alarms etc.</td>
<td>Automatic protection or alarms fail to operate on demand, thereby adversely affecting overall safety.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This problem may be seen as equivalent to a problem in a batch operation. The computer method is not designed to deal with this.

4) The start-up, shutdown and testing operations are similar to the maintenance point of 6.1.2 i.e. the computer method is not designed to pick these up.

6.3 Conclusions of the Comparison

In the comparisons of 6.1.2 and 6.2.2, the emphasis has been on finding the differences between the two sets of results and thus exposing any deficiencies in the computer method. However, terminology apart, there are considerably more points of agreement but these are obvious and do not require declaration.

There are a number of general points about the deficiencies of the computer method that are raised by the two sets of results. These are:

1) Concentration. This is a variable unlike pressure or flow in that it refers to the concentration of a particular component. In this respect it is similar to impurity. However, concentration differs from impurity in two respects. The first is that impurity is assumed to be always deviated high. The very term implies that there is no low level of impurity. The second difference is illustrated with the aid of an example. If two fluids, a and b, are mixed by a binary header, then the concentration of a in the mixed stream is deviated high if the flow of a is deviated high or if the flow of b is deviated low (and vice versa). An impurity in either inlet stream, on the other hand, simply results in that impurity in the outlet stream.
Although this computer method is able to deal with concentration on a simple basis (i.e. pure components mixing at vessels) it needs to be developed further.

Note that it is possible to deal with some concentration type problems on the basis of impurity.

2) Operability and Maintenance. At this stage, the computer method lacks the facility to deal with items of this nature.

3) Reverse flow and reverse pressure gradient. Again, this computer method lacks the facility to deal with these variable deviations.

4) Batch operation. The computer method lacks the facility to deal with modes other than the continuous operation mode.

5) Materials of construction. Although the comparison of results has not highlighted this problem, it is convenient to discuss this point here. In this computer method, only one materials of construction model has been specified for the entire plant configuration. Clearly, this is not very realistic. In a plant design, many different materials of construction may be specified. However, it was felt that restriction to one materials of construction model would both demonstrate the use of such a model and reduce the complexity of the plant configuration data. As is evidenced by the comparison of results, this has not significantly detracted from the conclusions of the computer method.

It must be stated that the advantage of any computer method is that it will perform the set task thoroughly, time and time again, without fatigue or error. There is always the danger, with a human HAZOP team that they may omit to look at
a particular deviation or omit to consider a process line and so on. These problems do not arise with a computer method. The human HAZOP team also has difficulty in following a process line from one P&ID to the next. This problem does not exist with a computer method. The whole plant is always on one "diagram".
7 Conclusions

This computer method, as has been demonstrated, performs a useful function in the process of identifying potential hazards in the design of process plant.

The method can be used as it stands or further developed to form a system that could be referred to as truly expert. In the latter case, it would form the core of the expert system. It already embodies some of the features of an expert system and so enhancement in the same philosophy could result in a credibly expert system.

The method does not attempt to deal with the recommendations that are made by the HAZOP study team. It is a tool purely to aid the process of the identification of hazards.

There are two general points to be noted. The first is that an extensive model library needs to be developed. The approach to this has been to create a model only when there is a use for it in a particular problem, thus ensuring that the model has been tested in use before it is put into the library. The second point is that the method needs further development in the area of the filtering/pruning of the results.

The conclusions are now divided into two sections. The first deals with the applications of the method with little or no modification and the second deals with the significant changes that could be made to the method.
7.1 Applications

There are a number of possible applications, but these are mainly variants on one of two themes:

1) Use by a process design team
2) Use by a HAZOP team

These categories are discussed in greater detail below.

7.1.1 Use by a Process Design Team

It is easy to envisage this computer method operating with an interface on the process design as it is being developed by the design team.

The procedure that is envisaged is that the design team would send on to the draughtsman the details on the design. These days, more often than not, the design is entered into a CAD package by the draughtsman. The CAD package is used to produce the P&IDs. The information utilised by the CAD package is almost all the information that is required by this computer-aided hazard identification method. Thus, an interface between the CAD package and this computer method would allow hazard identification on the design. This would be done overnight and the output would be delivered to the design team the following morning. The design team would make the necessary alterations and then proceed with the design.

The procedure suggested here would attempt to iron out the problems at the design stage. This is a cheaper and more convenient solution. Once the design is finalised, alterations to it may prove time consuming and expensive.
7.1.2 Use by a HAZOP Team

The output of the computer method could be used by the HAZOP team as a starting point for discussion. It is possible, though, that this may lead to the study team developing blind spots. Blind spots may take two forms. They may lead to the team either dismissing, out of hand, the results of the computer method or the team may rely too heavily on the results. One way of countering this may be to let only one of the HAZOP team, say the secretary, see the results of the computer method. These results can then be used to aid and spark off the discussion of the study team.

There is increasing demand in the chemical industry for a 'cheap' HAZOP study. Here, the HAZOP team consists of just one person. This one-man HAZOP is usually performed by someone who would fit the requirements of the leader of the more usual HAZOP team. These one-man HAZOP studies are not desirable in that they compromise the principles of HAZOP studies. However, they do offer a cost benefit. In this sort of cheap HAZOP the computer method would quickly prove a valuable aid. The results would be used by the person performing the HAZOP to ensure that the study is as comprehensive as possible.

7.2 Improvement and Enhancement

There are a few points relating to the improvement and enhancement of the computer method. These points will be discussed in turn.

1) Concentration

The utilities to handle concentration are already available but these need to be improved. Currently, the method can deal with simple concentration problems (two pure fluids mixing at a binary header, for example).
Improvements to the concentration utilities would include being able to handle mixing of fluids that contain more than one component and being able to follow the concentration change through the plant. It is felt that the structure of the method will be quite able to support the suggested improvements.

2) Materials of Construction

It has always been intended to allow the materials of construction facility to take a materials of construction model for each of the different units in the plant configuration. This is a relatively minor change.

3) Reverse Flow and Reverse Pressure Gradient

Reverse deviations are rather more complex. Unlike the other deviations, the causes of reverse deviations will generally be ANDed i.e. they will have the form "IF cause x AND cause y AND cause z THEN reverse flow". (Other deviations generally take the form "IF cause x OR cause y OR cause z THEN no flow").

Utilities to deal with these reverse deviations need to be developed.

4) Maintenance and Operability

It is felt that this section can be dealt with within the current framework. It would be possible to define a set of rules regarding the maintenance and operability aspects. For example, a rule could be added to check for the existence of drain and purge points in each process line. The drain points would also be checked to ensure that they were at the lowest point in the line. The purge points would be checked to ensure that there was access to purge
gas at that point.

Similar rules could be developed for the provision of positive isolation of process lines and vessels, the provision of manholes for vessels and so on.

5) Observability

Although this point is one that has not been raised previously, it is felt that this computer method could be improved by the addition of facilities to check the observability of process deviations. This would check on the instrumentation on the plant.

The form of the utilities for observability would be quite similar to those for maintenance and operability described above.

6) Batch Operation

This computer method is not designed for nor suited to batch operations.
8 User Manual

8.0 Introduction

This User Manual has been divided into five sections, each of which deals with one particular aspect of the computer method. These different aspects have already been described in Chapter 4 and are illustrated by the schematic diagram of Figure 4.2 in Chapter 4. Much of the material of this User Manual refers to the chapter on Modelling, Chapter 5.

8.1 The Creation of Process Unit Models

8.1.1 Developing a Process Unit Model

First, a unique and meaningful name is picked for the unit model. The name assigned to the unit model can be any length up to a maximum of 10 characters.

The next step is the actual modelling of the process unit. The modelling consists basically of four stages.

1) Assign port numbers to the different ports of the process unit
2) Develop propagation equations for the process unit
3) Develop initial event statements for the process unit
4) Develop terminal event statements for the process unit

These stages are described in greater detail below.

1) Assign port numbers to the different ports of the process unit. Ports are generally the points of connection to other process units. The convention followed here is that units with only one inlet and one outlet have port 1 as their inlet and port 2 as their outlet. Process vessels may, in addition to the points of connection to other process
units, need a port number assigned to the inside of the vessel. This is so that propagation equations and initial and terminal event statements can refer to the internal conditions of the process vessel. This point is dealt with in greater detail by the following three sections. It is envisaged that considerations of level or pressure will be the most frequent use of internal ports.

2) Develop propagation equations for the process unit. This is done by considering each process variable at each internal and outlet port and deliberating the effect, on that process variable at that port, of deviations in other process variables as they pass through the unit. This is probably best illustrated by an example. Consider a simple open isolation valve. Following the convention described in (1) above, the port numbers are assigned 1 to the inlet and 2 to the outlet as shown in the diagram.

![Figure 8.1 Open Isolation Valve](image)

If the variable under consideration is pressure and the port under consideration is the outlet port, port 2 (there is no internal port) then all the process variables are examined for any effect they may have on pressure at port 2. Clearly, the only variable affecting pressure at port 2 is pressure at port 1. Furthermore, the relation between these is that a deviation in pressure at port 1 will result in a deviation in pressure at port 2 of the same sign. A
more precise form of words is that deviations in pressure at port 2 are a function only of deviations of the same sign in pressure at port 1. The same reasoning can be applied to the variables flow and temperature with the same result. These propagational relationships are more briefly expressed by the use of propagation equations. Propagation equations for the relationships discussed above are as follows.

\[ p_2 = f(p_1) \]
\[ q_2 = f(q_1) \]
\[ t_2 = f(t_1) \]

The procedure for defining propagation equations for impurities is slightly different. Impurity needs to be expressed as that of a particular substance. The convention in this computer method is to use single letters of the alphabet to denote each one of the impurities. In this case, we may wish to express the fact that the impurity level of \( a \) at port 2 is a function of the impurity level of \( a \) at port 1. The propagation equation is

\[ i_a-2 = f(i_a-1) \]

Clearly, it is not very desirable to have to type in a propagation equation such as this for each of the impurities. Thus, it is possible to use a mnemonic to represent any of the impurities. This mnemonic is the underscore character. Thus, the general propagation equation for each of all the impurities is

\[ i_-2 = f(i_-1) \]

Let us now discuss a rather more complicated unit, a binary header. The port numbers are assigned as shown in the diagram below.
This time, for pressure at port 3 we conclude that it is a function of both the pressure at 1 and the pressure at 2. The same applies to the process variables flow and temperature. The propagation equations for these relationships are

\[ p_3 = f(p_1, p_2) \]
\[ q_3 = f(q_1, q_2) \]
\[ t_3 = f(t_1, t_2) \]

In the case of the propagation equations for impurities, it is clear that an impurity in either inlet will lead to the impurity in the outlet. The propagation equation for this relationship is

\[ i_{-_3} = f(i_{-_1}, i_{-_2}) \]

What about the process variable concentration? In the example of the isolation valve consideration of concentration was simple: the concentration out was a function of the concentration in. However, in the case of a binary header there may be two different fluids being
combined together and so consideration of concentration is rather more important. The definition of propagation expressions relating to concentration is complicated by a need to express concentration as that of a particular component. This problem has been overcome by the definition of the component as the component from inlet port 1 (or 2). Thus, we have the propagation expression: deviations in concentration of the component from port 1 at port 3 are a function of deviations of the same sign in concentration of the component from port 1 at port 1 and in flow at port 1 and of the opposite sign to deviations in flow at port 2.

This expression is perhaps made slightly clearer by the propagation equation form of the expression.

\[ x_{1-3} = f(x_{1-1}, q_1, -q_2) \]

The same reasoning also applies to the concentration at port 3 of the component from port 2. This results in the propagation equation

\[ x_{2-3} = f(x_{2-2}, q_2, -q_1) \]

This example of a binary header has demonstrated two points. The first point is the proposed way of dealing with the process variable concentration and the second is the use, in propagation equations, of a negative sign to denote deviations of the opposite sign.

The use of internal port numbers is illustrated in the case of a simple closed tank with one inlet and one outlet. Port numbers are assigned as shown in the diagram below.
Port 3 has been assigned to the level within the tank. With internal ports such as this, we also introduce the process variable level. When considering the variable level, we conclude that deviations in level are a function of deviations of the same sign in flow at port 1 and of the opposite sign to flow at port 2. This statement translates to the propagation equation below.

\[ l_3 = f(q_1, -q_2) \]

If the flow out of the vessel is gravitational then we may also conclude that deviations in flow at port 2 are a function of deviations of the same sign in level at port 3. i.e.

\[ q_2 = f(l_3) \]

3) Develop initial event statements for the process unit. Initial event statements are the means by which the causes of process deviations are attributed to faults within process units. The procedure to be followed is to consider all the possible faults that could arise within the unit and decide what process deviations would result from each of these faults. For example, let us consider the faults
that may arise in a centrifugal pump with ports numbered as shown in the diagram.

![Diagram of a Centrifugal Pump with Ports 1 and 2]

Figure 8.4 Centrifugal Pump

We could propose the possible set of faults given below.

- loss of NPSH
- rotation fault
- impeller fault
- cavitation
- blockage
- delivery shut in
- pump overheats

The next step is to write down, for each one of this set of fault conditions, all the process deviations that may be caused by the fault condition. Note that each of these resultant process deviations should actually be assigned to a specific port. In the case of the fault condition "delivery shut in" the resultant process deviations are "pressure at 2 high" and "flow at 2 none". The full set of fault conditions together with all of their respective resultant process deviations is
<table>
<thead>
<tr>
<th>Fault Condition</th>
<th>Resultant Process Deviations</th>
</tr>
</thead>
<tbody>
<tr>
<td>loss of NPSH</td>
<td>q2 low, q2 none</td>
</tr>
<tr>
<td>rotation fault</td>
<td>q2 low, q2 none</td>
</tr>
<tr>
<td>impeller fault</td>
<td>q2 low, q2 none</td>
</tr>
<tr>
<td>cavitation</td>
<td>q2 low, q2 none</td>
</tr>
<tr>
<td>blockage</td>
<td>q2 low, q2 none, p2 high</td>
</tr>
<tr>
<td>delivery shut in</td>
<td>q2 none, p2 high</td>
</tr>
<tr>
<td>pump fluid overheats</td>
<td>t2 high</td>
</tr>
</tbody>
</table>

It is clear from the above table that, because their sets of resultant process deviations are the same, the first four fault conditions could be lumped together. A more concise way of noting the information given in the table is in the form of initial event statements. These are

```
loss of NPSH, rotation fault, impeller fault, cavitation
   q2 low, q2 none

blockage
   q2 low, q2 none, p2 high

delivery shut in
   q2 none, p2 high

pump fluid overheats
   t2 high
```

Note that the left hand side of the equation (the cause) is separated from the right hand side (the effect) by "->". It is possible to have further causes of the fault conditions given above. If we had wanted to express the fact that "delivery shut in" could be a result of "operator error" then all that is required is the additional fault condition rule

```
operator error
   delivery shut in
```

176
In order to aid the user, the initial event statements are referred to by the following categories:

- Function
- Hydraulic
- Containment
- Environment

These categories have already been discussed in Chapter 5.

4) Develop terminal event statements for the process unit. Terminal event statements are the means by which the effects of process deviations on the process unit are expressed. The procedure to be followed is to consider all the relevant process deviations in turn and decide what the effects, if any, of each of the process deviations on the process unit would be. The relevant process deviations for process line units and for process vessels are discussed in 4.3.5.

Consideration of the centrifugal pump of (3) above gives a ready example of an effect rule. If the pump is running and there is no flow through it then the pump fluid is liable to overheat. This terminal event statement can be expressed in the usual manner as shown below.

q2 NONE > pump fluid overheats

Another example is that of an electrically heated steam kettle. This is illustrated with the appropriate port numbers below.
If the level should fall greatly, the heating element would be exposed, an obvious hazard. The terminal event statement in this case would be

13 LO>exposure of heating coil

8.1.2 Running the "unit model maker" Program

Once the process unit has been modelled on paper, the "unit_model_maker" program can be used to add the new process unit to the model library.

On running the program the following items of information are requested in order

i) Name of the unit model
ii) The propagation equations, one at a time
iii) The initial event statements, one at a time
iv) The terminal event statements, one at a time

Note that all the equations and statements must be entered enclosed in quotation marks.
The rules of syntax for the equations and statements are

1) Variables are denoted by the letters p, q, t, l, i, x

2) All variables including i and x are immediately followed by a port number

3) All occurrences of x, concentration, are followed by a port number, then a minus sign and then another port number. The first port number refers to the port through which the component, whose concentration is being defined, enters the unit. The second port number is the port at which the concentration is being defined.

4) All occurrences of i, impurity, are followed by a letter denoting the impurity, then a minus sign and then the port number. The mnemonic for any substance, "_", can be used to replace the impurity letter.

5) For propagation rules, the form of the equation is a variable/port combination followed by ",f(" followed by any number of items. (Variable/port combinations are described by (1), (2) and (3) above). Each term consists of an optional minus and a variable/port combination. Note that all negative terms should be placed together at the end of the equation. All terms are separated by commas and the equation is closed by ")".

6) For initial and terminal event statements, the form of the equation is left hand side, ",", right hand side. Each side of the equation consists of any number of terms separated by commas. Each term consists of either a piece of text (of maximum length 20 characters) or a variable/port combination followed by a space followed by a deviation code (LO, HI or NONE).

7) Extra spaces will not be tolerated.

8) Port numbers are restricted to numbers between 1 and 9 inclusively.
The examples of incorrect syntax given below will reinforce the rules described above.

\[ q_2 = f(q_1) \]  
Extra space between "2" and "="

\[ x_1 = f(q_1, q_2) \]  
Concentration term incorrect

\[ x_1 - l = f(-q_1, q_2) \]  
Negative term should appear after all positive terms

\[ q \mathrm{NONE}\overheating \]  
No port number after q

\[ \mathrm{blockage}\overrightarrow{Q2LO} \]  
Lower case q and space required between port number and deviation code

The following limitations have been declared on the variables used in the program.

i) The length of the name is restricted to a maximum of 10 characters

ii) The length of all the equations typed in is restricted to a maximum of 160 characters

iii) There is a maximum of 30 equations in total that can be entered

iv) The text terms in the event statements are limited to a maximum length of 20 characters

These limitations can be changed by altering the relevant declaration statements in the program.

8.1.3 Entering the Model in the Model Datafiles

Whenever a unit model is created, its details must be entered into one of two datafiles, "In_line_units_file" or "Vessels_file". The choice of datafile will depend on the classification of the process unit as an in-line unit or a vessel. These datafiles are used by the "configurator" program to divide up the plant configuration into process lines and
process vessels.

The process of entering the details of a unit is the same for both datafiles. The user steps to the end of the datafile and on a new line types in the name of the unit model, the number of inlet ports and the number of outlet ports. These three items are separated by spaces. On the line below this, the user enters the actual numbers assigned to all the inlet ports, again separated by spaces. If there are no inlet ports then a blank line is inserted. The actual numbers of the outlet ports are entered on the next line. If there are no outlet ports then a blank line is inserted. This is best illustrated with the aid of an example. Let us say we have modelled a heat exchanger, a process vessel, with two inlet ports, 1 and 3, and two outlet ports, 2 and 4. The name assigned to this model is, say, "hx". The relevant entry in "Vessels_file" will be as shown below.

hx 2 2
1 3
2 4

8.2 The Creation of Process Material Models

8.2.1 Developing a Process Material Model

The first thing to do (as with developing a process unit model) is to decide on a unique name for the model. This name can be any length up to a maximum of 10 characters.

The actual modelling of the process material consists of two stages:

1) Designation of the causes of keywords, its properties
2) Designation of the effects of process deviations, its susceptibilities
These stages are described in greater detail below.

1) **Designation of the causes of keywords.** This deals with the characteristics of the process material that will give rise to any of the set of keywords referred to in 4.3.4. The procedure to be followed is to consider each of the keywords in turn and decide which, if any, of the characteristics of the material being modelled can cause the keyword in question. All the relevant characteristics are noted and then the next keyword is reviewed.

Let us look at the modelling of a particular olefin. For the keyword BLOCKAGE it may be concluded that "polymerisation" of the olefin is possible and would cause BLOCKAGE. Similarly, "low boiling point impurity" may be deemed a possible cause of CAVITATION.

2) **Designation of the effects of process deviations.** This means the effects of process deviations upon the process material being modelled. Here, the procedure is to consider each process deviation in turn and decide what the effects of that process deviation on the process material would be. Attention must also be paid to the deviation "escape". In the case of "escape" the task is to decide what effects a leak of the material would have on the material itself and also the effects of the leak on the environment.

In the case of developing a model for liquid propane, the effects of "escape" on the material may be defined as "flashing off of liquid" and "fire/explosion hazard". If the material being modelled was water, the effect of the deviation "low temperature" would be defined as "freezing". More examples can be found in 4.3.4.
8.2.2 Running the "material modeller" Program

Running "material_modeller" will cause the user to be prompted for the following information.

i) Name of the process material model
ii) The causes, one at a time, of each of the keywords in turn
iii) The effects on the process material of each of the process deviations in turn

Note that the causes and effect input in (ii) and (iii) above must be enclosed in quotation marks if use is made of spaces.

There is a declared limitation of a maximum of 20 characters on the length of the causes and effects input by the user. The name assigned to the process material model is similarly limited to a maximum of 10 characters. These limitations can be changed by altering the relevant declarations in the program.

8.3 The Creation of a Process Plant Configuration

8.3.1 Producing a Process Plant Configuration

This section deals with the information required to create a file on the process plant configuration. Conventional HAZOP studies use a P&ID as the basis of the study (see Chapter 1). This computer method, however, is not sophisticated enough to be able to use the instrumentation information available on a P&ID. Although some control loops have been modelled and used (see Chapter 6), these have been rather simple. The quality of information required in the configuration file is reasonably basic. Thus, the input to this configuration file need not be as complex as that provided by a P&ID.
The information required by this computer method is

1) The name assigned to this particular study
2) Names of the units and the models representing those units
3) The links between the units
4) The process material contained in each process line
5) Model of the materials of construction
6) List of the impurities that are referenced

These different items of information are described in greater detail below.

1) The name assigned to this particular study. This name will be used to reference all three files pertaining to the study. The three files are the configuration file, the "identifier" output file and the "compacter" output file. These will all have the same name but will be followed, respectively, by the suffixes "conf", "pro_out" and "compact". The length of the name can be up to a maximum of 10 characters.

2) Names of the units and the models representing those units. The user is asked to name, uniquely, all the process units in the configuration. The user is also asked to assign process models to represent each of the units in the configuration. It is prudent to reiterate, at this stage, that this computer method automatically assigns a generic pipe unit (see 4.3.5 and 5.1.4) to each process line. This enables the user to dispense with the need to enter a unit and a model for each of the sections of piping between the process units.

3) The links between the units. The user needs to define all the links or connections between the process units. Each link is of the form Unit A port x is connected to Unit B
port y. These links need to take no account whatsoever of the generic pipe unit referred to in (2) above.

4) The process material contained in each line. The user needs to look at the flowsheet and decide on what process materials are flowing in which process lines. The process materials will, of course, have their corresponding models in the process material model library.

5) Model of the materials of construction. The user needs to enter a set of terminal event statements that model the materials of construction of the plant. These terminal events indicate the characteristics of the material in terms of its susceptibilities. For instance, if a material is susceptible to brittle fracture if the temperature is low, the following terminal event statement is required.

\[ \text{tl LO>brittle fracture} \]

These terminal event statements are used as part of the generic pipe unit.

6) List of impurities that are referenced. This is used by "identifier" to only run searches on the impurities that are actually referenced in the configuration. Note that the impurities are always denoted by the same letter, whether in the configuration or in the unit models.

8.3.2 Running the "configurator" Program

Running "configurator" will cause the user to be prompted for the following information.

i) The name assigned to the study
ii) The names, one by one, assigned to each of the various
units together with the unit model representing each unit. Note that the process models that are referenced must exist in "In_line_units_file" or "Vessels_file" for them to be accepted.

iii) The links between the units. These need to be entered as upstream unit, upstream port number, downstream unit, downstream port number.

iv) The process materials contained in each process line. "configurator" will have, from the information entered in (i), (ii) and (iii) above, worked out the routing of the various process lines. The user is asked, for each process line in turn, for the process material carried in that line.

v) The terminal event statements for the materials of construction

vi) The list of impurities that are referenced by the configuration

Note that the process unit datafiles, "In_line_units_file" and "Vessels_file" will be consulted in order to divide up the plant configuration into process lines and process vessels.

The configuration file created by "configurator" will have the suffix ".conf".

The following limitations have been declared on the variables used in the program.

i) The length of the filename to a maximum of 10 characters

ii) The length of the unit and model names to a maximum of 10 characters

iii) The number of units in the configuration to a maximum of 40

iv) The number of different models referenced to a maximum of 40
v) The number of vessels to 20
vi) The number of process lines to 20
vii) The number of links to 40
viii) The number of different process materials referenced to 20
ix) The number of different impurities referenced to 20
x) The number of materials of construction terminal event statements to 20

These limitations can be easily altered by changing the relevant declarations in the program.

8.4 Performing a Computer Hazard Identification; Running the "identifier" Program

The only user inputs at this stage are the entry of the name assigned to the study and the selection of an output option. The different output options have already been described in 4.3.5.

The running of "identifier" is a very simple matter. The user enters the PROLOG environment, loads up and runs "identifier". "identifier" will first prompt the user for the name assigned to the study and then, after loading the configuration file, will ask for an output option. The normal output option is "b", that is, output only to a file.

"identifier" takes just over 20 cpu minutes on a Honeywell Multics DPS8 to run the water separator example. Thus, it is suggested that "identifier" is run overnight by an absentee request.

In the cases of the output option selected as "b" or "c", "identifier" produces a file with the suffix ".pro_out". This file can then be used as input by "compacter".
8.5 Producing a HAZOP-like Document; Running the "compacter" Program

The "configurator" program needs as input, the file produced by "identifier". This file will not have been produced if output option a was chosen on running "identifier".

The only user input at this stage is the entry of the name of the study. The "compacter" program then calls up the "identifier" output file (suffix ".pro_out") and the configuration file (suffix ".conf") and uses these to create a HAZOP-like document in a file with suffix ".compact". This file can then be printed out.

The limitations on the variables are the same as those in the other FORTRAN programs, "unit_model_maker", "material_modeller" and "configurator".
References


11 Pyman, M.A.F and Mitchell, F.R. "Identifying Major Process Hazards at the Conceptual Stage" in Design '82, Institution of Chemical Engineers, Rugby, 1982


16 Austin, D.G. and Jeffries, G.V. "The Manufacture of Methyl Ethyl Ketone from 2-Butanol", Institution of Chemical Engineers, Rugby, 1979


34 Coelho, H., Cotta, J.C. and Pereira, L.M. "How to Solve it with Prolog", Laboratorio Nacional de Engenharia Civil, Lisbon, 1980


37 American Institute of Chemical Engineers "Guidelines for Hazard Evaluation Procedures, New York, 1985
APPENDIX A

Listing of "unit_model_maker" Program
unit_model_maker.fortran

program unit_model_maker

Set up and declare variable types

c
character*10 Uname
character*1 Var(6)
character*14 Var_name(6)
character Rule_type(6)
character*14 Rlname(6)
character Reply
character*160 Egn,Eqnarray(30)
character*200 Prolog_eqn,Prolarray(30)
logical Flag
integer Rlnumber,Eqn_count,j,1
character*80 Description
integer P_eqn_count,I_event_count,T_event_count

c
Set up common blocks

common /Variables/ Var
common /prop_text/ Uname,Egn,Prolog_eqn
common /prop_logl/ Flag
common /text_rule_text/ Var_name,Rlname
common /text-rule_nos/ Rlnumber

c
Set up and initialise arrays

Var(1)="p"
Var(2)="q"
Var(3)="t"
Var(4)="l"
Var(5)="x"
Var(6)="i"

Var_name(1)="pressure"
Var_name(2)="flow"
Var_name(3)="temperature"
Var_name(4)="level"
Var_name(5)="concentration"
Var_name(6)="impurity"

Rule_type(1)="f"
Rule_type(2)="h"
Rule_type(3)="c"
Rule_type(4)="i"
Rule_type(5)="e"
Rule_type(6)="effect"

Rlname(1)="functional"
Rlname(2)="hydraulic"
Rlname(3)="containment"
Rlname(4)="environmental"
Rlname(5)="impurity"
Rlname(6)="effect"

Read in model name

1010 write(0, ′("Type in the name of the unit model ... ",$′) )
read(0,*) Uname
open(8, file=Uname, status='NEW', err=9000)
write(0,'(/, "Type in a line of descriptive text (enclosed in quotes) ... ";")
read(0,*) Description
Egn_count=1
1040 write(0,:'/,'"Define a propagation rule ? (y or n) ... ",")')
read(0,*) Reply
if(Reply.eq. "n") goto 1480
if(Reply.eq."y") then
  write(0,:'/,"Type in propagation equation (enclosed in quotes) ... ");")'
read(0,*) Eqn
call propagation
c
print*, Flag
print*, Prolog_eqn
c
if(Flag) then
  Egnarray(Eqn_count)=Eqn
  Prolarray(Eqn_count)=Prolog_eqn
  Eqn_count=Eqn_count+1
endif
goto 1040
1480 P_eqn_count=Eqn_count
write(0,'(/, "Initial Event Statements")')
write(0,'/,'"--------------------------")')
1500 do 1600, Rlnumber=1,5
  length(Rlname(Rlnumber))
1520 write(0,="/,'"Define a "",a," rule ? (y or n) ... ",")')
read(0,*) Reply
if(Reply.eq. "n") goto 1600
if(Reply.eq."y") then
  write(0,="/,"Type in the equation (enclosed in quotes) ... ");")'
read(0,*) Eqn
call text-rules
c
print*, Flag
print*, Prolog_eqn
c
if(Flag) then
  Egnarray(Eqn_count)=Eqn
  Prolarray(Eqn_count)=Prolog_eqn
  Eqn_count=Eqn_count+1
endif
goto 1520
1600 continue
I_event_count=Eqn_count
write(0,="/,'"Terminal Event Statements")')
write(0,'/,'"--------------------------")')
1620 write(0, '"Define an effect rule ? (y or n) ... ",")')
read(0,*) Reply
if(Reply.eq. "n") goto 1650
if(Reply.eq."y") then
  write(0, '/,"Type in the equation (enclosed in quotes) ... ");")'
read(0,*) Eqn
call text_rules
c
print*, Flag
print*, Prolog_eqn
c
if(Flag) then
Egnarray(Eqn_count)=Eqn
Prolarray(Eqn_count)=Prolog_eqn
Eqn_count=Eqn_count+1
endif
endif
goto 1620
c
1650 T_event_count=Eqn_count
c
write(0,’(/,"Writing to file ...")’)
write(8,’("*/")’)
write(8,’{"Model : ",a’}) Uname
write(8,’("/")’)
write(8,’(a’) Description(1:Length(Description))
write(8,’("Propagation Equations")’)
write(8,’("------------------------")’)
do 1700,j=1,P_egn_count-1
write(8,’(/,a’) Egnarray(j)(1:Length(Egnarray(j)))
1700 continue
write(8,’(/,"Initial Event Statements")’)
write(8,’("------------------------")’)
do 1702,j=P_egn_count,I_event_count-1
write(8,’(/,a’) Egnarray(j)(1:Length(Egnarray(j)))
1702 continue
write(8,’(/,"Terminal Event Statements")’)
write(8,’("------------------------")’)
do 1704,j=I_event_count,T_event_count-1
write(8,’(/,a’) Egnarray(j)(1:Length(Egnarray(j)))
1704 continue
write(8,’("*/")’)
write(8,’("*/")’)
do 1720,j=1,Eqn_count-1
write(8,’(/,a’) Prolarray(j)(1:Length(Prolarray(j)))
1720 continue
write(8,’(*")’)
close(8)
write(0,’(/,"Finished Write !")’)
stop
9000 write(0,’("This model already exists !")’)
stop
end

subroutine propagation
c
Set up and declare variable types
c
character*5 Variable,Port*1
character*10 Uname
character*1 Var(6)
character*200 Eqn
character*200 Prolog_eqn
logical Flag
character Pos_flag,Neg_flag
integer i,i1,i2,ie,i9

c
Set up common blocks
c
common /Variables/ Var
common /prop_text/ Uname,Eqn,Prolog_eqn
common /prop_logl/ Flag
Initialise main flags

First_pos="y"
First_neg="y"
Flag=.false.
i8=1

Set up the first part of the Prolog equation
Prolog_eqn(1:i2)="rule("
i2=Length(Uname)+5
Prolog_eqn(5:i2)=Uname(1:Length(Uname))
i1=i2+1
i2=i1+3
Prolog_eqn(i1:i2)=",p,"
i1=i2+1

Get the resultant variable and port
call get_variable(Variable,Eqn,i8,Flag)
if(. not. Flag) goto 4100
i2=i1+Length(Variable)-1
Prolog_eqn(i1:i2)=Variable(1:Length(Variable))
i1=i2+1
i2=i1
Prolog_eqn(i1:i2)=","
i1=i2+1
call get-port(Port,Eqn,i8,Flag)
if(. not. Flag) goto 4100
i2=i1
Prolog_eqn(i1:i2)=Port
i1=i2+1
i2=i1+2
Prolog_eqn(i1:i2)=","
i1=i2+1
i2=i1+2
if(Eqn(i8:i9).ne."=f(" then
  Flag=.false.
goto 4100
i8=i9+1

Start the collection of the positive terms of the equation

Check if either the end of the equation or the start or the negative
items has been reached
if(Eqn(i8:i8).eq."".or.Eqn(i8:i8+1).eq."".or.Eqn(i8:i8).eq.""") goto 2

Check if there is a need to read a comma in the original equation
if((First_pos.eq."y".and.First_neg.eq."y") or.Eqn(i8:i8).eq."","") then
  Flag=.true.
  if(.not.(First_pos.eq."y".and.First_neg.eq."y")) then
    if(Eqn(i8:i8).eq."",") then
      Flag=.true.
      i8=i8+1
    else
      Flag=.false.
    endif
  else
    Flag=.false.
  endif
else
  Flag=.false.
endif
if(. not. Flag) goto 4100

Collect a positive term
call get_variable(Variable, Eqn, i8, Flag)
if(.not. Flag) goto 4100

call get_port(Port, Eqn, i8, Flag)
if(.not. Flag) goto 4100

The first item of the list does not need a comma preceding it

if(First_pos.eq."y") then
  i2=i1
  Prolog_eqn(i1:i2)="[
  i1=i2+1
  First_pos="n"
else
  i2=i1+1
  Prolog_eqn(i1:i2)=", ["
  i1=i2+1
endif

i2=i1+Length(Variable)-1
Prolog_eqn(i1:i2)=Variable(1:Length(Variable))
i1=i2+1
i2=i1
Prolog_eqn(i1:i2)=""

i1=i2+1
i2=i1
Prolog_eqn(i1:i2)=Port
i1=i2+1
i2=i1
Prolog_eqn(i1:i2)="]"
i1=i2+1

Go back and collect another positive term if possible
goto 1000

Close list of positive items and open list of negative items
2000 i2=i1+2
Prolog_eqn(i1:i2)=], ["
i1=i2+1

Start collecting negative items

First, check if there are any negative items
2200 if(Eqn(i8:i8).eq."\)") goto 4000

Read the comma and the negative sign preceding the term. Note that the
comma will not be present if there have been no positive terms and this
is the first negative term.
if((First_pos eq."n".and.First_neg eq."y").or.First_neg eq."n") then
  i8=i8+1
else
  Flag=.false.
  goto 4100
endif
endif

if(Eqn(i8:i8).eq."-") then
  i8=i8+1
  Flag=.true.
else
  Flag=.false.
  goto 4100
endif

Collect a variable and a port

call get_variable(Variable, Eqn, i8, Flag)
if(.not. Flag) goto 4100

call get_port(Port, Eqn, i8, Flag)
if(.not. Flag) goto 4100
The first negative term does not need a comma preceding it

```fortran
if(First_neg.eq."y") then
   i2=i1
   Prolog_eqn(il:i2)="[
   il=i2+1
   First_neg="n"
else
   i2=i1+1
   Prolog_eqn(il:i2)="],["n"
   il=i2+1
end if
```

Write the variable and term into the equation

```fortran
i2=i1+Length(Variable)-1
Prolog_eqn(il:i2)=Variable(1:Length(Variable))
il=i2+1
i2=i1
Prolog_eqn(il:i2)="",
il=i2+1
i2=i1
Prolog_eqn(il:i2)=Port
il=i2+1
i2=i1
Prolog_eqn(il:i2)="]"
il=i2+1
```

Go back and collect another negative item if possible

goto 2200

Close list of negative items

```fortran
i2=i1+2
Prolog_eqn(il:i2)="]")."
il=i2+1
```

If the translation failed blank out the whole equation.

```fortran
if(Flag) then
   do 4200, i=il, 200
      Prolog_eqn(i:i)=" "
   continue
4200
do 4300, i=1, i8
   write(0, '(a, $)') " 
4300   continue
   write(0, '(a)') " 
   write(0, '("Translation failed ! ")')
do 4400, i=1,200
   Prolog_eqn(i:i)=" 
4400   continue
endif
```

Return

End

```

```fortran
subroutine text-rules
```

```fortran
character*10 Uname
character*1 Var(6),Variable*5,Deviation*4,Text*20,Port
character*14 Var_name(6)
character*14 Rlname(6)
character*160 Eqn
```

c Start the counter for the Equation at 1
i8=1
c Dump the header of the rule into the Prolog equation
Prolog_eqn(1:5)="rule("
i2=5+Length(Uname)
Prolog_eqn(6:i2)=Uname(1:Length(Uname))
i1=i2+1
i2=i1
Prolog_eqn(i1:i2)=","
i1=i2+1
i2=i1+Length(Rlname(Rlnumber))-1
Prolog_eqn(i1:i2)=Rlname(Rlnumber)(1:Length(Rlname(Rlnumber)))
i1=i2+1
i2=i1+1
Prolog_eqn(i1:i2)=",["
i1=i2+1

c Collect the items on the left hand side of the equation
orig_i8=i8
call get_variable(Variable, Eqn, i8, Flag)
c If the flag is false go to the text collection section
if(.not. Flag) then
i8=orig_i8
goto 2000
endif
call get_port(Port, Eqn, i8, Flag)
c If there is no port then backspace and then go to the text collection section
if(.not. Flag) then
i8=orig_i8
goto 2000
endif
c Check for a space separating the port and the deviation. If a space does not exist then backspace over the two items that were read and go to the text collection section.
if(Eqn(i8:i8).eq. " ") then
i8=i8+1
else
i8=orig_i8
goto 2000
endif
call get_deviation(Deviation, Eqn, i8, Flag)
if(.not. Flag) goto 6000
Prolog_eqn(i1:i1+4+Length(Variable)+Length(Deviation))="[/Variable(1:Length(Variable)),/Port(1:1),/Deviation(1:Length(Deviation))]
i1=i1+5+Length(Variable)+Length(Deviation)
c Check if a comma follows. If it does then advance equation counter by one, insert a separating comma into the Prolog equation and go and collect another item for this side of the list.
if(Eqn(i8:i8).eq. ",") then
i8=i8+1
i2=i1
Prolog_eqn(i1:i2)=","
i1=i2+1
goto 1000
endif

c Check if there is a > sign at the end of this list. If there is then
go and collect the items on the r.h.s. If there is no > sign then fail.
if(Eqn(i8:i8).eq."">") then
  i8=i8+1
  goto 3000
else
  goto 6000
endif

c Text collection
2000 call get_text(Text,Eqn,i8,Flag)
if(.not.Flag) goto 6000
Prolog_eqn(i1:i1+11+Length(Text))="["Text(1:Length(Text))//",","

i1=i1+12+Length(Text)
c Check if a comma follows. If it does then advance equation counter by
one, insert separating comma in the Prolog equation and go and collect
another item.
if(Eqn(i8:i8).eq."","") then
  i8=i8+1
  i2=i1
  Prolog_eqn(i1:i2)=","
  i1=i2+1
  goto 1000
endif

c Check if there is a > sign. If there is, go and collect the items on the
r.h.s. If there is no > then fail.
if(Eqn(i8:i8).eq."">") then
  i8=i8+1
  goto 3000
else
  goto 6000
endif

c Close l.h.s. list and open r.h.s. list.
3000 i2=i1+2
Prolog_eqn(i1:i2)="]","["

i1=i2+1

c Start collecting the right hand side items.
3200 orig_i8=i8
call get_variable(Variable,Eqn,i8,Flag)
c If the flag is false go to the text collection section
if(.not.Flag) then
  i8=orig_i8
  goto 4000
endif
call get_port(Port,Eqn,i8,Flag)
c If there is no port then backspace and then go to the text collection
section
if(.not.Flag) then
  i8=orig_i8
  goto 4000
endif
c Check for a space separating the port and the deviation. If a space does
not exist then backspace over the two items that were read and go to the
text collection section.
if(Eqn(i8:i8).eq." ") then
  i8=i8+1
else
  i8=orig_i8
goto 4000
endif
call get_deviation(Deviation, Eqn, i8, Flag)
if(.not. Flag) goto 6000
Prolog_egn(i1: i1+4+Length(Variable)+Length(Deviation))="["//Variable(1:Length(Variable))//", "//Port(1:1)//", "//Deviation(1:Length(Deviation))//"]"
i1=i1+5+Length(Variable)+Length(Deviation)
c Check if a comma follows. If it does then advance the equation counter
c by one, insert a separating comma into the Prolog equation and
c collect another item for this side of the list.
if(Eqn(i8: i8).eq.,") then
  i8=i8+1
  i2=i1
  Prolog_eqn(i1: i2)=","
i1=i2+1
goto 3200
c Check if this is the end of the list. If it is then go and close the
c r.h.s. If it is not the end of the list then we have a failure.
if(i8.ge.Length(Eqn)) then
goto 5000
else
goto 6000
dedn

c Text collection
4000 call get_text(Text, Eqn, i8, Flag)
if(.not. Flag) goto 6000
Prolog_egn(i1: i1+11+Length(Text))="['//Text(1:Length(Text))//', ' ', ' ']
i1=i1+12+Length(Text)
c Check if a comma follows. If it does then advance the equation counter
c by one, insert a separating comma into the Prolog equation and go and
c collect another item.
if(Eqn(i8: i8).eq.,") then
  i8=i8+1
  i2=i1
  Prolog_eqn(i1: i2)=","
i1=i2+1
goto 3200
dend

c Check if this is the end of the list. If it is then go and close the
c r.h.s. If it is not the end of the list then we have a failure.
if(i8.ge.Length(Eqn)) then
goto 5000
else
goto 6000
dend

c Close the r.h.s. of the list.
5000 i2=i1+2
Prolog_eqn(i1: i2)="])."
i1=i2+1
c Blank out the rest
do 5500, i=i1, 200
Prolog_eqn(i1: i)=" "
5500 continue
Flag=.true.
goto 7000
c We have a failure!
6000 Flag=.false.
c Blank out the whole equation
do 6500, i=1, 200
Prolog

6500 continue
* Tab to the correct spot and point to the syntax error
do 6600, i=1, i8
   write(0, '(a, $)') " "
   write(0, '(a)') " "
   write(0, "(Translation failed !")')
7000 return

subroutine get_variable(Variable, Eqn, i8, Flag)

character*1 Var(6)
character Variable*5, Eqn*160
integer i, i8, i8_orig
logical Flag

common /Variables/ Var

c Initialise the variable to blank to avoid transferring ctl chars at the
c end of the string
Variable=" "
c
Save the value of the counter so that if this routine fails then the
c counter can be rest to its original value
i8_orig=i8
do 1000, i=1,6
   if(Eqn(i8:i8).eq. Var(1)) then
      Flag=.true.
      Variable(1:1)=Var(1)
      i=6
      i8=i8+1
   else
      Flag=.false.
      endif
1000 continue

Special treatment for the concentration variable
if(Variable(1:1).eq."x") then
   Variable(1:3)="[x, ">
   if(ichar(Eqn(i8:i8)).ge.49.and.ichar(Eqn(i8:i8)).le.57) then
      Variable(4:4)=Eqn(i8:i8)
      Variable(5:5)="]"
      Flag=.true.
      i8=i8+1
   else
      Flag=.false.
      endif
endif

Special treatment for the impurity variable
if(Variable(1:1).eq."i") then
Variable(1:3)="[i,"
Check that the impurity referred to is a lower case letter of the alphabet or that it is the generic dummy impurity
if(Eqn(i8:i8).eq."_".or.(ichar(Eqn(i8:i8)).ge.97.and.ichar(Eqn(i8:i8))<122)) then
  Variable(4:4)=Eqn(i8:i8)
  Variable(5:5)="]"
  Flag=.true.
  i8=i8+1
else
  Read the minus sign following the impurity term
  if(Eqn(i8:i8).eq."-" ) then
    i8=i8+1
    Flag=.true.
  else
    Flag=.false.
  endif
else
  Flag=.false.
endif
endif
If this goal has failed then reset the value of i8
if(.not.Flag) then
  i8=i8_orig
endif
returnend

subroutine get_port(Port, Eqn, i8, Flag)
character*1 Port
character Eqn*160
integer i,i8
logical Flag
if(ichar(Eqn(i8:i8)).ge.49.and.ichar(Eqn(i8:i8)).le.57) then
  Port=Eqn(i8:i8)
  Flag=.true.
  i8=i8+1
else
  Flag=.false.
endif
return
end

subroutine get_deviation(Deviation, Eqn, i8, Flag)
character Eqn*160, Deviation*4
integer i8
logical Flag
if(Eqn(i8:i8+1).eq."HI") then
  Deviation="high"
  i8=i8+2
  Flag=.true.
  goto 1000
endif
if(Eqn(i8:i8+1).eq."LO") then
  Deviation="low"
  i8=i8+2
endif
Flag=.true.
goto 1000
dpng 
if(Eqni(i8:i8+3).eq."NONE") then 
  Deviation="none"
  i8=i8+4
  Flag=.true.
goto 1000 
endif

There is no deviation!
Flag=.false.

1000 return
end

subroutine get_text(Text, Eqn, i8, Flag)
character Eqn*160, Text*20
integer i8, i9
logical Flag

 initialise Text to blanks so that no ctl chars at the end of the text 
are sent through to main 
Text=""
i9=i8+1
200 if(Eqn(i9:i9).eq.,") then 
  Text=Eqn(i8:i9-1)
  i8=i9
  goto 1000
endif
if(Eqn(i9:i9).eq.">") then 
  Text=Eqn(i8:i9-1)
  i8=i9
  goto 1000
endif
if(i9.ge.Length(Eqn)) then 
  Text=Eqn(i8:)
  i8=i9
  goto 1000
endif

Advance counter by one and check again for the end of the text 
i9=i9+1
1000 Flag=.true.
return
end

integer function Length(String)
character String*(*)
integer i
i=len(String)
1 if(String(i:i).ne." ") goto 2
i=i-1
if(i.eq.0) goto 2
goto 1
2 Length=i
return
end
APPENDIX B

Listing of "material_modeller" Program
program material_modeller

c  integer No_kw_cause, No_kw_effect, i
character*20 Filename*10, Key_cause(10), Key_effect(10,2)
character*20 Worda, First*3

c  Key_cause(1)="blockage"
Key_cause(2)="cavitation"
No_kw_cause=2
Key_effect(1,1)="escape"
Key_effect(1,2)=""
Key_effect(2,1)="t"
Key_effect(2,2)="high"
Key_effect(3,1)="t"
Key_effect(3,2)="low"
Key_effect(4,1)="p"
Key_effect(4,2)="high"
Key_effect(5,1)="p"
Key_effect(5,2)="low"
No_kw_effect=5

c  write(0,'(/, "Type in material name ... ", "$")')
read(0,*1 Filename
open(8, file=Filename, status="NEW", err=8000)
do 1300, i=1, No_kw_cause
First="yes"
  write(8, '"material_cause(", a, ", ", a, ", ", "$",) Filename(1:Length(Filename))
  Key_effect(i)(1:Length(Key_effect(i)))
write(0,'(/, "Type in all the material characteristics giving rise to " 
\cme)') Key_cause(i)(1:Length(Key_cause(i)))
write(0,'(" <CR> after each cause and "end" to stop ")")')
1200 read(0,*1 Worda
  if(Worda.ne."end" and Worda.ne."End" and Worda.ne."END") then
    if(First.ne."yes") then
      write(8, cursor "","$")
    endif
  endif
write(8, '(["", a, "", ]", "$") Worda(1:Length(Worda))
First="no"
goto 1200
endif
write(8, '(["", "")')
1300 continue

1500, i=1, No_kw_effect
First="yes"
if(Key_effect(i,1).eq."escape") then
write(8, '("material_effect("a", escape, "", "$") Filename(1:Length(Filename))
else
write(8, '("material_effect("a", ", a", ", a", ", "$") Filename(1:Length(Filename)), Key_effect(i,1)(1:Length(Key_effect(i,1))), Key_effect(i,2)(1:Length(Key_effect(i,2))))
endif
if(Key_effect(i,1).eq."escape") then
write(0,'(/, "Type in all the effects on the material of escape")'
else
write(0,'(/, "Type in all the effects on the material of ", a, ", a") 
\c Key_effect(i,1)(1:Length(Key_effect(i,1))), Key_effect(i,2)(1:Length(Key_effect(i,2))"),"
endif
write(0,'("<CR> after each effect and 'end' to stop")')
read(0,*) Worda
if(Worda.ne."end".and.Worda.ne."End".and.Worda.ne."END") then
  if(First.ne."yes") then
    write(8,'("",",")')
  endif
  write(8,'([",",","",","",","",","]",")') Worda(1:Length(Worda))
  First="no"
  goto 1400
endif
write(8,'("].")')
1500 continue
stop
8000 write(0,'(/,"Material file already exists !",/')
stop
end
APPENDIX C

Listing of "configurator" Program
program configurator

Unit(_) is the list of units in the configuration input by the user in the
read_units subroutine.

Model(_) is the unit model that Unit(_) refers to. This is also input in
read_units subroutine.

Link_unit(_,_) is a list of the upstream and downstream units of the links
in the configuration.

Link_port(_,_) is a list of the upstream and downstream ports of the links
in the configuration. The ports refer to the units of Link_unit(_, _).

Line_vessel(_,_) is a list of the source and drain vessels of all the lines
in the configuration.

Line_port(_, _) is a list of the source and drain ports of all the lines in
the configuration. The ports refer to the units of Line_vessel(_, _).

Vessel_list(_) is a list of all the unit models that are designated as
vessels.

Vessel_inlet_ports(x,y)
Vessel_outlet_ports(x,y) These are lists of the inlet and outlet ports of
the vessel in Vessel_list(x). In position (x, 1) is an integer giving the
number of ports that are listed.

J is the total number of units in the configuration.

L is the total number of links in the configuration.

M is the number of different unit models that have been referenced.

Z is the number of lines calculated to be in the configuration.

Proc_mat(x) is the name of the process material in process line x
List_proc_mats(_) is a list of all the different material models referenced
Imp_code(x) is the code for impurity x
Imp_name(x) is the name of impurity x
Number_imps is the number of impurities referenced

Eqn(x) is the list of equations describing the material of construction
Pipe_eqn(x) is the list of prolog translated equations of Eqn(x)
No_pipe_eqns is the number of equations defined

character*20 Filename*15, Inname*10, Date, User, Version*2
character*10 Unit(40), Model(40), Vessel_list(20), Line_vessel(20, 2)
character*10 Unit_list(40)
integer Link_port(2, 40), Line_port(20, 2)
integer Vessel_inlet_ports(20, 6), Vessel_outlet_ports(20, 6), No_of_vessels
integer Unit_inlet_ports(20, 6), Unit_outlet_ports(20, 6), No_of_units
character*10 Link_unit(2, 40)
integer J, L, M, Z, i5
character*10 Proc_mat(20), List_proc_mats(20)
integer No_of_mats
character Imp_code(20)*1, Imp_name(20)*30
integer Number_imps
character Eqn(20)*160, Pipe_eqn(20)*200
integer No_pipe_eqns

common /Units_text/ Unit, Model
common /Units numb/ J
common /Links_text/ Link_unit
common /Links numb/ L, Link_port
common /File_data/ Filename, Version, Date, User
common /List_models_text/ List_models
common /List_models numb/ M
common /Vessel_data_text/ Vessel_list
common /Vessel_data numb/ Vessel_inlet_ports, Vessel_outlet_ports, No_of_ve

```
common /Unit_data_text/ Unit_list
common /Unit_data_numb/ Unit_inlet_ports, Unit_outlet_ports, No_of_units
common /Lines_text/ Line_vessel
common /Lines_numb/ Line_port, Z
common /Proc_materials_text/ Proc_mat, List_proc_mats
common /Proc_materials_numb/ No_of_mats
common /Imps_text/ Imp_code, Imp_name
common /Imps_numb/ Number_imps
common /Mat_cons_text/ Egn, Pipe_eqn
common /Mat_cons_numb/ No_pipe_eqns

Read in the vessel and model datafiles

open(1, file="Vessels_file", status="OLD", err=8010)
open(2, file="In_line_units_file", status="OLD", err=8020)
rewind 1
rewind 2
c
i5 is the counter for the number of vessels or units read in
15=1
100 read(1,*,end=500) Vessel_list(i5), Vessel_inlet_ports(i5,1), Vessel_outlet_ports(i5,1)
   if(Vessel_inlet_ports(i5,1).gt.0) then
      read(1,*) (Vessel_inlet_ports(i5, j+1), j=1, Vessel_inlet_ports(i5,1))
   endif
   if(Vessel_outlet_ports(i5,1).gt.0) then
      read(1,*) (Vessel_outlet_ports(i5, j+1), j=1, Vessel_outlet_ports(i5,1))
   endif
   No_of_vessels=i5
   i5=i5+1
   goto 100
500 i5=1
550 read(2,*,end=1000) Unit_list(i5), Unit_inlet_ports(i5,1), Unit_outlet_ports(i5,1)
   if(Unit_inlet_ports(i5,1).gt.0) then
      read(2,*) (Unit_inlet_ports(i5, j+1), j=1, Unit_inlet_ports(i5,1))
   endif
   if(Unit_outlet_ports(i5,1).gt.0) then
      read(2,*) (Unit_outlet_ports(i5, j+1), j=1, Unit_outlet_ports(i5,1))
   endif
   No_of_units=i5
   i5=i5+1
   goto 550
c
1000 write(0,'("Configuration filename ? ... ",$,"")
   read(0,*) Inname
   Filename=Inname(1:Length(Inname))".conf"
   open(8, file=Filename, status="NEW", err=8000)
   write(0,'(" Version number ? ... ",$,"")
   read(0,*) Version
   write(0,'(" Date ? ... ",$,"")
   read(0,*) Date
   write(0,'(" User Id ? ... ",$,"")
   read(0,*) User
call read_units
call list_unit_models
call read_links
call list_lines
call process_materials
call impurities
call mats_of_construction
call output_configuration
stop
8000 write(0,'("File already exists !!","")
   goto 1000
8010 write(0,'(/,"Unable to read vessel datafile!",/)
stop
8020' write(0,'(/,"Unable to read unit datafile!",/)
stop
end

subroutine read_units
character*10 Unit(40), Model(40)
integer J
logical Check
common /Units_text/ Unit, Model
common /Units_num/ J

c write(0,'(/,"Type in the units, one at a time, with the corresponding mo
odel. Type")
write(0, "(END DATA to stop entering units.")
write(0,'(/, " Unit name Model name")
J=1
1000 write(0, '(i2, " ")') J
read(0, *) Unit(J), Model(J)
if(Unit(J). ne. "end". and. Unit(J). ne. "END") then
    call verify_model(Model(J), Check)
    if(. not. Check) then
        write(0, '("Error ! Model name not found in Unit or Vessel datafile
\cs")
        write(0, '("Please retype!")
        goto 1000
     endif
     J=J+1
     goto 1000
end if
Unit(J)= " 
J=J-1
return
end

subroutine read_links
integer Link_port(2,40)
character*10 Link_unit(2,40)
integer L
logical Link_check
common /Links_text/ Link_unit
common /Links_num/ L, Link_port

c write(0,'(/,"Type in, one at a time, the links between the units. Type")
write(0, "(END x x x to stop.")
write(0,'(/, " UPSTREAM DOWNSTREAM")
write(0,'(" UNIT PORT UNIT PORT")
L=1
1000 write(0, '(i2, " ")') L
read(0, *) Link_unit(1,L), Link_port(1,L), Link_unit(2,L), Link_port(2,L)
if(Link_unit(1,L). ne. "END". and. Link_unit(1,L). ne. "end") then
    call verify_link(Link_unit(1,L), Link_port(1,L), Link_unit(2,L), Link_port(2,L), Link_check)
    if(. not. Link_check) then
write(0, ' ("Retype link i")')
endif
L=L+1
goto 1000
endif
Link_unit(1, L)=" 
Link_unit(2, L)=" 
Link_port(1, L)=" 
Link_port(2, L)=" 
L=L-1
return
dend

\014
subroutine output_configuration
character*20 Filename*15, Version*2, Date, User
character*10 Unit(40), Model(40), List_models(40)
integer Link_port(2, 40), Line_port(20, 2), Z, No_of_mats
character*10 Link_unit(2, 40), Line_vessel(20, 2), Proc_mat(20), List_proc_mat
character*20 Imp_code(20)*1, Imp_name(20)*30
integer Number_imps
character*10 Egn(20)*160, Pipe_egn(20)*200
integer No_pipe_egns
c
common /Units_text/ Unit, Model
common /Units numb/ J
common /Links_text/ Link_unit
common /Links numb/ L, Link_port
common /File_data/ Filename, Version, Date, User
common /List_models_text/ List_models
common /List_models numb/ M
common /Lines_text/ Line_vessel
common /Lines numb/ Line_port, Z
common /Proc_materials_text/ Proc_mat, List_proc_mats
common /Proc_materials numb/ No_of_mats
common /Imps_text/ Imp_code, Imp_name
common /Imps numb/ Number_imps
common /Mat_cons_text/ Eqn, Pipe_eqn
common /Mat_cons numb/ No_pipe_eqns
c
write(8, ' (" /*", //")')
write(8, ' ("Filename = ", a20)') Filename
write(8, ' (" Version ", a2)') Version
write(8, ' (" Created on ", a20)') Date
write(8, ' (" by ", a20, //")') User
write(8, ' (" UNIT NAME MODEL NAME")')
do 1000, i=1, J
write(8, '(i2, " ", a10, " ", a10)') i, Unit(i), Model(i)
1000 continue
write(8, ' (" ")')
write(8, ' (" UPSTREAM DOWNSTREAM")')
write(8, ' (" UNIT PORT UNIT PORT")')
do 1500, i1=1, L
write(8, '(i2, ", a10, ", a10, ", a10, ", i2)') i, Link_unit(1, i)
\c, Link_port(1, i), Link_unit(2, i), Link_port(2, i)
1500 continue
write(8, ' (" ")')
write(8, ' ("Line SOURCE DRAIN PROCESS")')
write(8,'("No. VESSEL PORT VESSEL PORT MATERIAL")')
do 1600,i=1,Z
i,L cine_vessel(i,1),Line_port(i,1),Line_vessel(i,2),Line_port(i,2),Proc_mat(i)
1600 continue
write(8,'(/, "Impurities")')
write(8,'("Code Name")')
do 1700, i=1,Number_imps
    write(6,'("%a", "")') Imp_code(i), Imp_name(i)(1: Length(Imp_name(i)))
1700 continue
write(8,'(/, "Materials of Construction Terminal Events")')
do 1800, i=1,No_pipe_egns
    write(8,'("%a")') Egn(i)(1: Length(Egn(i)))
1800 continue
write(8,'(/, "*/", ")")
i1=Length(Filename)
write(8,'("conf_filename("", a, ").")') Filename(1:i1)
i1=Length(Version)
write(8,'("conf_version("", a, ").")') Version(1:i1)
i1=Length(Date)
write(8,'("conf_date("", a, ").")') Date(1:i1)
i1=Length(User)
write(8,'("conf_who("", a, ").")') User(1:i1)
write(8,'("/")')
write(8,'("unit_models([", ")")')
do 2000, i=1,M
    if(i.ne.1) write(8,'(", ")")')
i1=Length(List_models(i))
    write(8,'("%a")') List_models(i)(1:i1)
2000 continue
write(8,'("]. ")")')
doi=1,J
    i1=Length(Unit(i))
i2=Length(Model(i))
    write(8,'("key("", a, ", ", a, ").")') Unit(i)(1:i1), Model(i)(1:i2)
2200 continue
write(8,'("/")')
do 2400, i=1,Z
    i1=Length(Line_vessel(i,1))
i2=Length(Line_vessel(i,2))
    write(8,'("line(", a, ", ", a, ", ", i2, ", ", a, ", ", i2, ").")') i,Line_vessel(i,1:11),Line_vessel(i,1:13),Line_port(i,2)
2400 continue
write(8,'("/")')
do 3000, i=1,Z
    write(8,'("material_in_line("", i2, ", ", a, ").")') i,Proc_mat(i)(1:Length(Proc_mat(i)))
3000 continue
write(8,'("/")')
write(8,'("process_material_models([", "]")')
do 3500, i=1,No_of_mats
    if(i.ne.1) then
        write(8,'(", "]")')
    endif
write(8, '(a,$)') List_proc_mats(i)(1:Length(List_proc_mats(i)))
3500 continue
write(8, '(['),"")
write(8, '(/)')
do 3600, i=1, Number_imps
write(8, '("impurity("a,"."))') Imp_code(i)
3600 continue
write(8, '(/)')
do 3700, i=1, No_pipe_egns
write(8, '(a)') Pipe_eqn(i)(1:Length(Pipe_eqn(i)))
3700 continue
write(8, '(/)')
return
end

subroutine list_unit_models
c
character*10 Unit(40), Model(40), List_models(40), Flag*1
integer J, M, i, ii
common /Units_text/ Unit, Model
common /Units_numb/ J
common /List_models_text/ List_models
common /List_models_numb/ M
M=0
do 2000, i=1, J
Flag="n"
do 1000, il=1, M
if(Model(i).eq.List_models(il)) then
Flag="y"
i1=M
disable
1000 continue
if(Flag. eq. "n") then
M=M+1
List_models(M)=Model(i)
endif
2000 continue
return
end

subroutine list_lines
c
character*10 Unit(40), Model(40), Vessel_list(20), Model_of_unit
character*10 Link_unit(2,40), Line_vessel(20,2)
integer Link_port(2, 40), L, Vessel_inlet_ports(20,6), Vessel_outlet_ports(20)
c,6), Drain_link_no, No_of_vessels
integer Line_port(20,2), i, i1, i2, i3, Z
logical Model_check
c
common /Vessel_data_text/ Vessel_list
common /Vessel_data_numb/ Vessel_inlet_ports, Vessel_outlet_ports, No_of_vessels
common /Links_text/ Link_unit
common /Links_numb/ L, Link_port
common /Lines_text/ Line_vessel
common /Lines_numb/ Line_port, Z
Z=0
do 3000, i=1, L
  call model_from_unit(1, 1, 1, Model_of_unit, Model_check)
  do 2500, il=1, 20
    i3 is the number of outlet ports of the vessel
    do 2000, i2=2, i3+1
      if(Link_port(1, i). eq. Vessel_outlet_ports(il, i2)) then
        call follow_to_drain_vessel(i, Drain_link_no)
        Z=Z+1
        Line_vessel(Z, 1)=Link_unit(1, i)
        Line_vessel(Z, 2)=Link_unit(2, Drain_link_no)
        Line_port(Z, 1)=Link_port(1, i)
        Line_port(Z, 2)=Link_port(2, Drain_link_no)
      endif
    2000 continue
  endif
  2500 continue
3000 continue
return
end

subroutine follow_to_drain_vessel(First_link, Drain_link)
integer i, Up_link, Drain_link, First_link
character*10 Flag*3, Up_unit, Down_unit, Vessel_list(20), Link_unit(2, 40), Mod

cell_of_unit
integer L, Link_port(2, 40), Up_port, Down_port
logical Model_check

common /Vessel_data_text/ Vessel_list
common /Links_text/ Link_unit
common /Links_numb/ L, Link_port

Up_link=First_link
10 Up_unit=Link_unit(1, Up_link)
Up_port=Link_port(1, Up_link)
Down_unit=Link_unit(2, Up_link)
Down_port=Link_port(2, Up_link)
Flag="not"
do 1000, i=1, 20
  if(Vessel_list(i). eq. Model_of_unit) then
    Drain_link=Up_link
    i=20
    Flag="end"
  endif
1000 continue
if(Flag. ne. "end") then
  if(Drain_unit=Not a vessel) then
    call model_from_unit(Drain_unit, Model_of_unit, Model_check)
    if(Vessel_list(i). eq. Model_of_unit) then
      Drain_link=Up_link
      i=20
      Flag="end"
    endif
  endif
  do 2000, i=1, L
if(Link_unit(1, i).eq.Down_unit) then
Up_1ink=i
goto 10
endif
2000 continue
endif
return
end

subroutine model_from_unit(A_unit, A_model, Check)
character*10 Unit(40), Model(40), A_unit, A_model
integer i, J
logical Check
common /Units_text/ Unit, Model
common /Units_numb/ J
do 1000, i=1, J
  if(Unit(i).eq.A_unit) then
    A_model=Model(i)
    i=J
    Check=.true.
  endif
1000 continue
return
end

subroutine verify_link(Up_unit, Up_port, Down_unit, Down_port, Check)
character*10 Unit_list(40), Vessel_list(20), Up_unit, Down_unit
character*10 Up_model, Down_model
integer Vessel_inlet_ports(20,6), Vessel_outlet_ports(20,6), No_of_vessels
integer Unit_inlet_ports(20,6), Unit_outlet_ports(20,6), No_of_units
integer Up_port, Down_port
integer i, ii
logical Check, Up_unit_check, Up_port_check, Down_unit_check, Down_port_check
common /Vessel_data_text/ Vessel_list
common /Vessel_data_numb/ Vessel_inlet_ports, Vessel_outlet_ports, No_of_vessels
common /Unit-data_text/ Unit-list
common /Unit-data_numb/ Unit_inlet_ports, Unit_outlet_ports, No_of_units

Up_unit_check=.false.
Up_port_check=.false.
Down_unit_check=.false.
Down_port_check=.false.
Check=.false.
call model_from_unit(Up_unit, Up_model, Up_unit_check)
if(.not.Up_unit_check) goto 2200
Up_unit_check=.false.
do 1000, i=1, No_of_vessels
  if(Up_model.eq.Vessel_list(i)) then
    Up_unit_check=.true.
  endif
1000 continue

do 500, i1=2, Vessel_outlet_ports(i, 1)+1
   if(Up_port.eq.Vessel_outlet_ports(i, i1)) then
      Up_port_check=true.
      i1=Vessel_outlet_ports(i, 1)+1
   endif
500   continue
   i=No_of_vessels
endif
1000  continue
   if(.not. Up_unit_check) then
      do 2000, i=1, No_of_units
         if(Up_model.eq.Unit_list(i)) then
            Up-unit-check=true.
            do 1500, i1=2, Unit_outlet_ports(i, 1)+1
               if(Up_port.eq.Unit_outlet_ports(i, i1)) then
                  Up-port-check=true.
                  i1=Unit_outlet_ports(i, 1)+1
               endif
            1500   continue
            i=No_of_units
         endif
      2000   continue
   endif
2200  call model_from_unit(Down_unit, Down_model, Down_unit_check)
   if(.not. Down_unit_check) goto 5200
   Down_unit_check=false.
   do 4000, i=1, No_of_vessels
      if(Down_model.eq.Vessel_list(i)) then
         Down_unit_check=true.
         do 3500, i1=2, Vessel_inlet_ports(i, 1)+1
            if(Down_port.eq.Vessel_inlet_ports(i, i1)) then
               Down-port-check=true.
               i1=Vessel_inlet_ports(i, 1)+1
            endif
         3500   continue
         i=No_of_vessels
      endif
   4000   continue
   if(.not. Down_unit_check) then
      do 5000, i=1, No_of_units
         if(Down_model.eq.Unit_list(i)) then
            Down_unit_check=true.
            do 4500, i1=2, Unit_inlet_ports(i, 1)+1
               if(Down_port.eq.Unit_inlet_ports(i, i1)) then
                  Down_port_check=true.
                  i1=Unit_inlet_ports(i, 1)+1
               endif
            4500   continue
            i=No_of_units
         endif
      5000   continue
   endif
5200  if(Up_unit_check) then
      if(.not. Up_port_check) then
         write(0, 'Error ! Upstream unit is not known')
      endif
      if(Down_unit_check) then
         write(0, 'Error ! No such outlet port for model specified for ups 
stream unit')
      endif
e else
      write(0, 'Error ! Upstream unit is not known')
endif
if(Down_unit_check) then

if(.not.Down_port_check) then
  write(0, '("Error ! No such inlet port for model specified for down 
\cstream unit")')
endif
else
  write(0, '("Error ! Downstream unit is not known")')
endif
if(Up_unit_check. and. Up_port_check. and. Down_unit_check. and. Down_port_check) then
  Check=. true.
else
  Check= .false.
endif
return
end

 subroutine verify_model(Unit_model, Check)
character*10 Unit_model, Vessel_list(20), Unit_list(40)
integer Vessel_inlet_ports(20,6), Vessel_outlet_ports(20,6), No_of_vessels
integer Unit_inlet_ports(20,6), Unit_outlet_ports(20,6), No_of_units
integer i
logical Check
common /Vessel_data_text/ Vessel_list
common /Vessel-data-numb/ Vessel_inlet_ports, Vessel_outlet_ports, No_of_vessels
common /Unit-data-text/ Unit_list
common /Unit-data_numb/ Unit_inlet_ports, Unit_outlet_ports, No_of_units
Check=.false.
do 1000, i=1, No_of_units
  if(Unit_model. eq. Unit_list(i)) then
    Check=. true.
    i=No_of_units
  endif
1000 continue
if(. not. Check) then
  do 2000, i=1, No_of_vessels
    if(Unit_model. eq. Vessel_list(i)) then
      Check=. true.
      i=No_of_vessels
    endif
2000 continue
endif
return
end

 subroutine process_materials
character*10 Proc_mat(20), List_proc_mats(20), Line_vessel(20,2)
integer Z, Line_port(20,2), i, ii, No_of_mats
character Flag
common /Lines_text/ Line_vessel
common /Lines_numb/ Line_port, Z
common /Proc_materials_text/ Proc_mat, List_proc_mats
common /Proc_materials_numb/ No_of_mats

No_of_mats=0
write(0, "/, "Input of process materials")"
write(0,'(---------------------------')
do 2000,i=1,Z
c,"/)) i,Line_vessel(i,1)(1:Length(Line_vessel(i,1))),Line_port(i,1),Line_vessel(i,2)(1:Length(Line_vessel(i,2))),Line_port(i,2)
write(0,'("Process material? ",',$.')')
read(0,*) Proc_mat(i)
Flag="n"
do 1000,i1=1,No_of_mats
if(List_proo_mats(i1).eq.Proc_mat(i)) then
   Flag="y"
i1=No_of_mats
endif
1000 continue
if(Flag.eq."n") then
   No_of_mats=No_of_mats+1
   List_proc_mats(No_of_mats)=Proc_mat(i)
endif
2000 continue
write(0,'(/, "End of input of process materials",/))
return
end

subroutine impurities

character*20 Letter*3, Name, Imp_code(20)*1, Imp_name(20)*30
integer Number_imps

common /Imps_text/ Imp_code, Imp_name
common /Imps_numb/ Number_imps

write(0,'("Impurity Definition")')
write(0,'(-------------------")')
write(0,'(/, "Type in, for one impurity at a time, the letter code and nam')

c of each of")')
write(0,'("the impurities that are referenced in this configuration. Term')
cinate the")')
write(0,'("list with 'end list" ")')
write(0,'(/, "Letter Impurity")')
write(0,'(" code name")')
Number_imps=0
1000 read(0,*),Letter,Name
if(.not.(Letter.eq."End",or.Letter.eq."end",or.Letter.eq."END")) then
   if((Length(Letter).eq.1.and.ichar(Letter(1:1)).ge.97.and.ichar(Letter(1')
\c:1)).lt.122) then
      Number_imps=Number_imps+1
      Imp_code(Number_imps)=Letter
      Imp_name(Number_imps)=Name
   else
      write(0,'("Letter code incorrect! Retype!")')
      write(0,'("The letter code must be a lower case letter of the alph')
cabet")')
   endif
   goto 1000
endif
return
end

subroutine mats_of_construction


c Set up and initialise arrays

c
Var(1)="p"
Var(2)="q"
Var(3)="t"
Var(4)="1"
Var(5)="x"
Var(6)="i"
No_pipe_eqns=0
i8=1

c
write(0,'(//,"Materials of Construction")')
write(0,'("--------------------------")')

100 write(0,'("Define a materials of construction terminal event ? (y or n) &")')
read(0,* Reply)
if(Reply.eq."n") goto 6000
if(Reply.ne."y") goto 100
No_pipe_eqns=No_pipe_eqns+1
write(0,'("Type in the equation (enclosed in quotes) ...")')
read(0,* Eqn(No_pipe_eqns))
Pipe_egn(No_pipe_eqns)(1:18)="rule(pipe,effect,"
ml=19
i8=1

... 

"","")
c Read in the variable, port and deviation of the "cause"
call get_variable(Variable,Eqn(No_pipe_eqns),i8,Flag)
if(. not.Flag) goto 3000
call get_port(Port,Eqn(No_pipe_eqns),i8,Flag)
if(. not.Flag) goto 3000
c Read in the space after the variable and port
if(Eqn(No_pipe_eqns)(i8:i8).eq." ") then
i8=i8+1
else
Flag=.false.
goto 3000
endif
c Read in the ">" that signifies the end of the cause side of the equation
if(Eqn(No_pipe_eqns)(i8:i8).eq." ") then
i8=i8+1
else
Flag=.false.
goto 3000
endif
c Open up the list of consequences
m2=m1
Pipe_eqn(No_pipe_eqns)(m1:m2)="["/
\ct(1:1)/",="/Deviation(1:Length(Deviation))/"]",
m1=m2+1

c Get the next consequence
1000 orig_i8=i8
   call get_variable(Variable, Eqn(No_pipe_eqns), i8, Flag)
   if(.not.Flag) then
     i8=orig_i8
     goto 2000
   endif
   call get_port(Port, Eqn(No_pipe_eqns), i8, Flag)
   if(.not.Flag) then
     i8=orig_i8
     goto 2000
   endif
   c Read the space between port and deviation. If this space does not exist
   c then backspace over the "port" read and go to text collection
   if(Eqn(No_pipe_eqns)(i8:i8). eq. " ") then
     i8=i8+1
   else
     i8=orig_i8
     goto 2000
   endif
   call get_deviation(Deviation, Eqn(No_pipe_eqns), i8, Flag)
   if(.not.Flag) goto 3000
   c Drop the current consequence into the Pipe_eqn
   m2=m1+4+Length(Variable)+Length(Deviation)
   Pipe_eqn(No_pipe_eqns)(m1:m2)="["//Variable(1: Length(Variable))//", "//Deviation(1: Length(Deviation))//"]"
   m1=m2+1
   c Skip the text collection section
   goto 2500
2000 call get_text(Text, Eqn(No_pipe_eqns), i8, Flag)
   if(.not.Flag) goto 3000
   c Drop the current consequence into the Pipe_eqn
   m2=m1+11+Length(Text)
   Pipe_eqn(No_pipe_eqns)(m1:m2)="["//Text(1: Length(Text))//", ", "
   m1=m2+1
   c If a comma follows then another consequence is to be collected
2500 if(Eqn(No_pipe_eqns)(i8:i8). eq. ", ") then
   i8=i8+1
   c Drop a comma to separate the cons in the Pipe_eqn
   m2=m1
   Pipe_eqn(No_pipe_eqns)(m1:m2)=",";
   m1=m2+1
   goto 1000
   endif
   c Close the Pipe_eqn
   m2=m1+2
   Pipe_eqn(No_pipe_eqns)(m1:m2)="]")."
   m1=m2+1
   c Blank out the rest of the Pipe_eqn
   do 2800, i=m1, 200
     Pipe_eqn(No_pipe_eqns)(i:i)= " "
2800 continue
   c print*, Pipe_eqn(No_pipe_eqns)
   c Go and see if another equation is to be collected
   goto 100
   c We have a failure so point to the failure and reduce the count of
c equations by one
3000 do 4000 i=1,18
  write(0,'(a,$)') " "
4000 continue
write(0,'(a)') ""
write(0,>('Translation failed !')')
No_pipe_eqns=No_pipe_eqns-1
goto 100

C
6000 return
derend

\014

subroutine get_variable(Variable, Eqn, i8, Flag)
character*1 Var(6)
character Variable*5, Eqn*150
integer i, i8, i8_orig
logical Flag

c common /Variables/ Var

c Initialise the variable to blank to avoid transferring ctl chars at the
end of the string
Variable=" "

I Save the value of the counter so that if this routine fails then the
counter can be reset to its original value
i8_orig=i8

do 1000, i=1,6
  if(Eqn(i8:i8).eq.Var(i)) then
    Flag=.true.
    Variable(1:1)=Var(i)
    i=6
    i8=i8+1
  else
    Flag=.false.
  endif
1000 continue

c Special treatment for the concentration variable
if(Variable(1:1).eq."x") then
  Variable(1:3)="[x,"
  if(ichar(Eqn(i8:i8)).ge.49.and.ichar(Eqn(i8:i8)).le.57) then
    Variable(4:4)=Eqn(i8:i8)
    Variable(5:5)="]"
    Flag=.true.
    i8=i8+1
  else
    Flag=.false.
  endif
end if

c Special treatment for the impurity variable
if(Variable(1:1).eq."i") then
Variable(1:3)="[i,"

Check that the impurity referred to is a lower case letter of the alphabet or that it is the generic dummy impurity

if(Eqn(i8:i8).eq."_".or.ichar(Eqn(i8:i8)).ge.97.and.ichar(Eqn(i8:i8))<122)) then
  Variable(4:4)=Eqn(i8:i8)
  Variable(5:5)="]"
  Flag=.true.
  i8=i8+1

Read the minus sign following the impurity term
if(Eqn(i8:i8).eq."-") then
  i8=i8+1
  Flag=.true.
else
  Flag=.false.
endif
else
  Flag=.false.
endif
endif
c
If this goal has failed then reset the value of i8
if(.not.Flag) then
  i8=i8_orig
endif
return
end

subroutine get_port(Port, Eqn, i8, Flag)
character*1 Port
character Eqn*160
integer i,i8
logical Flag
C
if(ichar(Eqn(i8:i8)).ge.49.and.ichar(Eqn(i8:i8)).le.57) then Port=Eqn(i8:i8)
  Flag=.true.
  i8=i8+1
else
  Flag=.false.
endif
return
end

subroutine get_deviation(Deviation, Eqn, i8, Flag)
character Eqn*160, Deviation*4
integer i8
logical Flag
C
if(Eqn(i8:i8+1).eq."HI") then
  Deviation="high"
  i8=i8+2
  Flag=.true.
  goto 1000
endif
if(Eqn(i8:i8+1).eq."LO") then
  Deviation="low"
  i8=i8+2
Flag=.true.
goto 1000
endif
if(Egn(i8:i8+3).eq. 'NONE') then
  Deviation="none"
i8=i8+4
  Flag=.true.
goto 1000
endif

There is no deviation!
Flag=.false.
1000 return
end

subroutine get_text(Text, Egn, i8, Flag)
character Eqn*160, Text*20
integer i8, i9
logical Flag

 initialise Text to blanks so that no ctl_chars at the end of the text
Text=""
i9=i8+1
200 if(Eqn(i9:i9).eq. ',' ) then
     Text=Eqn(i8:i9-1)
i8=i9
     goto 1000
endif
if(Eqn(i9:i9).eq. ']' ) then
     Text=Eqn(i8:i9-1)
i8=i9
     goto 1000
endif
if(i9.ge.Length(Eqn)) then
     Text=Eqn(i8:)
i8=i9
     goto 1000
endif

 advance counter by one and check again for the end of the text
i9=i9+1
goto 200

End of text
1000 Flag=.true.
return
end
APPENDIX D

Listing of "identifier" Program
identifier

/* ------------------------------------------------------------------ /* identifier - Hazard Identification Program */
/* ------------------------------------------------------------------ */

NOTES:

1) This version has a working cause and effect identifier to search for effects through the unit models in the line.

2) The steps followed by the cause and effect identifier are as follows...

   i) Find all the causes and effects of the top event at the source vessel
   ii) Find all possible causes of the top event at the unit under consideration
   iii) Find all possible effects of the top event at the unit under consideration
   iv) Repeat (ii) and (iii) for every unit in the line
   v) Follow the top event through the propagation rules at the drain vessel and save all effects
   vi) Repeat (i), (ii), (iii), (iv) and (v) for all the lines
   vii) Conduct cause and effect searches on all the vessels

3) This system also employs a generic unit, a pipe, in each process line. This results in minimum duplication of output.

4) This version utilises the cause and effect searches on the vessels after the searches on the lines have been completed.

5) This version utilises process material models to check the validity of the causes and to check for any effects of the process deviation on the process material.

6) This version has a concentration change utility.

7) This version has a routine to deal with lines that are normally closed to flow.

8) The printing stage has been amended so as to change non-valid NONE variables to LOW. The only valid NONE variables are FLOW and LEVEL. All other variables have NONE deviations changed to LOW.

9) There is a utility available to handle impurities as a variable deviation.

10) All the source vessels are assumed to be sources for all deviations apart from escape and impurities.

*/

/* STANDARD CLAUSES -- member, append,prt, etc. */

member(X,[X;_]) :- /* standard member clause */
member(X,Y).

append([],K,K).               /* standard append clause */
append([X|K1],K2,[X|K3]) :-
    append(K1,K2,K3).

delete(_,[],[]).            /* standard delete clause */
delete(X,[X|L],M) :-
    !,
    delete(X,L,M).
delete(X,[Y|L1],[Y|L2]) :-
    delete(X,L1,L2).

islist([H;T]).               /* standard list test */
islist([]).

prt([]):-
    nl.
prt([H;T]) : -
    write(H), tab(1), prt(T).

fileprt([]): -
    nl.
fileprt([H;T]) : -
    put(039), write(H), put(039), tab(1), fileprt(T).

plist([]): -
    nl.
plist([H;T]) : -
    write(H), nl, plist(T).

user_output(Output_list) :-
    (   output_option(a)
        ;
        output_option(c)
    ),
    !,
    prt(Output_list).

user_output(Output_list) : -
    !.
user_write(Item) : -
    (   output_option(a)
        ;
        output_option(c)
    ),
    !,
    write(Item).

user_write(Item) : -
    !.

file_output(Output_list) :-
    (   output_option(b)
        ;
        output_option(c)
    ),
    !,
    outfile(Outfile),
    tell(Outfile),
    fileprt(Output_list),
tell(user).

file_output(Output_list) :-
    !.

file_plist([]). /* output list to file if opted for */

file_plist([H|T]) :-
    file_output(H),
    file_output([]),
    file_plist(T).

user_tab(Tab) :- /* tab on right output option */
    ( output_option(a);
      output_option(c)
    ),!
    tab(Tab).

user_tab(Tab) :-
    !.

file_tab(Tab) :-
    ( output_option(b);
      output_option(c)
    ),!
    tab(Tab).

file_tab(Tab) :-
    !.

last_of_list(X,[X]). /* obtain last element of list */

last_of_list(X,[_;Y]) :-
    last_of_list(X,Y).

/* ----------------------------------------------- */
/* CLAUSES FOR OPTION RECOGNITION AND MANIPULATION OF FLOWLINES */
/* ----------------------------------------------- */

deviations([none, high, low]). /* clauses for available options */

deviation_of(none, high).

deviation_of(low, high).

deviation_of(high, low).

variables([flow, pressure, temperature]).

/* List of valid variable deviation combinations */

variable_deviation(flow, none).

variable_deviation(flow, low).

variable_deviation(flow, high).

variable_deviation(pressure, low).

variable_deviation(pressure, high).

variable_deviation(temperature, low).

variable_deviation(temperature, high).

variable_deviation(escape, ' ').

variable_deviation([i, _], high). /* Impurity variable deviation */
vessel_variable_deviation(level, low).
vessel_variable_deviation(level, high).
vessel_variable_deviation(pressure, low).
vessel_variable_deviation(pressure, high).

/*
Clauses to convert variable names into their keys
*/
variable_key(flow, q) :- !.
variable_key(temperature, t) :- !.
variable_key(pressure, p) :- !.
variable_key(level, l) :- !.
variable_key(Anyother, Anyother) :- !.

/*
Key clause for the generic pipe unit. This is to allow for the fact that the automatic selection of the pipe unit would mean that the configuration file would not have a key for the pipe unit.
*/
key(pipe, pipe).

/*
Clauses to enable the recognition of certain keywords in the causes.
*/
keyword(blockage).
keyword(cavitation).

/* --------------------------------------------- */
/* CONTROLLING CLAUSE FOR EXPERT HAZARD IDENTIFICATION */
/* --------------------------------------------- */

/*
This is the clause controlling the different stages of the cause and effect search. At a later stage it may prove expedient to insert a deny clause in order to invoke this clause automatically.
*/
start :-
filename, !,
load_up_unit_models, !,
load_up_process_material_models, !,
output_configuration_info, !,
conduct_search.

/*
The conduct_search goal is used to separate the two stages of the search - the line search and the vessel search. The line searches are failed in order to proceed through all the lines. The second invocation of conduct_search is used to initiate the vessel searches at the source vessels. The third invocation of conduct_search is used to continue the search at the drain vessels. The fourth goal is the catchall.
*/
conduct_search :-
(line(Linenumber, S_vessel, S_port, D_vessel, D_port),
variable_deviation(Var, Dev),
variable_key(Var, Kvar),
file_output([line, Linenumber]),
file_output([Dev,Kvar]),
start_line(Linenumber,S_vessel,S_port,D_vessel,D_port,Var,Dev),
file_output([end_of_list])
),
fail.

conduct_search :-
{ line(Linenumber,S_vessel,S_port,D_vessel,D_port),
  assert(linenumber(O)),
  search_vessel_variables(S_vessel,S_port),
  retractall(linenumber(_))
},
fail.

conduct_search :-
line(Ln,S_vessel,S_port,D_vessel,D_port),
search_downstream_vessel(D_vessel,D_port),
fail.

conduct_search.

search_downstream_vessel(Vessel,Port) :-
line(_,Vessel,_,_,_),
!,
fail.

search_downstream_vessel(Vessel,Port) :-
assert(linenumber(O)),
search_vessel_variables(Vessel,Port),
retractall(linenumber(O)),
!,
fail.

/*
Start the line search by first checking the propagation steps at the source vessel. If there are any vessel deviations that could give rise to the current line deviation then this is to be developed. The search of the line can proceed after the source vessel check.
*/
start_line(Ln,S_vess,S_port,D_vess,D_port,Var,Dev) :-
!,
line_var_dev_assert(Ln,Var,Dev),
!,
variable_key(Var,Kvar),
!,
reverse_propagation_steps_at_source(S_vess,S_port,Kvar,Dev),
!,
cause_and_effect_search,
!.

/*
This clause checks for the existence of propagation rules that may give rise to the current deviation at the source vessel.
*/
reverse_propagation_steps_at_source(Vessel,Port,Var,Dev) :-
key(Vessel,Vess_model),
rule(Vess_model,p,[Var,Port],Proplist,Inproplist),
strip_prop_list_into_causes(Vessel,Port,Var,Dev,Proplist),
strip_inprop_list_into_causes(Vessel,Port,Var,Dev,Inproplist),
fail.

/*
This clause checks for the existence of propagation rules that will indicate
the effect that the current deviation will have on the other variables at the vessel.

reverse_propagation_steps_at_source(Vessel, Port, Var, Dev) :-
  key(Vessel, Vess_model),
  rule(Vess_model, p, [Outvar, Outport], Proplist, Inproplist),
  Outvar \= [x, _],
  (member([Var, Port], Inproplist),
   assert(effect(_, Vessel, [Var, Port, Dev], [[Outvar, Outport, Dev]])),
   save_effect_at_vessel(Vessel, Var, Dev, Port, Outvar, Dev, Outport)
),
  fail.

This clause checks for the existence of propagation rules that will indicate
the inverse effects that the current deviation will have on the other
variables at the vessel.

reverse_propagation_steps_at_source(Vessel, Port, Var, Dev) :-
  opposite_of(Dev, Oppdev),
  key(Vessel, Vess_model),
  rule(Vess_model, p, [Outvar, Outport], Proplist, Inproplist),
  Outvar \= [x, _],
  (member([Var, Port], Inproplist),
   assert(effect(_, Vessel, [Var, Port, Dev], [[Outvar, Outport, Oppdev]])),
   save_effect_at_vessel(Vessel, Var, Dev, Port, Outvar, Oppdev, Outport)
),
  fail.

Catchall clause for the case where there are no propagation rules referring to
the current deviation.

reverse_propagation_steps_at_source(Vessel, Port, Var, Dev).

Break up the list of proportional entries and flag them as causes.

strip_prop_list_into_causes(Vessel, Port, Var, Dev, [[Pvar, Pport]; Restlist]) :-
  assert(cause(_, Vessel, [Var, Port, Dev], [[Pvar, Pport, Dev]])),
  strip_prop_list_into_causes(Vessel, Port, Var, Dev, Restlist).

strip_prop_list_into_causes(Vessel, Port, Var, Dev, []).

Break up the list of inversely proportional entries and flag them as causes.

strip_inprop_list_into_causes(Vessel, Port, Var, Dev, [[Pvar, Pport]; Restlist]) :-
  opposite_of(Dev, Opposite_dev),
  assert(cause(_, Vessel, [Var, Port, Dev], [[Pvar, Pport, Opposite_dev]])),
  strip_inprop_list_into_causes(Vessel, Port, Var, Dev, Restlist).

strip_inprop_list_into_causes(Vessel, Port, Var, Dev, []).

Clause to control the search at the vessels. The vessel variables are
considered in turn. Once the vessel variables have been dealt with, the
vessels are searched for the concentration changes. This is done by the second
invocation of the search_vessel_variables clause. Note that an uninstantiated
port number is used for the calls to "reverse_propagation_steps_at_source"
and "source_vessel_search". This is so that all the ports of the vessel are considered for the causes and consequences of the vessel variable deviation.

```prolog
search_vessel_variables(S_vessel, S_port) :-
    vessel_variable_deviation(Var, Dev),
    variable_key(Var, Kvar),
    file_output([line, 0]),
    file_output([Dev, Kvar, S_vessel]),
    reverse_propagation_steps_at_source(S_vessel, _S_port, Kvar, Dev),
    source_vessel_search(Kvar, Dev, S_vessel, S_port), /* _S_port=S_port */
    file_output([end_of_list]),
    fail.
```

The vessels are first searched for the occurrence of high concentrations and then for the occurrence of low concentrations.

```prolog
search_vessel_variables(S_vessel, S_port) :-
    retractall(deviation(_)),
    assert(deviation(high)),
    file_output([line, 0]),
    file_output([high, x, S_vessel]),
    get_conc_changes(S_vessel),
    print_conc_changes(S_vessel),
    file_output([end_of_list]),
    retractall(deviation(_)),
    !,
    assert(deviation(low)),
    file_output([line, 0]),
    file_output([low, x, S_vessel]),
    get_conc_changes(S_vessel),
    print_conc_changes(S_vessel),
    file_output([end_of_list]),
    retractall(deviation(_)),
    !.
```

/* INPUT OF \* CASE ID, CONFIGURATION FILE AND UNIT & PROPERTY MODELS */
/* --------- \* PROGRAM VARIABLES \* --------- */

This stage takes in the configuration file name and calls up all the related unit model files. Note the "repeat" goal in the clause "filename". This will allow backtracking to resatisfy the selection of the output option.

```prolog
filename :-
    nl, nl,
    write('Type in the case identification name ... '),
    nl,
    get_io_filenames(Inname, Inputfile),!,
    consult(Inputfile), nl,
    nl, nl,
    write('Output options are ... '), nl, nl,
    write('a) Full continuous reporting to terminal'), nl,
    write('b) Brief report to file'), nl,
    write('c) Full report to terminal AND brief report to file'), nl, nl,
    repeat.
```
write('Option ? ... '),
    read(Option),
    nl
),
check_output_option(Option),
!,
assert(output_option(Option)).

/*
Append ".conf" to the study name to obtain the input filename and ".pro_out" to the study name to obtain the output filename.
*/
get_io_filenames(Name, Infile) :-
    name(Name, List),
    append(List, [046,099,111,110,102], Inlist),
    append(List, [046,112,114,111,095,111,117,116], Outlist),
    name(Infile, Inlist),
    name(Outfile, Outlist),
    assert(outfile(Outfile)).

check_output_option(a).
check_output_option(b).
check_output_option(c).
check_output_option(_) :-
    !,
write('Invalid Option !'), nl, fail.

/*
Load up all the models for the units referenced in the configuration file plus the model for the generic pipe unit.
*/
load_up_unit_models :-
    unit_models(J),
    append(J,[pipe],J1), !,
    load(J1).

load([H;T]) :-
    consult(H),
    load(T).
load([]).

/*
Load up all the models for the properties of the materials contained in the configuration file.
*/
load_up_process_material_models :-
    process_material_models(J),
    load(J).

/* ----------------------------------------------- */
/* OUTPUT CONFIGURATION INFORMATION AND SELECT LINE FOR INVESTIGATION */
/* ----------------------------------------------- */

/*
This stage prints out brief details of the configuration. The name of the configuration datafile together with the date on which it was written is printed out.
*/
output_configuration_info :-
    nl, nl,
    conf_filename(Name),
    conf_date(Date),
    conf_who(Who),
    write('Configuration file : '), write(Name), nl,
    write(' created on : '), write(Date), nl,
    write(' by : '), write(Who), nl.

/* PROPERTIES SELECTION AND GUIDEWORD FOR SUBSEQUENT INVESTIGATION */

This stage allows the assertion of variable and deviation for the cause
and effect search.
The "variable_key" goal translates from the variable word to the variable
symbol used in the search.

/* */

line_var_dev_assert(L, V, D) :-
    retractall(variable(_)),
    retractall(deviation(_)),
    retractall(linenumber(_)),
    variable_key(V, Kvar),
    assert(variable(Kvar)),
    assert(deviation(D)),
    assert(linenumber(L)).

/* SEARCH FOR CAUSES AND EFFECTS OF HIGH IMPURITY LEVELS */

This stage conducts a cause and effect search for high impurity levels
before the general cause and effect search of process deviations. Each
process line is checked, this includes the source and drain vessels.

/* */

cause_and_effect_search :-
    retractall(ref(_)),
    assert(ref(0)),
    linenumber(Ln),
    line(Ln, S_vessel, S_port, D_vessel, D_port),
    variable([i, _]),
    deviation(high),
    link(S_vessel, S_port, Next_unit, Next_port),
    check_line_is_open(Next_unit, Next_port, D_vessel, D_port),
    collect_line_impurities(S_vessel, S_port, Next_unit, Next_port, D_vessel, D_port).

collect_line_impurities(S_vessel, S_port, Next_unit, Next_port, D_vessel, D_port) :-
    impurity(I),
    search_next_impurity(I, S_vessel, S_port, Next_unit, Next_port, D_vessel, D_port),
    fail.

search_next_impurity(I, S_vessel, S_port, Next_unit, Next_port, D_vessel, D_port) :-
    impurity_cause_search(I, S_vessel, S_port),
    ...
impurity_effect_search(I, S_vessel, S_port),
impurity_print(I, S_vessel, S_port),
impurity_cause_search(I, pipe, 1),
impurity_effect_search(I, pipe, 1),
impurity_print(I, pipe, 1),
impurity_search_on_line(I, Next_unit, Next_port, D_vessel, D_port),!

/*
* Catchall for when the impurities are exhausted
*/
collect_line_impurities(S_vessel, S_port, Next_unit, Next_port, D_vessel, D_port) :-

impurity_search_on_line(I, D_vessel, D_port, D_vessel, D_port) :-
  impurity_cause_search(I, D_vessel, D_port),
  impurity_effect_search(I, D_vessel, D_port),
  impurity_print(I, D_vessel, D_port).

impurity_search_on_line(I, A_unit, A_port, D_vessel, D_port) :-
  impurity_cause_search(I, A_unit, A_port),
  impurity_effect_search(I, A_unit, A_port),
  impurity_print(I, A_unit, A_port),
  link(A_unit, _, Next_unit, Next_port),
  impurity_search_on_line(I, Next_unit, Next_port, D_vessel, D_port).

impurity_cause_search(I, Unit, _) :-
  key(Unit, Kunit),
  (  rule(Kunit, Ruletype, Causes, Events),
      Ruletype \= effect,
      member([i, I], _, high], Events),
      assert(cause(_, Unit, [[i, I], ' ', high], Causes))
  ),
  fail.

impurity_cause_search(I, Unit, _) :-
  key(Unit, Kunit),
  (  rule(Kunit, Ruletype, Events, Effects),
      member([i, I], _, high], Events),
      assert(effect(_, Unit, [[i, I], ' ', high], Effects)
  ),
  fail.

impurity_effect_search(I, Unit, _) :-
  !.

impurity_effect_search(I, Unit, _) :-
  key(Unit, Kunit),
  (  rule(Kunit, Ruletype, Events, Effects),
      member([i, I], _, high], Events),
      assert(effect(_, Unit, [[i, I], ' ', high], Effects)
  ),
  fail.

impurity_print(I, A_unit, _) :-
  user_output([[]]),
  linenumber(Ln),
  user_output(['Line ', Ln, ',', ' A_unit, Impurity ', 'I, high']),
  user_output(['--------------------------']),
  user_output([[]]),
  user_output(['Causes']),
  impurity_print_causes(I, A_unit),
  user_output([[]]),
  user_output(['Consequences']),
  impurity_print_effects(I, A_unit).

impurity_print_causes(I, Unit) :-

{ cause(R, Unit, [[i, I], ' ', high], Causes),
  retract(cause(R, Unit, [[i, I], ' ', high], Causes)),
  print_out_impurity_causes(Causes, Unit, I)
},
fail.

impurity_print_causes(I, Unit).

impurity_print_effects(I, Unit) :-
  { effect(R, Unit, [[i, I], ' ', high], Effects),
    retract(effect(R, Unit, [[i, I], ' ', high], Effects)),
    print_out_impurity_effects(Effects, Unit, I)
  },
fail.

impurity_print_effects(I, Unit).

print_out_impurity_causes([[Var, Port, Dev]|Tail], Unit, I) :-
  file_output([I, Unit, cause, Var, Port, Dev]),
  user_output([Var, Port, Dev]),
  print_out_impurity_causes(Tail, Unit, I).

print_out_impurity_effects([], Unit, I).

print_out_impurity_effects([[Var, Port, Dev]|Tail], Unit, I) :-
  file_output([I, Unit, effect, Var, Port, Dev]),
  user_output([Var, Port, Dev]),
  print_out_impurity_effects(Tail, Unit, I).

print_out_impurity_effects([], Unit, I).

/* ------------------------------------------------------------------ */
/* SEARCH FOR CAUSES AND EFFECTS OF USER SPECIFIED VARIABLE DEVIATION */
/* ------------------------------------------------------------------ */

This stage controls the mechanics of the search for causes and effects. The line is considered one unit at a time and for each unit the search for causes at the unit is invoked followed by the invocation of the search for effects of the particular cause.

/* */

/* Controlling clause for the cause and effect search of a variable deviation on a line. The process line is checked to see if it is open to flow. In the case of the line being closed the search is directed by the second invocation of cause_and_effect_search. The source vessel is first consulted followed by consultation of the generic pipe unit. The search is then directed along the process line. */

cause_and_effect_search :-
  retractall(ref(_)),
  assert(ref(0)),
  linenumber(Ln),
  line(Ln, S_vessel, S_port, D_vessel, D_port),
  variable(Var),
  deviation(Dev),
Clause to check that the process line is open. This consists of travelling down the process line consulting the propagation rules at each unit. There are two catchalls for this clause. The first is for the variable "escape" for which the line is assumed open and the second is for when the end of the line has been reached.

\[
\text{check\_line\_is\_open}(\text{Unit}, \text{Port}, \text{D\_vessel}, \text{D\_port}) :- /* Catchall for the variable "escape" */ \text{variable(escape)}. \\
\text{check\_line\_is\_open}(\text{D\_vessel}, \text{D\_port}, \text{D\_vessel}, \text{D\_port}). /* Catchall for end of line */ \\
\text{check\_line\_is\_open}(\text{Up\_unit}, \text{Up\_in\_port}, \text{D\_vessel}, \text{D\_port}) :- \text{link(Up\_unit, Up\_out\_port, Down\_unit, Down\_in\_port)}, \text{variable(Var), key(Up\_unit, Model)}, \text{rule(Model, p, [Var, Up\_out\_port], Pos\_inputs, Neg\_inputs)}, \text{member([Var, Up\_in\_port], Pos\_inputs)}, \text{check\_line\_is\_open(Down\_unit, Down\_in\_port, D\_vessel, D\_port)}. \\
\]

Clause to pick up all the saved effects at the source vessel

\[
\text{source\_vessel\_search}(\text{Var}, \text{Dev}, \text{S\_vessel}, \text{S\_port}) :- \\
\text{retract(saved\_effect(S\_vessel, Cause\_var, Cause\_dev, Cause\_port, Var, D\_ev, S\_port))}, \\
\text{assert(cause(_, S\_vessel, [Var, S\_port, Dev], [[Cause\_var, Cause\_port, Cause\_dev]]))}, \\
\text{fail}. \\
\]

Catchall clause to conduct cause and effect search on the vessel after all saved effects at the vessel have been picked up. Note that there is one automatic cause, that the vessel supplies the variable deviation.

\[
\text{source\_vessel\_search}(\text{Var}, \text{Dev}, \text{S\_vessel}, \text{S\_port}) :- \\
*/ Cut to prevent backtracking past this point */ \\
( \text{Var = escape ; /* Automatic cause for escape not set up */} \text{var(S\_port) ; /* Auto cause for vessel v'bles not set up */} \text{assert(cause(_, S\_vessel, [Var, S\_port, Dev], [[Var, S\_port, Dev]]))} ). \\
\]
Var = escape, /* Do not need a port number for escape */
> cause_search_on_unit(Var, Dev, S_vessel, _),
>
> cause_search_on_unit(Var, Dev, S_vessel, S_port)
>
> effect_search_on_unit(Var, Dev, S_vessel, S_port),
>
> ( Var = escape, /* Do not need a port number for escape */
>     print_causes_and_effects(Var, Dev, S_vessel, _),
>   )
>     print_causes_and_effects(Var, Dev, S_vessel, S_port)
>
> /* Pick up the next unit in the line and conduct cause and effect searches on the
unit */

> line_units_search(Ln, Up_unit, Up_port, Down_vess, Down_port, Var, Dev) :-
>     link(Up_unit, Up_port, Next_unit, Next_port),
>     Next_unit \= Down_vess,
>     cause_search_on_unit(Var, Dev, Next_unit, _),
>     effect_search_on_unit(Var, Dev, Next_unit, _),
>     print_causes_and_effects(Var, Dev, Next_unit, _),
>     link(Next_unit, Out_port, _, _), /* The outlet port is used for the next
stage of the line search */
>     line_units_search(Ln, Next_unit, Out_port, Down_vess, Down_port, Var, Dev).

/*
The case of the next unit in the line being the drain vessel. In this case
conduct cause and effect searches on the drain vessel and then step through
the propagation rules in order to save effects at the vessel.
*/

line_units_search(Ln, Up_unit, Up_port, Down_vess, Down_port, Var, Dev) :-
    link(Up_unit, Up_port, Down_vess, Down_port),
    ( Var = escape, /* Do not need a port number for escape */
        cause_search_on_unit(Var, Dev, Down_vess, _),
      )
    cause_search_on_unit(Var, Dev, Down_vess, Down_port)
    ),
    ( Var = escape ;
      assert(effect(_, Down_vess, [Var, Down_port, Dev], [[Var, Down_port, Dev]
      
]))
    effect_search_on_unit(Var, Dev, Down_vess, Down_port),
    (  
      
    )
Var = escape, /* Do no need a port number for escape */
print_causes_and_effects(Var, Dev, Down_vess, _),

print_causes_and_effects(Var, Dev, Down_vess, Down_port),
!
material_effect_search(Ln, Var, Dev, Mat),
!
print_causes_and_effects(Var, Dev, Mat, _no_port),
!
doPropagationSteps(Var, Dev, Down_vess, Down_port).

/*
Check if the current variable deviation will be transformed to another. If so,
print and save the effect and check for more propagation rules that affect
the current variable deviation.
*/
doPropagationSteps(Var, Dev, Down_vess, Down_port) :-
key(Down_vess, K_down_vess),
rule(K_down_vess, p, [Out_var, Out_port], Pos_inputs, Neg_inputs),
Out_var \= [x, _], /* No prop' rules for concn to be selected */
( member([Var, Down_port], Pos_inputs),
  assert(effect(_, Down_vess, [Var, Down_port, Dev], [Out_var, Out_port, De
  cv])),
  print_causes_and_effects(Var, Dev, Down_vess, Down_port),
  effect_search_on_unit(Out_var, Dev, Down_vess, Out_port),
  print_causes_and_effects(Out_var, Dev, Down_vess, Out_port),
  save_effect_at_vessel(Down_vess, Var, Dev, Down_port, Out_var, Dev, Out_p
  
cort)
),
fail.

/*
Check if the current variable deviation will be inversely transformed to
another. If so, print and save the effect and check for more propagation rules
that contain the current variable deviation.
*/
doPropagationSteps(Var, Dev, Down_vess, Down_port) :-
key(Down_vess, K_down_vess),
rule(K_down_vess, p, [Out_var, Out_port], Pos_inputs, Neg_inputs),
Out_var \= [x, _], /* No prop' rules for concn to be selected */
( member([Var, Down_port], Neg_inputs),
  opposite_of(New_dev, Dev),
  assert(effect(_, Down_vess, [Var, Down_port, Dev], [Out_var, Out_port, Ne
  ow_dev])),
  print_causes_and_effects(Var, Dev, Down_vess, Down_port),
  effect_search_on_unit(Out_var, New_dev, Down_vess, Out_port),
  print_causes_and_effects(Out_var, New_dev, Down_vess, Out_port),
  save_effect_at_vessel(Down_vess, Var, Dev, Down_port, Out_var, New_dev, Out_p
  
cort)
),
fail.

/*
Catchall for when there is no propagation of the current variable deviation.
*/
doPropagationSteps(Var, Dev, Down_vess, Down_port).
Check if this effect is already saved at the vessel. If so, stop and discard this additional saved effect. If not, then save the effect in the form of a "saved_effect" predicate.

save_effect_at_vessel(Vessel, In_var, In_dev, In_port, Out_var, Out_dev, Out_port) :-
    saved_effect(Vessel, In_var, In_dev, In_port, Out_var, Out_dev, Out_port), !.

save_effect_at_vessel(Vessel, In_var, In_dev, In_port, Out_var, Out_dev, Out_port) :-
    !,
    assert(saved_effect(Vessel, In_var, In_dev, In_port, Out_var, Out_dev, Out_port)).

This second invocation of cause-and-effect-search is to deal with the process lines where no flow is the normal state. In such lines a search is made for the causes and effects of HI flow (i.e. SOME flow when there should be NONE). Correspondingly, this cause-and-effect-search goal only deal with HI flow. The other process deviations are dealt with by the catchall because there is no action required.

cause-and-effect-search : -
    retractall(ref(_)),
    retractall(cause(_, _, _, _)), /* No causes/effects from previous */
    retractall(effect(_, _, _, _)), /* vessel searches on dev'ns other */
    assert(ref(0)), /* than q none are to be saved */
    linenum(Ln),
    line(Ln, S_vessel, S_port, D_vessel, D_port),
    variable(q), /* Note that ONLY flow is to be considered */
    deviation(high), /* Note that ONLY high is to be considered */
    !,
    link(S_vessel, S_port, Next_unit, Next_port),
    find_closed_units_in_line(Next_unit, Next_port, D_vessel, D_port).

cause_end_effect_search. /* Catchall for closed lines when Q HI is not the d
\eviation */

/* Find the closed units in the line and check for the causes of HI flow. Only
the closed units are considered because faults in ONLY those units will lead
to flow. With all the other units this is a normal condition. */

find_closed_units_in_line(D_vessel, D_port, D_vessel, D_port). /* Catchall for th
\ce end of the line */

find_closed_units_in_line(Unit, Port, D_vessel, D_port) :- /* This unit is OPEN */
    link(Unit, Out_port, Next_unit, Next_port),
    key(Unit, Model),
    rule(Model, p, [q, Out_port], Pos_inputs, Neg_inputs),
    member([q, Port], Pos_inputs),
    !,
    fold_closed_units_in_line(Next_unit, Next_port, D_vessel, D_port).

find_closed_units_in_line(Unit, Port, D_vessel, D_port) :- /* This unit is CLOSED */
link(Unit, Out_port, Next_unit, Next_port),

cause_search_on_unit(q, high, Unit, _Port),

effect_search_on_unit(q, high, Unit, _Port),

print_causes_and_effects(q, high, Unit, _Port),

find_closed_units_in_line(Next_unit, Next_port, D_vessel, D_port).

/* ------------------------------------------------------------------ /*
/* CAUSE SEARCH ON A PARTICULAR UNIT IN THE RELEVANT PROCESS LINE */
/* ------------------------------------------------------------------ */

This stage performs a search for causes of a specific top event and follows the "fault tree" right down to the base events at the unit. The "effect" ruletypes are excluded from being used in the cause search.

*/
cause_search_on_unit(Var, Dev, Unit, Port) :-
    key(Unit, Kunit),
    (rule(Kunit, Ruletype, Causes, Events),
     Ruletype \= effect,
     member([Var, Port, Dev], Events),
     assert(cause(_, Unit, [Var, Port, Dev], Causes)),
     develop_cause_at_unit(Unit, Causes)
    ),
    fail.

cause_search_on_unit(Var, Dev, Unit, Port) :-

develop_cause_at_unit(Unit, [[Var, Port, Dev]; Tail]) :-
    check_for_cause_keywords(Var, Port, Dev, Unit),
    cause_search_on_unit(Var, Dev, Unit, Port),
    develop_cause_at_unit(Unit; Tail).

develop_cause_at_unit(Unit, []) :-

/*
Note that the blank items reject any causes that are other process deviations.
*/
check_for_cause_keywords(Var,' ',' ',Unit) :-
    keyword(Var),
    linenumber(Ln),
    Ln \= 0, /* Ensure that the unit is not a vessel */
    material_in_line(Ln, Mat),
    material_cause(Mat, Var, Causes),
    assert(cause(_, Unit, [Var, ' ',' ',''],Caus es)).
/*
Catchall for cases where there is no keyword, unit is a vessel and where there are no material causes.
*/
check_for_cause_keywords(Var, Port, Dev, Unit).
/* ----------------------------------------------- */
/* SEARCH FOR THE EFFECT OF THE DEVIATION ON THE PROCESS MATERIAL */
/* ----------------------------------------------- */
material_effect_search(Ln, Var, Dev, Mat) :-
    material_in_line(Ln, Mat),
    material_effect(Mat, Var, Dev, Effects),
    assert(effect(_, Mat, [Var, ' ', Dev], Effects)).
material_effect_search(Ln, Var, Dev, Mat) :-
    material_in_line(Ln, Mat).
/* ----------------------------------------------- */
/* SEARCH FOR EFFECTS OF A PARTICULAR SCENARIO CAUSING THE "TOP EVENT"*/
/* ----------------------------------------------- */

This stage incorporates a search for the effects of a particular "scenario", i.e. the other effects that may result from the "cause tree" which will lead to the variable deviation being investigated. The search is followed through right to the "event statements" in each unit.
/*
Treat the search for the effects of escape as a special case - the effects are only checked for on the process material. Conduct this search for the effects on the process material only after the whole of the line has been searched for causes of escape.
*/
effect_search_on_unit(escape, ' ', Down_vess, Port) :-
    linenumber(Ln),
    line(Ln, S_vess, S_port, Down_vess, Port),
    material_effect_search(Ln, escape, ' ', Mat).

/* Catchall for when the current unit is not the last to be searched for causes. In this case, omit the search for effects of escape and go on. */
effect_search_on_unit(escape, ' ', Unit, Port) :-
    !.

/* Find an effect at the unit and develop that effect at the unit as far as possible. */
effect_search_on_unit(Var, Dev, Unit, Port) :-
    key(Unit, Kunit),
    ( rule(Kunit, Ruletype, Events, Effects),
      Ruletype = effect,
      member([Var, Port, Dev], Events),
      assert(effect(_, Unit, [Var, Port, Dev], Effects)),
      develop_effects_at_unit(Unit, Effects)
    ),
    fail.
/*
No effects to be found
*/
effect_search_on_unit(Var, Dev, Unit, Port) :- !.

/*
Develop further the first effect in the list of effects passed down and then
develop the tail of the list.
*/
develop_effects_at_unit(Unit, [[Var, Port, Dev]; Tail]) :-
effect_search_on_unit(Var, Dev, Unit, Port),
develop_effect_at_unit(Unit, Tail).

/*
End of the list of effects to be developed.
*/
develop_effects_at_unit(Unit, []).

/* ---------------------------------------------- */
/* PRINTING STAGE FOR ONE PARTICULAR SET OF CAUSES AND EFFECTS */
/* ---------------------------------------------- */

/*
This stage prints out the results of a cause and effect search on a
unit together with the results of this particular cause. The head goal
of this stage is called by the "cause_and_effect_cut" goal in the cause
search stage. This goal is repeatedly called for every successful cause
search. The clauses for "cause" and "effect" are retracted during this
print stage.
*/

/*
Clause controlling the printing out of all the causes and effects of a
variable deviation at a particular unit.
*/
print_causes_and_effects(Var, Dev, Unit, Port) :-
  linenumber(0), !,
  user_output([]),
  user_output([]),
  user_output(['Vessel search on ', Unit]),
  user_output(['------------------------']),
  user_output([]),
  user_output([Dev, Var]),
  user_output([]),
  user_output(['Causes']),
  user_output(['-----']),
  cause_structure_print(Var, Dev, Unit, Port),
  user_output([]),
  user_output(['Effects']),
  user_output(['-----']),
  effect_structure_print(Var, Dev, Unit, Port).  

print_causes_and_effects(Var, Dev, Unit, Port) :-
  linenumber(Ln),
  user_output([]),
cause_structure_print(Var, Dev, Unit, Port) :-
    cause(R, Unit, [Var, Port, Dev], Allcauses),
    retract(cause(R, Unit, [Var, Port, Dev], Allcauses)),
    print_out_causes(Allcauses, Unit, Port, Dev),
    fail.

cause_structure_print(Var, Dev, Unit, Port).

effect_structure_print(Var, Dev, Unit, Port) :-
    effect(R, Unit, [Var, Port, Dev], Effects),
    retract(effect(R, Unit, [Var, Port, Dev], Effects)),
    print_out_effects(Effects, Unit, Port, Dev),

effect_structure_print(Var, Dev, Unit, Port).
fail.

/*
 * Catchall for no more effects.
 */

effect_structure_print(Var, Dev, Unit, Port).

/*
 * Clause to structure the output of effects. Tab to the next stop, output the 
 * first of the list of effects, then go and expand the first of the list, then 
 * recall this clause with the tail of the list.
 */

print_out_effects([[Evar, Eport, Edev]; Tail], Unit, Port, Dev) :-
  check_valid_none_items(Evar, Edev, Checked_dev),
  file_output([Unit, port, Port, effect, Evar, Eport, Checked_dev]),
  user_output([port, Port, ' ', Evar, Eport, Checked_dev]),
  !
  effect_structure_print(Evar, Edev, Unit, Eport),
  !,
  print_out_effects(Tail, Unit, Port, Dev).

/*
 * Finish printing in the event of an empty list.
 */

print_out_effects([], Unit, Port, Dev).

/*
 * Clauses to check the validity of the items that have NONE deviations. The 
 * only NONE deviations that are allowed are with the variables FLOW and LEVEL. 
 * All other NONE deviations should read LOW. Deviations other than NONE are 
 * unaffected.
 */

check_valid_none_items(q, none, none).

check_valid_none_items(l, none, none).

check_valid_none_items(Var, none, low).

check_valid_none_items(Var, Anydev, Anydev).

/*----------------------------------------------------------------------*/
/* CLAUSES FOR COLLECTING CONCENTRATION CHANGES AT VESSELS */
/*----------------------------------------------------------------------*/

/*
 * Controlling clause for the search at a vessel. The different input lines 
 * are dealt with one at a time. The fail ensures that every line coming into 
 * the vessel is considered.
 */

get_conc_changes(Vessel) :-
  get_material_in_an_input_line(Vessel, Vport, Mat),
  ensure_mat_diff_to_that_in_vessel(Vessel, Vport, Mat),
  fail.

/*
 * Catchall for when all the incoming lines have been considered.
 */

get_conc_changes(Vessel).
If the material coming into the vessel is the same as that in one of the lines going out of the vessel then this particular line is discarded. The reason for this is that there will be no sensible concentration changes to collect for a material that is in actual fact just being recycled.

/*

ensure_mat_diff_to_that_in_vessel(Vessel, Vport, Mat) :-
  line(Ln,Vessel,Port,_,_),
  material_in_line(Ln,Mat),
  !,
  fail.

/*

The material coming into the vessel in this line is different to that in all of the lines leaving the vessel. Thus, go and and collect the causes of concentration change for this line.

/*

ensure_mat_diff_to_that_in_vessel(Vessel, Vport, Mat) :-
  drop_off_all_conc_propagations(Vessel, Vport, Mat),
  !,
  fail.

/*

Clause to enable the calling up of a material coming into a specific port on a particular vessel.

get_material_in_an_input_line(Vessel, Port, Material) :-
  line(Ln,_,_,Vessel,Port),
  material_in_line(Ln,Material).

/*

Clause to control the search for causes of concentration changes of the component coming into the specific part of the vessel.

drop_off_all_conc_propagations(Vessel, Vport, Mat) :-
  key(Vessel,Vessel_mod),
  rule(Vessel_mod,p,[[x,Vport],Portout],Pos_list,Neg_list),
  drop_off_positive_conc_propagations(Vessel,[[x,Vport],Portout],Pos_list),
  drop_off_negative_conc_propagations(Vessel,[[x,Vport],Portout],Neg_list).

/*

Clause to deal with the causes of concentration change that are themselves concentration changes of the same deviation.

drop_off_positive_conc_propagations(Vessel,[[x,Vport],Portout],[[[x,Nport],V_in\c_port],Restlist]) :-
  line(Ln,_,_,Vessel,V_in_port),
  material_in_line(Ln,Newmat),
  line(Lna,_,_,Vessel,Vport),
  material_in_line(Lna,Mat),
  deviation(Dev),
  assert(cause(_,Vessel,[[x,Mat],Portout,Dev],[[x,Newmat],V_in_port,Dev\c])),
  drop_off_positive_conc_propagations(Vessel,[[x,Vport],Portout],Restlist).

/*

Clause to deal with the causes of concentration change that are process deviations.

drop_off_positive_conc_propagations(Vessel,[[x,Vport],Portout],[[Var,V_in_port]\c!Restlist]) :-
  line(Lna,_,_,Vessel,Vport),
material_in_line(Lna,Mat),
deviation(Dev),
assert(cause(_,Vessel,[[x,Mat],Portout,Dev],[Var,V_in_port,Dev]),
drop_off_positive_conc_propagations(Vessel,[[x,Vport],Portout],Restlist)
/*
Catchall for the end of the list.
*/
drop_off_positive_conc_propagations(Vessel,[[x,Vport],Portout],[]).

/*
Clause to deal with the causes of concentration change that are themselves concentration changes but with the deviation of the opposite sign.
*/
drop_off_negative_conc_propagations(Vessel,[[x,Vport],Portout],[[x,Nport],V_in_c_port];Restlist) :-
   line(Ln,...,Vessel,V_in_port),
   material_in_line(Ln,New_mat),
   line(Lna,...,Vessel,Vport),
   material_in_line(Lna,Mat),
   deviation(Dev),
   opposite_of(Dev,Opp_dev),
   assert(cause(_,Vessel,[[x,Mat],Portout,Dev],[[x,Newmat],V_in_port,Opp_c_dev]),
drop_off_negative_conc_propagations(Vessel,[[x,Vport],Portout],Restlist).

/*
Clause to deal with the causes of concentration change that are process deviations of the opposite sign to the deviation to the concentration change.
*/
drop_off_negative_conc_propagations(Vessel,[[x,Vport],Portout],[[Var,V_in_port];Restlist]) :-
   line(Lna,...,Vessel,Vport),
   material_in_line(Lna,Mat),
   deviation(Dev),
   opposite_of(Dev,Opp_dev),
   assert(cause(_,Vessel,[[x,Mat],Portout,Dev],[Var,V_in_port,Opp_dev]),
drop_off_negative_conc_propagations(Vessel,[[x,Vport],Portout],Restlist).

/*
Catchall for the end of the list.
*/
drop_off_negative_conc_propagations(Vessel,[[x,Vport],Portout],[]).

/*
Controlling clause for the printing out of all the concentration changes that have been found for one particular vessel.
*/
print_conc_changes(Vessel) :-
   user_output([]),
   user_output(['Concentration change searches on vessel : ',Vessel]),
   user_output(['-----------------------------------------------']),
   user_output([]),
   print_next_conc_change(Vessel).

/*
Print the concentration changes of one component at a time. Select a component for which there is a concn change and then go and print the relevant causes. The fail ensures that all the components are printed out.
*/
print_next_conc_change(Vessel) :-
    cause(_, Vessel, [[x, Component], Port, Dev], Cause),
    user_output(['Causes of concentration of ', Component, ', ', Dev, ' at port ', Port]),
    print_the_one_component_conc_change(Vessel, Component),
    fail.

/* Catchall for no more components.
*/
print_next_conc_change(Vessel).

/* Print out all the causes of concentration change of the particular component that are themselves concentration changes.
*/
print_the_one_component_conc_change(Vessel, Comp) :-
    cause(_, Vessel, [[x, Comp], Port, Dev], [[x, Pcomp], Pport, Pdev]),
    retract(cause(_, Vessel, [[x, Comp], Port, Dev], [[x, Pcomp], Pport, Pdev])

\c),
    user_output(['x of ', Pcomp, ' ', Pdev, ' at port ', Pport]),
    file_output([Comp, Vessel, Port, cause, x, Pcomp, Pport, Pdev])
),
    fail.

/* Print out all the causes of concentration change of the particular component that are variable deviations.
*/
print_the_one_component_conc_change(Vessel, Comp) :-
    cause(_, Vessel, [[x, Comp], Port, Dev], [Cvar, Cport, Cdev]),
    retract(cause(_, Vessel, [[x, Comp], Port, Dev], [Cvar, Cport, Cdev])

\c),
    user_output([Cvar, Cport, Cdev]),
    file_output([Comp, Vessel, Port, cause, Cvar, ', ', Cport, Cdev])
),
    fail.

/* Catchall for no more causes of concentration change for the particular component.
*/
print_the_one_component_conc_change(Vessel, Comp).
APPENDIX E

Listing of "compacter" Program
Integer L_number, L, V, Vds, V_c_no(20,5), V_e_no(20,5)
Integer Lds, L_c_no(20,12), L_e_no(20,12)
Character*5 Line_devs(12,2), Vessel_devs(5,2)
Character*20 Fname*10, Infile, Outfile, Confile
Character*20 Words, Wordint*5
Character*20 Vessel*10, Vessel_devs(5,2)
Character*5 V_c_v(20,5,20)*25, V_c_p(20,5,20), V_c_d(20,5,20)
Character*5 V_c_u(20,10,20), V_c_v(20,10,20), V_c_d(20,10,20)
Character*5 L_c_u(20,12,100), L_c_v(20,12,100)
Character*5 L_e_u(20,12,100), L_e_v(20,12,100)
Character*5 L_imp_c_unit(20,20,120)*20, L_imp_c_v(20,20,120)*25, L_imp_c_d(20,20,120)
Character*5 L_imp_e_unit(20,20,120)*20, L_imp_e_v(20,20,120)*25, L_imp_e_d(20,20,120)
Character*10 Impurity_list(20)
Integer No_of_imps, L_imp_c_no(20,20), L_imp_e_no(20,20)
Common /Vessel_names/ Vessel
Common /No_of_vessels/ V
Common /Vessel_deviations/ Vessel_devs
Common /Vessel_deviations/ V
Common /No_vessel_devs/ Vds
Common /No_of_V_ces/ V_c_no, V_e_no
Common /V_cause_list/ V_c_v, V_c_p, V_c_d
Common /V_effect_list/ V_e_v, V_e_p, V_e_d
Common /No_of_lines/ L
Common /Line_deviations/ Line_devs
Common /No_line_devs/ Lds
Common /No_of_l_ces/ L_c_no, L_e_no
Common /L_cause_list/ L_c_u, L_c_port
Common /L_effect_list/ L_e_u, L_e_port
Common /L_effect_list/ L_e_v, L_e_p, L_e_d
Common /Vessel_conc_nos/ V_c_conc_no, No_of_comps
Common /Vessel_conc_txt/ V_c_conc_in_v, V_c_conc_in_c, V_c_conc_in_d, V_c_conc_out_p
Common /Component_names/ Component
Common /Impurity_neuses/ L_imp_c_unit, L_imp_c_v, L_imp_c_d
Common /Impurity_neuses/ L_imp_e_unit, L_imp_e_v, L_imp_e_d
Common /Impurities/ Impurity_list
Common /Impurity_neuses/ No_of_imps, L_imp_c_no, L_imp_e_no

Initialise all variables

V=0
do 130, i=1,20
   Vessel(i)= " 
do 120, i1=1,5
  V_c_no(i, il)=0
  V_e_no(i, il)=0
do 110, i2=1,20
  V_c_v(i, il, i2)=
  V_c_p(i, il, i2)=
  V_c_d(i, il, i2)=
  V_e_v(i, il, i2)=
  V_e_p(i, il, i2)=
  V_e_d(i, il, i2)=
110 continue
120 continue
130 continue

L=0
No_of_imps=0

do 160, i=1,20
  do 150, i1=1,12
    L_c_no(i, il)=0
    L_e_no(i, il)=0
    do 140, i2=1,100
      L_c_u(i, il, i2)=
      L_c_port(i, il, i2)=
      L_c_v(i, il, i2)=
      L_c_p(i, il, i2)=
      L_c_d(i, il, i2)=
      L_e_u(i, il, i2)=
      L_e_port(i, il, i2)=
      L_e_v(i, il, i2)=
      L_e_p(i, il, i2)=
      L_e_d(i, il, i2)=
    140 continue
  150 continue
140 continue

do 158, i1=1,20
  Impurity_list(il)=
  L_imp_c_no(i, il)=0
  L_imp_e_no(i, il)=0
  do 156, 12=1,40
    L_imp_c_unit(i, i1, i2)=
    L_imp_c_v(i, il, i2)=
    L_imp_c_p(i, il, i2)=
    L_imp_c_d(i, il, i2)=
    L_imp_e_unit(i, il, i2)=
    L_imp_e_v(i, il, i2)=
    L_imp_e_p(i, il, i2)=
    L_imp_e_d(i, il, i2)=
  156 continue
  158 continue
158 continue
160 continue

Line_devs(1,1)="q"
Line_devs(1,2)="none"
Line_devs(2,1)="q"
Line_devs(2,2)="low"
Line_devs(3,1)="q"
Line_devs(3,2)="high"
Line_devs(4,1)="p"
Line_devs(4,2)="low"
Line_devs(5,1)="p"
Line_devs(5,2)="high"
Line_devs(6,1)="t"
Line_devs(6,2)="low"
Line_devs(7,1)="t"
Line_devs(7,2)="high"
Line_devs(8,1)="escape"
Line_devs(8,2)=" "
Lds=8

Vessel_devs(1,1)="p"
Vessel_devs(1,2)="low"
Vessel_devs(2,1)="p"
Vessel_devs(2,2)="high"
Vessel_devs(3,1)="1"
Vessel_devs(3,2)="low"
Vessel_devs(4,1)="1"
Vessel_devs(4,2)="high"
Vds=4

Read in the case identification name

write(0,'("Type in the case identification name ... ",")')
read(0,*), Fname
Infile=Fname(1: Length(Fname))//".pro_out"
Outfile=Fname(1: Length(Fname))//".compact"
Confile=Fname(1: Length(Fname))//".conf"

Open input filename, check status and rewind

open(9, file=Infile, status="OLD", err=9010)
rewind 9

Open output file and check status

open(8, file=Outfile, status="NEW", err=9020)

Open configuration file, check status and rewind

open(7, file=Confile, status="OLD", err=9030)

Write header to output file

write(8,'("HAZOP file created from prolog run file : ",",a")') Infile
write(8,'(''''))
write(8,'("----------------------------------------------------------")')

Start running through the prolog output file

1050 read(9,*,end=1100) Worda,Wordint
read(Wordint,'(i5)') Lnumber
if(Lnumber.eq.0) then
  call vessel_routine
else
  call line_routine
endif

goto 1050
1100 call prune_line_doubled_items
call prune_vess_doubled_items
call cut_out_doubled_units
call unit_pruning
call separate_line_items
call print_out
stop

Error statements

Opening input file error
9010 write(0,'("Input file does not exist !")')
goto 9040
subroutine vessel_routine
   
   integer V, Vds, V_c_no(20, 5), V_e_no(20, 5)
   character*6 Vessel_devs(5, 2)
   character*20 Vessel(20)
   character*5 V_c_v(20, 5, 20)*25, V_c_p(20, 5, 20), V_c_d(20, 5, 20)
   character*5 V_e_v(20, 5, 20)*25, V_e_p(20, 5, 20), V_e_d(20, 5, 20)
   character Dev*6, Var*6, Vess*20, Flag*5
   integer i, Vess_no, Loc_b, No, Ne
   character*25 Worda, Wordb, Wordc, Wordd
   character*25 Unit*20, Port*2, Causeff*25
   
   integer V_c_conc_no(20, 2, 10), Hilo, NOf_comp, Comp_no
   character*25 V_c_conc_in_v(20, 2, 10, 10), V_c_conc_in_c(20, 2, 10, 10), Comp
   character*2 V_c_conc_in_p(20, 2, 10, 10), V_c_conc_out_p(20, 2, 10, 10)
   character Component(10)*25, V_c_conc_in_d(20, 2, 10, 10)*2
   
   common /Vessel_names/ Vessel
   common /No_of_vessels/ V
   common /Vessel_deviations/ Vessel_devs
   common /No_vessel_devs/ Vds
   common /No_of_V_ces/ V_c_no, V_e_no
   common /V_cause_list/ V_c_v, V_c_p, V_c_d
   common /V_effect_list/ V_e_v, V_e_p, V_e_d
   
   common /Vessel_conc_nos/ V_c_conc_no, NOf_comp
   common /Vessel_conc_txt/ V_c_conc_in_v, V_c_conc_in_c, V_c_conc_in_p, V_c_c
   \n
   read(9, *) Dev, Var, Vess
   Flag="false"
   do 1000, i=1, V
      if(Vessel(i).eq.Vess) then
         Vess_no=i
         i=V
         Flag="true"
      endif
   1000 continue
   if(Flag.eq."false") then
      Vess_no=V+1
      V=V+1
      Vessel(V)=Vess
   endif
   
   Check if this is a concentration cause. If it is then direct it to the
   latter part of this vessel_routine subroutine.
   
   if(Var.eq."x") goto 2000
   
   do 1200, i=1, Vds
      if(Vessel_devs(i, 1).eq.Var. and. Vessel_devs(i, 2).eq.Dev) then
         Loc_b=i
i=Vds
endif

1200 continue
Nc=V_c_no(Vess_no, Loc_b)
Ne=V_e_no(Vess_no, Loc_b)

1500 read(9, *) Worda
if(Worda ne "end_of_list") then
  backspace 9
  read(9, *) Unit, Worda, Port, Causeff, Wordb, Wordc, Wordd
  if(Causeff eq "cause") then
    Nc=Nc+1
    V_c_v(Vess_no, Loc_b, No)=Wordb
    V_c_p(Vess_no, Loc_b, No)=Wordc
    V_c_d(Vess_no, Loc_b, No)=Wordd
  else
    Ne=Ne+1
    V_e_v(Vess_no, Loc_b, Ne)=Wordb
    V_e_p(Vess_no, Loc_b, Ne)=Wordc
    V_e_d(Vess_no, Loc_b, Ne)=Wordd
  endif
  goto 1500
endif
V_c_no(Vess_no, Loc_b)=Nc
V_e_no(Vess_no, Loc_b)=Ne
c Now skip the concentration routine and go to the end
goto 3000

c Section dealing with the causes of concentration changes
c
2000 read(9, *) Worda
if(Worda ne "end_of_list") then
  backspace 9
  read(9, *) Comp, Unit, Port, Causeff, Worda, Wordb, Wordc, Wordd
  Flag="false"
  do 2500, i=1, No_of_comps
    if(Comp eq Component(i)) then
      Comp_no=i
      i=No_of_comps
      Flag="true"
    endif
  2500 continue
  if(Flag eq "false") then
    No_of_comps=No_of_comps+1
    Comp_no=No_of_comps
    Component(Comp_no)=Comp
  endif
  if(Dev eq "low") Hilo=1
  if(Dev eq "high") Hilo=2
  V_c_conc_no(Vess_no, Hilo, Comp_no)=V_c_conc_no(Vess_no, Hilo, Comp_no)+1
  V_c_conc_in_v(Vess_no, Hilo, Comp_no, V_c_conc_no(Vess_no, Hilo, Comp_no))=Worda
  V_c_conc_in_p(Vess_no, Hilo, Comp_no, V_c_conc_no(Vess_no, Hilo, Comp_no))=Wordb
  V_c_conc_in_d(Vess_no, Hilo, Comp_no, V_c_conc_no(Vess_no, Hilo, Comp_no))=Wordc
  V_c_conc_out_p(Vess_no, Hilo, Comp_no, V_c_conc_no(Vess_no, Hilo, Comp_no))=Wordd
  goto 2000
endif
3000 return
subroutine line_routine

integer L, Lds, L_c_no(20, 12), L_e_no(20, 12)
character*6 Line_devs(12, 2)
character*5 L_c_u(20, 12, 100), L_c_port(20, 12, 100)*2, L_c_v(20, 12, 100)*25
\c,L_c_p(20, 12, 100, L_c_d(20, 12, 100)
character*5 L_e_u(20, 12, 100)*20, L_e_port(20, 12, 100)*2, L_e_v(20, 12, 100)*25
\c,L_e_p(20, 12, 100), L_e_d(20, 12, 100)
character*25 Worda, Wordb, Wordd, Wordd, Wordint*5
character Unit*20, Port*2, Causeff*25
character Dev*8, Var*8
integer i, Ln, Loc_b, No, Ne

character*5 L_imp_c_unit(20, 20, 40)*20, L_imp_c_v(20, 20, 40)*25, L_imp_c_p(20
\c, 20, 40), L_imp_c_d(20, 20, 40)
character*5 L_imp_e_unit(20, 20, 40)*20, L_imp_e_v(20, 20, 40)*25, L_imp_e_p(20
\c, 20, 40), L_imp_e_d(20, 20, 40)
character*10 Imp, Impurity_list(20), Imp_flag, Impurities_read*4
integer No_of_imps, Imp_number, L_imp_c_no(20, 20), L_imp_e_no(20, 20), Nic, Nie

common /No_of_lines/ L
common /Line_deviations/ Line_devs
common /No_line_devs/ Lds
common /No_of_l_ces/ L_c_no, L_e_no
common /L_cause_list_1/ L_c_u, L_c_port
common /L_cause_list_2/ L_c_v, L_c_p, L_c_d
common /L_effect_list_1/ L_e_u, L_e_port
common /L_effect_list_2/ L_e_v, L_e_p, L_e_d

common /Impurity Causes/ L_imp_c_unit, L_imp_c_v, L_imp_c_p, L_imp_c_d
common /Impurity_effects/ L_imp_e_unit, L_imp_e_v, L_imp_e_p, L_imp_e_d
common /Impurities/ Impurity_list
common /Impurity_nos/ No_of_imps, L_imp_c_no, L_imp_e_no

backspace 9
read(9, *) Worda, Wordint
read(Wordint, '(15)') Ln
if(Ln gt. L) L=Ln
read(9, *) Dev, Var

if the variable deviation is an impurity one then go to latter section
if(Dev. eq. "high". and. Var(1: 5). eq. "[i, _") goto 2000
do 1000, i=1, Lds
if(Line_devs(i, 1). eq. Var. and. Line_devs(i, 2). eq. Dev) then
  Loc_b=i
1000 continue
Ne=L_e_no(Ln, Loc_b)
Nc=L_c_no(Ln, Loc_b)
1200 read(9, *) Worda
if(Worda ne. "end_of_list") then
  backspace 9
  read(9, *) Unit, Worda, Port, Causeff, Wordb, Wordc, Wordd
  if(Causeff.eq. "cause") then
    Nc=Nc+1
    L_c_u(Ln, Loc_b, Nc)=Unit
    L_c_port(Ln, Loc_b, Nc)=Port
    L_c_v(Ln, Loc_b, Nc)=Wordb
    L_c_p(Ln, Loc_b, Nc)=Wordc
    L_c_d(Ln, Loc_b, Nc)=Wordd
  else
...
Ne=Ne+1
L_e_u(Ln, Loc_b, Ne)=Unit
L_e_port(Ln, Loc_b, Ne)=Port
L_e_v(Ln, Loc_b, Ne)=Wordb
L_e_p(Ln, Loc_b, Ne)=Wordc
L_e_d(Ln, Loc_b, Ne)=Wordd
endif
goto 1200
endif
L_c_no(Ln, Loc_b)=Nc
L_e_no(Ln, Loc_b)=Ne
return

Impurity section
2000 Impurities_read="none"
read(9,*) Worda
if(Worda.ne."end_of_list") then
  Impurities_read="some"
  backspace 9
  read(9,*) Imp, Unit, Causeff, Wordb, Wordc, Wordd
  Imp_flag="not_found"
do 2100, i=1, No_of_imps
  if(Imp.eq.Impurity_list(i)) then
    Imp_number=i
    i=No_of_imps
    Imp_flag="found"
  endif
2100 continue
if(Imp_flag.ne."found") then
  No_of_imps=No_of_imps+1
  Impurity_list(No_of_imps)=Imp
  Imp_number=No_of_imps
endif
Nic=L_imp_c_no(Ln, Imp_number)
Nie=L_imp_e_no(Ln, Imp_number)
if(Causeff.eq."cause") then
  Nic=Nic+1
  L_imp_c_unit(Ln, Imp_number, Nic)=Unit
  L_imp_c_v(Ln, Imp_number, Nic)=Wordb
  L_imp_c_p(Ln, Imp_number, Nic)=Wordc
  L_imp_c_d(Ln, Imp_number, Nic)=Wordd
else
  Nie=Nie+1
  L_imp_e_unit(Ln, Imp_number, Nie)=Unit
  L_imp_e_v(Ln, Imp_number, Nie)=Wordb
  L_imp_e_p(Ln, Imp_number, Nie)=Wordc
  L_imp_e_d(Ln, Imp_number, Nie)=Wordd
endif
L_imp_c_no(Ln, Imp_number)=Nic
L_imp_e_no(Ln, Imp_number)=Nie
goto 2010
endif
return
end

subroutine Prune_line_doubled_items
character*5 L_c_u(20,12,100)*20, L_c_port(20,12,100)*2, L_c_v(20,12,100)*25
&, L_c_p(20,12,100), L_c_d(20,12,100)
character*5 L_e_u(20,12,100)*20, L_e_port(20,12,100)*2, L_e_v(20,12,100)*25
&, L_e_p(20,12,100), L_e_d(20,12,100)
integer L_c_no(20,12), L_e_no(20,12), i, 11, i2, i3, i4

common /No_of_1_ces/ L_c_no, L_e_no
common /L_cause_list_1/ L_c_u, L_c_port
common /L_cause_list_2/ L_c_v, L_c_p, L_c_d
common /L_effect_list_1/ L_e_u, L_e_port
common /L_effect_list_2/ L_e_v, L_e_p, L_e_d

C Step through all the process lines
do 3000, i=1, 20
C Step through all the process deviations
do 2000, ii=1, 12
C Step through the lists of causes
do 1000, i2=1, L_c_no(i, i1)-1
do 500, i3=12+1, L_c_no(i, i1)
if(i3 .le. L_c_no(i, i1)) then
  if(L_c_u(i, i1, i3) .eq. L_c_u(1, i1, i2) .and. L_c_v(1, i1, i3). eq. L_c_v(i, i1, i2) .and. L_c_p(i, i1, i3). eq. L_c_p(i, i1, i2) .and. L_c_d(i, i1, i3). eq. L_c_d(i, i1, i2)) then
    do 300, i4=i3, L_c_no(i, 11)-1
    L_c_u(i, i1, i4)=L_c_u(i, i1, i4+1)
    L_c_port(i, i1, i4)=L_c.port(i, i1, i4+1)
    L_c_v(i, i1, i4)=L_c.v(i, i1, i4+1)
    L_c_p(i, i1, i4)=L_c.p(i, i1, i4+1)
    L_c_d(i, i1, i4)=L_c.d(i, i1, i4+1)
  endif
enddo 300
L_c_no(i, ii)=L_c_no(i, ii)-1
i3=i3-1
endif
enddo 500
enddo 1000

C Step through the lists of effects
C
do 1900, i2=1, L_e_no(i, i1)-1
do 1500, i3=12+1, L_e_no(i, i1)
if(i3 .le. L_e_no(i, i1)) then
  if(L_e_u(i, i1, i3) .eq. L_e_u(i, i1, i2) .and. L_e_v(i, i1, i3) .eq. L_e_v(i, i1, i2) .and. L_e_p(i, i1, i3) .eq. L_e_p(i, i1, i2) .and. L_e_d(i, i1, i3) .eq. L_e_d(i, i1, i2)) then
    do 1300, i4=i3, L_e_no(i, 11)-1
    L_e_u(i, i1, i4)=L_e_u(i, i1, i4+1)
    L_e_port(i, i1, i4)=L_e.port(i, i1, i4+1)
    L_e_v(i, i1, i4)=L_e.v(i, i1, i4+1)
    L_e_p(i, i1, i4)=L_e.p(i, i1, i4+1)
    L_e_d(i, i1, i4)=L_e.d(i, i1, i4+1)
  endif
enddo 1300
L_e_no(i, ii)=L_e_no(i, ii)-1
i3=i3-1
endif
enddo 1500
enddo 1900
enddo 3000
end

subroutine prune_vess_doubled_items
character*5 V_c_v(20,5,20)*25,V_c_p(20,5,20),V_c_d(20,5,20)
character*5 V_e_v(20,5,20)*25,V_e_p(20,5,20),V_e_d(20,5,20)
integer V_c_no(20,5), V_e_no(20,5), i, i1, i2, i3, i4

common /No_of_V_ces/ V_c_no, V_e_no
common /V_cause_list/ V_c_v, V_c_p, V_c_d
common /V_effect_list/ V_e_v, V_e_p, V_e_d

Step through all the vessels
do 3000, i=1,20

Step through all the deviations
do 2000, i1=1,5

Step through the lists of causes
do 1000, i2=1, V_c_no(i, i1)-1
   do 500, i3=i2+1, V_c_no(i, i1)
      if(i3. le. V_c_no(i, i1)) then
         if(V_c_v(i, i1, i3) .eq. V_c_v(i, i1, i2) .and. V_c_p(i, i1, i3) .eq. V_c_p(i, i1, i2) .and. V_c_d(i, i1, i3) .eq. V_c_d(i, i1, i2)) then
            do 300, i4=i3, V_c_no(i, i1)-1
               V_c_v(i, i1, i4)=V_c_v(i, i1, i4+1)
               V_c_p(i, i1, i4)=V_c_p(i, i1, i4+1)
               V_c_d(i, i1, i4)=V_c_d(i, i1, i4+1)
            300 continue
            V_c_no(i, i1)=V_c_no(i, i1)-i
            i3=i3-1
         endif
      500 continue
   1000 continue

Step through the lists of effects

do 1900, i2=1, V_e_no(i, i1)-1
   do 1500, i3=i2+1, V_e_no(i, i1)
      if(i3. le. V_e_no(i, i1)) then
         if(V_e_v(i, i1, i3) .eq. V_e_v(i, i1, i2) .and. V_e_p(i, i1, i3) .eq. V_e_p(i, i1, i2) .and. V_e_d(i, i1, i3) .eq. V_e_d(i, i1, i2)) then
            do 1300, i4=i3, V_e_no(i, i1)-1
               V_e_v(i, i1, i4)=V_e_v(i, i1, i4+1)
               V_e_p(i, i1, i4)=V_e_p(i, i1, i4+1)
               V_e_d(i, i1, i4)=V_e_d(i, i1, i4+1)
            1300 continue
            V_e_no(i, i1)=V_e_no(i, i1)-i
            i3=i3-1
         endif
      1500 continue
   1900 continue

2000 continue
3000 continue

end subroutine cut_out_doubled_units

character*5 L_c_u(20,12,100)*20, L_c_port(20,12,100)*2, L_c_v(20,12,100)*25
\c, L_c_p(20,12,100), L_c_d(20,12,100)
character*5 L_e_u(20,12,100)*20, L_e_port(20,12,100)*2, L_e_v(20,12,100)*25
\c, L_e_p(20,12,100), L_e_d(20,12,100)
integer L_c_no(20,12), L_e_no(20,12), i, i1, i2, i3, j, k

common /No_of_L_ces/ L_c_no, L_e_no
common /L_cause_list/ L_c_u, L_c_port
do 3000, i=1,20
   do 2000, i1=1,12
      do 1000, i2=1,L_c_no(i,i1)
         j=0
         k=0
         j=k+1
         k=j
      100 if(k.ge.L_c_no(i,i1)) goto 180
         if(L_c_u(i,i1,k).eq.L_c_u(i,i1,k+1)) then
            k=k+1
            goto 150
         endif
      180 if(j.eq.k and k.lt.L_c_no(i,i1)) goto 100
         L_c_u(i,i1,j)=" "
         goto 200
      200 continue
      if(k.lt.L_c_no(i,i1)) goto 100
      continue
   1000 do 1990, i2=1,L_e_no(i,i1)
      j=0
      k=0
      j=k+1
      k=j
      if(k.ge.L_e_no(i,i1)) goto 1180
      if(L_e_u(i,i1,k).eq.L_e_u(i,i1,k+1)) then
         k=k+1
         goto 1150
      endif
      1180 if(j.eq.k and k.lt.L_e_no(i,i1)) goto 1100
         L_e_u(i,i1,j)=" "
         goto 1200
      1200 continue
      if(k.lt.L_e_no(i,i1)) goto 1100
      continue
   1990 continue
   2000 continue
   3000 continue
end

subroutine unit_pruning

integer L, Unitnumber
character*15 Unit(40), Model(40), Worda, Wordb, Wordint*6
character*15 A_unit, A_model, Model1, Model2
character*6 Line_devs(12,2)
integer Lds, L_c_no(20,12), L_e_no(20,12)
c character*5 L_c_u(20,12,100)*20, L_c_port(20,12,100)*2, L_c_v(20,12,100)*25
\c,L_c_p(20,12,100),L_c_d(20,12,100)
c character*5 L_e_u(20,12,100)*20, L_e_port(20,12,100)*2, L_e_v(20,12,100)*25
\c,L_e_p(20,12,100),L_e_d(20,12,100)
c common /No_of_lines/ L
common /Line deviations/ Line_devs
common /No_line_devs/ Lds
common /No_of_1 ces/ L_c_no, L_e_no
common /L-cause_list_1/ L_c_u, L_c_port
common /L-cause_list_2/ L_c_v, L_c_p, L_c_d
common /L_effect_list_1/ L_e_u, L_e_port
common /L_effect_list_2/ L_e_v, L_e_p, L_e_d
Read the unit and model names from the configuration file

1000 read(7, *, end=9000) Worda, Wordb
   if(Worda.ne."UNIT" and Wordb.ne."NAME") goto 1000

Located the point from which to start reading the units and models

1010 read(7, *, end=9000) Wordint, Worda, Wordb
   if(Worda.ne."DOWNSTREAM") then
      read(Wordint,'(i6)') Unitnumber
      Unit(Unitnumber)=Worda
      Model(Unitnumber)=Wordb
      goto 1010
   endif

End of the unit and model identification part of the configuration file

Add on the generic pipe unit

Unitnumber=Unitnumber+1
Unit(Unitnumber)="pipe"
Model(Unitnumber)="pipe"

Step through the lines

do 5000, i=1,20

Step through the line deviation variables

do 4000, i1=1,12

12=0
12=12+1
   if(12.gt.L_c_no(i1)) goto 4000

Get the next non-blank unit. This will be where the item in the unit array AND the item in the variable array are non-blank.
   if(L_c_u(i1,i2).eq." " or L_c_v(i1,i2).eq." ") goto 2500

i2 is the marker for the first non-blank unit
   A_unit=L_c_u(i1,i2)
call get_model_of_unit
   Modell=A_model
   i3=i2

2550 i3=i3+1
   if(i3.gt.L_c_no(i1)) goto 2500
   if(L_c_u(i1,i3).eq." ") goto 2550

i3 is the marker for the next non-blank unit
   A_unit=L_c_u(i1,i3)
call get_model_of_unit
   Modell2=A_model
   if(Modell.ne.Modell2) goto 2550

The models ARE the same

Temporarily increase arrays by one
   L_c_no(i1)=L_c_no(i1)+1

Shuffle all arrays down one
   do 2600, i4=L_c_no(i1)-1,12,-1
      L_c_u(i1,i4+1)=L_c_u(i1,i4)
      L_c_port(i1,i4+1)=L_c_port(i1,i4)
      L_c_v(i1,i4+1)=L_c_v(i1,i4)
2500 continue

c Insert the "same" unit in the space made by shuffling down
L_c_u(i, il, i3)=L_c_u(i, il, i3+1)
L_c_port(i, il, i3)=" 
L_c_v(i, il, i3)=" 
L_c_p(i, il, i3)=" 
L_c_d(i, il, i3)=" 

2600 do 2650, i4=i3+1

Start i4 off at i3+1 because the array has been shuffled down, the
previous i3 now being equal to i3+1

i4=i3+1

2650 i4=i4+1

If i4 has exceeded the set length of the array then go to the shuffling
up stage

if(i4.gt.L_c_no(i, il)) then
  goto 2660
endif

2700 do 2750, i5=i3+1, L_c_no(i, il)

L_c_u(i, il, i5)=L_c_u(i, il, i5+i4-i3-1)
L_c_port(i, il, i5)=L_c_port(i, il, i5+i4-i3-1)
L_c_v(i, il, i5)=L_c_v(i, il, i5+i4-i3-1)
L_c_p(i, il, i5)=L_c_p(i, il, i5+i4-i3-1)
L_c_d(i, il, i5)=L_c_d(i, il, i5+i4-i3-1)

2800 continue

Shuffle array up by as much space as the duplicated unit had taken up

2900, i5=i3+1, L_c_no(i, il)
L_c_u(i, il, i5)=L_c_u(i, il, i5+14-13-1)
L_c_port(i, il, i5)=L_c_port(i, il, i5+14-13-1)
L_c_v(i, il, i5)=L_c_v(i, il, i5+14-13-1)
L_c_p(i, il, i5)=L_c_p(i, il, i5+14-13-1)
L_c_d(i, il, i5)=L_c_d(i, il, i5+14-13-1)

3000 continue

3100 continue

3200 continue

3300 continue

3400 continue

3500 continue

3600 continue

3700 continue

3800 continue

3900 continue

4000 continue

4100 continue

4200 continue

4300 continue

4400 continue

4500 continue

4600 continue

4700 continue

4800 continue

4900 continue

5000 continue

5100 continue

5200 continue

5300 continue

5400 continue

5500 continue

5600 continue

5700 continue

5800 continue

5900 continue

6000 continue

6100 return

9000 write(0,'//("Error in reading configuration file ! ")')
stop
subroutine separate_line_items
character*5 L_c_u(20,12,100)*20, L_c_port(20,12,100)*2, L_c_v(20,12,100)*25 \c,L_c_p(20,12,100),L_c_d(20,12,100)
\c,L_e_u(20,12,100)*20, L_e_port(20,12,100)*2, L_e_v(20,12,100)*25 \c,L_e_p(20,12,100),L_e_d(20,12,100)
integer L_c_no(20,12),L_e_no(20,12),i,11,i2,i3,i4
common /No_of_l_ces/ L_c_no, L_e_no
common /L_cause_list_1/ L_c_u, L_c_port
common /L_cause_list_2/ L_c_v, L_e_p, L_c_d
common /L_cause_list_3/ L_e_u, L_e_port
common /L_cause_list_4/ L_e_v, L_e_p, L_e_d

Step through all the process lines
do 3000,i=1,20
Step through all the process deviations
do 2000,i1=1,12
Step through the lists of causes
100 do 1000,i2=1,L_c_no(i, i1)
If the CURRENT variable is non-blank and the following UNIT is non-blank then insert a blank line and shuffle the array down and then increase counter of causes by one.
if(L_c_v(i, i1,i2).ne." ".and.L_c_u(i, i1,i2+1).ne." ") then
    do 500,i3=L_c_no(i, i1),i2+1,-1 L_c_u(i, i1,i3+1)=L_c_u(i, i1,i3)
        L_c_port(i, i1,i3+1)=L_c_port(i, i1,i3)
        L_c_v(i, i1,i3+1)=L_c_v(i, i1,i3)
        L_e_p(i, i1,i3+1)=L_e_p(i, i1,i3)
        L_c_d(i, i1,i3+1)=L_c_d(i, i1,i3)
    500 continue
    L_c_u(i, i1,i2+1)=" "
    L_c_port(i, i1,i2+1)=" "
    L_c_v(i, i1,i2+1)=" "
    L_e_p(i, i1,i2+1)=" "
    L_c_d(i, i1,i2+1)=" "
    L_c_no(i, i1) =L_c_no(i, i1) +1
endif
1000 continue
if(L_c_no(i, i1).gt.1) then
    if(L_c_v(i, i1,L_c_no(i, i1)-1).ne." ".and.L_c_u(i, i1,L_c_no(i, i1)-1)\c).ne." ") goto 100
endif
Step through the lists of effects
1100 do 1900,i2=1,L_e_no(i, i1)
If the CURRENT variable is non-blank and the following UNIT is non-blank then insert a blank line and shuffle the array down and then increase counter of effects by one.
if(L_e_v(i, i1,i2).ne." ".and.L_e_u(i, i1,i2+1).ne." ") then
    do 1500,i3=L_e_no(i, i1),i2+1,-1 L_e_u(i, i1,i3+1)=L_e_u(i, i1,i3)
        L_e_port(i, i1,i3+1)=L_e_port(i, i1,i3)
        L_e_v(i, i1,i3+1)=L_e_v(i, i1,i3)
    1500 continue
L_e_p(i, i1, i3+1)=L_e_p(i, i1, i3)
L_e_d(i, i1, i3+1)=L_e_d(i, i1, i3)

continue
L_e_u(i, i1, i2+1)=" 
L_e_port(i, i1, i2+1)=" 
L_e_v(i, i1, i2+1)=" 
L_e_p(i, i1, i2+1)=" 
L_e_d(i, i1, i2+1)=" 
L_e_no(i, i1)=L_e_no(i, i1)+1
endif

1900 continue

if(L_e_no(i, i1).gt. 1) then
  if(L_e_v(i, i1, L_e_no(i, i1)-1).ne. " ".and. L_e_u(i, i1, L_e_no(i, i1)-1).ne. " ") goto 1100
endif

2000 continue

3000 continue

end subroutine printout

c integer V, Vds, V_c_no(20, 5), V_e_no(20, 5)
integer L, Lds, L_c_no(20, 12), L_e_no(20, 12)
integer Maxnumber

character*6 Line_devs(12, 2), Vessel_devs(5, 2)
character*3 C_flag, E_flag
character*20 Vessel(20)
character*5 V_c_v(20, 5, 20)*25, V_c_p(20, 5, 20), V_c_d(20, 5, 20)
character*5 V_e_v(20, 5, 20)*25, V_e_p(20, 5, 20), V_e_d(20, 5, 20)
character*5 L_c_u(20, 12, 100)*20, L_c_port(20, 12, 100)*2, L_c_v(20, 12, 100)*25
\c, L_c_p(20, 12, 100), L_c_d(20, 12, 100)
character*5 L_e_u(20, 12, 100)*20, L_e_port(20, 12, 100)*2, L_e_v(20, 12, 100)*25
\c, L_e_p(20, 12, 100), L_e_d(20, 12, 100)
c integer V_c_conc_no(20, 2, 10), Hilo, No_of_comps, Comp_no
character*25 V_c_conc_in_v(20, 2, 10, 10), V_c_conc_in_c(20, 2, 10, 10), Comp
character*2 V_c_conc_in_p(20, 2, 10, 10), V_c_conc_out_p(20, 2, 10, 10)
character Component(10)*25, V_c_conc_in_d(20, 2, 10, 10)*5

c character*5 L_imp_c_unit(20, 20, 40)*20, L_imp_c_v(20, 20, 40)*25, L_imp_c_p(20
\c, 20, 40), L_imp_c_d(20, 20, 40)
character*10 Impurity_list(20)
integer No_of_imps, L_imp_c_no(20, 20), L_imp_e_no(20, 20)
c integer k
character*4 Conc_items_flag, combine_strings*39
c
common /Vessel_names/ Vessel
common /No_of_vessels/ V
common /Vessel_deviations/ Vessel_devs
common /No_vessel_devs/ Vds
common /No_of_V_ces/ V_c_no, V_e_no
common /V_cause_list/ V_c_v, V_c_p, V_c_d
common /V_effect_list/ V_e_v, V_e_p, V_e_d
common /No_of_lines/ L
common /Line_deviations/ Line_devs
common /No/devs/ Lds
common /No_of_v_ces/ L_c_no, L_e_no
Print out vessel details

Step through all the process vessels
do 1000, i=1,V

Step through all the vessel process deviations
do 900, il=1, Vds

Check if there are any causes or effects for the current deviation. If there are none then print a message to that effect and skip the cause and effect printing stage.
if(V_c_no(i, il).eq.0. and. V_e_no(i, il).eq.0) then
  write(8, '(/, "No causes and effects of ", a, " ", a, " found on vessel ", a)') Vessel_devs(il, l), Vessel_devs(il, 2), Vessel(i)
else
  write(8, '("CTL-L")')
  write(8, '(/, " -----------------------------------------------------")')
  write(8, '(/, "Search on vessel : ", a)') Vessel(i)
  write(8, '(/, a, " ", a, /)') Vessel_devs(il, l), Vessel_devs(il, 2)
  write(8, '("CAUSES", t67, "CONSEQUENCES")')
  number=V_c_no(i, il)
  do 700, i2=1, Maxnumber
    if(V_e_no(i, il).ge. i2) then
      write(8, '(a, " ", a)') combine_strings(V_c_v(i, il, i2), V_c_p(i, il, i2), V_c_d(i, il, i2))
    endif
  end if
  if(V_e_no(i, il).ge. i2) then
    write(8, '(" ")')
  endif
  if(V_e_no(i, il).gt. Maxnumber) Maxnumber=V_e_no(i, il)
do 700 continue
endif
900 continue
do 980, il=1,2

Check if there any concentration change causes or effects to be printed. If there are none, then skip the printing section.
Conc_items_flag="none"
do 950, k=1, No_of_comps
if(V_c_conc_no(i, il, k).gt. 0) Conc_items_flag="some"
if(Conc_items_flag.eq."none") k=No_of_comps
950 continue
if(Conc_items_flag.eq."none") then
  c write(8,"(""No concentration ",","$)"")
  c if(il.eq.1) write(8,"(""low",","$)"")
  c if(il.eq.2) write(8,"(""high",","$)"")
  c write(8,"("" causes or effects found on vessel ","a")"") Vessel(i)
else
  write(8,"(/,"")
  960 continue
endif
write(8,"("///")"
write(8,"("-----------------------------

c----")")
write(8,"(/,""Concentration search on vessel : ","a,/,")"") Vessel(i)
do 970,i2=1, No_of_comps
  write(8,"(/,""Causes of concentration of ","a," ",","$)"") Compone
  do 980,i3=1, V_c_conc_no(i, il, i2)
    write(8,"(/,"")
    960 continue
  endif
write(8,"(/)"
970 continue
1000 continue

c Print out line details

c Step through all the lines
do 3000, i=1, L
  c Step through all the line deviations
do 1900, i1=1, Lds
write(8,"(""CTL-L")"
write(8,"(//)"
write(8,"("-----------------------------
c---")")

c Check if there are any causes or effects for the current deviation. If there are none then print a message to that effect and skip the cause and effect printing stage.
if(L_c_no(i, il).eq.0 .and. L_e_no(i, il).eq.0) then
  write(8,"(""No causes and effects of ","a," ","a," found on line ","i2")"") Line_devs(il, l), Line_devs(il, 2), i
else
  write(8,"(""Search on line ","i2")"
  write(8,"(/,""a," ","a")"") Line_devs(i1, l), Line_devs(i1, 2)
  write(8,"(/,""CAUSES",t67,"CONSEQUENCES",/)"
  Maxnumber=L_c_no(i, il)
  if(L_e_no(i, il).gt. Maxnumber) Maxnumber=L_e_no(i, il)
do 1700, i2=1, Maxnumber
    if(L_c_no(i, il).ge. i2) then
      write(8,"(/,"")
      endif
if(L_e_no(i,il).ge.i2) then
  write(8,'(a,"","a","a")') L_e_u(i,i1,i2),L_e_port(i,i1,i2),
  combine_strings(L_e_v(i,i1,i2),L_e_p(i,i1,i2),L_e_d(i,i1,i2))
else
  write(8,'("")')
endif
1700 continue
endif
1900 continue
c
  Print out details on impurities for each line
  write(8,'(//)')
write(8,'(-----------------------------------------------------------
-----------------------------------------------------------------
......)
write(8,'("Line ",i2," IMPURITIES ")') i
write(8,'(//)')
do 2800,i1=1,No_of_imps
  write(8,'("Impurity ",a," high")') Impurity_list(il)(1:Length(Impur
  city_list(il)))
  write(8,'("Causes",t67,"Consequences")')
  Maxnumber=L_imp_c_no(i,il)
  if(Maxnumber.lt.L_imp_e_no(i,il)) Maxnumber=L_imp_e_no(i,il)
  do 2400,i2=1,Maxnumber
    if(L_imp_c_no(i,il).ge.i2) then
      write(8,'(a,"","a")') L_imp_c_unit(i,il,i2),combine_strings(L
      c_imp_c_v(i,i1,i2),L_imp_c_p(i,i1,i2),L_imp_c_d(i,i1,i2))
    else
      write(8,'("")')
    endif
  end if
  if(L_imp_e_no(i,i1).ge.i2) then
    write(8,'(a,"",a,"","$")') L_imp_e_unit(i,il,i2),combine_strings(L
    c_imp_e_v(i,i1,i2),L_imp_e_p(i,i1,i2),L_imp_e_d(i,i1,i2))
  else
    write(8,'("")')
  endif
2400 continue
write(8,'(//)')
2800 continue
write(8,'(//)')
3000 continue
return
end

subroutine get_model_of_unit
character*15 Unit(40),Mode1(40),A_unit,A_model
integer i,J,Unitnumber
common /Units_text/ Unit,Model
common /Units numb/ Unitnumber
common /Unit_Model/ A_unit,A_model

do 1000,i=1,Unitnumber
  if(Unit(i).eq.A_unit) then
    A_model=Mode1(i)
  endif
1000 continue
return
character*39 function combine_strings(stra, strb, strc)
character stra(*), strb(*), strc(*)
character*2 Spaces
integer ia, ib, ic, lena, lenb, lenc

ia=len(stra)
ib=len(strb)
ic=len(strc)
lena=Length(stra)
lenb=Length(strb)
lenc=Length(strc)
Spaces=" 
if(lena.eq.0) then
   combine_strings=" 
else
   combine_strings=stra(1:lena)
endif
if(lenb.gt.0) then
   combine_strings=combine_strings(1:Length(combine_strings))//Spaces(1:2}//strb(1:lenb)
endif
if(lenc.gt.0) then
   combine_strings=combine_strings(1:Length(combine_strings))//Spaces(1:2}//strc(1:lenc)
endif
return
end

integer function Length(String)
character String(*)
integer i
i=len(String)
1 if(String(i:i).ne." ") goto 2
i=i-1
if(i.eq.0) goto 2
goto 1
2 Length=i
return
end
APPENDIX F

Configuration File for Water Separator Example
UNIT NAME   MODEL NAME
1 source1  hyd_source
2 pump2  pumpset_a
3 nrv3  nrv
4 meter4  fg_c_bypas
5 pipeline5  pipeline
6 cntrl_v16  cv_c_bypas
7 valve7  valve
8 settler8  settler
9 pump9  pumpset_a
10 divider10  divider
11 orifice11  orifice
12 valve12  valve
13 cntrl_v113  cv_c_bypas
14 exchang14  ht_exchg_a
15 sink15  sink
16 source16  source
17 sink17  sink
18 valve18  valve
19 sink19  sink

UPSTREAM   PORT   DOWNSTREAM   PORT
1 source1  1   pump2  1
2 pump2  2   nrv3  1
3 nrv3  2   meter4  1
4 meter4  2   pipeline5  1
5 pipeline5  2   cntrl_v16  1
6 cntrl_v16  2   valve7  1
7 valve7  2   settler8  1
8 settler8  2   pump9  1
9 pump9  2   divider10  1
10 divider10  3   orifice11  1
11 orifice11  2   valve12  1
12 valve12  2   settler8  3
13 divider10  2   cntrl_v113  1
14 cntrl_v113  2   exchang14  1
15 exchang14  2   sink15  1
16 source16  1   exchang14  3
17 exchang14  4   sink17  1
18 settler8  4   valve18  1
19 valve18  2   sink19  1

Line   SOURCE   DRAIN   PROCESS
No. VESSEL PORT VESSEL PORT MATERIAL
1 source 1 settler 1 olefin
2 settler 2 divider 1 olefin
3 divider 3 settler 3 olefin
4 divider 2 exchanger 1 olefin
5 exchanger 2 sink 1 olefin
6 source 1 exchanger 3 olefin
7 exchanger 4 sink 1 olefin
8 settler 4 sink 1 olefin

Impurities
--------------
Code Name
a water in olefin
b lower alkanes/alkene
c organic acids - olef
d unreacted monomer
e olefin in water

Materials of Construction Terminal Events
----------------------------------------
t1 LO>leak (major), brittle fracture
ib-1 HI>pressure high
to-1 HI>corrosion
*/

conf_filename('Water_sep. conf').
conf_version('1').
conf_date('4/8/86').
conf_who('jcp').

unit_models([hyd_source, pumpset_a, nrv, fq_c_bypas, pipeline, cv_c_bypas, valve, sett
clear, divider, orifice, ht_exchg_a, sink, source]).

key(source1, hyd_source).
key(pump2, pumpset_a).
key(nrv3, nrv).
key(meter4, fq_c_bypas).
key(pipeline5, pipeline).
key(cntrl_v16, cv_c_bypas).
key(valve7, valve).
key(settler8, settler).
key(pump9, pumpset_a).
key(divider10, divider).
key(orifice11, orifice).
key(valve12, valve).
key(cntrl_v13, cv_c_bypas).
key(exchanger4, ht_exchg_a).
key(sink15, sink).
key(source16, source).
key(sink17, sink).
key(valve18, valve).
key(sink19, sink).
link(source1, 1, pump2, 1).
link(pump2, 2, nrv3, 1).
link(nrv3, 2, meter4, 1).
link(meter4, 2, pipeline5, 1).
link(pipeline5, 2, ctrl_v16, 1).
link(ctrl_v16, 2, valve7, 1).
link(valve7, 2, settler8, 1).
link(settler8, 2, pump9, 1).
link(pump9, 2, divider10, 1).
link(divider10, 3, orifice11, 1).
link(orifice11, 2, valve12, 1).
link(valve12, 2, settler8, 3).
link(divider10, 2, ctrl_v113, 1).
link(ctrl_v113, 2, exchanger14, 1).
link(exchanger14, 2, sink15, 1).
link(source16, 1, exchanger14, 3).
link(exchanger14, 4, sink17, 1).
link(settler8, 4, valve18, 1).
link(valve18, 2, sink19, 1).

material_in_line( 1, olefin).
material_in_line( 2, olefin).
material_in_line( 3, olefin).
material_in_line( 4, olefin).
material_in_line( 5, olefin).
material_in_line( 6, olefin).
material_in_line( 7, olefin).
material_in_line( 8, water).

process_material_models([olefin, water]).

impurity(a).
impurity(b).
impurity(c).
impurity(d).
impurity(e).

rule(pipe, effect, [[t, l, low]], [['leak (major)', ' ', ' '], ['brittle fracture', ' ', ' '], ['corrosion', ' ']]).
rule(pipe, effect, [[[i, b], 1, high]], [['pressure high', ' ', ' ']]).
rule(pipe, effect, [[[i, c], 1, high]], [['corrosion', ' ', ' ']]).
APPENDIX G

Unit and Process Material Models for Water Separator Example
cv_c_bypas

/*
Model : cv_c_bypas

Control loop (control valve, controller, sensor) and bypass

Propagation Equations
---------------------
p2=f(p1)
q2=f(q1)
t2=f(t1)
x1-2=f(x1-1)
i_-2=f(i_-1)

Initial Event Statements
------------------------
loop fails closed, isolatn valve closed>q2 NONE, p1 HI
loop fails pt-olsd, isolatn vlve pt-olsd>q2 LO
loop fails open, bypass fails, bypass directed open>q2 HI
sensor fails closed, cntrlr fails closed, ctl-vl fails closed, set-pnt mvd closed>
\cloop fails closed
sensor fails pt-olsd, cntrlr fails pt-olsd, ctl-vl fails pt-olsd, set-pnt mvd pt-o
\olsd>loop fails pt-olsd
sensor fails open, cntrlr fails open, ctl-vl fails open, set-pnt mvd open>loop fai
\cls open
blockage>q2 LO
leak>escape

Terminal Event Statements
-------------------------
*/

rule(cv_c_bypas,p,[p,2],[[p,1]],[]).
rule(cv_c_bypas,p,[q,2],[[q,1]],[]).
rule(cv_c_bypas,p,[t,2],[[t,1]],[]).
rule(cv_c_bypas, p, [[x, 1], 2], [[x, 1], 1], []). 
rule(cv_c_bypas, p, [[_, 1], 2], [[_, 1], 1], []). 
rule(cv_c_bypas, functional, [[‘loop fails closed’, ‘’, ’’], [‘isolatn valve close \
cd’, ‘’], [q, 2, none], [p, 1, high]]). 
rule(cv_c_bypas, functional, [[‘loop fails pt-clsd’, ‘’, ’’], [‘isolatn vlve pt-cl \ncsd’, ‘’], [q, 2, low]]). 
rule(cv_c_bypas, functional, [[‘loop fails open’, ‘’, ‘’], [‘bypass fails’, ‘’, ‘’ \c], [‘bypass directed open’, ‘’, ‘’], [q, 2, high]]). 
rule(cv_c_bypas, functional, [[‘sensor fails closed’, ‘’, ‘’], [‘cntrlr fails clos \ced’, ‘’], [‘ctl-vl fails closed’, ‘’], [‘set-pnt mvd closed’, ‘’], \c[[‘loop fails closed’, ‘’]]). 
rule(cv_c_bypas, functional, [[‘sensor fails open’, ‘’], [‘cntrlr fails open’, \c’], [‘ctl-vl fails open’, ‘’], [‘set-pnt mvd open’, ‘’], [‘loop \cfails open’, ‘’], [q, 2, low]]). 
rule(cv_c_bypas, hydraulic, [[‘blockage’, ‘’], [q, 2, low]]). 
rule(cv_c_bypas, containment, [[‘leak’, ‘’], [‘escape’, ‘’]]).
/*
Model : divider

Binary divider - no flow distribution specified

Propagation Equations
---------------------
\[ p_2 = f(p_1) \]
\[ p_3 = f(p_1) \]
\[ q_2 = f(q_1) \]
\[ q_3 = f(q_1) \]
\[ t_2 = f(t_1) \]
\[ t_3 = f(t_1) \]
\[ x_{\ell-2} = f(x_{\ell-1}) \]
\[ x_{\ell-3} = f(x_{\ell-1}) \]
\[ i_\ell-2 = f(i_{\ell-1}) \]
\[ i_\ell-3 = f(i_{\ell-1}) \]

Initial Event Statements
------------------------
blockage > p_2 HI, p_3 HI
blockage, leak (major) > q_2 LO, q_3 LO, q_2 NONE, q_3 NONE
blockage - other leg > q_2 HI, q_3 HI
leak > escape

Terminal Event Statements
-------------------------
*/

rule(divider,p,[p,2],[[p,1],[[]].
rule(divider,p,[p,3],[[p,1],[[]].
rule(divider,p,[q,2],[[q,1],[[]].
rule(divider,p,[q,3],[[q,1],[[]].

divider
rule(divider, p, [t, 2], [[t, 1]], []). 
rule(divider, p, [t, 3], [[t, 1]], []). 
rule(divider, p, [[x, 1], 2], [[[x, 1], 1]], []). 
rule(divider, p, [[x, 1], 3], [[[x, 1], 1]], []). 
rule(divider, p, [[i, _], 2], [[[i, _], 1]], []). 
rule(divider, p, [[i, _], 3], [[[i, _], 1]], []). 
rule(divider, hydraulic, [['blockage', ' ', ' ']], [[[p, 2, high], [p, 3, high]]]). 
rule(divider, hydraulic, [['blockage', ' ', ' '], ['leak (major)', ' ', ' ']], [[[q, 2, low], [q, 3, low], [q, 2, none], [q, 3, none]]]. 
rule(divider, hydraulic, [['blockage - other leg', ' ', ' ']], [[[q, 2, high], [q, 3, high \c]]]. 
rule(divider, containment, [['leak', ' ', ' ']], [['escape', ' ', ' ']]).
/*
Model : fq_c_bypas

Flowmeter and bypass

Propagation Equations
-----------------------
p2=f(p1)
q2=f(q1)
t2=f(t1)
x1-2=f(x1-1)
i_2=f(i_1)

Initial Event Statements
------------------------
bypass fails, bypass directed open>q2 HI
leak>escape

Terminal Event Statements
-------------------------
q2 LO, q2 HI>inaccurate measurement
*/

rule(fq_c_bypas, p, [p, 2], [[p, 1]], []).
rule(fq_c_bypas, p, [q, 2], [[q, 1]], []).
rule(fq_c_bypas, p, [t, 2], [[t, 1]], []).
rule(fq_c_bypas, p, [[x, 1], 2], [[[x, 1], 1]], []).
rule(fq_c_bypas, p, [[i, _], 2], [[[i, _], 1]], []).
rule(fq_c_bypas, functional, [['bypass fails', ' ', ''], ['bypass directed open', ' \o', '']], [[q, 2, high]]).
rule(fq_c_bypas, containment, [['leak', ' ', ''], ['escape', ' ', '']], []).
rule(fq_c_bypas, effect, [[q, 2, low], [q, 2, high]], [['inaccurate measurement', ' ', ''], \c]).
ht_exchg_a

/*
Model : ht_exchg_a

Heat Exchanger - tube side higher press, shell to tube transfer, no phase chg

Propagation Equations
---------------------
p2=f(p1)
q2=f(q1)
t2=f(t1,t3,q3,-q1)
x1-2=f(x1-1)
i-2=f(i-1)
p4=f(p3)
q4=f(q3)
t4=f(t1,t3,q3,-q1)
x3-4=f(x3-3)
i-4=f(i-3)

Initial Event Statements
------------------------
tube fouling>t2 LO,t4 HI,q2 LO,p1 HI
shell leak (major)>q4 NONE,q4 LO,p4 LO
tube leak (major)>q2 LO,p2 LO,p4 HI
tube leak>id-4 HI
leak>escape

Terminal Event Statements
-------------------------
*/

rule(ht_exchg_a,p,[p,2],[[p,1]],[]).
rule(ht_exchg_a,p,[q,2],[[q,1]],[]).
rule(ht_exchg_a,p,[t,2],[[t,1],[t,3],[q,3]],[[q,1]]).
rule(ht_exchg_a, p, [[x, 1], 2], [[[x, 1], 1]], []).  
rule(ht_exchg_a, p, [[i, _], 2], [[[i, _], 1]], []).  
rule(ht_exchg_a, p, [p, 4], [[p, 3]], []).  
rule(ht_exchg_a, p, [q, 4], [[q, 3]], []).  
rule(ht_exchg_a, p, [t, 4], [[t, 1], [t, 3], [q, 3]], [[q, 1]]).  
rule(ht_exchg_a, p, [[x, 3], 4], [[[x, 3], 3]], []).  
rule(ht_exchg_a, p, [[i, _], 4], [[[i, _], 3]], []).  
rule(ht_exchg_a, functional, [["tube fouling", "", "]", [[t, 2, low], [t, 4, high], [q, \\c2, low], [p, 1, high]]).  
rule(ht_exchg_a, containment, [["shell leak (major)"],",", "]", [[q, 4, none], [q, 4, 1 \\cow], [p, 4, low]]).  
rule(ht_exchg_a, containment, [["tube leak (major)"],",", "]", [[q, 2, low], [p, 2, low \\o], [p, 4, high]]).  
rule(ht_exchg_a, containment, [["tube leak", ",", "]", [[i, d], 4, high]]).  
rule(ht_exchg_a, containment, [["leak", ",", "]", [[’escape’, ",", ""])].
**Model : hyd_source**

Hydrocarbon source - incorporates impurities

Propagation Equations
---------------------

**Initial Event Statements**
----------------------------

source empty>p1 LO, q1 NONE
impurity a in source>ia-1 HI
impurity b in source>ib-1 HI
impurity c in source>ic-1 HI

**Terminal Event Statements**
----------------------------

*/

rule(hyd_source, functional, [['source empty', '', '']], [[p, 1, low], [q, 1, none]]).
rule(hyd_source, impurity, [['impurity a in source', '', '']], [[i, a, 1, high]]).
rule(hyd_source, impurity, [['impurity b in source', '', '']], [[i, b, 1, high]]).
rule(hyd_source, impurity, [['impurity c in source', '', '']], [[i, c, 1, high]]).
Model : nrv

Non-return valve

Propagation Equations
---------------------
\[ P_2 = f(P_1) \]
\[ q_2 = f(q_1) \]
\[ t_2 = f(t_1) \]
\[ x_{1-2} = f(x_{1-1}) \]
\[ i_{-2} = f(i_{-1}) \]

Initial Event Statements
------------------------
valve stuck clsd \( \rightarrow \) \( q_2 \) NONE
valve fails pt. clsd \( \rightarrow \) \( q_2 \) LO
leak \( \rightarrow \) escape

Terminal Event Statements
-------------------------

\* /

rule(nrv, p, \( [p, 2] \), \( [[p, 1]] \), []).
rule(nrv, p, \( [q, 2] \), \( [[q, 1]] \), []).
rule(nrv, p, \( [t, 2] \), \( [[t, 1]] \), []).
rule(nrv, p, \( [[x, 1], 2] \), \( [[[x, 1], 1]] \), []).
rule(nrv, p, \( [[i, _], 2] \), \( [[[i, _], 1]] \), []).
rule(nrv, functional, \( ['valve stuck clsd', ',' ', '] \), \( [[q, 2, none]] \)).
rule(nrv, functional, \( ['valve fails pt. clsd', ',' ', '] \), \( [[q, 2, low]] \)).
rule(nrv, containment, \( ['leak', ',', ' '] \), \( ['escape', ',', ' '] \)).
olefin

material_cause(olefin, blockage, []).
material_cause(olefin, cavitation, [['low b.pt. material', '', ''], ['gassy material', '', '']]).
material_cause(olefin, escape, [['flamable release', '', '']]).
material_effect(olefin, t, high, [['pressure high', '', ''], ['blockage', '', ''], ['polymerisation', '', '']]).
material_effect(olefin, t, low, []).
material_effect(olefin, p, high, []).
material_effect(olefin, p, low, [['liquid flashes', '', '']]).
orifice

/*
Model : orifice

Orifice plate

Propagation Equations
---------------------
p2 = f(p1)
q2 = f(q1)
t2 = f(t1)
x1-2 = f(x1-1)
i_2 = f(i_1)

Initial Event Statements
------------------------
worn orifice > p2 HI, q2 HI
blocked orifice > p2 LO, q2 LO, q2 NONE
leak > escape

Terminal Event Statements
-------------------------
*/

rule(orifice, p, [p, 2], [[p, 1]], []).  
rule(orifice, p, [q, 2], [[q, 1]], []).  
rule(orifice, p, [t, 2], [[t, 1]], []).  
rule(orifice, p, [[x, 1], 2], [[[x, 1], 1]], []).  
rule(orifice, p, [[i, _], 2], [[[i, _], 1]], []).  
rule(orifice, functional, [['worn orifice', ' ', ' ']], [[p, 2, high], [q, 2, high]]).  
rule(orifice, functional, [['blocked orifice', ' ', ' ']], [[p, 2, low], [q, 2, low], [q, 2 \ '0', none]]).  
rule(orifice, containment, [['leak', ' ', ' ']], [['escape', ' ', ' ']]).
pipe
/*
Model : pipe

Generic Pipe Model

Propagation Equations
---------------------
\[ P_2 = f(P_1) \]
\[ q_2 = f(q_1) \]
\[ t_2 = f(t_1) \]
\[ x_{l-2} = f(x_{l-1}) \]
\[ i_{-2} = f(i_{-1}) \]

Initial Event Statements
------------------------
blockage, leak (major) > q_2 LO, q_2 NONE
blockage, ht to blocd-in fluid > p_1 HI
leak (major) > p_2 LO
leak > escape
solar heat, fire > ht to blocd-in fluid
external cold source > t_2 LO
external heat source > t_2 HI
cold weather, lagging loss > external cold source
hot weather, fire > external heat source

Terminal Event Statements
-------------------------
p_1 HI > leak (major), pipe rupture, flange leak
*/

rule(pipe, p, [p, 2], [[p, 1]], []).
rule(pipe, p, [q, 2], [[q, 1]], []).
rule(pipe, p, [t, 2], [[t, 1]], []).
rule(pipe, p, [[x, 1], 2], [[x, 1], 1]).
rule(pipe, p, [[i, _], 2], [[i, _], 1]).
rule(pipe, hydraulic, [['blockage', ' ', ' '], ['leak (major)', ' ', ' ']], [[q, 2, low], \c(q, 2, none)].
rule(pipe, hydraulic, [['blockage', ' ', ' '], ['ht to blokd-in fluid', ' ', ' ']], [[p \c, 1, high]].
rule(pipe, containment, [['leak (major)', ' ', ' ']], [[p, 2, low]]).
rule(pipe, containment, [['leak', ' ', ' ']], [['escape', ' ', ' ']]).
rule(pipe, environmental, [['solar heat', ' ', ' '], ['fire', ' ', ' ']], [['ht to blok \cd-in fluid', ' ', ' ']]).
rule(pipe, environmental, [['external cold source', ' ', ' ']], [[t, 2, low]]).
rule(pipe, environmental, [['external heat source', ' ', ' ']], [[t, 2, high]]).
rule(pipe, environmental, [['cold weather', ' ', ' '], ['lagging loss', ' ', ' ']], [['\c external cold source', ' ', ' ']]).
rule(pipe, environmental, [['hot weather', ' ', ' '], ['fire', ' ', ' ']], [['\c external heat source', ' ', ' ']]).
rule(pipe, effect, [[p, 1, high]], [['leak (major)', ' ', ' '], ['pipe rupture', ' ', ' '], ['flange leak', ' ', ' ']]).
pipeline

/*
Model : pipeline

Pipeline

Propagation Equations
----------------------
\[ p_2 = f(p_1) \]
\[ q_2 = f(q_1) \]
\[ t_2 = f(t_1) \]
\[ x_{1-2} = f(x_{1-1}) \]
\[ i_{-2} = f(i_{-1}) \]

Initial Event Statements
------------------------
\( \text{blockage, leak (major)} \rightarrow q_2 \text{ NONE, } q_2 \text{ LO} \)
\( \text{blockage, hammer/surge, ht to blckd-in fluid} \rightarrow p_2 \text{ HI} \)
\( \text{leak} \rightarrow \text{escape} \)
\( \text{solar heat, fire} \rightarrow \text{ht to blckd-in fluid} \)

Terminal Event Statements
-------------------------
*/

rule(pipeline, p, [p, 2], [[p, 1]], []).
rule(pipeline, p, [q, 2], [[q, 1]], []).
rule(pipeline, p, [t, 2], [[t, 1]], []).
rule(pipeline, p, [[x, 1], 2], [[x, 1], 1], []).
rule(pipeline, p, [[i, _], 2], [[i, _], 1], []).
rule(pipeline, functional, [['blockage', ' ', ' '], ['leak (major)', ' ', ' ']], [q, 2, \cnone], [q, 2, low])].
rule(pipeline, hydraulio, [['blockage', ' ', ' '], ['hammer/surge', ' ', ' ']], ['ht to \blckd-in fluid', ' ', ' '], [p, 2, high])].
rule(pipeline, containment, [['leak', ' ', ' ']], [['escape', ' ', ' ']]).
rule(pipeline, environmental, [['solar heat', ' ', ' '], ['fire', ' ', ' ']], ['ht to \blckd-in fluid', ' ', ' ']).
pumpset_a
/*
 Model : pumpset_a

Pumpset - 2 100% pumps, one running and the other on standby

Propagation Equations
---------------------
p2=f(p1)
q2=f(q1)
t2=f(t1)
x1-2=f(x1-1)
i_2=f(i_-1)

Initial Event Statements
------------------------
loss of NPSH, rotation fault, impeller fault, cavitation>q2 LO, q2 NONE
delivery shut in>q2 NONE, p2 HI
delivery part shut>q2 LO
no change to stndby, stndby fail on dmand, maloperatn of valves>q2 NONE
blockage>q2 NONE, q2 LO, p2 HI
pump fluid overheatst2 HI
leak (major)>q2 NONE, q2 LO
leak>escape
ht to blckd-in fluid>p2 HI
solar heat, fire>ht to blckd-in fluid

Terminal Event Statements
-------------------------
q2 NONE>pump fluid overheat
*/
rule(pumpset_a,p,[p,2],[[p,1]],[]).
rule(pumpset_a,p,[q,2],[[q,1]],[]).
rule(pumpset_a, p, [t, 2], [[t, 1]], []).  
rule(pumpset_a, p, [[x, 1], 2], [[[x, 1]], 1]], []).  
rule(pumpset_a, p, [[i, _], 2], [[[i, _]], 1]], []).  
rule(pumpset_a, hydraulic, [['loss of NPSH', ', ', ''], ['rotation fault', ', ', ''], ['c'impeller fault', ', ', ''], ['cavitation', ', ', ''], [[q, 2, low], [q, 2, none]]).  
rule(pumpset_a, hydraulic, [['delivery shut in', ', ', ''], [q, 2, none], [p, 2, high]])  
rule(pumpset_a, hydraulic, [['delivery part shut', ', ', '']],[q, 2, low])).  
rule(pumpset_a, hydraulic, [['no change to stdby', ', ', ''], ['stdby fail on dman cd', ', ', ''], ['maloperatn of valves', ', ', '']],[[q, 2, none]]).  
rule(pumpset_a, hydraulic, [['blockage', ', ', '']],[q, 2, none], [q, 2, low], [p, 2, high]).  
rule(pumpset_a, hydraulic, [['pump fluid overheats', ', ', '']],[t, 2, high]]).  
rule(pumpset_a, containment, [['leak (major)', ', ', '']],[q, 2, none], [q, 2, low]]).  
rule(pumpset_a, containment, [['leak', ', ', '']],[['escape', ', ', ']]).  
rule(pumpset_a, environmental, [['ht to blckd-in fluid', ', ', '']],[p, 2, high]]).  
rule(pumpset_a, environmental, [['solar heat', ', ', ''], ['fire', ', ', '']],[['ht to c blckd-in fluid', ', ', ']]).  
rule(pumpset_a, effect, [[q, 2, none]], [['pump fluid overheats', ', ', ']]).
Model: settler

Buffer/Settling tank - 1 inlet, 1 outlet, pump kickback, water sump/drain

Propagation Equations
------------------------
\[ 15 = f(q_1, -q_2, -q_4) \]
\[ t_5 = f(t_1) \]
\[ x_{1-5} = f(x_{1-1}) \]
\[ q_2 = f(15, p_5) \]
\[ p_2 = f(p_5) \]
\[ t_2 = f(t_5) \]
\[ q_4 = f(15, p_5) \]
\[ p_4 = f(p_5) \]
\[ t_4 = f(t_5) \]
\[ x_{1-2} = f(x_{1-1}) \]
\[ i_{-5} = f(i_{-1}) \]
\[ i_{-2} = f(i_{-5}) \]

Initial Event Statements
------------------------
pressure cntrl fault, relief valve lifts, loss of N2 blanket > p_5 LO
pressure cntrl fault > p_5 HI
drain open > 15 LO
leak > escape
sump drained too far > i_4-4 HI

Terminal Event Statements
-------------------------
q_1 NONE, 15 NONE > outflow(s) none
q_1 LO, 15 LO > outflow(s) low
15 HI> liquid enters vent
q1 HI, q2 HI> poor settling, ia-2 HI

Is-1 HI> draining, sump freeze up

Ia-1 HI> poor separation of water, water sump overfills

Ic-1 HI> corrosion

Ib-1 HI> pressure high

P1 HI> overpressure, leak (major)

*/

rule(settler, p, [1, 5], [[q, 1], [q, 2], [q, 4]]).
rule(settler, p, [t, 5], [[t, 1], []]).
rule(settler, p, [[x, 1], 5], [[x, 1], 1], []).
rule(settler, p, [q, 2], [[1, 5], [p, 5]], []).
rule(settler, p, [p, 2], [[p, 5]], []).
rule(settler, p, [t, 2], [[t, 5]], []).
rule(settler, p, [q, 4], [[1, 5], [p, 5]], []).
rule(settler, p, [p, 4], [[p, 5]], []).
rule(settler, p, [t, 4], [[t, 5]], []).
rule(settler, p, [[x, 1], 2], [[x, 1], 1], []).
rule(settler, p, [[[i, _], 5], [[[i, _], 1], []]].
rule(settler, p, [[[i, _], 2], [[[i, _], 5]], []].
rule(settler, hydraulic, [['pressure cntrl fault', ' ', ' '], ['relief valve lifts', 'c', ' '], ['loss of N2 blanket', ' ', ' ']], [[p, 5, low]]).
rule(settler, hydraulic, [['pressure cntrl fault', ' ', ' ']], [[p, 5, high]]).
rule(settler, hydraulic, [['drain open', ' ', ' ']], [[1, 5, low]]).
rule(settler, containment, [['leak', ' ', ' ']], [['escape', ' ', ' ']]).
rule(settler, impurity, [['sump drained too far', ' ', ' ']], [[[i, e], 4, high]]).
rule(settler, effect, [[q, 1, none], [1, 5, none]], [['outflow(s) none', ' ', ' ']]).
rule(settler, effect, [[q, 1, low], [1, 5, low]], [['outflow(s) low', ' ', ' ']]).
rule(settler, effect, [[1, 5, high]], [['liquid enters vent', ' ', ' ']]).
rule(settler, effect, [[q, 1, high], [q, 2, high]], [['poor settling', ' ', ' ']], [[[i, a], 2 \\c, high]]).
rule(settler, effect, [[t, 5, low]], [['drain/sump freeze-up', ' ', ' ']]).
rule(settler, effect, [[[i, a], 1, high]], [['poor separation of water', ' ', ' ']], ['water \\c, sump overfills', ' ', ' ']]).
rule(settler, effect, [[[i, c], 1, high]], [['corrosion', ' ', ' ']]).
rule(settler, effect, [[[i, b], 1, high]], [['pressure high', ' ', ' ']]).
rule(settler, effect, [[[p, 1, high]], [['overpressure', ' ', ' ']], [['leak (major)', ' ', ' c', ' ']]).
sink

/*
Model : sink

Sink - one inlet

Propagation Equations
------------------------

Initial Event Statements
------------------------
sink Pull>pl HI,q1 LO,q1 NONE

Terminal Event Statements
------------------------

*/

rule(sink,functional,[["sink full","",""],[p,1,high],[q,1,low],[q,1,none]])
\c.
Model : source

Source - one outlet

Propagation Equations
---------------------

Initial Event Statements
------------------------

source empty;p1 LO,q1 NONE

Terminal Event Statements
------------------------

*/

rule(source, functional, ['source empty', ', ', ''], [[p, 1, low], [q, 1, none]]).
valve

/*
Model : valve

Isolation valve (normally state open)

Propagation Equations
------------------------
p2=f(p1)
q2=f(q1)
t2=f(t1)
x1-2=f(x1-1)
i_2=f(i_1)

Initial Event Statements
------------------------
valve directed clsd>q2 NONE,p1 HI
valve partly clsd>q2 LO
leak>escape

Terminal Event Statements
------------------------
*/

rule(valve,p,[p,2],[[p,1]],[]).
rule(valve,p,[q,2],[[q,1]],[]).
rule(valve,p,[t,2],[[t,1]],[]).
rule(valve,p,[[x,1],2],[[[x,1],1]],[]).
rule(valve,p,[[i, _,2],[[i, _,1]],[]]).
rule(valve,functional,[[valve directed clsd',' ' ', '']],[[q,2,none],[p,1,high]] \
|).
rule(valve,functional,[[valve partly clsd',' ' ', ']],[[q,2,low]]).
rule(valve,containment,[['leak',' ' ', ']],[['escape',' ' ', ']]).
water

material_cause(water, blockage, []).  
material_cause(water, cavitation, []).  
material_effect(water, escape, ' ', []).  
material_effect(water, t, high, []).  
material_effect(water, t, low, [['formation of ice', ' ', ' ', '']]).  
material_effect(water, p, high, []).  
material_effect(water, p, low, []).
APPENDIX H

"compacter" Output for Water Separator Example
HAZOP file created from prolog run file: WATER_sea.org.out

Search on vessel 1 source
p
CAUSES
source empty

Search on vessel 1 settlers
p
CAUSES
CONSEQUENCES
p 5 low
pressure control fault
relief valve lifts
loss of H2 blanket
p 5 high
pressure control fault
overpressure
leak (major)

Search on vessel 1 settlers
l
CAUSES
CONSEQUENCES
q 1 low
q 2 high
q 4 high
drain open

Search on vessel 1 settlers
l
CAUSES
CONSEQUENCES
q 1 low
q 2 high
q 4 high
outflow(s) low
<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>p 1 high</td>
<td>a 2 high</td>
</tr>
<tr>
<td>q 1 high</td>
<td>a 4 high</td>
</tr>
<tr>
<td>q 2 low</td>
<td>Liquid enters vent</td>
</tr>
<tr>
<td>q 4 low</td>
<td></td>
</tr>
<tr>
<td>q 2 none</td>
<td></td>
</tr>
<tr>
<td>q 4 none</td>
<td></td>
</tr>
</tbody>
</table>

Search on vessel 1: divider10

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>p 1 low</td>
<td>o 2 low</td>
</tr>
<tr>
<td>o 3 low</td>
<td>o 4 low</td>
</tr>
</tbody>
</table>

Search on vessel 1: exchanger14

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>o 1 low</td>
<td>o 2 low</td>
</tr>
<tr>
<td>o 3 low</td>
<td>o 4 low</td>
</tr>
</tbody>
</table>

Search on vessel 1: exchanger14

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>p 1 high</td>
<td>o 2 high</td>
</tr>
<tr>
<td>o 3 high</td>
<td>o 4 high</td>
</tr>
<tr>
<td>tube leak (major)</td>
<td></td>
</tr>
</tbody>
</table>

Search on vessel 1: source16
<table>
<thead>
<tr>
<th>d</th>
<th>low</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
</tr>
<tr>
<td>source empty</td>
<td></td>
</tr>
</tbody>
</table>

---

Search on vessel: sink15

<table>
<thead>
<tr>
<th>d</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
</tr>
<tr>
<td>sink full</td>
<td></td>
</tr>
</tbody>
</table>

---

Search on vessel: sink17

<table>
<thead>
<tr>
<th>d</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
</tr>
<tr>
<td>sink full</td>
<td></td>
</tr>
</tbody>
</table>

---

Search on vessel: sink19

<table>
<thead>
<tr>
<th>d</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
</tr>
<tr>
<td>sink full</td>
<td></td>
</tr>
</tbody>
</table>

---

Search on Line 1

<table>
<thead>
<tr>
<th>d</th>
<th>none</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
</tr>
<tr>
<td>source1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>source empty</td>
</tr>
<tr>
<td>oloe</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>leak (major)</td>
</tr>
<tr>
<td>pump2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>rotation fault</td>
</tr>
<tr>
<td>2</td>
<td>impeller fault</td>
</tr>
<tr>
<td>2</td>
<td>cavitation</td>
</tr>
<tr>
<td>low head</td>
<td>material</td>
</tr>
<tr>
<td>gassy material</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>delivery shut in</td>
</tr>
<tr>
<td>2</td>
<td>no change to standby</td>
</tr>
<tr>
<td>2</td>
<td>standby fail on demand</td>
</tr>
</tbody>
</table>
Search on line 1

q  low

CAUSES

source1  1 q 1 low

penc  2 blockage
     2 leak (major)

pump2  2 loss of head
       2 rotation fault
       2 impeller fault
       2 cavitation
       2 gassy material
       2 delivery part shut
       2 blockage
       2 leak (major)

cntrvlvl6  2 loop fails at-clsd
            2 sensor fails at-clsd
            2 cntrvlvl fails at-clsd
            2 set-pt and cvd at-clsd
            2 isolatn valve at-clsd
            2 blockage

valve7  2 valve partly clsd

CONSEQUENCES

meter4  2 inaccurate measurement

settler9  1 q 1 low
         1 outflow(s) low
         1 t 5 low
<table>
<thead>
<tr>
<th><strong>a</strong> high</th>
<th><strong>CONSEQUENCES</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>source1</strong></td>
</tr>
<tr>
<td></td>
<td>1 a 1 high</td>
</tr>
<tr>
<td><strong>meter4</strong></td>
<td>2 bypass fails</td>
</tr>
<tr>
<td></td>
<td>2 bypass directed open</td>
</tr>
<tr>
<td><strong>ctrl_v16</strong></td>
<td>2 loop fails open</td>
</tr>
<tr>
<td></td>
<td>sensor fails open</td>
</tr>
<tr>
<td></td>
<td>cntlr fails open</td>
</tr>
<tr>
<td></td>
<td>cntl-v1 fails open</td>
</tr>
<tr>
<td></td>
<td>set-point out of open</td>
</tr>
<tr>
<td></td>
<td>2 bypass fails</td>
</tr>
<tr>
<td></td>
<td>2 bypass directed open</td>
</tr>
<tr>
<td><strong>CONSEQUENCES</strong></td>
<td>meter4</td>
</tr>
<tr>
<td></td>
<td>2 inaccurate measurement</td>
</tr>
<tr>
<td></td>
<td><strong>settler9</strong></td>
</tr>
<tr>
<td></td>
<td>1 a 1 high</td>
</tr>
<tr>
<td></td>
<td>1 poor settling</td>
</tr>
<tr>
<td></td>
<td>1 (leak a) 2 high</td>
</tr>
<tr>
<td></td>
<td>1 L 5 high</td>
</tr>
<tr>
<td></td>
<td>5 liquid enters vent</td>
</tr>
</tbody>
</table>

---

Search on line 1

<table>
<thead>
<tr>
<th><strong>p</strong> low</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
</tr>
<tr>
<td><strong>source1</strong></td>
</tr>
<tr>
<td>1 a 1 low</td>
</tr>
<tr>
<td>1 source faulty</td>
</tr>
<tr>
<td><strong>pipe</strong></td>
</tr>
<tr>
<td>2 leak (major)</td>
</tr>
<tr>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td><strong>settler9</strong></td>
</tr>
<tr>
<td>1 a 1 low</td>
</tr>
<tr>
<td><strong>olefin</strong></td>
</tr>
<tr>
<td>liquid flashes</td>
</tr>
</tbody>
</table>

---

Search on line 1

<table>
<thead>
<tr>
<th><strong>p</strong> high</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
</tr>
<tr>
<td><strong>source1</strong></td>
</tr>
<tr>
<td>1 a 1 high</td>
</tr>
<tr>
<td><strong>pipe</strong></td>
</tr>
<tr>
<td>1 blockage</td>
</tr>
<tr>
<td>1 ht to block-in fluid</td>
</tr>
<tr>
<td>1 solar heat</td>
</tr>
<tr>
<td>1 fire</td>
</tr>
<tr>
<td><strong>dumo2</strong></td>
</tr>
<tr>
<td>2 delivery shut in</td>
</tr>
<tr>
<td>2 blockage</td>
</tr>
<tr>
<td>2 ht to block-in fluid</td>
</tr>
<tr>
<td>2 solar heat</td>
</tr>
<tr>
<td>2 fire</td>
</tr>
<tr>
<td><strong>pipeline5</strong></td>
</tr>
<tr>
<td>2 blockage</td>
</tr>
<tr>
<td>2 hammer/surge</td>
</tr>
<tr>
<td>2 ht to block-in fluid</td>
</tr>
<tr>
<td>2 solar heat</td>
</tr>
<tr>
<td>2 fire</td>
</tr>
<tr>
<td><strong>ctrl_v16</strong></td>
</tr>
<tr>
<td>1 loop fails closed</td>
</tr>
<tr>
<td>sensor fails closed</td>
</tr>
<tr>
<td>cntlr fails closed</td>
</tr>
<tr>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td><strong>pio9</strong></td>
</tr>
<tr>
<td>1 leak (major)</td>
</tr>
<tr>
<td>1 olean rupture</td>
</tr>
<tr>
<td>1 flange leak</td>
</tr>
<tr>
<td><strong>settler9</strong></td>
</tr>
<tr>
<td>1 a 1 high</td>
</tr>
<tr>
<td>1 overpressure</td>
</tr>
<tr>
<td>1 leak (major)</td>
</tr>
<tr>
<td>Search on Line 1</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td><strong>CAUSES</strong></td>
</tr>
<tr>
<td>source1</td>
</tr>
<tr>
<td>1 t 1 low</td>
</tr>
<tr>
<td>pipe</td>
</tr>
<tr>
<td>2 external cold source</td>
</tr>
<tr>
<td>cold weather</td>
</tr>
<tr>
<td>leakage loss</td>
</tr>
<tr>
<td>pump2</td>
</tr>
<tr>
<td>2 pump fluid overheats</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on Line 1</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td></td>
</tr>
<tr>
<td>source1</td>
<td></td>
</tr>
<tr>
<td>1 t 1 high</td>
<td>settler4</td>
</tr>
<tr>
<td>pipe</td>
<td></td>
</tr>
<tr>
<td>2 external heat source</td>
<td></td>
</tr>
<tr>
<td>hot weather</td>
<td>olefin</td>
</tr>
<tr>
<td>fire</td>
<td>pressure high</td>
</tr>
<tr>
<td></td>
<td>place case</td>
</tr>
<tr>
<td></td>
<td>polymerisation</td>
</tr>
<tr>
<td>pump2</td>
<td></td>
</tr>
<tr>
<td>2 pump fluid overheats</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on Line 1</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td></td>
</tr>
<tr>
<td>aloe</td>
<td></td>
</tr>
<tr>
<td>pump2</td>
<td></td>
</tr>
<tr>
<td>nrv3</td>
<td></td>
</tr>
<tr>
<td>meter4</td>
<td></td>
</tr>
<tr>
<td>diesel line5</td>
<td></td>
</tr>
<tr>
<td>cntrl valves3</td>
<td></td>
</tr>
<tr>
<td>valve7</td>
<td></td>
</tr>
<tr>
<td>settler3</td>
<td></td>
</tr>
</tbody>
</table>

- t1 falls closed
- set proud and closed
- isolaton valve closed
- valve 1 valve directed closed
- leak (minor)
- brittle fracture
### Line 1: IMPURITIES

<table>
<thead>
<tr>
<th>Impurity a high Causes source</th>
<th>Consequences</th>
</tr>
</thead>
</table>
| Impurity a in source           | Poor settling of water
|                                | Water tank overflows  |

<table>
<thead>
<tr>
<th>Impurity b high Causes source</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Impurity b in source</td>
<td>High pressure</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Impurity c high Causes source</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Impurity c in source</td>
<td>Corrosion</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Impurity e high Causes Settle</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supernatant too far</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Impurity d high Causes</th>
<th>Consequences</th>
</tr>
</thead>
</table>

---

### Search on line 2

<table>
<thead>
<tr>
<th>a</th>
<th>none</th>
</tr>
</thead>
</table>

### CAUSES

| Settle 9 | 2 l 5 none
|          | 2 l 5 low
|          | 2 l 2 none
| Pumpe    | 2 blockage
|          | 2 leak (major)
|          | 2 low NPSH
|          | 2 rotation fault
|          | 2 Moter fault
|          | 2 cavitation
|          | 2 material
|          | 2 gassy material
|          | 2 delivery shut in
|          | 2 no change to supply
|          | 2 end of day
|          | 2 motor rel. of valves
|          | 2 blockage
|          | 2 leak (major)
|          | 2 l 5 none
|          | 2 l 2 none
|          | 2 l 3 none

#
<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>settle9</td>
<td>settle9</td>
</tr>
<tr>
<td>2 l 5 low</td>
<td>2 l 5 low</td>
</tr>
<tr>
<td>2 a 5 low</td>
<td>2 a 5 low</td>
</tr>
<tr>
<td>2 q 2 low</td>
<td>2 q 2 low</td>
</tr>
<tr>
<td>pipe</td>
<td>divider10</td>
</tr>
<tr>
<td>2 blockage</td>
<td>1 a 1 low</td>
</tr>
<tr>
<td>2 leak (major)</td>
<td>1 a 2 low</td>
</tr>
<tr>
<td>2 leak (major)</td>
<td>1 a 3 low</td>
</tr>
</tbody>
</table>

Search on line 2

q high

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>settle9</td>
<td>settle9</td>
</tr>
<tr>
<td>2 l 5 high</td>
<td>2 l 5 low</td>
</tr>
<tr>
<td>2 a 5 high</td>
<td>2 a 5 low</td>
</tr>
<tr>
<td>2 q 2 high</td>
<td>2 q 2 low</td>
</tr>
<tr>
<td>divider10</td>
<td></td>
</tr>
<tr>
<td>1 a 1 high</td>
<td></td>
</tr>
<tr>
<td>1 a 2 high</td>
<td></td>
</tr>
<tr>
<td>1 a 3 high</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 2

q low

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>settle9</td>
<td>divider10</td>
</tr>
<tr>
<td>2 a 5 low</td>
<td>1 a 1 low</td>
</tr>
<tr>
<td>2 q 2 low</td>
<td>1 a 2 low</td>
</tr>
<tr>
<td>pipe</td>
<td>divider10</td>
</tr>
<tr>
<td>2 leak (major)</td>
<td>1 a 3 low</td>
</tr>
<tr>
<td>olefin</td>
<td></td>
</tr>
<tr>
<td></td>
<td>liquid flashes</td>
</tr>
</tbody>
</table>

Search on line 2

q high

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>settle9</td>
<td>oleo</td>
</tr>
<tr>
<td>2 a 5 high</td>
<td>1 leak (major)</td>
</tr>
<tr>
<td>2 q 2 high</td>
<td>1 oleo rupture</td>
</tr>
</tbody>
</table>
### Search on line 2

#### t low

**CAUSES**
- settler9
  - 2 t 5 low
  - 2 t 2 low
- aloe
  - 2 external cold source
  - cold weather
  - lagging loss

**CONSEQUENCES**
- aloe
  - 1 leak (major)
  - 1 small fracture

### Search on line 2

#### t high

**CAUSES**
- settler9
  - 2 t 5 high
  - 2 t 2 high
- aloe
  - 2 external heat source
  - hot weather
  - fire
- pump9
  - 2 pump fluid overheats

**CONSEQUENCES**
- aloe
  - 1 t 1 high
  - 1 t 2 high
  - 1 t 3 high

### Search on line 2

**escape**

**CAUSES**
- settler9
  - leak
- aloe
  - leak
- pump9
  - leak
- divider10
  - leak

**CONSEQUENCES**
- aloe
  - olefin
  - flammable release

### Search on line 2

**place**

**CAUSES**
- blockage
  - 1 ht to block-in fluid
  - solar heat
  - fire

**CONSEQUENCES**
- divider10
  - 1 a 1 high
  - 1 a 2 high
  - 1 a 3 high
<table>
<thead>
<tr>
<th>Line 2</th>
<th>IMPURITIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Impurity a high Causes</td>
<td>Consequences</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Impurity b high Causes</td>
<td>Consequences</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Impurity c high Causes</td>
<td>Consequences</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Impurity e high Causes</td>
<td>Consequences</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Impurity d high Causes</td>
<td>Consequences</td>
</tr>
</tbody>
</table>

**Search on line 3**

<table>
<thead>
<tr>
<th>Causes</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>none</td>
</tr>
<tr>
<td>Causes</td>
<td>Causes</td>
</tr>
<tr>
<td>divider10</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>q 3 none</td>
</tr>
<tr>
<td>3</td>
<td>blockage</td>
</tr>
<tr>
<td>3</td>
<td>leak (major)</td>
</tr>
<tr>
<td>pipe</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>leak (major)</td>
</tr>
<tr>
<td>orifice11</td>
<td>2</td>
</tr>
<tr>
<td>valve12</td>
<td>2</td>
</tr>
</tbody>
</table>

**Search on line 3**

<table>
<thead>
<tr>
<th>Causes</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>low</td>
</tr>
<tr>
<td>Causes</td>
<td>Causes</td>
</tr>
<tr>
<td>divider10</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>q 3 low</td>
</tr>
<tr>
<td>3</td>
<td>blockage</td>
</tr>
<tr>
<td>3</td>
<td>leak (major)</td>
</tr>
<tr>
<td>pipe</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>leak (major)</td>
</tr>
</tbody>
</table>
oriifice11  2  blocked orifice

oriifice12  2  valve partly cld

Search on line 3
a  high

CAUSES
divider10  3  a 1 high
  3  a 3 high
  3  blockage - other led

oriifice11  2  worn orifice

Search on line 3
a  low

CAUSES
divider10  3  a 1 low
  3  a 3 low

oriifice11  2  worn orifice

Search on line 3
a  high

CAUSES
divider10  3  a 1 high
  3  a 3 high
  3  blockage

oriifice11  2  worn orifice

valve12  1  valve directed cld

Search on line 3
t  low
### Causes

<table>
<thead>
<tr>
<th>Cause</th>
<th>Consequences</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Divider 10</td>
<td>3 t 1 low</td>
<td>1 Leak (major)</td>
</tr>
<tr>
<td>Divider 10</td>
<td>3 t 3 low</td>
<td>1 Brittle fracture</td>
</tr>
<tr>
<td>Ploe</td>
<td>2 external cold source</td>
<td>3 t 3 low</td>
</tr>
<tr>
<td></td>
<td>Cold weather</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Leaking loss</td>
<td></td>
</tr>
<tr>
<td>Ploe</td>
<td>3 t 3 high</td>
<td></td>
</tr>
<tr>
<td>Orifice 11</td>
<td>Leak</td>
<td>3 t 3 high</td>
</tr>
<tr>
<td>Valve 12</td>
<td>Leak</td>
<td>4 Pressure high</td>
</tr>
<tr>
<td>Settler 8</td>
<td>Leak</td>
<td>Blockage</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Polymerisation</td>
</tr>
</tbody>
</table>

### Search on Line 3

#### Causes

<table>
<thead>
<tr>
<th>Cause</th>
<th>Consequences</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orifice 11</td>
<td>Leak</td>
<td>3 t 3 high</td>
</tr>
<tr>
<td>Valve 12</td>
<td>Leak</td>
<td>4 Pressure high</td>
</tr>
<tr>
<td>Settler 8</td>
<td>Leak</td>
<td>Blockage</td>
</tr>
</tbody>
</table>

### Search on Line 3

#### Causes

<table>
<thead>
<tr>
<th>Cause</th>
<th>Consequences</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orifice 11</td>
<td>Leak</td>
<td>3 t 3 high</td>
</tr>
<tr>
<td>Valve 12</td>
<td>Leak</td>
<td>4 Pressure high</td>
</tr>
<tr>
<td>Settler 8</td>
<td>Leak</td>
<td>Blockage</td>
</tr>
</tbody>
</table>

### Line 3 Impurities

#### Impurity a high

<table>
<thead>
<tr>
<th>Cause</th>
<th>Consequences</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orifice 11</td>
<td>Leak</td>
<td>3 t 3 high</td>
</tr>
<tr>
<td>Valve 12</td>
<td>Leak</td>
<td>4 Pressure high</td>
</tr>
<tr>
<td>Settler 8</td>
<td>Leak</td>
<td>Blockage</td>
</tr>
</tbody>
</table>

### Impurity b high

<table>
<thead>
<tr>
<th>Cause</th>
<th>Consequences</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orifice 11</td>
<td>Leak</td>
<td>3 t 3 high</td>
</tr>
<tr>
<td>Valve 12</td>
<td>Leak</td>
<td>4 Pressure high</td>
</tr>
<tr>
<td>Settler 8</td>
<td>Leak</td>
<td>Blockage</td>
</tr>
</tbody>
</table>

### Impurity c high

<table>
<thead>
<tr>
<th>Cause</th>
<th>Consequences</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orifice 11</td>
<td>Leak</td>
<td>3 t 3 high</td>
</tr>
<tr>
<td>Valve 12</td>
<td>Leak</td>
<td>4 Pressure high</td>
</tr>
<tr>
<td>Settler 8</td>
<td>Leak</td>
<td>Blockage</td>
</tr>
</tbody>
</table>

### Impurity d high

<table>
<thead>
<tr>
<th>Cause</th>
<th>Consequences</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orifice 11</td>
<td>Leak</td>
<td>3 t 3 high</td>
</tr>
<tr>
<td>Valve 12</td>
<td>Leak</td>
<td>4 Pressure high</td>
</tr>
<tr>
<td>Settler 8</td>
<td>Leak</td>
<td>Blockage</td>
</tr>
<tr>
<td>Impurity e high Causes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------</td>
<td>--------------</td>
</tr>
<tr>
<td>settlers</td>
<td>sum drained too far</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Impurity d high Causes</th>
<th></th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Search on line 4 q none</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
</tr>
<tr>
<td>divider10</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>pipe</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>cntrl_vl13</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 4 q low</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
</tr>
<tr>
<td>divider10</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>pipe</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>cntrl_vl13</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 4 q high</th>
</tr>
</thead>
</table>

| Exchanger of 1         | 1 q 1 none   |
|                        | 1 q 2 none   |

| Exchanger of 2         | 1 q 1 low    |
|                        | 1 q 2 low    |
|                        | 1 q 3 high   |
|                        | 1 q 4 high   |

| Exchanger of 3         | 2 blockage   |

| Exchanger of 4         | 2 blockage   |
### Causes
- **divider10**
  - 2: q1 high
  - 2: q2 high
  - 2: blockage - other tech.
- **cntrl1_vl13**
  - 2: loop fails open
  - sensor fails open
  - cntrlr fails open
  - ctrl-vl fails open
  - set-ont and onen
  - 2: bypass fails
  - 2: bypass directed open

### Consequences
- **exchanc14**
  - 1: q1 high
  - 1: q2 high
  - 1: t1 low
  - 1: t4 low

---

#### Search on Line 4

**p**

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>divider10</strong></td>
<td>2: p1 low</td>
</tr>
<tr>
<td></td>
<td>2: p2 low</td>
</tr>
<tr>
<td><strong>pipe</strong></td>
<td>2: leak (major)</td>
</tr>
</tbody>
</table>

**Consequences**
- **exchanc14**
  - 1: p1 low

---

#### Search on Line 4

**p**

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>divider10</strong></td>
<td>2: p1 high</td>
</tr>
<tr>
<td></td>
<td>2: p2 high</td>
</tr>
<tr>
<td><strong>pipe</strong></td>
<td>1: blockage</td>
</tr>
<tr>
<td></td>
<td>1: hot to block-in fluid</td>
</tr>
<tr>
<td></td>
<td>solar heat</td>
</tr>
<tr>
<td></td>
<td>fire</td>
</tr>
<tr>
<td><strong>cntrl1_vl13</strong></td>
<td>1: loop fails closed</td>
</tr>
<tr>
<td></td>
<td>sensor fails closed</td>
</tr>
<tr>
<td></td>
<td>cntrlr fails closed</td>
</tr>
<tr>
<td></td>
<td>ctrl-vl fails closed</td>
</tr>
<tr>
<td></td>
<td>set-ont and onen</td>
</tr>
<tr>
<td><strong>exchanc14</strong></td>
<td>1: isolan valve closed</td>
</tr>
</tbody>
</table>

**Consequences**
- **exchanc14**
  - 1: p1 high
  - 1: p2 high

---

#### Search on Line 4

**t**

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>exchanc14</strong></td>
<td>1: tube fouling</td>
</tr>
</tbody>
</table>

---
<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>divider10</td>
<td>plae</td>
<td>1</td>
</tr>
<tr>
<td>t 1 low</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t 2 low</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td>2 external cold source</td>
<td>1</td>
</tr>
<tr>
<td>cold weather tagging loss</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exchange14</td>
<td>1 t 1 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 t 2 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 t 4 low</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 4

t high

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>divider10</td>
<td>exchange14</td>
<td>1 t 1 high</td>
</tr>
<tr>
<td>t 1 high</td>
<td>olefin</td>
<td>pressure high</td>
</tr>
<tr>
<td>t 2 high</td>
<td>blockage</td>
<td>polymerisation</td>
</tr>
<tr>
<td>pipe</td>
<td>exchange14</td>
<td>1 t 4 high</td>
</tr>
<tr>
<td>2 external heat source hot weather fire</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Search on line 4

escape

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>divider10</td>
<td>leak</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td>olefin</td>
<td>flammable release</td>
</tr>
<tr>
<td>cntrl_v13</td>
<td>leak</td>
<td></td>
</tr>
<tr>
<td>exchange14</td>
<td>leak</td>
<td></td>
</tr>
</tbody>
</table>

Line 4 IMPURITIES

<table>
<thead>
<tr>
<th>Impurity a high Causes</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Impurity b high Causes</td>
<td>Consequences</td>
</tr>
<tr>
<td>Impurity c high Causes</td>
<td>Consequences</td>
</tr>
<tr>
<td>Impurity e high Causes</td>
<td>Consequences</td>
</tr>
<tr>
<td>Impurity d high</td>
<td>Causes</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------</td>
</tr>
<tr>
<td></td>
<td>exchng14</td>
</tr>
</tbody>
</table>

Search on line 5

<table>
<thead>
<tr>
<th>q none</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>exchng14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 a 1 none</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 a 2 none</td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipe</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 blockage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 leak (major)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sink15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 sink full</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 5

<table>
<thead>
<tr>
<th>q low</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>exchng14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 a 1 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 a 2 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipe</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 blockage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 leak (major)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sink15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 sink full</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 5

<table>
<thead>
<tr>
<th>q high</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>exchng14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 a 1 high</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 a 2 high</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 5

<table>
<thead>
<tr>
<th>q low</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>exchng14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 a 1 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 a 2 low</td>
<td></td>
</tr>
<tr>
<td>Causes</td>
<td>Consequences</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------</td>
<td></td>
</tr>
<tr>
<td>Exchanger.</td>
<td>2 tube leak (major)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 leak (major)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Olefin</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Liquid flashes</td>
<td></td>
</tr>
<tr>
<td>Pipe.</td>
<td>1 leak (major)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 pipe rupture</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 flange leak</td>
<td></td>
</tr>
<tr>
<td>Blockage</td>
<td>1 nt to click-in fluid</td>
<td></td>
</tr>
<tr>
<td>Solar heat</td>
<td>1 sink full</td>
<td></td>
</tr>
<tr>
<td>Fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sink.</td>
<td>1 sink full</td>
<td></td>
</tr>
</tbody>
</table>

**Search on line 5**

<table>
<thead>
<tr>
<th>Causes</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exchanger.</td>
<td>2 t 1 low</td>
</tr>
<tr>
<td></td>
<td>2 t 3 low</td>
</tr>
<tr>
<td></td>
<td>2 q 3 low</td>
</tr>
<tr>
<td></td>
<td>2 q 1 high</td>
</tr>
<tr>
<td></td>
<td>2 t 2 low</td>
</tr>
<tr>
<td></td>
<td>2 tube fouling</td>
</tr>
<tr>
<td></td>
<td>2 external cold source</td>
</tr>
<tr>
<td></td>
<td>cold weather</td>
</tr>
<tr>
<td></td>
<td>lagging loss</td>
</tr>
<tr>
<td>Pipe.</td>
<td>1 leak (major)</td>
</tr>
<tr>
<td></td>
<td>1 brittle fracture</td>
</tr>
<tr>
<td></td>
<td>1 t 1 low</td>
</tr>
</tbody>
</table>

**Search on line 5**

<table>
<thead>
<tr>
<th>Causes</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exchanger.</td>
<td>2 t 1 high</td>
</tr>
<tr>
<td></td>
<td>2 t 3 high</td>
</tr>
<tr>
<td></td>
<td>2 q 3 high</td>
</tr>
<tr>
<td></td>
<td>2 q 1 low</td>
</tr>
<tr>
<td></td>
<td>2 t 2 high</td>
</tr>
<tr>
<td></td>
<td>2 external heat source</td>
</tr>
<tr>
<td></td>
<td>hot weather</td>
</tr>
<tr>
<td></td>
<td>fire</td>
</tr>
<tr>
<td>Pipe.</td>
<td>1 t 1 high</td>
</tr>
<tr>
<td></td>
<td>Olefin</td>
</tr>
<tr>
<td></td>
<td>pressure high</td>
</tr>
<tr>
<td></td>
<td>blockage</td>
</tr>
<tr>
<td></td>
<td>polymerisation</td>
</tr>
<tr>
<td>Sink.</td>
<td>2 sink full</td>
</tr>
</tbody>
</table>
### Search on line 3

#### CAUSES
- exchange
- place

#### CONSEQUENCES
- olefin
- flammable release

### Line 5 IMPURITIES

<table>
<thead>
<tr>
<th>Impurity</th>
<th>Causes</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>a high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c high</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d high</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Search on line 6

#### CAUSES
- source16
- place

#### CONSEQUENCES
- exchange
- leak

<table>
<thead>
<tr>
<th>Source</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>source16</td>
<td>1</td>
</tr>
<tr>
<td>place</td>
<td>2</td>
</tr>
</tbody>
</table>

### Search on line 6

#### CAUSES
- source16
- place

#### CONSEQUENCES
- exchange

<table>
<thead>
<tr>
<th>Source</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>source16</td>
<td>1</td>
</tr>
<tr>
<td>place</td>
<td>2</td>
</tr>
<tr>
<td>Search on line 6</td>
<td>CAUSES</td>
</tr>
<tr>
<td>------------------</td>
<td>--------</td>
</tr>
<tr>
<td>a high</td>
<td>source16 1 a 1 high</td>
</tr>
<tr>
<td></td>
<td>3 a 2 high</td>
</tr>
<tr>
<td></td>
<td>3 a 4 high</td>
</tr>
<tr>
<td>3 t 4 high</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>a low</td>
<td>source16 1 a 1 low</td>
<td>exchang14 3 a 3 low</td>
</tr>
<tr>
<td></td>
<td>3 a 2 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 a 4 low</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>a high</td>
<td>source16 1 a 1 high</td>
<td>exchang14 3 a 3 high</td>
</tr>
<tr>
<td></td>
<td>source empty</td>
<td>olefin 3 a 3 low</td>
</tr>
<tr>
<td></td>
<td>3 a 4 low</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>t low</td>
<td>source15 1 t 1 low</td>
<td>oleo 1 leak (major)</td>
</tr>
<tr>
<td></td>
<td>oleo 1 oleo rupture</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 flange leak</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>t low</td>
<td>source15 2 external cold source</td>
<td>exchang14 3 t 3 low</td>
</tr>
<tr>
<td></td>
<td>cold weather</td>
<td></td>
</tr>
<tr>
<td></td>
<td>lagging loss</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 t 2 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 t 4 low</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------</td>
<td></td>
</tr>
<tr>
<td>high</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source15</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>high</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Plate</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 external heat source hot weather fire</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Plate</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>leak</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Plate</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>leak</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Line 6 IMPURITIES</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Impurity a high</td>
<td></td>
</tr>
<tr>
<td>Causes</td>
<td></td>
</tr>
<tr>
<td>Impurity b high</td>
<td></td>
</tr>
<tr>
<td>Causes</td>
<td></td>
</tr>
<tr>
<td>Impurity c high</td>
<td></td>
</tr>
<tr>
<td>Causes</td>
<td></td>
</tr>
<tr>
<td>Impurity d high</td>
<td></td>
</tr>
<tr>
<td>Causes</td>
<td></td>
</tr>
<tr>
<td>Impurity e high</td>
<td></td>
</tr>
<tr>
<td>Causes</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 7</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td></td>
</tr>
<tr>
<td>exchanger14</td>
<td></td>
</tr>
<tr>
<td>4 a 3 none</td>
<td></td>
</tr>
<tr>
<td>2 blockage</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td></td>
</tr>
<tr>
<td>exchanger14</td>
<td></td>
</tr>
<tr>
<td>leak</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td></td>
</tr>
<tr>
<td>otefin</td>
<td></td>
</tr>
<tr>
<td>pressure high</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td></td>
</tr>
<tr>
<td>exchanger14</td>
<td></td>
</tr>
<tr>
<td>3 a 2 high</td>
<td></td>
</tr>
<tr>
<td>3 a 4 high</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td></td>
</tr>
<tr>
<td>exchanger14</td>
<td></td>
</tr>
<tr>
<td>leak</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on line 6</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td></td>
</tr>
<tr>
<td>otefin</td>
<td></td>
</tr>
<tr>
<td>flammable release</td>
<td></td>
</tr>
<tr>
<td>Search on Line 7</td>
<td>CAUSES</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>q low</td>
<td>exchang14 4 a 3 low</td>
</tr>
<tr>
<td></td>
<td>exchang14 4 a 4 low</td>
</tr>
<tr>
<td></td>
<td>exchang14 4 shell leak (major)</td>
</tr>
<tr>
<td></td>
<td>pipe 2 blockage</td>
</tr>
<tr>
<td></td>
<td>pipe 2 leak (major)</td>
</tr>
<tr>
<td></td>
<td>sink17 1 sink full</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on Line 7</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>q high</td>
<td>exchang14 4 a 3 high</td>
<td>sink17 1 q 1 high</td>
</tr>
<tr>
<td></td>
<td>exchang14 4 a 4 high</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on Line 7</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>p low</td>
<td>exchang14 4 a 3 low</td>
<td>sink17 1 q 1 low</td>
</tr>
<tr>
<td></td>
<td>exchang14 4 a 4 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td>exchang14 4 shell leak (major)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipe 2 leak (major)</td>
<td>olefin liquid flashes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on Line 7</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>p high</td>
<td>exchang14 4 a 3 high</td>
<td>olefin 1 leak (major)</td>
</tr>
<tr>
<td></td>
<td>exchang14 4 a 4 high</td>
<td>olefin 1 oleo rupture</td>
</tr>
<tr>
<td></td>
<td>exchang14 4 tube leak (major)</td>
<td>olefin 1 flange leak</td>
</tr>
<tr>
<td></td>
<td>pipe 2 blockage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>pipe 2 leak (major)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sink17 1 sink full</td>
<td></td>
</tr>
<tr>
<td>Causes</td>
<td>Consequences</td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>Imoirtity b high</td>
<td>Causes</td>
<td>Consequences</td>
</tr>
<tr>
<td>tube leak</td>
<td>Causes</td>
<td>Consequences</td>
</tr>
<tr>
<td>Search on line 8</td>
<td>a</td>
<td>none</td>
</tr>
<tr>
<td>Causes</td>
<td>Consequences</td>
<td></td>
</tr>
<tr>
<td>settlers</td>
<td>settlers 4</td>
<td>5</td>
</tr>
<tr>
<td>pipe</td>
<td>blockage</td>
<td>leak (major)</td>
</tr>
<tr>
<td>valve18</td>
<td>valve directed closed</td>
<td></td>
</tr>
<tr>
<td>sink19</td>
<td>sink full</td>
<td></td>
</tr>
<tr>
<td>Search on line 8</td>
<td>a</td>
<td>low</td>
</tr>
<tr>
<td>Causes</td>
<td>Consequences</td>
<td></td>
</tr>
<tr>
<td>settlers</td>
<td>settlers 4</td>
<td>5</td>
</tr>
<tr>
<td>pipe</td>
<td>blockage</td>
<td>leak (major)</td>
</tr>
<tr>
<td>valve18</td>
<td>valve partly closed</td>
<td></td>
</tr>
<tr>
<td>sink19</td>
<td>sink full</td>
<td></td>
</tr>
<tr>
<td>Search on line 8</td>
<td>a</td>
<td>high</td>
</tr>
<tr>
<td>Causes</td>
<td>Consequences</td>
<td></td>
</tr>
<tr>
<td>Scenario 1</td>
<td>Scenario 2</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td></td>
</tr>
<tr>
<td>Settlers</td>
<td>Settlers</td>
<td></td>
</tr>
<tr>
<td>4 l 5 high</td>
<td>4 l 5 low</td>
<td></td>
</tr>
<tr>
<td>4 o 5 high</td>
<td>4 o 4 low</td>
<td></td>
</tr>
<tr>
<td>aloe</td>
<td>sink19</td>
<td></td>
</tr>
<tr>
<td>2 leak (major)</td>
<td>1 o 1 high</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 8

CAUSES
- aloe
- sink19

CONSEQUENCES
- leak (major)
- aloe rupture
- flange leak

Search on line 8

p high

CAUSES
- aloe
- sink19

CONSEQUENCES
- blockage
- ht to alkali fluid
- solar heat
- fire
- valve18
- valve directed close
- sink full

Search on line 8

t low

CAUSES
- aloe
- sink19

CONSEQUENCES
- leak (major)
- brittle fracture

Search on line 8

t high

CAUSES

CONSEQUENCES
- external cold source
- cold water
- lagging loss
- water
- formation of ice
<table>
<thead>
<tr>
<th>cause</th>
<th>high</th>
<th>consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>settlerA</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>settlerB</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>aloe</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>valveV</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>hot weather</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fire</td>
</tr>
</tbody>
</table>

**Search on line 8**

**Escape CAUSES**

<table>
<thead>
<tr>
<th>cause</th>
<th>consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>settlerG</td>
<td>leak</td>
</tr>
<tr>
<td>aloe</td>
<td>leak</td>
</tr>
</tbody>
</table>

**Line 8 IMPURITIES**

**Impurity a**

<table>
<thead>
<tr>
<th>cause</th>
<th>consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>settlerG</td>
<td>leak</td>
</tr>
</tbody>
</table>

**Impurity b**

<table>
<thead>
<tr>
<th>cause</th>
<th>consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>settlerG</td>
<td>leak</td>
</tr>
<tr>
<td>aloe</td>
<td>pressure high</td>
</tr>
</tbody>
</table>

**Impurity c**

<table>
<thead>
<tr>
<th>cause</th>
<th>consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>settlerG</td>
<td>leak</td>
</tr>
<tr>
<td>aloe</td>
<td>pressure high</td>
</tr>
</tbody>
</table>

**Impurity e**

<table>
<thead>
<tr>
<th>cause</th>
<th>consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>settlerG</td>
<td>leak</td>
</tr>
<tr>
<td>aloe</td>
<td>corrosion</td>
</tr>
</tbody>
</table>

**Impurity d**

<table>
<thead>
<tr>
<th>cause</th>
<th>consequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>settlerG</td>
<td>leak</td>
</tr>
<tr>
<td>aloe</td>
<td>sumo drained too far</td>
</tr>
</tbody>
</table>
APPENDIX 1

Configuration File for Propane Pipeline Example
Propane_ln.conf

/*

Filename = Propane_ln.conf
Version 1
Created on 29/5/86 by jcp

UNIT NAME MODEL NAME
1 tanks storg_tnk
2 valve2 valve
3 pump3 pump
4 nrv4 nrv
5 divider5 divider
6 cntrl_v16 cv_n_bypas
7 valve7 valve
8 exchang8 ht_exchang
9 nrv9 nrv
10 trip10 trip Valve
11 blk_bld11 d_blick_bld
12 pipeline12 pipeline
13 blk_bld13 d_blick_bld
14 cntrl_v114 cv_n_bypas
15 trip15 trip Valve
16 nrv16 nrv
17 valve17 valve
18 tank18 buffer_tnk
19 valve19 valve
20 sink20 sink
21 source21 source
22 cntrl_v122 cv_n_bypas
23 exchang23 ht_exchang
24 drum24 conds_drum
25 cntrl_v125 cv_n_bypas
26 sink26 sink
27 tank27 surge_tnk
28 pump28 pump
29 nrv29 nrv
30 valve30 valve
31 tw_valve31 thr_way_vl
32 valve32 valve
33 valve33 valve
34 mixer34 mixer

<table>
<thead>
<tr>
<th>UPSTREAM</th>
<th>UNIT</th>
<th>PORT</th>
<th>UNIT</th>
<th>PORT</th>
<th>DOWNSTREAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>tank1</td>
<td>2</td>
<td>valve2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>valve2</td>
<td>2</td>
<td>pump3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>pump3</td>
<td>2</td>
<td>nrv4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>nrv4</td>
<td>2</td>
<td>divider5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>divider5</td>
<td>3</td>
<td>cntrl_v16</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>cntrl_v16</td>
<td>2</td>
<td>tank1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>divider5</td>
<td>2</td>
<td>valve7</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>valve7</td>
<td>2</td>
<td>exchang8</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

*
9 exchang8 2 nrv9 1
10 nrv9 2 trip10 1
11 trip10 2 blick_bld11 1
12 blick_bld11 2 pipeline12 1
13 pipeline12 2 blick_bld13 1
14 blick_bld13 2 cntrl_v114 1
15 cntrl_v114 2 trip15 1
16 trip15 2 nrv16 1
17 nrv16 2 valve17 1
18 valve17 2 tank18 1
19 tank18 2 valve19 1
20 valve19 2 sink20 1
21 source21 1 cntrl_v122 1
22 cntrl_v122 2 exchang23 3
23 exchang23 2 drum24 1
24 drum24 2 cntrl_v125 1
25 cntrl_v125 2 sink26 1
26 sink26 2 valve30 1
27 valve30 2 tw_valve31 1
28 tw_valve31 2 valve32 1
29 valve32 2 exchang8 3
30 exchang8 2 mixer34 1
31 mixer34 2 mixer33 1
32 mixer33 2 mixer32 1
33 mixer32 2 tank27 1

Line SOURCE DRAIN PROCESS
No. VESSEL PORT VESSEL PORT MATERIAL
1 tank1 2 divider5 1 propane
2 divider5 3 tank1 1 propane
3 divider5 2 exchang8 1 propane
4 exchang8 2 tank18 1 propane
5 tank18 2 sink26 1 propane
6 source21 1 exchang23 3 water
7 exchang23 4 drum24 1 water
8 drum24 2 sink26 1 water
9 tank27 2 exchang23 1 glycol
10 exchang23 2 tw_valve31 1 glycol
11 tw_valve31 2 exchang3 3 glycol
12 exchang8 4 mixer34 1 glycol
13 tw_valve31 3 mixer34 2 glycol
14 mixer34 3 tank27 1 glycol

Impurities

Code Name
p propane in glycol
s glycol in water

Materials of Construction Terminal Events

t1 LOxleak (major), brittle fracture
conf_filename('Propane_ln.conf').
conf_version('1').
conf_date('6/8/86').
conf_who('Jop').

unit_models([storg_tnk, valve, pumpset_b, nrv, divider, cv_c_bypas, ht_exchg_b, ht_exchg_c, trip_valve, d_blck_bld, pipeline, buffer_tnk, sink, source, conds_drum, surge_tnk, thr_way_vl, mixer]).

key(tank1, storg_tnk).
key(valve2, valve).
key(pump3, pumpset_b).
key(nrv4, nrv).
key(divider5, divider).
key(cntrl_v16, cv_c_bypas).
key(valve7, valve).
key(exchang8, ht_exchg_b).
key(nrv9, nrv).
key(trip10, trip_valve).
key(blick_bld11, d_blck_bld).
key(pipeline12, pipeline).
key(blick_bld13, d_blck_bld).
key(cntrl_v114, cv_c_bypas).
key(trip15, trip_valve).
key(nrv18, nrv).
key(valve17, valve).
key(tank18, buffer_tnk).
key(valve19, valve).
key(sink20, sink).
key(source21, source).
key(cntrl_v122, cv_n_bypas).
key(exchang23, ht_exchg_c).
key(drums24, conds_drum).
key(cntrl_v125, cv_o_bypas).
key(sink28, sink).
key(tank27, surge_tnk).
key(pump28, pumpset_b).
key(nrv29, nrv).
key(valve30, valve).
key(tw_valve31, thr_way_vl).
key(valve32, valve).
key(valve33, valve).
key(mixer34, mixer).

link(tank1, 2, valve2, 1).
link(valve2, 2, pump3, 1).
link(pump3, 2, nrv4, 1).
link(nrv4, 2, divider5, 1).
link(divider5, 3, cntrl_v18, 1).
link(cntrl_v18, 2, tank1, 1).
link(divider5, 2, valve7, 1).
link(valve7, 2, exchang8, 1).
link(exchang8, 2, nrv9, 1).
link(nrv9, 2, trip10, 1).
link(trip10, 2, blick_bld11, 1).
link(blick_bld11, 2, pipeline12, 1).
link(pipeline12, 2, blick_bld13, 1).
link(blick_bld13, 2, cntrl_v114, 1).
APPENDIX J

Unit and Process Material Models for Propane Pipeline Example
buffer_tnk

/*
Model : buffer_tnk

Buffer Tank

Propagation Equations
---------------------
\[ 13 = f(q_1, -q_2) \]
\[ p_2 = f(p_3) \]
\[ q_2 = f(13, p_3) \]
\[ x_1-2 = f(x_1-1) \]
\[ i_-2 = f(i_-1) \]

Initial Event Statements
------------------------
[...]

Terminal Event Statements
------------------------
[...]

*/

rule(buffer_tnk, p, [1,3], [[q, 1]], [[q, 2]]).
rule(buffer_tnk, p, [p, 2], [[p, 3]], []).
rule(buffer_tnk, p, [q, 2], [[1,3], [p, 3]], []).
rule(buffer_tnk, p, [[x, 1], 2], [[[x, 1], 1]], []).  
rule(buffer_tnk, p, [[i, _], 2], [[[i, _], 1]], []).  
rule(buffer_tnk, functional, [[[press ctrl fail-open', ' ', ' '], ['rv lifts', ' ', ' ']], [[[x, 1], 1]], [])].  
rule(buffer_tnk, functional, [[[sensor fail-open', ' ', ' '], ['ctrl valve fail-open', ' ', ' ']], [[[x, 1], 1]], [])].  
rule(buffer_tnk, functional, [[[press ctrl fail-open', ' ', ' ']], [[[p, 3, low]]]).  
rule(buffer_tnk, containment, [[[leak (major)', ' ', ' ']][[l, 3, low], [p, 3, low], (q, c2, low), [q, 2, none]]].  
rule(buffer_tnk, containment, [[[leak', ' ', ' ']], [[[escape', ' ', ' ']].  
rule(buffer_tnk, effect, [[[l, 3, low]]][['loss of outflow(s)', ' ', ' ']].  
rule(buffer_tnk, effect, [[[l, 3, none]]][['outflow(s) none', ' ', ' ']].  
rule(buffer_tnk, effect, [[[l, 3, high]]][['tank overfilled', ' ', ' '], ['liquid enter cs vent', ' ', ' ']]).
cond_drum

/*
 Model : cond_drum

Condensate drum

Propagation Equations
---------------------
13=f(q1,-q2)
p2=f(p3)
q2=f(13)
t3=f(t1)
t2=f(t3)
x1-2=f(x1-1)
i-2=f(i-1)

Initial Event Statements
------------------------
leak (major)>13 LO,p3 LO,q2 LO,q2 NONE
leak>escape

Terminal Event Statements
-------------------------
13 LO>loss of liquid seal
13 HI>drum overfilled
*/

rule(conds_drum,p,[1,3],[[q,1]],[[q,2]]).
rule(conds_drum,p,[p,2],[[p,3]],[]).
rule(conds_drum,p,[q,2],[[1,3]],[]).
rule(conds_drum,p,[t,3],[[t,1]],[]).
rule(conds_drum,p,[t,2],[[t,3]],[]).
rule(conds_drum,p,[[x,1],2],[[[x,1],1]],[]).
rule(conds_drum,p,[[i,-],2],[[[i,-],1]],[]).

rule(conds_drum,containment,[[['leak (major)'',' ',' ']],[[1,3,low],[p,3,low],[q,
c2,low],[q,2,none]]).
rule(conds_drum,containment,[[['leak','','']],[['escape','','']]]).
rule(conds_drum,effect,[[1,3,low]],[['loss of liquid seal','','']]).
rule(conds_drum,effect,[[1,3,high]],[['drum overfilled','','']]).
Model : cv_o_bypass

Control loop (control valve, controller, sensor) and bypass

Propagation Equations

\[ p_2 = f(p_1) \]
\[ q_2 = f(q_1) \]
\[ t_2 = f(t_1) \]
\[ x_{i-2} = f(x_{i-1}) \]
\[ i_{-2} = f(i_{-1}) \]

Initial Event Statements

1. Loop fails closed, isolation valve closed > \( q_2 \) NONE, \( p_1 \) HI
2. Loop fails pt-clsd, isolation valve pt-clsd > \( q_2 \) LO
3. Loop fails open, bypass fails, bypass directed open > \( q_2 \) HI
4. Sensor fails closed, cntrlr fails closed, ctl-vl fails closed, set-pnt mvd closed > loop fails closed
6. Sensor fails open, cntrlr fails open, ctl-vl fails open, set-pnt mvd open > loop fai cls open
7. Blockage > \( q_2 \) LO
8. Leak > escape

Terminal Event Statements

/*

rule(cv_o_bypass, p, [p, 2], [[p, 1]], []).
rule(cv_o_bypass, p, [q, 2], [[q, 1]], []).
rule(cv_o_bypass, p, [t, 2], [[t, 1]], []).
rule(cv_c_bypas, p, [[x, 1], 2], [[x, 1], 1], []). 

rule(cv_c_bypas, p, [[i, _, 1], 2], [[i, _, 1], 1], []). 

rule(cv_c_bypas, functional, [[loop fails closed', '', ''], ['isolation valve closed', '', ''], [[q, 2], none], [p, 1, high]]. 

rule(cv_c_bypas, functional, [[loop fails pt-clsd', '', ''], ['isolation valve pt-clsd', '', ''], [[q, 2], low]]. 

rule(cv_c_bypas, functional, [[loop fails open', '', ''], ['bypass fails', '', ''], ['bypass directed open', '', ''], [[q, 2], high]]. 

rule(cv_c_bypas, functional, [[sensor fails closed', '', ''], ['controller fails closed', '', ''], ['ctl-vl fails closed', '', ''], ['set-pnt mvd closed', '', ''], [[loop fails closed', '', ']]]. 


rule(cv_c_bypas, functional, [[sensor fails open', '', ''], ['controller fails open', '', ''], ['ctl-vl fails open', '', ''], ['set-pnt mvd open', '', ''], [[loop fails open', '', ']]]. 

rule(cv_c_bypas, hydraulic, ['blockage', '', ''], [[q, 2], low]]. 

rule(cv_c_bypas, containment, ['leak', '', ''], ['escape', '', ']]).
d_blck_bld

/*
 Model : d_blck_bld

 Double block and bleed

 Propagation Equations
 ---------------------
 . p2=f(p1)
   q2=f(q1)
   t2=f(t1)
   x1-2=f(x1-1)
   i_-2=f(i_-1)

 Initial Event Statements
 ------------------------
 isolatn vl clsd>q2 NONE, p1 HI
 isolatn vl pt. clsd>q2 LO, p1 HI
 leak>escape
 leak (major), drain valve open>q2 LO, q2 NONE, p2 LO

 Terminal Event Statements
 -------------------------
 */

 rule(d_blck_bld, p, [p, 2], [p, 1], []).  
 rule(d_blck_bld, p, [q, 2], [q, 1], []).  
 rule(d_blck_bld, p, [t, 2], [t, 1], []).  
 rule(d_blck_bld, p, [x, 1], 2, [[x, 1], 1], []).  
 rule(d_blck_bld, p, [i, _], 2, [[i, _], 1], []).  
 rule(d_blck_bld, functional, [['isolatn vl clsd', '' ', '']], [q, 2, none], [p, 1, high]).  
 rule(d_blck_bld, functional, [['isolatn vl pt. clsd', '' ', '']], [q, 2, low], [p, 1, high]).  
 rule(d_blck_bld, containment, [['leak', '' ', '']], [['escape', '' ', ']]).  
 rule(d_blck_bld, containment, [['leak (major)', '' ', ']], ['drain valve open', '' ', ']], [q, 2, low], [q, 2, none], [p, 2, low])
Binary divider - no flow distribution specified

Propagation Equations
---------------------
\[ p_2 = f(p_1) \]
\[ p_3 = f(p_1) \]
\[ q_2 = f(q_1) \]
\[ q_3 = f(q_1) \]
\[ t_2 = f(t_1) \]
\[ t_3 = f(t_1) \]
\[ x_{l-2} = f(x_{l-1}) \]
\[ x_{l-3} = f(x_{l-1}) \]
\[ i_{-2} = f(i_{-1}) \]
\[ i_{-3} = f(i_{-1}) \]

Initial Event Statements
------------------------
blockage > p_2 HI, p_3 HI
blockage, leak (major) > q_2 LO, q_3 LO, q_2 NONE, q_3 NONE
blockage - other leg > q_2 HI, q_3 HI
leak > escape

Terminal Event Statements
-------------------------
*/

rule(divider, p, [p, 2], [[p, 1]], []).
rule(divider, p, [p, 3], [[p, 1]], []).
rule(divider, p, [q, 2], [[q, 1]], []).
rule(divider, p, [q, 3], [[q, 1]], []).
rule(divider, p, [[t, 2], [[t, 1]], []]).
rule(divider, p, [[t, 3], [[t, 1]], []]).
rule(divider, p, [[x, 1], 2], [[[x, 1], 1]], []).
rule(divider, p, [[x, 1], 3], [[[x, 1], 1]], []).
rule(divider, p, [[i, _], 2], [[[i, _], 1]], []).
rule(divider, p, [[i, _], 3], [[[i, _], 1]], []).
rule(divider, hydraulic, [["blockage", ",", ","], [[p, 2, high], [p, 3, high]]).
rule(divider, hydraulic, [["blockage", ",", ","], ["leak (major)", ",", ","], [[q, 2, low], [q, 3, low], [q, 2, none], [q, 3, none]]).
rule(divider, hydraulic, [["blockage - other leg", ",", ","], [[q, 2, high], [q, 3, high]]).
rule(divider, containment, [["leak", ",", ","], [["escape", ",", ","]]).
material_cause(glycol, blockage, []).
material_cause(glycol, impurity, []).
material_cause(glycol, cavitation, []).
material_effect(glycol, escape, _, []). 
material_effect(glycol, 'concentration change', _, []). 
material_effect(glycol, t, high, []). 
material_effect(glycol, t, low, []). 
material_effect(glycol, p, high, []). 
material_effect(glycol, p, low, []).
ht_exchg_b

/*
Model : ht_exchg_b

Heat Exchanger - Shell to tube transfer, tube side higher pressure

Propagation Equations
---------------------
p2=f(p1)
q2=f(q1)
t2=f(t1,t3,q3,-q1)
x1-2=f(x1-1)
i_2=f(i_-1)
p4=f(p3)
q4=f(q3)
t4=f(t1,t3,q3,-q1)
x3-4=f(x3-3)
i_-4=f(i_-3)

Initial Event Statements
------------------------
tube fouling\(>t2\) LO, t4 HI, q2 LO, p1 HI
shell leak (major)>q4 NONE, q4 LO, p4 LO

tube leak (major)>q2 LO, p2 LO, p4 HI

Terminal Event Statements
-------------------------
\(i-4\) HI\(>p4\) HI, t4 LO

*/

rule(ht_exchg_b,p,[p,2],[[p,1],[[]]]).
rule(ht_exchg_b,p,[q,2],[[q,1],[[]]].
rule(ht_exchg_b, p, [t, 2], [[t, 1], [t, 3], [q, 3]], [[q, 1]]).
rule(ht_exchg_b, p, [[x, 1], 2], [[[x, 1], 1]], []).
rule(ht_exchg_b, p, [[i, _], 2], [[[i, _], 1]], []).
rule(ht_exchg_b, p, [p, 4], [[p, 3]], []).
rule(ht_exchg_b, p, [q, 4], [[q, 3]], []).
rule(ht_exchg_b, p, [t, 4], [[t, 1], [t, 3], [q, 3]], [[q, 1]]).
rule(ht_exchg_b, p, [[x, 3], 4], [[[x, 3], 3]], []).
rule(ht_exchg_b, p, [[i, _], 4], [[[i, _], 3]], []).
rule(ht_exchg_b, functional, [['tube fouling', ' ', ' ']], [[t, 2, low], [t, 4, high], [q, \c2, low], [p, 1, high]]).
rule(ht_exchg_b, containment, [['shell leak (major)', ' ', ' ']], [[q, 4, none], [q, 4, low \cow], [p, 4, low]]).
rule(ht_exchg_b, containment, [['tube leak (major)', ' ', ' ']], [[t, 2, low], [p, 2, low \cow], [p, 4, high]]).
rule(ht_exchg_b, containment, [['tube leak', ' ', ' ']], [[[i, p], 4, high]]).
rule(ht_exchg_b, containment, [['leak', ' ', ' ']], [['escape', ' ', ' ']])
rule(ht_exchg_b, effect, [[[i, p], 4, high]], [[p, 4, high], [t, 4, low]]).
ht_exchg_c

/*
Model : ht_exchg_c

Heat Exchanger - shell to tube transfer, condensing steam

Propagation Equations
----------------------
p2=f(p1)
q2=f(q1)
t2=f(t1,t3,q3,-q1)
x1-2=f(x1-1)
i_-2=f(1_-1)
p4=f(p3)
q4=f(q3)
t4=f(t1,t3,q3,-q1)
x3-4=f(x3-3)
i_-4=f(i_-3)

Initial Event Statements
------------------------
tube fouling>t2 LO,t4 HI,q2 LO,p1 HI
shell leak (major)>q4 NONE,q4 LO,p4 LO
tube leak (major)>q2 LO,p2 LO,p4 HI
tube leak>1g-4 HI
leak>escape

Terminal Event Statements
-------------------------
*/

rule(ht_exchg_c,p,[p,2],[[p,1],[[p,1]]]).
rule(ht_exchg_c,p,[q,2],[[q,1],[[q,1]]]).
rule(ht_exchg_c,p,[t,2],[[t,1],[t,3],[q,3],[[q,1]]]).
rule(ht_exchg_c, p, [[x, 1], 2], [[[x, 1], 1]], []).
rule(ht_exchg_c, p, [[i, _], 2], [[[i, _], 1]], []).
rule(ht_exchg_c, p, [p, 4], [[p, 3]], []). 
rule(ht_exchg_c, p, [q, 4], [[q, 3]], []). 
rule(ht_exchg_c, p, [t, 4], [[t, 1], [t, 3], [q, 3]], [[q, 1]]).
rule(ht_exchg_c, p, [[x, 3], 4], [[[x, 3], 3]], []). 
rule(ht_exchg_c, p, [[i, _], 4], [[[i, _], 3]], []). 
rule(ht_exchg_c, functional, [['tube fouling', ' ', ' ']], [[t, 2, low], [t, 4, high], [q, \c2, low], [p, 1, high]]).
rule(ht_exchg_c, containment, [['shell leak (major)', ' ', ' ']], [[q, 4, none], [q, 4, 1 \cow], [p, 4, low]]).
rule(ht_exchg_c, containment, [['tube leak (major)', ' ', ' ']], [[q, 2, low], [p, 2, low \c], [p, 4, high]]).
rule(ht_exchg_c, containment, [['tube leak', ' ', ' ']], [[i, g, 4, high]]).
rule(ht_exchg_c, containment, [['leak', ' ', ' ']], [['escape', ' ', ' ']]).
mixer

/*
Model : mixer

Binary mixer - conc rules assume pure fluids, no flow ratio assumed

Propagation Equations
-----------------------

\[ p_3 = f(p_1, p_2) \]
\[ q_3 = f(q_1, q_2) \]
\[ t_3 = f(t_1, t_2) \]
\[ x_{1-3} = f(x_{1-1}, q_1, -q_2) \]
\[ x_{2-3} = f(x_{2-2}, q_2, -q_1) \]
\[ i_{-3} = f(i_{-1}, i_{-2}) \]

Initial Event Statements
------------------------

leak > escape

Terminal Event Statements
-------------------------

rule(mixer, p, [p, 3], [[p, 1], [p, 2]], []).
rule(mixer, p, [q, 3], [[q, 1], [q, 2]], []).
rule(mixer, p, [t, 3], [[t, 1], [t, 2]], []).
rule(mixer, p, [[x, 1], 3], [[[x, 1], 1], [q, 1]], [[q, 2]]).
rule(mixer, p, [[x, 2], 3], [[[x, 2], 2], [q, 2]], [[q, 1]]).
rule(mixer, p, [[i, _], 3], [[[i, _], 1], [[i, _], 2]], []).
rule(mixer, containment, [['leak', ' ', ' '], [['escape', ' ', ' ']]).
/*
Model : nrv

Non-return valve

Propagation Equations
---------------------
p2=f(p1)
q2=f(q1)
t2=f(t1)
x1-2=f(x1-1)
i_-2=f(i_-1)

Initial Event Statements
------------------------
valve stuck clsd>q2 NONE
valve fails pt. clsd>q2 LO
leak>escape

Terminal Event Statements
-------------------------
*/

rule(nrv,p,[[p,2],[[p,1]],[]]).
rule(nrv,p,[[q,2],[[q,1]],[]]).
rule(nrv,p,[[t,2],[[t,1]],[]]).
rule(nrv,p,[[x,1],2],[[[x,1],1]],[]).
rule(nrv,p,[[i,1],2],[[[i,1],1]],[]).
rule(nrv,functional,[[valve stuck clsd, ' ', ' ']],[[q,2,none]]).
rule(nrv,functional,[[valve fails pt. clsd, ' ', ' ']],[[q,2,low]]).
rule(nrv,containment,[[leak, ' ', ' ']],[['escape', ' ', ' ']]).
pipe

/*
Model : pipe

Generic Pipe Model

Propagation Equations
----------------------
p2=f(p1)
q2=f(q1)
t2=f(t1)
x1-2=f(x1-1)
i-2=f(i-1)

Initial Event Statements
------------------------
blockage, leak (major)>q2 LO, q2 NONE
blockage, ht to blckd-in fluid>p1 HI
leak (major)>p2 LO
leak>escape
solar heat, fire>ht to blckd-in fluid
external cold source>t2 LO
external heat source>t2 HI
cold weather, lagging loss>external cold source
hot weather, fire>external heat source

Terminal Event Statements
-------------------------
p1 HI>leak (major), pipe rupture, flange leak
*/
rule(pipe,p,[p,2],[[p,1]],[]).
rule(pipe,p,[q,2],[[q,1]],[]).
rule(pipe,p,[t,2],[[t,1]],[]).
rule(pipe, p, [[x, 1], 2], [[[x, 1], 1]], []).  
rule(pipe, p, [[i, _], 2], [[[i, _], 1]], []).  
rule(pipe, hydraulic, [['blockage', ' ', ' '], ['leak (major)', ' ', ' ']], [[q, 2, low], c[q, 2, none]]).  
rule(pipe, hydraulic, [['blockage', ' ', ' '], ['ht to block-in fluid', ' ', ' ']], [[p \ c, 1, high]]).  
rule(pipe, containment, [['leak (major)', ' ', ' ']], [[p, 2, low]]).  
rule(pipe, containment, [['leak', ' ', ' ']], [['escape', ' ', ' ']]).  
rule(pipe, environmental, [['solar heat', ' ', ' '], ['fire', ' ', ' ']], [['ht to block \ cd-in fluid', ' ', ' ']]).  
rule(pipe, environmental, [['external cold source', ' ', ' ']], [[t, 2, low]]).  
rule(pipe, environmental, [['external heat source', ' ', ' ']], [[t, 2, high]]).  
rule(pipe, environmental, [['cold weather', ' ', ' '], ['lagging loss', ' ', ' ']], [['external \ heat source', ' ', ' ']]).  
rule(pipe, environmental, [['hot weather', ' ', ' '], ['fire', ' ', ' ']], [['external \ heat source', ' ', ' ']]).  
rule(pipe, effect, [[p, 1, high]], [['leak (major)', ' ', ' '], ['pipe rupture', ' ', ' '], [\ c], ['flange leak', ' ', ' ']]).
pipeline

/**
 Model : pipeline

 Pipeline

 Propagation Equations
 ---------------------
 p2=f(p1)
 q2=f(q1)
 t2=f(t1)
 x1-2=f(x1-1)
 i_-2=f(i_-1)

 Initial Event Statements
 ------------------------
 blockage, leak (major)>q2 NONE, q2 LO
 blockage, hammer/surge, ht to blckd-in fluid>p2 HI
 leak>escape
 solar heat, fire>ht to blckd-in fluid

 Terminal Event Statements
 -------------------------
 */

 rule(pipeline,p,[[p,2]],[[p,1]],[]).
 rule(pipeline,p,[[q,2]],[[q,1]],[]).
 rule(pipeline,p,[[t,2]],[[t,1]],[]).
 rule(pipeline,p,[[x,1],[x,1]],[]).
 rule(pipeline,p,[[i,1],[i,1]],[]).
 rule(pipeline,functional,[['blockage', ' '],['leak (major)', ' '],[[q,2, 
none],[q,2,low]]).
 rule(pipeline,hydraulic,[['blockage', ' '],['hammer/surge', ' '],['ht to 
blckd-in fluid', ' '],[[p,2,high]]).
 rule(pipeline,containment,[['leak', ' '],['escape', ' ']]).
 rule(pipeline,environmental,[['solar heat', ' '],['fire', ' ']],['ht to 
blckd-in fluid', ' '])


material_cause(propane, blockage, []).  
material_cause(propane, cavitation, ['pressure low', '', ''], ['temperature high °C', ',', ', ']).  
material_effect(propane, escape, _, ['fire hazard', '', ''], ['explosion hazard', '°C', ',', ', ']).  
material_effect(propane, t, high, ['vapour formation', '', '']).  
material_effect(propane, t, low, []).  
material_effect(propane, p, high, []).  
material_effect(propane, p, low, ['liquid flashes', '', '\']).
Model: pumpset_b

Pumpset – Two modes 1) one running/one on standby 2) both running

Propagation Equations

\[ p_2 = f(p_1) \]
\[ q_2 = f(q_1) \]
\[ t_2 = f(t_1) \]
\[ x_{1-2} = f(x_{1-1}) \]
\[ i_{-2} = f(i_{-1}) \]

Initial Event Statements

- Loss of NPSH, rotation fault, impeller fault, cavitation > q2 LOW, q2 NONE
- Delivery shut in > q2 NONE, p2 HIGH
- Delivery part shut > q2 LOW
- No change to standby, standby fail on demand, maloperation of valves > q2 NONE
- Only 1 on when 2 reqd > q2 LOW
- No rseat of dlvry rv > q2 NONE
- Blockage > q2 NONE, q2 LOW, p2 HIGH
- Pump fluid overheats > t2 HIGH
- Leak (major) > q2 NONE, q2 LOW
- Leak > escape
- Heat to blckd-in fluid > p2 HIGH
- Solar heat, fire > heat to block-in fluid

Terminal Event Statements

- q2 NONE > pump fluid overheats

*/
rule(pumpset_b, p, [p, 2], [[p, 1]], []).  
rule(pumpset_b, p, [q, 2], [[q, 1]], []).  
rule(pumpset_b, p, [t, 2], [[t, 1]], []).  
rule(pumpset_b, p, [[x, 1], 2], [[[x, 1]], 1]], []).  
rule(pumpset_b, p, [[[x, _], 2], [[[x, _]], 1]], []).  
rule(pumpset_b, hydraulic, [['loss of NPSH', ''], ['rotation fault', ''], ['impeller fault', ''], ['cavitation', ''], ['low'], [[q, 2], [q, 2, none]]).  
rule(pumpset_b, hydraulic, [['delivery shut in', ''], ['delivery part shut', ''], ['delivery part shut', ''], [[q, 2], [q, 2, low]]).  
rule(pumpset_b, hydraulic, [['only 1 on whn 2 reqd', ''], ['maloperatn of valves', ''], ['maloperatn of valves', ''], [[q, 2], [q, 2, none]]).  
rule(pumpset_b, hydraulic, [['blockage', ''], ['blockage', ''], [[q, 2], [q, 2, none], [q, 2, low], [p, 2, high]]).  
rule(pumpset_b, hydraulic, [['pump fluid overheats', ''], ['pump fluid overheats', ''], [[t, 2], [t, 2, high]]).  
rule(pumpset_b, containment, [['leak (major)', ''], ['leak (major)', ''], [[q, 2], [q, 2, none], [q, 2, low]]).  
rule(pumpset_b, containment, [['leak', ''], ['leak', ''], [['escape', ''], ['escape', ''], [[p, 2], [p, 2, high]]].  
rule(pumpset_b, environmental, [['ht to blckd-in fluid', ''], ['ht to blckd-in fluid', ''], [[p, 2], [p, 2, high]]).  
rule(pumpset_b, environmental, [['solar heat', ''], ['solar heat', ''], [['fire', ''], ['fire', ''], [[p, 2], [p, 2, high]]]).  
rule(pumpset_b, effect, [[q, 2], [q, 2, none], [['pump fluid overheats', ''], ['pump fluid overheats', '']]).
sink

/*
Model : sink
*/

Sink - one inlet

Propagation Equations
-----------------------

Initial Event Statements
-------------------------
sink full>p1 HI, q1 LO, q1 NONE

Terminal Event Statements
--------------------------

*/

rule(sink, functional, [['sink full', ' ', ' ']], [[p, 1, high], [q, 1, low], [q, 1, none]])
\c.
Model : source

Source - one outlet

Propagation Equations
---------------------

Initial Event Statements
------------------------
source empty>pl LO, q1 NONE

Terminal Event Statements
-------------------------

rule(source, functional, [['source empty', ' ', ' ']], [[p, 1, low], [q, 1, none]]).
storg_tnk

/*
Model : storg_tnk

Storage Tank - one outlet (2) and kickback port (1)

Propagation Equations
---------------------
13=f(-q2)
p2=f(13,p3)
q2=f(13)
t2=f(t3)
x3-2=f(x3-3)
i-2=f(i-3)

Initial Event Statements
------------------------
tank overfilled>13 HI
tank nearing empty>13 LO
drain open>13 LO, escape
leak>escape

Terminal Event Statements
-------------------------
13 HI>tank overflows
13 LO>loss of outflow(s), q2 NONE, q2 LO
*/

rule(storg_tnk, p, [1,3], [], [[q,2]]).
rule(storg_tnk, p, [p,2], [[1,3],[p,3]], []).
rule(storg_tnk, p, [q,2], [[1,3]], []).
rule(storg_tnk, p, [t,2], [[t,3]], []).
rule(storg_tnk, p, [[x,3],2], [[[x,3],3]], []).
rule(storg_tnk, p, [[i,1],2], [[[i,1],3]], []).
rule(storg_tnk, functional, [['tank overfilled', ', ', '']], [[1,3, high]]).
rule(storg_tnk, functional, [['tank nearing empty', ', ', '']], [[1,3, low]]).
rule(storg_tnk, containment, [['drain open', ', ', '']], [[1,3, low], ['escape', ', ', '\c']]).
rule(storg_tnk, containment, [['leak', ', ', '']], [['escape', ', ', '']]).
rule(storg_tnk, effect, [[1,3, high]], [['tank overflows', ', ', ']]).
rule(storg_tnk, effect, [[1,3, low]], [['loss of outflow(s)', ', ', ']], [q, 2, none], [q \c, 2, low]].
surge_tnk

/*
Model : surge_tnk

Surge Tank - one inlet and one outlet

Propagation Equations
---------------------
\[ l_3 = f(q_1, -q_2) \]
\[ p_2 = f(l_3, p_3) \]

Initial Event Statements
------------------------
strt up level insuff\(>l_3\) LO
leak>escape

Terminal Event Statements
-------------------------
rule(surge_tnk, p, \([1,3]\), \([[q,1]], [[q,2]]\))
rule(surge_tnk, p, \([2]\), \([[1,3],[p,3]]\), \([\]\))
rule(surge_tnk, functional, \([\text{'strt up level insuff'},',',']\), \([1,3,\text{low}]\))
rule(surge_tnk, containment, \([\text{'leak'},',',']\), \([\text{'escape'},',',']\)).
Model: thr_way_v1

Three way valve

Propagation Equations

\[ q_2 = f(q_1, -q_3) \]
\[ q_3 = f(q_1, -q_2) \]
\[ p_2 = f(p_1) \]
\[ p_3 = f(p_1) \]
\[ t_2 = f(t_1) \]
\[ t_3 = f(t_1) \]

Initial Event Statements

stuck to other port, set point error>q_2 LO, q_2 NONE, q_3 LO, q_3 NONE
stuck to this port, set point error>q_2 HI, q_3 HI
leak>escape

Terminal Event Statements

```c
rule(thr_way_v1, p, [q, 2], [[q, 1]], C[q, 3]).
rule(thr_way_v1, p, [q, 3], [[q, 1]], [q, 2]).
rule(thr_way_v1, p, [p, 2], [[p, 1]], []).
rule(thr_way_v1, p, [p, 3], [[p, 1]], []).
rule(thr_way_v1, p, [t, 2], [[t, 1]], []).
rule(thr_way_v1, p, [t, 3], [[t, 1]], []).
rule(thr_way_v1, functional, [['stuck to other port', ''], ['set point error', ''], [[q, 2, low], [q, 2, none], [q, 3, low], [q, 3, none]]).
rule(thr_way_v1, functional, [['stuck to this port', ''], ['set point error', ''], [[q, 2, high], [q, 3, high]]).
rule(thr_way_v1, containment, [['leak', ''], ['escape', '']]).
```
trip_valve

/*

Model : trip_valve

Trip valve

Propagation Equations

\[
\begin{align*}
    p_2 &= f(p_1) \\
    q_2 &= f(q_1) \\
    t_2 &= f(t_1) \\
    x_{i-2} &= f(x_{i-1}) \\
    i_{-2} &= f(i_{-1})
\end{align*}
\]

Initial Event Statements

spurious closure > q_2 NONE
blockage > q_2 LO, q_2 NONE, p_1 HI
leak > escape

Terminal Event Statements

rule(trip_valve, p, [p, 2], [C, p, 1], []).
rule(trip_valve, p, [q, 2], [q, 1], []).
rule(trip_valve, p, [t, 2], [t, 1], []).
rule(trip_valve, p, [[x, 1], 2], [[x, 1], 1], []).
rule(trip_valve, p, [[i, _], 2], [[i, _], 1], []).
rule(trip_valve, functional, ['spurious closure', '', ''], [[q, 2, none]]).
rule(trip_valve, functional, ['blockage', '', ''], [[q, 2, low], [q, 2, none], [p, 1, hi \cgh]]).
rule(trip_valve, containment, ['leak', '', ''], ['escape', '', ''])..
valve

/*
Model : valve
Isolation valve (normally state open)

Propagation Equations
---------------------
p2=f(p1)
q2=f(q1)
t2=f(t1)
x1-2=f(x1-1)
i_2=f(i_1)

Initial Event Statements
------------------------
valve directed clsd>q2 NONE, p1 HI
valve partly clsd>q2 LO
leak>escape

Terminal Event Statements
-------------------------
*/

rule(valve, p, [p, 2], [[p, 1]], []).  
rule(valve, p, [q, 2], [[q, 1]], []).  
rule(valve, p, [t, 2], [[t, 1]], []).  
rule(valve, p, [[x, 1], 2], [[[x, 1], 1]], []).  
rule(valve, p, [[i, _], 2], [[[i, _], 1]], []).  
rule(valve, functional, ['valve directed clsd', ',', ''], [[q, 2, none], [p, 1, high]]).  
rule(valve, functional, ['valve partly clsd', ',', ''], [[q, 2, low]]).  
rule(valve, containment, ['leak', ',', ''], ['escape', ',', '']).
water

material_cause(water, blockage, []).  
material_cause(water, cavitation, []).  
material_effect(water, escape, ' ', []).  
material_effect(water, t, high, []).  
material_effect(water, t, low, ["formation of ice", ",", ","]).  
material_effect(water, p, high, []).  
material_effect(water, p, low, []).
APPENDIX K

"compacter" Output for Propane Pipeline Example
SEARCH ON VESSEL: TANK1

CAUSES

\( p \) low

CONSEQUENCES

\( l \) 3 low
\( p \) 3 low

SEARCH ON VESSEL: TANK1

CAUSES

\( p \) high

CONSEQUENCES

\( l \) 3 high
\( p \) 3 high

SEARCH ON VESSEL: TANK1

CAUSES

\( q \) 2 high

CONSEQUENCES

Tank nearing empty
Gran open

Loss of outflow(s)
\( q \) 2 none

SEARCH ON VESSEL: TANK1

CAUSES

\( l \) low

CONSEQUENCES

\( q \) 2 low

SEARCH ON VESSEL: TANK1

CAUSES

\( q \) 2 high

CONSEQUENCES

\( q \) 2 low

Tank overflows

SEARCH ON VESSEL: DIVIDERS

CAUSES

\( p \) low

CONSEQUENCES

\( p \) 1 low

\( p \) 2 low
Search on vessel 1: divider5

CAUSES

p 1 high
blockage

CONSEQUENCES

q 2 high
q 3 high

Search on vessel 1: exchanger

CAUSES

p 1 low

CONSEQUENCES

p 2 low
p 4 low

shell leak (major)
tube leak (major)

Search on vessel 1: exchanger

CAUSES

p high

CONSEQUENCES

p 2 high
p 4 high

tube fouling
tube leak (major)

Search on vessel 1: tank18

CAUSES

p 3 low

dress ctrl fall-open
sensor fall-open
cvrtr fall-open
ctrl valve fall-open
set at wvd open
rv lifts
leak (major)

CONSEQUENCES

q 2 low

Search on vessel 1: tank18
### CAUSES

- p 3 high  
- press ctrl fall-c1sd  
- sensor fall-c1sd  
- cntrr fall-c1sd  
- ctrl valve fall-c1sd  
- set at eva c1sd

### CONSEQUENCES

- a 2 high

---

### Search on vessel: tank18

#### 1. Low

### CAUSES

- a 1 low
- a 2 high

### CONSEQUENCES

- a 2 low

---

### Search on vessel: tank18

#### 2. High

### CAUSES

- a 1 high
- a 2 low
- a 2 none

### CONSEQUENCES

- a 2 high
- tank overfilled
- liquid enters vent

---

### Search on vessel: source21

#### 1. Low

### CAUSES

- source empty

### CONSEQUENCES

---

### Search on vessel: exchanger23

#### 1. Low

### CAUSES

- p 1 low
- p 3 low
- shell leak (major)
- tuec leak (major)

### CONSEQUENCES

- a 2 low
- a 4 low
<table>
<thead>
<tr>
<th>Search on vessel: exchanger23</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>p</strong> high</td>
<td></td>
</tr>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>p 1 high</td>
<td>p 2 high</td>
</tr>
<tr>
<td>tube fouling</td>
<td>p 4 high</td>
</tr>
<tr>
<td>p 3 high</td>
<td></td>
</tr>
<tr>
<td>tube leak (major)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on vessel: drum24</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>p</strong> low</td>
<td></td>
</tr>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>p 3 low</td>
<td>p 2 low</td>
</tr>
<tr>
<td>leak (major)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on vessel: drum24</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>p</strong> high</td>
<td></td>
</tr>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>p 3 high</td>
<td>p 2 high</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on vessel: drum24</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>l</strong> low</td>
<td></td>
</tr>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>q 1 low</td>
<td>p 2 low</td>
</tr>
<tr>
<td>q 2 high</td>
<td>loss of liquid seal</td>
</tr>
<tr>
<td>leak (major)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Search on vessel: drum24</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>l</strong> high</td>
<td></td>
</tr>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>q 1 high</td>
<td>p 2 high</td>
</tr>
<tr>
<td>q 2 low</td>
<td>drum overfilled</td>
</tr>
<tr>
<td>q 2 none</td>
<td></td>
</tr>
</tbody>
</table>

| Search on vessel: tank27      |  |
o low
CAUSES
l 3 low
p 3 low

CONSEQUENCES
o 2 low

Search on vessel: tank27
p high
CAUSES
l 3 high
p 3 high

CONSEQUENCES
o 2 high

Search on vessel: tank27
l low
CAUSES
q 1 low
q 2 high
stir up level insuff

CONSEQUENCES
o 2 low

Search on vessel: tank27
l high
CAUSES
q 1 high
q 2 low
q 2 none

CONSEQUENCES
o 2 high

Search on vessel: tw_valve31
p low
CAUSES
p 1 low

CONSEQUENCES
o 2 low
o 3 low

Search on vessel: tw_valve31
p high
CAUSES

CONSEQUENCES
<table>
<thead>
<tr>
<th>Q1 high</th>
<th>Q2 high</th>
<th>Q3 high</th>
</tr>
</thead>
</table>

**Search on vessel: mixer36**

<table>
<thead>
<tr>
<th>Q1 low</th>
<th>Q2 low</th>
</tr>
</thead>
</table>

**CAUSES**

<table>
<thead>
<tr>
<th>Q1 low</th>
<th>Q2 low</th>
</tr>
</thead>
</table>

**CONSEQUENCES**

<table>
<thead>
<tr>
<th>Q3 low</th>
</tr>
</thead>
</table>

**Search on vessel: mixer36**

<table>
<thead>
<tr>
<th>Q3 high</th>
</tr>
</thead>
</table>

**CAUSES**

<table>
<thead>
<tr>
<th>Q1 high</th>
<th>Q2 high</th>
</tr>
</thead>
</table>

**CONSEQUENCES**

<table>
<thead>
<tr>
<th>Q3 high</th>
</tr>
</thead>
</table>

**Search on vessel: sink20**

<table>
<thead>
<tr>
<th>Q3 high</th>
</tr>
</thead>
</table>

**CAUSES**

<table>
<thead>
<tr>
<th>sink full</th>
</tr>
</thead>
</table>

**CONSEQUENCES**

<table>
<thead>
<tr>
<th>sink full</th>
</tr>
</thead>
</table>

**Search on vessel: sink26**

<table>
<thead>
<tr>
<th>Q3 high</th>
</tr>
</thead>
</table>

**CAUSES**

<table>
<thead>
<tr>
<th>sink full</th>
</tr>
</thead>
</table>

**CONSEQUENCES**

<table>
<thead>
<tr>
<th>sink full</th>
</tr>
</thead>
</table>

**Search on line 1**

<table>
<thead>
<tr>
<th>Q1 none</th>
</tr>
</thead>
</table>

**CAUSES**

<table>
<thead>
<tr>
<th>tank1</th>
<th>2 l 3 none</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 q 2 none</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>doe</th>
<th>2 blockage</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 leak (major)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>tank1</td>
</tr>
<tr>
<td>qwo3</td>
</tr>
<tr>
<td>divider5</td>
</tr>
<tr>
<td>1 q 2 none</td>
</tr>
<tr>
<td>Valve 2</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td>pino3</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>nruv6</td>
</tr>
</tbody>
</table>

**Search on line 1**

<table>
<thead>
<tr>
<th>q</th>
<th>low</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td></td>
</tr>
<tr>
<td>tank1</td>
<td>2 l 3 low</td>
</tr>
<tr>
<td></td>
<td>2 q 2 low</td>
</tr>
<tr>
<td>pipe</td>
<td>2 blockage</td>
</tr>
<tr>
<td></td>
<td>leak (major)</td>
</tr>
<tr>
<td>valve2</td>
<td>2 valve partly closed</td>
</tr>
<tr>
<td>pino3</td>
<td>2 loss of NPSH</td>
</tr>
<tr>
<td></td>
<td>2 rotation fault</td>
</tr>
<tr>
<td></td>
<td>2 impeller fault</td>
</tr>
<tr>
<td></td>
<td>2 cavitation</td>
</tr>
<tr>
<td></td>
<td>pressure low</td>
</tr>
<tr>
<td></td>
<td>temperature high</td>
</tr>
<tr>
<td></td>
<td>delivery part shut</td>
</tr>
<tr>
<td></td>
<td>only 1 on when 2 read</td>
</tr>
<tr>
<td></td>
<td>blockage</td>
</tr>
<tr>
<td></td>
<td>leak (major)</td>
</tr>
<tr>
<td>nruv6</td>
<td>2 valve fails at, closed</td>
</tr>
</tbody>
</table>

**Search on line 1**

<table>
<thead>
<tr>
<th>q</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td></td>
</tr>
<tr>
<td>tank1</td>
<td>2 l 3 high</td>
</tr>
<tr>
<td></td>
<td>2 q 2 high</td>
</tr>
<tr>
<td>divider5</td>
<td>1 q 1 high</td>
</tr>
<tr>
<td></td>
<td>1 q 2 high</td>
</tr>
<tr>
<td></td>
<td>1 q 3 high</td>
</tr>
</tbody>
</table>
### Search on Line 1

**p**  
**low**

**CAUSES**
- tank1  
  - 2 l 3 low  
  - 2 o 3 low  
  - 2 o 2 low  
- ploe  
  - 2 leak (major)

**CONSEQUENCES**
- divider5  
  - 1 o 1 low  
- propane  
  - liquid flashes

### Search on Line 1

**p**  
**high**

**CAUSES**
- tank1  
  - 2 l 3 high  
  - 2 o 3 high  
  - 2 o 2 high  
- pipe  
  - 1 blockage  
  - 1 hot to block-in fluid  
  - solar heat  
  - fire  
- valve2  
  - 1 valve directed clsd  
- pump3  
  - 2 delivery snout in  
  - 2 blockage  
  - 2 hot to block-in fluid  
  - solar heat  
  - fire

**CONSEQUENCES**
- ploe  
  - 1 leak (major)  
  - 1 ploe rupture  
  - 1 flange leak

### Search on Line 1

**t**  
**low**

**CAUSES**
- tank1  
  - 2 t 3 low  
  - 2 t 2 low  
- ploe  
  - 2 external cold source  
  - cold weather  
  - lagging loss

**CONSEQUENCES**
- ploe  
  - 1 leak (major)  
  - 1 brittle fracture

### Search on Line 1
<table>
<thead>
<tr>
<th></th>
<th>t high</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>tank1</td>
<td>t 3 high</td>
<td>divider5 1 t 1 high</td>
</tr>
<tr>
<td></td>
<td>t 2 high</td>
<td>propane gas</td>
</tr>
<tr>
<td>pipe</td>
<td>external heat source</td>
<td>vapour formation</td>
</tr>
<tr>
<td></td>
<td>hot weather</td>
<td>divider5 1 t 2 high</td>
</tr>
<tr>
<td></td>
<td>fire</td>
<td>divider5 1 t 3 high</td>
</tr>
<tr>
<td>pumo3</td>
<td>pumo fluid overheats</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 1
escape

<table>
<thead>
<tr>
<th></th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>tank1</td>
<td>drain open</td>
<td>propanone gas fire hazard</td>
</tr>
<tr>
<td></td>
<td>leak</td>
<td>explosion hazard</td>
</tr>
<tr>
<td>pipe</td>
<td>Leak</td>
<td>propanone gas fire hazard</td>
</tr>
<tr>
<td>valve2</td>
<td>Leak</td>
<td>explosion hazard</td>
</tr>
<tr>
<td>pumo3</td>
<td>Leak</td>
<td>propanone gas fire hazard</td>
</tr>
<tr>
<td>nrv4</td>
<td>Leak</td>
<td>explosion hazard</td>
</tr>
<tr>
<td>divider5</td>
<td>Leak</td>
<td>propanone gas fire hazard</td>
</tr>
</tbody>
</table>

Line 1 IMPURITIES

<table>
<thead>
<tr>
<th>Impurity a high</th>
<th>Causes</th>
<th>Consequences</th>
</tr>
</thead>
</table>

Search on line 2

<table>
<thead>
<tr>
<th></th>
<th>q none</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>divider5</td>
<td>q 1 none</td>
<td>tank1 1 q 1 none</td>
</tr>
<tr>
<td></td>
<td>q 3 none</td>
<td>divider5 1 q 3 none</td>
</tr>
<tr>
<td></td>
<td>blockage</td>
<td>divider5 1 blockage</td>
</tr>
<tr>
<td></td>
<td>leak (major)</td>
<td>pumo3 2 leak (major)</td>
</tr>
<tr>
<td></td>
<td>blockage</td>
<td>pipe 2 blockage</td>
</tr>
<tr>
<td></td>
<td>leak (major)</td>
<td>pipe 2 leak (major)</td>
</tr>
<tr>
<td></td>
<td>loop fails closed</td>
<td>pumo3 2 loop fails closed</td>
</tr>
<tr>
<td>Sensor fails closed</td>
<td>Consequences</td>
<td></td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>Motor fails closed</td>
<td>1 o 1 low</td>
<td></td>
</tr>
<tr>
<td>Contact fails closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CTI-VL fails closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set-ant mvd closed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Isolatn valve closed</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Search on line 2

**CAUSES**

<table>
<thead>
<tr>
<th>Divider5</th>
<th>3 o 1 low</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>o 3 low</td>
</tr>
<tr>
<td></td>
<td>blockage</td>
</tr>
<tr>
<td></td>
<td>leak (major)</td>
</tr>
<tr>
<td>Pipeline</td>
<td>2 blockage</td>
</tr>
<tr>
<td></td>
<td>leak (major)</td>
</tr>
<tr>
<td>Ctrl VL6</td>
<td>2 Loop fails at-c1sls</td>
</tr>
<tr>
<td></td>
<td>sensor fails at-c1sls</td>
</tr>
<tr>
<td></td>
<td>CtrlVL fails at-c1sls</td>
</tr>
<tr>
<td></td>
<td>CTI-VL fails at-c1sls</td>
</tr>
<tr>
<td></td>
<td>set-ant mvd at-c1sls</td>
</tr>
<tr>
<td></td>
<td>Isolatn valve at-c1sls</td>
</tr>
<tr>
<td></td>
<td>blockage</td>
</tr>
</tbody>
</table>

### Search on line 2

**CAUSES**

<table>
<thead>
<tr>
<th>Divider5</th>
<th>3 o 1 high</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>o 3 high</td>
</tr>
<tr>
<td></td>
<td>blockage - other leg</td>
</tr>
<tr>
<td>Ctrl VL6</td>
<td>2 Loop fails open</td>
</tr>
<tr>
<td></td>
<td>sensor fails open</td>
</tr>
<tr>
<td></td>
<td>CtrlVL fails open</td>
</tr>
<tr>
<td></td>
<td>CTI-VL fails open</td>
</tr>
<tr>
<td></td>
<td>set-ant mvd open</td>
</tr>
<tr>
<td></td>
<td>bypass fails</td>
</tr>
<tr>
<td></td>
<td>bypass directed open</td>
</tr>
</tbody>
</table>

### Search on line 2

**CAUSES**

<table>
<thead>
<tr>
<th>Divider5</th>
<th>3 o 1 low</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>o 3 low</td>
</tr>
<tr>
<td>pipe</td>
<td>2</td>
</tr>
<tr>
<td>------</td>
<td>----</td>
</tr>
</tbody>
</table>

**Search on line 2**

| 0  | high |

**CAUSES**

<table>
<thead>
<tr>
<th>divider5</th>
<th>3</th>
<th>a 1 high</th>
<th>a 3 high</th>
<th>blockage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td>leak (major)</td>
<td>1</td>
<td>fire</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>pipe</th>
<th>1</th>
<th>tank1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>a 1 high</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| cntrl vl6 | 1  | loop fails closed |
|           |    | sensor fails closed |
|           |    | cntrl vl fails closed |
|           |    | ctl vl fails closed |
|           |    | set-ont wmo closed |
|           |    | isoltn valve closed |

---

**Search on line 2**

| t  | low |

**CAUSES**

<table>
<thead>
<tr>
<th>divider5</th>
<th>3</th>
<th>t 1 low</th>
<th>t 3 low</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>leak (major)</td>
<td>brittle fracture</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>pipe</th>
<th>2</th>
<th>tank1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>t 1 low</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Search on line 2**

| t  | high |

**CAUSES**

<table>
<thead>
<tr>
<th>divider5</th>
<th>3</th>
<th>t 1 high</th>
<th>t 3 high</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>leak (major)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>pipe</th>
<th>2</th>
<th>propane</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>vapor formation</td>
</tr>
</tbody>
</table>

---

**Search on line 2**
<table>
<thead>
<tr>
<th></th>
<th>escape</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CAUSES</td>
<td></td>
<td>CONSEQUENCES</td>
</tr>
<tr>
<td>divser5</td>
<td>leak</td>
<td>propane</td>
</tr>
<tr>
<td>ploe</td>
<td>leak</td>
<td></td>
</tr>
<tr>
<td>ctrl_vl6</td>
<td>leak</td>
<td></td>
</tr>
<tr>
<td>tank1</td>
<td>drain use</td>
<td>leak</td>
</tr>
</tbody>
</table>

**Line 2 IM PURITIES**

**Impurity q high**

**Causes**

**Impurity q high**

**Causes**

**Search on Line 3**

**q none**

**CAUSES**

**CONSEQUENCES**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>divser5</td>
<td>2</td>
<td>a 1 none</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>a 2 none</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>blockage</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>leak (major)</td>
</tr>
<tr>
<td>ploe</td>
<td>2</td>
<td>blockage</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>leak (major)</td>
</tr>
<tr>
<td>valve7</td>
<td>2</td>
<td>valve directed clsd</td>
</tr>
</tbody>
</table>

**Search on Line 3**

**q low**

**CAUSES**

**CONSEQUENCES**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>divser5</td>
<td>2</td>
<td>a 1 low</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>a 2 low</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>blockage</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>leak (major)</td>
</tr>
<tr>
<td>ploe</td>
<td>2</td>
<td>blockage</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>leak (major)</td>
</tr>
<tr>
<td>valve7</td>
<td>2</td>
<td>valve partly clsd</td>
</tr>
</tbody>
</table>
Search on line 3

\( q \) high

\text{CAUSES}

- divider5
  - 2 q 1 high
  - 2 q 2 high
  - 2 blockage - other leg

\text{CONSEQUENCES}

- exchange
  - 1 q 1 high
  - 1 q 2 high
  - 1 t 2 low
  - 1 t 4 low

Search on line 3

\( p \) low

\text{CAUSES}

- divider5
  - 2 p 1 low
  - 2 p 2 low

- pipe
  - 2 leak (major)

\text{CONSEQUENCES}

- exchange
  - 1 q 1 low

- propano
  - liquid flashes

- exchange
  - 1 q 2 low

Search on line 3

\( p \) high

\text{CAUSES}

- divider5
  - 2 p 1 high
  - 2 p 2 high
  - 2 blockage

- pipe
  - 1 blockage
  - 1 ht to block-in fluid
  - 1 solar heat
  - 1 fire

- valve7
  - 1 valve directed else

- exchange
  - 1 tube fouling

\text{CONSEQUENCES}

- pipe
  - 1 leak (major)
  - 1 pipe rupture
  - 1 flange leak

- exchange
  - 1 q 1 high
  - 1 q 2 high

Search on line 3

\( t \) low

\text{CAUSES}

- divider5
  - 2 t 1 low
  - 2 t 2 low

- pipe
  - 2 external cold source

\text{CONSEQUENCES}

- pipe
  - 1 leak (major)
  - 1 brittle fracture

- exchange
  - 1 t 1 low
<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>cold weather</td>
<td>1 t 2 low</td>
</tr>
<tr>
<td>legacy loss</td>
<td>1 t 4 low</td>
</tr>
</tbody>
</table>

**Search on Line 3**

<table>
<thead>
<tr>
<th>t</th>
<th>high</th>
</tr>
</thead>
</table>

**CONSEQUENCES**

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>divider5</td>
<td>exchange</td>
</tr>
<tr>
<td>2 t 1 high</td>
<td>1 t 1 high</td>
</tr>
<tr>
<td>2 t 2 high</td>
<td>ethane</td>
</tr>
<tr>
<td>olee</td>
<td>exchange</td>
</tr>
<tr>
<td>2 external heat source</td>
<td>vapor formation</td>
</tr>
<tr>
<td>hot weather</td>
<td>1 t 2 high</td>
</tr>
<tr>
<td>fire</td>
<td>1 t 4 high</td>
</tr>
</tbody>
</table>

**Search on Line 3**

<table>
<thead>
<tr>
<th>escape</th>
<th></th>
</tr>
</thead>
</table>

**CONSEQUENCES**

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>divider5</td>
<td>leak</td>
</tr>
<tr>
<td>olee</td>
<td>propane</td>
</tr>
<tr>
<td>valve7</td>
<td>fire hazard</td>
</tr>
<tr>
<td>exchang5</td>
<td>explosion hazard</td>
</tr>
</tbody>
</table>

**Line 3 IMPURITIES**

**Impurity p high**

<table>
<thead>
<tr>
<th>Causes</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchang5</td>
<td>p 4 high</td>
</tr>
<tr>
<td>tube leak</td>
<td></td>
</tr>
</tbody>
</table>

**Impurity g high**

<table>
<thead>
<tr>
<th>Causes</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchang5</td>
<td></td>
</tr>
<tr>
<td>g 4 high</td>
<td></td>
</tr>
</tbody>
</table>

**Search on Line 4**

<table>
<thead>
<tr>
<th>q</th>
<th>none</th>
</tr>
</thead>
</table>

**CONSEQUENCES**

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchang5</td>
<td>tank19</td>
</tr>
<tr>
<td>2 q 1 none</td>
<td>q 1 none</td>
</tr>
<tr>
<td>2 q 2 none</td>
<td>q 2 none</td>
</tr>
<tr>
<td>olee</td>
<td></td>
</tr>
<tr>
<td>2 blockage</td>
<td></td>
</tr>
<tr>
<td>nrv15</td>
<td></td>
</tr>
</tbody>
</table>
nrv9  2  valve stuck clsd
trio15  2  sourious closure
trio10  2  blockage
bick_bld13  2  isoltn vl clsd
bick_bld11  2  leak (major)
2  drain valve open
pipeline12  2  blockage
2  leak (major)
entrl_vl14  2  leak (major)
2  sensor fails clsd
2  ctnl_vl fails clsd
2  set-ont wvd clsd
2  isoltn valve clsed
valve17  2  valve directed clsd

Search on line 4
q  low

CAUSES
exchang  2  q  1 low
2  q  2 low
2  tube fouling
2  tube less (major)
place  2  blockage
2  leak (major)
nrv16  2  valve fails pt, clsd
nrvi9  2  valve stuck clsd
trio15  2  blockage
bick_bld13  2  isoltn vl clsd
bick_bld11  2  leak (major)
2  drain valve open
pipeline12  2  blockage
2  leak (major)
entrl_vl14  2  leak (major)
2  sensor fails clsd
2  ctnl_vl fails clsd
2  set-ont wvd clsd
2  isoltn valve clsed
2  blockage

CONSEQUENCES
q  1 low
1  q  3 low
3  loss of outfls
valve 17
2 valve partly clsd

---

Search on line 4
q high

CAUSES
exchanger
2 q 1 high
2 q 2 high

contrl_vt14
2 loop fails open
sensor fails open
contrl fails open
evt_vt fails open
set point and open
2 bypass falls
2 bypass directed open

---

Search on line 4
p low

CAUSES
exchanger
2 p 1 low
2 p 2 low
2 tube leak (major)

pipe
2 leak (major)

bleck_bld13
2 leak (major)
2 drain valve open

---

Search on line 4
o high

CAUSES
exchanger
2 o 1 high
2 o 2 high

pipe
1 blockage
1 nt to block-in fluid
solar heat
fire

tria15

---

Search on line 4
q 1 high

CONSEQUENCES
tank18
1 q 1 high
1 l 3 high
3 tank overfilling
3 liquid enters vent

---

CONSEQUENCES
tank13
1 o 1 low

CONSEQUENCES
pipe
1 leak (major)
1 pipe rupture
1 flange leak

---

CONSEQUENCES
tank11
1 o 1 high

---

CONSEQUENCES
Search on line 4

t

CAUSES

exchanners

pipe

CONSEQUENCES

ploe

1 loss (major)

1 brittle fracture

tank14

1 t 1 high

Search on line 4

t

CAUSES

exchanners

pipe

CONSEQUENCES

ploe

1 external cold source

cold weather

1 t 1 high

1 vapour formation

Search on line 4

ecape
<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchanger5</td>
<td>leak</td>
</tr>
<tr>
<td>plooe</td>
<td>leak</td>
</tr>
<tr>
<td>nrw18</td>
<td>leak</td>
</tr>
<tr>
<td>nrw9</td>
<td>leak</td>
</tr>
<tr>
<td>trio19</td>
<td>leak</td>
</tr>
<tr>
<td>trio10</td>
<td>leak</td>
</tr>
<tr>
<td>block_bld13</td>
<td>leak</td>
</tr>
<tr>
<td>block_bld11</td>
<td>leak</td>
</tr>
<tr>
<td>pipeline12</td>
<td>leak</td>
</tr>
<tr>
<td>ctrcl_vli14</td>
<td>leak</td>
</tr>
<tr>
<td>valve17</td>
<td>leak</td>
</tr>
<tr>
<td>tank18</td>
<td>leak</td>
</tr>
</tbody>
</table>

**Line 4 IMPURITIES**

<table>
<thead>
<tr>
<th>Impurity o high Causes</th>
<th>exchanger5</th>
<th>tube leak</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchanger5</td>
<td>o 4 high</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Impurity g high Causes</th>
<th>exchanger5</th>
<th>tube leak</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchanger5</td>
<td>g 4 low</td>
<td></td>
</tr>
</tbody>
</table>

**Search on line 5**

<table>
<thead>
<tr>
<th>q</th>
<th>none</th>
</tr>
</thead>
</table>

**CAUSES**

<table>
<thead>
<tr>
<th>tank18</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>plooe</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>valve19</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>sink20</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

**Search on line 5**

<table>
<thead>
<tr>
<th>q</th>
<th>low</th>
<th>2 l 3 high</th>
</tr>
</thead>
<tbody>
<tr>
<td>sink18</td>
<td>2</td>
<td>3 none</td>
</tr>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>--------------------</td>
<td></td>
</tr>
<tr>
<td>tank18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2 l 3 low</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2 o 3 low</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2 q 2 low</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>leak (major)</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>blockage</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>leak (major)</td>
<td></td>
</tr>
<tr>
<td>valve19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>valve partly clsd</td>
<td></td>
</tr>
<tr>
<td>sink20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>sink full</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search on line 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>q</td>
<td>high</td>
<td></td>
</tr>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
<td></td>
</tr>
<tr>
<td>tank18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2 l 3 high</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2 o 3 high</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2 q 2 high</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sink20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 o 1 high</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search on line 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>q</td>
<td>low</td>
<td></td>
</tr>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
<td></td>
</tr>
<tr>
<td>tank18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2 o 3 low</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2 q 2 low</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sink20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 o 1 low</td>
<td></td>
</tr>
<tr>
<td>opropane</td>
<td>liquid flashes</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search on line 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>q</td>
<td>high</td>
<td></td>
</tr>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
<td></td>
</tr>
<tr>
<td>tank18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2 o 3 high</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2 q 2 high</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td></td>
<td></td>
</tr>
<tr>
<td>valve19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>valve directed clsd</td>
<td></td>
</tr>
<tr>
<td>sink20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>sink full</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Search on line 5

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>tank18</td>
<td>2 t 2 low</td>
</tr>
<tr>
<td>pipe</td>
<td>2 external cold source</td>
</tr>
<tr>
<td></td>
<td>cold weather</td>
</tr>
<tr>
<td></td>
<td>lagging too</td>
</tr>
<tr>
<td>tank18</td>
<td>2 t 2 high</td>
</tr>
<tr>
<td>aloe</td>
<td>2 external heat source</td>
</tr>
<tr>
<td></td>
<td>not weather</td>
</tr>
<tr>
<td></td>
<td>fire</td>
</tr>
<tr>
<td>esc30e</td>
<td></td>
</tr>
<tr>
<td>tank19</td>
<td>leak</td>
</tr>
<tr>
<td>pipe</td>
<td>leak</td>
</tr>
<tr>
<td>valve19</td>
<td>leak</td>
</tr>
</tbody>
</table>

### Line 5 IMPURITIES

<table>
<thead>
<tr>
<th>Impurity p high Causes</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Impurity g high Causes</th>
<th>Consequences</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

No causes and effects of a none found on line 6

No causes and effects of a low found on line 6
No causes and effects of \( q \) high found on line 6

No causes and effects of \( q \) low found on line 6

No causes and effects of \( t \) high found on line 6

No causes and effects of \( t \) low found on line 6

Search on line 6

escape

CAUSES

pipe leak

exchange23 leak

Line 6 IMPURITIES

Impurity \( q \) high Causes

Impurity \( q \) high Causes

Search on line 7

q none

CAUSES

exchange23 4 q 3 none
drum2 1 q 1 none

pipe 4 q 4 none

shell leak (major)

Search on line 7

q low
<table>
<thead>
<tr>
<th><strong>CAUSES</strong></th>
<th><strong>CONSEQUENCES</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>exch23</td>
<td>drun=24</td>
</tr>
<tr>
<td>4 q 3 low</td>
<td>1 q 1 low</td>
</tr>
<tr>
<td>4 q 4 low</td>
<td>1 l 3 low</td>
</tr>
<tr>
<td>4 shell leak (major)</td>
<td>3 loss of liquid seal</td>
</tr>
<tr>
<td>aloa</td>
<td></td>
</tr>
<tr>
<td>2 blockage</td>
<td></td>
</tr>
<tr>
<td>2 leak (major)</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 7

p high

<table>
<thead>
<tr>
<th><strong>CAUSES</strong></th>
<th><strong>CONSEQUENCES</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>exch23</td>
<td>drun=24</td>
</tr>
<tr>
<td>4 q 3 high</td>
<td>1 q 1 high</td>
</tr>
<tr>
<td>4 q 4 high</td>
<td>1 l 3 high</td>
</tr>
<tr>
<td>4 shell leak (major)</td>
<td>3 drum overfilled</td>
</tr>
<tr>
<td>aloa</td>
<td></td>
</tr>
<tr>
<td>2 blockage</td>
<td></td>
</tr>
<tr>
<td>2 leak (major)</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 7

p low

<table>
<thead>
<tr>
<th><strong>CAUSES</strong></th>
<th><strong>CONSEQUENCES</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>exch23</td>
<td>drun=24</td>
</tr>
<tr>
<td>4 p 3 low</td>
<td>1 q 1 low</td>
</tr>
<tr>
<td>4 p 4 low</td>
<td>1 l 3 low</td>
</tr>
<tr>
<td>4 shell leak (major)</td>
<td>3 drum overfilled</td>
</tr>
<tr>
<td>aloa</td>
<td></td>
</tr>
<tr>
<td>2 blockage</td>
<td></td>
</tr>
<tr>
<td>2 leak (major)</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 7

p high

<table>
<thead>
<tr>
<th><strong>CAUSES</strong></th>
<th><strong>CONSEQUENCES</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>exch23</td>
<td>drun=24</td>
</tr>
<tr>
<td>4 p 3 high</td>
<td>1 q 1 high</td>
</tr>
<tr>
<td>4 p 4 high</td>
<td>1 l 3 high</td>
</tr>
<tr>
<td>4 tube leak (major)</td>
<td>3 drum overfilled</td>
</tr>
<tr>
<td>aloa</td>
<td></td>
</tr>
<tr>
<td>1 blockage</td>
<td></td>
</tr>
<tr>
<td>1 nt to stick-in fluid</td>
<td>1 flange leak</td>
</tr>
<tr>
<td>1 solar heat</td>
<td></td>
</tr>
<tr>
<td>1 fire</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 7

r low

<table>
<thead>
<tr>
<th><strong>CAUSES</strong></th>
<th><strong>CONSEQUENCES</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>exch23</td>
<td>drun=24</td>
</tr>
<tr>
<td>4 p 3 high</td>
<td>1 q 1 high</td>
</tr>
<tr>
<td>4 p 4 high</td>
<td>1 l 3 high</td>
</tr>
<tr>
<td>4 tube leak (major)</td>
<td>3 drum overfilled</td>
</tr>
<tr>
<td>aloa</td>
<td></td>
</tr>
<tr>
<td>1 blockage</td>
<td></td>
</tr>
<tr>
<td>1 nt to stick-in fluid</td>
<td>1 flange leak</td>
</tr>
<tr>
<td>1 solar heat</td>
<td></td>
</tr>
<tr>
<td>1 fire</td>
<td></td>
</tr>
<tr>
<td>locale</td>
<td>1 leak (main)</td>
</tr>
<tr>
<td>--------</td>
<td>---------------</td>
</tr>
<tr>
<td>pipe</td>
<td>2 external colc source</td>
</tr>
<tr>
<td>pipe</td>
<td>2 external heat source</td>
</tr>
</tbody>
</table>

**Search on Line 7**

t  high

**CAUSES**

<table>
<thead>
<tr>
<th>locale</th>
<th>1 t 1 low</th>
<th>1 t 3 low</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchange23</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>water</td>
<td>formation of ice</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td>2 t w 1 heat source</td>
<td>heating</td>
</tr>
</tbody>
</table>

**Search on Line 8**

eescne

**CAUSES**

<table>
<thead>
<tr>
<th>locale</th>
<th>1 leak</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchange23</td>
<td>leak</td>
</tr>
<tr>
<td>pipe</td>
<td>leak</td>
</tr>
<tr>
<td>drum24</td>
<td>leak</td>
</tr>
</tbody>
</table>

**Line 7 IMPURITIES**

<table>
<thead>
<tr>
<th>impurity</th>
<th>1 high</th>
<th>1 low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exchange23</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Search on Line 13**

g  none
<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>drum24</td>
<td></td>
</tr>
<tr>
<td>2 l 3 none</td>
<td>drum24 2 l 3 high</td>
</tr>
<tr>
<td>2 q 2 none</td>
<td></td>
</tr>
<tr>
<td>2 Leak (major)</td>
<td>sink25 1 q 1 none</td>
</tr>
<tr>
<td>pipe</td>
<td></td>
</tr>
<tr>
<td>2 BLockage</td>
<td></td>
</tr>
<tr>
<td>2 Leak (major)</td>
<td></td>
</tr>
<tr>
<td>cntrl_v128</td>
<td></td>
</tr>
<tr>
<td>2 Loop falls closed</td>
<td></td>
</tr>
<tr>
<td>sensor falls closed</td>
<td></td>
</tr>
<tr>
<td>cntrlr falls closed</td>
<td></td>
</tr>
<tr>
<td>ctrl-wl falls closed</td>
<td></td>
</tr>
<tr>
<td>set-ant and set-ctl</td>
<td></td>
</tr>
<tr>
<td>2 isolan valve closed</td>
<td></td>
</tr>
<tr>
<td>sink26</td>
<td></td>
</tr>
<tr>
<td>1 sink full</td>
<td></td>
</tr>
</tbody>
</table>

Search on Line 8

q low

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>drum24</td>
<td></td>
</tr>
<tr>
<td>2 l 3 Low</td>
<td>drum24 2 l 3 high</td>
</tr>
<tr>
<td>2 q 2 Low</td>
<td></td>
</tr>
<tr>
<td>2 Leak (major)</td>
<td>sink25 1 q 1 low</td>
</tr>
<tr>
<td>pipe</td>
<td></td>
</tr>
<tr>
<td>2 BLockage</td>
<td></td>
</tr>
<tr>
<td>2 Leak (major)</td>
<td></td>
</tr>
<tr>
<td>cntrl_v128</td>
<td></td>
</tr>
<tr>
<td>2 Loop falls at-clsd</td>
<td></td>
</tr>
<tr>
<td>sensor falls at-clsd</td>
<td></td>
</tr>
<tr>
<td>cntrlr falls at-clsd</td>
<td></td>
</tr>
<tr>
<td>ctrl-wl falls at-clsd</td>
<td></td>
</tr>
<tr>
<td>set-ant and set-ctl</td>
<td></td>
</tr>
<tr>
<td>2 isolan valve at-clsd</td>
<td></td>
</tr>
<tr>
<td>2 BLockage</td>
<td></td>
</tr>
<tr>
<td>sink26</td>
<td></td>
</tr>
<tr>
<td>1 sink full</td>
<td></td>
</tr>
</tbody>
</table>

Search on Line 8

q high

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>drum24</td>
<td></td>
</tr>
<tr>
<td>2 l 3 high</td>
<td>drum24 2 l 3 low</td>
</tr>
<tr>
<td>2 q 2 high</td>
<td></td>
</tr>
<tr>
<td>cntrl_v128</td>
<td></td>
</tr>
<tr>
<td>2 Loop falls open</td>
<td></td>
</tr>
<tr>
<td>sensor falls open</td>
<td></td>
</tr>
<tr>
<td>cntrlr falls open</td>
<td></td>
</tr>
<tr>
<td>ctrl-wl falls open</td>
<td></td>
</tr>
<tr>
<td>set-ant and set-ctl</td>
<td></td>
</tr>
<tr>
<td>2 isolan valve open</td>
<td></td>
</tr>
<tr>
<td>2 BLockage</td>
<td></td>
</tr>
<tr>
<td>sink26</td>
<td></td>
</tr>
<tr>
<td>1 sink full</td>
<td></td>
</tr>
</tbody>
</table>
Search on line 8

CAUSES

drum24  2 t 3 low
2 t 2 low
pipe  2 external cold source
       cold weather
       lagging loss

sink26  1 t 1 low
       water
formation of ice

Search on line 8

CAUSES

drum24  2 o 3 high
2 o 2 high
pipe  1 blockage
       1 nt to bleed-in fluid
       solar heat
       fire

cntrl_vl25  1 loop fails closed
            sensor fails closed
            cntrl vl fails closed
            cntl vl fails closed
            set-rant vwd closed
            1 isolatn valve closed

sink26  1 sink full

Search on line 8

CAUSES

drum24  2 a 3 low
2 a 2 low
pipe  2 leak (major)

CONSEQUENCES

sink26  1 a 1 low

Search on line 8

CAUSES

drum24  2 o 3 low
2 o 2 low
pipe  2 leak (major)

CONSEQUENCES

sink26  1 o 1 low
### CAUSES

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td>drum24</td>
<td>2</td>
<td>t</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>external heat source</th>
</tr>
</thead>
<tbody>
<tr>
<td>pipe</td>
<td>2</td>
<td>hot weather</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fire</td>
</tr>
</tbody>
</table>

### CONSEQUENCES

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td>sink2b</td>
<td>1</td>
<td>t</td>
<td>1</td>
</tr>
</tbody>
</table>

### Search on line 8

**escape**

**CAUSES**

<table>
<thead>
<tr>
<th></th>
<th>Leak</th>
</tr>
</thead>
<tbody>
<tr>
<td>drum24</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td></td>
</tr>
<tr>
<td>ctrvl125</td>
<td></td>
</tr>
</tbody>
</table>

### Line 8 IMPURITIES

<table>
<thead>
<tr>
<th></th>
<th>Causes</th>
</tr>
</thead>
<tbody>
<tr>
<td>purity g high</td>
<td></td>
</tr>
</tbody>
</table>

**Consequences**

<table>
<thead>
<tr>
<th></th>
<th>Causes</th>
</tr>
</thead>
<tbody>
<tr>
<td>purity g high</td>
<td></td>
</tr>
</tbody>
</table>

**Consequences**

### Search on line 9

**q none**

**CAUSES**

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3</th>
<th>none</th>
</tr>
</thead>
<tbody>
<tr>
<td>tank27</td>
<td>2</td>
<td>q</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>blockage</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>leak (major)</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>loss of NPS4</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>rotation fault</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>imeller fault</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>cavitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>delivery shut in</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>no chance to stop</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>stoppage fail on demand</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>malopofit of valves</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>no reset of valve rw</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>alakage</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>leak (major)</th>
</tr>
</thead>
<tbody>
<tr>
<td>alve</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

### Sunk27

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>1 3 high</th>
</tr>
</thead>
<tbody>
<tr>
<td>tank27</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

### Oma29

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>1 1 none</th>
</tr>
</thead>
<tbody>
<tr>
<td>pumna29</td>
<td>2</td>
<td>fluid overheats</td>
</tr>
</tbody>
</table>

### Exchange23

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2 none</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchano23</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

### Vrv29

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>valve stuck clsd</th>
</tr>
</thead>
<tbody>
<tr>
<td>nrv29</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
Search on Line 9
q  low

CAUSES
tank27  2  q  2  low
piso  2  olookup  2  leak (major)
pump29  2  loss of NPSH  2  rotation fault  2  flowmeter fault  2  cavitPe  2  delivery osrt shut  2  only 1 on whn 2 reqd  2  olookup  2  leak (major)
nrv29  2  valve fails or, clsd

CONSEQUENCES
tank27  2  l  3  high
exhange23  1  q  1  low  1  q  2  low  1  t  4  high

Search on Line 9
q  high

CAUSES
tank27  2  q  2  high

CONSEQUENCES
tank27  2  l  3  low
exhange23  1  q  1  high  1  q  2  high  1  t  2  low  1  t  6  low

Search on Line 9
q  low

CAUSES
tank27  2  l  3  low  2  q  3  low  2  o  2  low
piso  2  leak (major)

CONSEQUENCES
exhange23  1  q  1  low  1  q  2  low

Search on Line 9
q  high

CAUSES

CONSEQUENCES
<table>
<thead>
<tr>
<th>tank27</th>
<th>2 t 3 high</th>
<th>oileo</th>
<th>1 leak (major)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 o 3 high</td>
<td></td>
<td>1 o lor rupture</td>
</tr>
<tr>
<td></td>
<td>2 p 2 high</td>
<td></td>
<td>1 flange leak</td>
</tr>
<tr>
<td>oileo</td>
<td>1 blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 ht to oile-v fluid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 solar heat</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pumo28</td>
<td>2 deliver shut in</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 blockage</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 ht to oile-v fluid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 solar heat</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2 fire</td>
<td></td>
<td></td>
</tr>
<tr>
<td>exch423</td>
<td>1 tube fouling</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Search on line 9**

- **t low**
  - **CAUSES**
    - tank27 2 t 2 low
    - oileo
    - 2 external cold source
      - cold weather
      - lagging loss
    - exch423
    - 1 t 1 low
    - 1 t 2 low
    - 1 t 4 low

---

**Search on line 9**

- **t high**
  - **CAUSES**
    - tank27 2 t 2 high
    - exch423
    - 1 t 2 high
    - 1 t 4 high
    - pumo28 2 pumo fluid overheats

---

**Search on line 9**

- **escape**
  - **CAUSES**
    - tank27 leak
    - oileo leak
Leak
Leak
Leak

Line 9 IMPURITIES

Impurity q high
Causes
Consequences

Impurity q high
Causes
Consequences

exchang23
Tube leak

Search on Line 10
q none

CAUSES
exchang23 2 q 1 none
tu_valve31

Search on Line 10
q low

CAUSES
exchang23 2 q 1 low
tu_valve31
tu_tube

Search on Line 10
q high

CAUSES
exchang23 2 q 1 high
tu_valve31
Search on line 10
\( p \) low

**CAUSES**
- exchAg23
  - 2 \( p \) 1 low
  - 2 \( p \) 2 low
  - 2 leak (major)
- olde
  - 2 leak (major)

**CONSEQUENCES**
- tu valve31
  - 1 \( o \) 1 low
  - 1 \( o \) 2 low
  - 1 \( o \) 3 low

---

Search on line 10
\( p \) high

**CAUSES**
- exchAg23
  - 2 \( p \) 1 high
  - 2 \( p \) 2 high
- olde
  - 1 blockage
  - 1 hit to block-in fluid
  - 1 solar heat
  - 1 fire
- valve30
  - 1 valve directed closed

**CONSEQUENCES**
- olde
  - 1 leak (major)
  - 1 olde rupture
  - 1 flange leak
- tu valve31
  - 1 \( o \) 1 high
  - 1 \( o \) 2 high
  - 1 \( o \) 3 high

---

Search on line 10
\( t \) low

**CAUSES**
- exchAg23
  - 2 \( t \) 1 low
  - 2 \( t \) 3 low
  - 2 \( q \) 3 low
  - 2 \( q \) 1 high
  - 2 tube fouling
- olde
  - 2 external cold source
  - 2 cold weather
  - 2 lagging loss

**CONSEQUENCES**
- olde
  - 1 leak (major)
  - 1 brittle fracture
- tu valve31
  - 1 \( t \) 1 low
  - 1 \( t \) 2 low
  - 1 \( t \) 3 low

---

Search on line 10
\( t \) high

---
CAUSES

exchangel3
2 t 1 high
2 t 3 high
2 q 3 high
2 q 1 low
2 t 2 high

plue
2 external heat source
hot weather
fire

CONSEQUENCES
tv_valve31
1 t 1 high
1 t 2 high
1 t 3 high

Search on line 10

escape

CAUSES

exchangel3 leak
plue leak

valve30 leak
tv_valve31 leak

CONSEQUENCES

Line 10 IMPURITIES

Impurity a high
Causes

Impurity g high
Causes

exchangel3 tuce leak

Search on line 11

a none

CAUSES
tv_valve31
2 a 1 none
2 a 3 high
2 a 2 none
2 stuck to other port
2 set point error
plue
2 blockage
2 leak (major)

valve32
2 valve directed clsd

CONSEQUENCES
tv_valve31
2 a 3 high

Search on line 11
CAUSES

`tv_valve31` 2 q 1 low
2 q 3 high
2 q 2 low
2 stuck to other part
2 set point error

`dloe` 2 blockage
2 leak (major)

`valve32` 2 valve partly clsd

CONSEQUENCES

`tv_valve31` 2 a 3 high

Search on line 11

q high

CAUSES

`tv_valve31` 2 a 1 high
2 q 3 low
2 q 2 high
2 stuck to this part
2 set point error

CONSEQUENCES

`tv_valve31` 2 a 3 low

Search on line 11

a low

CAUSES

`tv_valve31` 2 q 1 low
2 q 3 low
2 q 2 high
2 stuck to this part
2 set point error

CONSEQUENCES

`dloe` 3 a 3 low

Search on line 11

a high

CAUSES

`tv_valve31` 2 q 1 high
2 q 2 high

CONSEQUENCES

`dloe` 1 leak (major)
1 valve rupture
1 flange leak

`valve32` 1 blockage
1 hit to block-in fluid
1 solar heat
1 fire

`exchang9` 3 q 3 low
3 q 4 low
3 t 2 low
3 t 4 low
3 q 3 high
3 q 4 high
valve32
1 valve directed closed

Search on Line 11

t - low

CAUSES

tw_valve31
2 t 1 low
2 t 2 low

plug
2 external cold source
cold weather
lagging loss

CONSEQUENCES

role
1 leak (major)
1 brittle fracture

Search on Line 11

t - high

CAUSES

tw_valve31
2 t 1 high
2 t 2 high

plug
2 external heat source
hot weather
fire

CONSEQUENCES

exchanger
3 t 3 high
3 t 2 high
3 t 4 high

Search on Line 11

escape

CAUSES

tw_valve31
leak

plug
leak

valve32
leak

exchanger
leak

Line 11 IMPURITIES

Impurity @ high

Causes
exchanger
Consequences
tube leak

Impurity @ high

Causes
exchanger
Consequences
t @ high
### Search on line 12

<table>
<thead>
<tr>
<th>q</th>
<th>none</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>exchange</td>
<td>4 q 3 none</td>
</tr>
<tr>
<td>pipe</td>
<td>2 blockage</td>
</tr>
</tbody>
</table>

### Search on line 12

<table>
<thead>
<tr>
<th>q</th>
<th>low</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>exchange</td>
<td>4 q 3 low</td>
</tr>
<tr>
<td>pipe</td>
<td>2 blockage</td>
</tr>
</tbody>
</table>

### Search on line 12

<table>
<thead>
<tr>
<th>q</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>exchange</td>
<td>4 q 4 high</td>
</tr>
<tr>
<td>pipe</td>
<td>2 leak (major)</td>
</tr>
</tbody>
</table>

### Search on line 12

<table>
<thead>
<tr>
<th>q</th>
<th>low</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>exchange</td>
<td>4 q 4 low</td>
</tr>
<tr>
<td>pipe</td>
<td>2 leak (major)</td>
</tr>
</tbody>
</table>

### Search on line 12

<table>
<thead>
<tr>
<th>q</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td>exchange</td>
<td>4 q 3 high</td>
</tr>
<tr>
<td>pipe</td>
<td>2 leak (major)</td>
</tr>
<tr>
<td>CAUSES</td>
<td>CONSEQUENCES</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>exchanger</td>
<td>oileo</td>
</tr>
<tr>
<td>gas 3 high</td>
<td>1 leak (major)</td>
</tr>
<tr>
<td>gas 4 high</td>
<td>1 oileo rupture</td>
</tr>
<tr>
<td>tube leak (major)</td>
<td>1 freeze leak</td>
</tr>
<tr>
<td>line</td>
<td></td>
</tr>
<tr>
<td>blockage</td>
<td>mixer34</td>
</tr>
<tr>
<td>hot to cold-in fluid</td>
<td>1 gas 1 high</td>
</tr>
<tr>
<td>solar heat</td>
<td>1 gas 3 high</td>
</tr>
<tr>
<td>fire</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 12

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchanger</td>
<td>oileo</td>
</tr>
<tr>
<td>gas 3 low</td>
<td>1 leak (major)</td>
</tr>
<tr>
<td>gas 3 low</td>
<td>1 oileo rupture</td>
</tr>
<tr>
<td>gas 3 low</td>
<td>1 freeze leak</td>
</tr>
<tr>
<td>mixer34</td>
<td></td>
</tr>
<tr>
<td>2 external cold source</td>
<td></td>
</tr>
<tr>
<td>cold weather</td>
<td></td>
</tr>
<tr>
<td>lagging loss</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 12

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchanger</td>
<td>mixer34</td>
</tr>
<tr>
<td>gas 1 high</td>
<td>1 gas 1 high</td>
</tr>
<tr>
<td>gas 3 high</td>
<td>1 gas 3 high</td>
</tr>
<tr>
<td>tube fouling</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td></td>
</tr>
<tr>
<td>2 external heat source</td>
<td></td>
</tr>
<tr>
<td>hot weather</td>
<td></td>
</tr>
<tr>
<td>fire</td>
<td></td>
</tr>
</tbody>
</table>

Search on line 12

<table>
<thead>
<tr>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>exchanger</td>
<td>mixer34</td>
</tr>
<tr>
<td>leak</td>
<td></td>
</tr>
<tr>
<td>pipe</td>
<td>mixer34</td>
</tr>
<tr>
<td>leak</td>
<td></td>
</tr>
</tbody>
</table>
Line 12 IMPURITIES

Impurity 4 high Cause
exchanger1 Causes leak

Impurity 4 high Cause
Consequences
exchanger4 t 4 high

Search on Line 13
q 2

CAUSES

CONSEQUENCES

tw_valve31 3 q 1 none
tw_valve31 3 q 2 high
mixer34 2 q 2 none
mixer34 2 q 3 none

valve33 2 valve directed clsd

Search on Line 13
q 1

CAUSES

CONSEQUENCES

tw_valve31 3 q 1 low
tw_valve31 3 q 2 high
mixer34 2 q 2 low
mixer34 2 q 3 low

valve33 2 valve partly clsd

Search on Line 13
q 2

CAUSES

CONSEQUENCES

tw_valve31 3 q 1 none
tw_valve31 3 q 2 high
mixer34 2 q 2 none
mixer34 2 q 3 none

valve33 2 valve directed clsd
<table>
<thead>
<tr>
<th>Search on line 13</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td><code>tu_valve31</code> 3 o 1 high</td>
<td><code>mover34</code> 2 o 2 high</td>
</tr>
<tr>
<td><code>tu_valve31</code> 3 o 2 low</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>mover34</code> 2 o 3 high</td>
</tr>
<tr>
<td><code>mover31</code> 3 stuck to this port</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>mover34</code> 2 set point error</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Search on line 13</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td><code>tu_valve31</code> 3 o 1 low</td>
<td><code>mover34</code> 2 o 2 low</td>
</tr>
<tr>
<td><code>tu_valve31</code> 3 o 3 low</td>
<td></td>
</tr>
<tr>
<td><code>mover31</code> 2 leak (major)</td>
<td></td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Search on line 13</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td><code>tu_valve31</code> 3 o 1 high</td>
<td><code>mover31</code> 1 leak (major)</td>
</tr>
<tr>
<td><code>tu_valve31</code> 3 o 3 high</td>
<td><code>mover31</code> 1 leak rupture</td>
</tr>
<tr>
<td></td>
<td><code>mover31</code> 1 flange leak</td>
</tr>
<tr>
<td><code>mover31</code> 1 blockage</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>mover31</code> 1 ht to block-in fluid</td>
</tr>
<tr>
<td></td>
<td><code>mover31</code> 1 solar heat</td>
</tr>
<tr>
<td></td>
<td><code>mover31</code> 1 fire</td>
</tr>
<tr>
<td><code>mover31</code> 2 valve direct dtsd</td>
<td></td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Search on line 13</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td><code>tu_valve31</code> 3 t 1 low</td>
<td><code>mover31</code> 1 leak (major)</td>
</tr>
<tr>
<td><code>tu_valve31</code> 3 t 3 low</td>
<td><code>mover31</code> 1 brittle fracture</td>
</tr>
<tr>
<td><code>mover31</code> 2 external cold source</td>
<td></td>
</tr>
<tr>
<td><code>mover31</code> 2 cold weather</td>
<td></td>
</tr>
<tr>
<td><code>mover31</code> 2 lagging loss</td>
<td></td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Search on line 13</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CAUSES</strong></td>
<td><strong>CONSEQUENCES</strong></td>
</tr>
<tr>
<td><code>tu_valve31</code></td>
<td></td>
</tr>
<tr>
<td><code>mover31</code> 2 t 2 low</td>
<td></td>
</tr>
<tr>
<td><code>mover31</code> 2 t 3 low</td>
<td></td>
</tr>
</tbody>
</table>
CAUSES

tw_valve31 3 t 1 high
3 t 3 high
pipe 2 external heat source
hot weather
fire

CONSEQUENCES
mixer34 2 t 2 high
2 t 3 high

Search on Line 13
escape

CAUSES

tw_valve31 leak
pipe leak
valve33 leak
mixer34 leak

CONSEQUENCES

Line 13 IMPURITIES

Impurity a high
Causes

Impurity a high
Causes

Search on Line 14
q none

CAUSES

mixer34 3 q 1 none
3 q 2 none
3 q 3 none
pipe 2 blockage
2 leak (major)

CONSEQUENCES
tank27 1 q 1 none
1 q 3 none

Search on Line 16
q low

CAUSES

mixer34 3 q 1 low
3 q 2 low
3 q 3 low

CONSEQUENCES
tank27 1 q 1 low
1 q 3 low
<table>
<thead>
<tr>
<th>Place</th>
<th>Blockage</th>
<th>Leak (Major)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

**Search on Line 16**

<table>
<thead>
<tr>
<th>q (high)</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mixer34</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>q 1</td>
<td>high</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>q 2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>q 3</td>
</tr>
</tbody>
</table>

| tank27   | 1       | q 1          |
|          | 1       | q 3          |

**Search on Line 16**

<table>
<thead>
<tr>
<th>o (low)</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mixer34</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>o 1</td>
<td>low</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>o 2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>o 3</td>
</tr>
</tbody>
</table>

| tank27   | 1       | o 1          |

**Search on Line 16**

<table>
<thead>
<tr>
<th>o (high)</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mixer34</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>o 1</td>
<td>high</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>o 2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>o 3</td>
</tr>
</tbody>
</table>

| o (low)  | tank27  | 1             |
|          | o 1     | high          |
|          | fire    |               |
|          | flange  | leak (major)  |

<table>
<thead>
<tr>
<th>o (high)</th>
<th>CAUSES</th>
<th>CONSEQUENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mixer34</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>t 1</td>
<td>low</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>t 2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>t 3</td>
</tr>
</tbody>
</table>

| tank27   | 1       | t 1          |
|          | o (low) |               |
|          | brittle fracture |               |
pipe 2  external cold source
    cold weather
    lagging loss

Search on line 14
    t  high
CAUSES
    mixer34  3  t  1  high
               3  t  2  high
               3  t  3  high
pipe 2  external heat source
       not weather
       fire

Search on line 14
    escape
CAUSES
    mixer34  leak
    pipe    leak
    tank27  leak

Line 14  IMPURITIES
    Impurity a high
    Causes
    Consequences
    Impurity g high
    Causes
    Consequences