Computer-aided Applications in Process Plant Safety

Hong An

Hazid Technologies Ltd.
Suite 14B, Beeston Business Park
Technology Drive, Beeston
Nottingham
Loughborough University
NG9 2ND

Centre for Innovative and Collaborative Engineering
Department of Civil & Building Engineering
Loughborough University
Loughborough
Leicestershire, LE11 3TU

Loughborough University

Thesis Access Form

Copy No ……………………………… **Location**………………………………

Author Hong An

Title Computer-aided applications in process plant safety

Status of access ~~OPEN~~ / ~~RESTRICTED~~ / CONFIDENTIAL

Moratorium period: ……5… years, ending …30/01/ 2015………

Conditions of access proved by (CAPITALS): …………………………………………………

Director of Research (Signature) …………………………………………………………

Department of Computer Science

Author's Declaration: I agree the following conditions:

Author's signature ……………………………… **Date** ………………………

Users declaration: for signature during any Moratorium period (Not Open work):
I undertake to uphold the above conditions:

Date	Name (CAPITALS)	Signature	Address

Loughborough University

Certificate of Originality

This is to certify that I am responsible for the work submitted in this thesis, that the original work is my own except as specified in acknowledgments or in footnotes, and that neither the thesis nor the original work contained therein has been submitted to this or any other institution for a higher degree.

Author's signature ....

Date ....

COMPUTER-AIDED APPLICATIONS IN PROCESS PLANT SAFETY

Hong An

A dissertation thesis submitted in partial fulfilment of the requirements for the award of the degree Doctor of Engineering (EngD), at Loughborough University

January 2010

Hazid Technologies Ltd.
Suite 14B, Beeston Business Park
Technology Drive, Beeston
Nottingham
Loughborough University
NG9 2ND

Centre for Innovative and Collaborative Engineering
Department of Civil & Building Engineering
Loughborough University
Loughborough
Leicestershire, LE11 3TU

This thesis is respectfully dedicated to

Prof. Paul Chung, Dr. Colin Machin and Dr. David Styles

Without them, this EngD would never happen.

ACKNOWLEDGEMENTS

I would like to thank both my academic supervisors, Professor P.W.H. Chung and Dr. Colin Machin. I should like to thank Professor Chung for his guidance, advice and his patience on teaching me how to do research work and for his understanding during very difficult time of my study and personal life. I should like to thank Dr. Colin for his help and understanding of my hardship and for his help with my EngD thesis.

I also would like to thank my industrial supervisors, Mr. Joe McDonald and Dr. Steve McCoy, and my industrial bosses, Mr. Jim Madden and Mr. John Debrugha. I should also thank both my programming leaders, Mr. Brendan Reiley and Mr. Robert Lee, especially Mr. Brendan Reiley as they taught me my programming skills.

I am very grateful to my supervisors and bosses for their patience and consideration when I was on maternity leave and later was looking after my son alone.

I should also like to thank my fellow research engineer, Dr. Claire Palmer, for sharing with me her considerable experience in doing research work and her friendship.

I should also like to thank my family for their support during this project.

This work was supported by the Engineering and Physics Research Council, UK and Hazid Technologies Ltd., UK.

ABSTRACT

Process plants that produce chemical products through pre-designed processes are fundamental in the Chemical Engineering industry. The safety of hazardous processing plants is of paramount importance as an accident could cause major damage to property and/or injury to people. HAZID is a computer system that helps designers and operators of process plants to identify potential design and operation problems given a process plant design. However, there are issues that need to be addressed before such a system will be accepted for common use.

This research project considers how to improve the usability and acceptability of such a system by developing tools to test the developed models in order for the users to gain confidence in HAZID's output as HAZID is a model based system with a library of equipment models. The research also investigates the development of computer-aided safety applications and how they can be integrated together to extend HAZID to support different kinds of safety-related reasoning tasks.

Three computer-aided tools and one reasoning system have been developed from this project. The first is called "Model Test Bed", which is to test the correctness of models that have been built. The second is called "Safe Isolation Tool", which is to define isolation boundary and identify potential hazards for isolation work. The third is an Instrument Checker, which lists all the instruments and their connections with process items in a process plant for the engineers to consider whether the instrument and its loop provide safeguards to the equipment during the hazard identification procedure. The fourth is a cause-effect analysis system that can automatically generate cause-effect tables for the control engineers to consider the safety design of the control of a plant as the table shows process events and corresponding process responses designed by the control engineer.

The thesis provides a full description of the above four tools and how they are integrated into the HAZID system to perform control safety analysis and hazard identification in process plants.

KEYWORDS

Knowledge representation; Testing; Process system engineering; Process control engineering; Maintenance safety; Safe isolation; Cause-effect analysis; Safety analysis function evaluation chart; Hazard identification; Hazard and operability study.

PREFACE

This thesis represents the research conducted between 2005 and 2009 to fulfil the requirements of an Engineering Doctorate (EngD) at the Centre for Innovative and Collaborative Engineering (CICE), Loughborough University, UK. The research was undertaken within an industrial context and was sponsored by the Engineering and Physical Sciences Research Council (EPSRC), UK and Hazid Technologies Ltd., an engineering software company, and an Overseas Research Student Scholarship (ORS).

The essence of the Engineering Doctorate program is to produce doctoral graduates that can drive innovation in the engineering industry with the highest level of technical, managerial and business competence. It gives the student a chance to work in a real industry environment and solve the real industry problems and offers a combination of research and commercial skills. It helps the development of innovative thinking, while tackling real industrial problems.

The EngD is examined on the basis of a thesis of about 18,000-22,000 words, supported by publications or technical reports. This thesis is supported by a journal paper and three conference papers produced as part of the research. These papers are attached as appendices to the thesis. The papers are an integral part of, and should be read when referenced in conjunction with, the thesis.

USED ACRONYMS / ABBREVIATIONS

CAEX	Computer-Aided Engineering eXchange
CSV	Comma-Separated Values
DBB	Double Block and Bleed
HAZOP	HAZard and OPerability study
ISO	International Standardization Organization
OPC	Off –Page-Connector
PCE	Process Control Engineering
P&IDs	Piping and Instrumentation Diagrams
PRV	Pressure Relief Valve
SAFE	Safety Analysis Function Evaluation chart
SAX	Simple API for XML parser
SBB	Single Block and Bleed
SPPS	Smart Plant Process Safety
XML	eXtensible Markup Language

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF PAPERS

The following papers, included in the appendices, have been produced in partial fulfilment of the award requirements of the Engineering Doctorate during the course of the research.

PAPER 1 (SEE APPENDIX A)

An, H., McCoy,S.A., Chung, P.W.H., McDonald, J., and Madden, J., 2007. A model test bed to verify the correctness of safety-related behavioural knowledge in a system for automated hazard identification. *Proceedings of the 17th Advances in Risk and Reliability Technology Symposium (AR^{2}TS) (Ed L.Bartlett)*, Loughborough University, UK, 17-19 Apr, 2007, pp. 204-218.

PAPER 2 (SEE APPENDIX B)

An, H., Chung, P.W.H., McDonald, J., and Madden, J., 2008. A Computer Tool to Support Safe Isolation for Maintenance. *Proceedings of the 2nd World Conference on Safety of Oil and Gas Industry,* Texas, USA, 28-29 Oct 2008, pp.406-415.

PAPER 3 (SEE APPENDIX C)

An, H., Chung, P.W.H., McDonald, J., and Madden, J., 2009. Automated Cause-Effect Analysis for Process Plants. *Proceedings of the ninth International Conference on Chemical & Process Engineering (ICheaP-9),* Rome, Italy, 10-13 May 2009, pp.1281-1287.

This paper was then selected to publish with its full-length in the Italian Association of Chemical Engineering (AIDIC) Conference Series Vol.9, Reed Business International. The publication of the Volume is expected by the end of January 2010.

PAPER 4 (SEE APPENDIX D)

An, H., Chung, P.W.H., McDonald, J., and Madden, J., 2009. Computer-aided identification of isolation boundary for safe maintenance and Cause-Effect analysis for assessing safeguards. *Int. J. Process Systems Engineering*, 1(1), pp.29–45.

1 INTRODUCTION

This Engineering Doctorate project investigated and explored model verification, hazard identification for safe isolation, instrumentation identification and control design analysis. It developed and implemented three computer-aided tools and one automatic reasoning system. They are integrated with an existing knowledge-based system, HAZID, which is a product of the sponsor company.

The project collaboration is between Hazid Technologies Ltd. and the Computer Science Department, Loughborough University.

This introduction begins with the project aim follows by a brief description of the industrial sponsor, Hazid Technologies Ltd. A project overview and the structure of the thesis are then described.

1.1 AIM

Process plant safety is always a top priority in the process industry worldwide. Carrying out comprehensive hazard identification at both design and operation stages is essential. HAZard and OPerability (HAZOP) study is a method that is widely used for identifying hazard and operability problems. However, it is well known that HAZOP meetings are time-consuming, labour-intensive and expensive. These negative features make it a suitable candidate for computer support.

HAZID is a knowledge-based system that helps to identify potential design and operation problems given a process plant design. It does this by automating the traditional HAZOP study to make the HAZOP meetings more effective and shorter.

However, there are issues that need to be addressed before such a system will be accepted for common use.

The aim of this project is to improve the usability and acceptability of HAZID system by developing new tools for the existing system to make it more useful and more powerful.

The project considers:

- Developing a tool that can test the developed models in order for the users to gain confidence in HAZID's output as HAZID is a model based system with library of equipment models.
- Extending the use of electronic P&IDs for control safety analysis.
- Developing computer-aided safety applications that can extend the HAZID system to support different kinds of safety-related reasoning tasks.

These form the project's main contributions to knowledge. This thesis represents an attempt to develop and apply new computer-aided tools to model verification, process control safety analysis and maintenance-related hazard identification.

1.2 HAZID TECHNOLOGIES LTD.

Established in 2001 as an attempt to extend and commercialize a computer-aided tool for process plants originally developed at Loughborough University, Hazid Technologies Ltd. has grown to be an advanced technology company that leads the HAZOP automation technology worldwide. The organizational culture in Hazid Technologies Ltd. is future oriented, task focused and cooperation emphasized. It sets a high target and gives people freedom to explore the ways to achieve it. The working environment is relaxing yet commitment to work is required and respected. People are encouraged to build good and simple relationships and co-operate within a team.

The product of the company, HAZID, is a state-of-the-art knowledge-based system to provide facilities for automating the HAZOP process, a widely used hazard identification method throughout the process industry. It helps to improve speed, consistency, accuracy and completeness of HAZOP studies while reducing time consumed on comprehensive manual study.

HAZID system can be used by many different process industry sectors, e.g. oil, gas, petrochemical, chemical, nuclear, pharmaceuticals, power, and similar manufacturing operations etc. Its users can be process engineers, control engineers, process plant operators, plant managers, safety professionals, industrial consultants, insurers and regulators who have the responsibility to assess potential hazards for process plants. The mission statement of Hazid Technologies Ltd. is "Make HAZID the dominant technique for process safety analysis". The company's sales slogan is "Engineered for safety".

The company's collaboration with Intergraph Corporation in the US has given the company access to many potential customers. Intergraph is a leading global provider of Spatial Information Management (SIM) software. Intergraph's SmartPlant P&ID, which is an intelligent CAD system, has been integrated with the HAZID system as its main input. HAZID system is being marketed worldwide as Smart Plant Process Safety (SPPS) by the Intergraph Process & Marine division.

In order to keep a leading place in the HAZOP automation, the company is developing new products to further enhance its usability and acceptability, including the four tools developed in this research project, which are the main topics of this thesis.

1.3 PROJECT OVERVIEW

This research project consists of four "mini" projects. The first project is to develop an integrated tool with the existing Model Builder (MBuilder) in HAZID to test the correctness of the models built. The second project is to develop a tool to help with hazard identification for isolation work. The third project is to identify the instrument loop and its connection with the process items for the engineers to consider whether the instrument and its loop provide safeguards to the equipment. The fourth project is to develop a rule-based system to automate the generation of the cause-effect table for control engineers to consider the design of the control of a process plant.

As a result of these four "mini" projects, four computer-aided tools are developed and integrated with HAZID to make it more useful and powerful. The first is called "Model Test Bed", which is to test the correctness of models built. The second is called "Safe Isolation Tool", which is to define isolation boundary and identify potential hazards for islation work. The third is an Instrument Checker, which lists all the instruments and their connections with process items in a process plant. The fourth is a cause-effect analysis system that automatically generates cause-effect tables for process plants. All tools have been tested and results have been verified by the process engineers in the sponsor company.

Figure 1.1 shows the four tools and how they are related to each other. "Model Test Bed" serves to test the models built in the HAZID which will then call HAZOP analysis to perform the HAZOP results. Output of “Instrument Checker” is used to check the safeguards in the HAZID and also serves as input to the cause-effect analysis system. The isolation tool can be used in conjunction with the cause-effect analysis system to produce control safety analysis for the isolation boundary before actual isolation process is carried out.

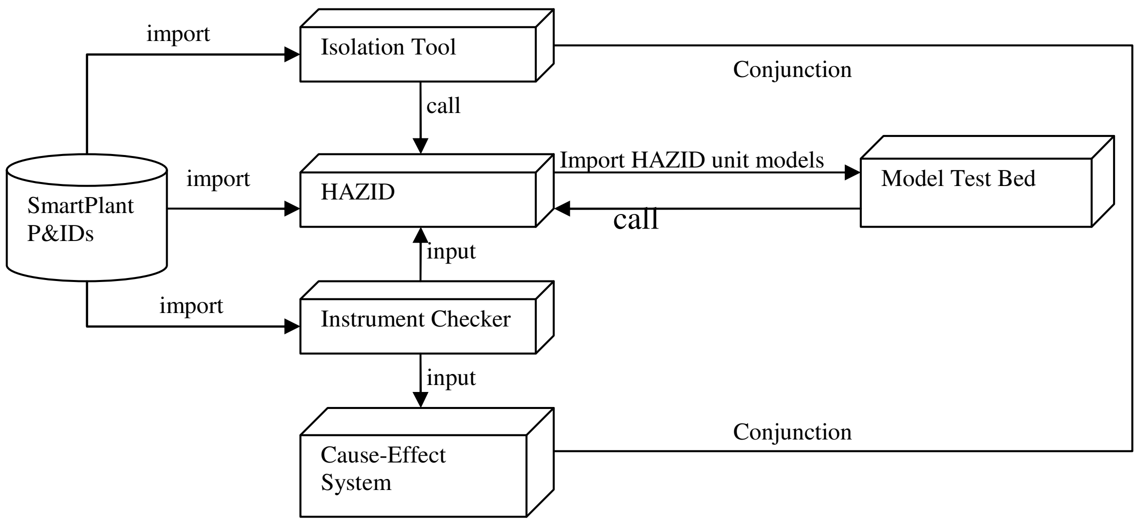


Figure 1.1 Project Overview

All the above tools make use of electronic Piping and Instrumentation Diagrams (P&IDs) that are increasingly being made available by CAD systems, thus extending intelligent CAD applications in the area of process safety. A P&ID is a Piping and Instrumentation diagram that illustrates a process plant in terms of its flow direction, equipment items and signal connection as well as control methods etc. Currently, HAZID takes input from SmartPlant P&ID. In principle it can take input from any intelligent CAD systems.

The four tools are developed using C++, Php, Oracle database and SQL. CLIPS inference engine as a general purpose rule engine (Giarratano & Riley 1994) is used for the cause-effect analysis system.

1.4 STRUCTURE OF THE THESIS

Chapter 2 describes the conventional HAZOP technique. Computer-aided HAZOP automation is discussed. The HAZID system is introduced. A brief comparison with other HAZOP automation tools is given.

Chapter 3 reviews the current modelling technology and introduces in detail the modelling verification tool “Model Test Bed”.

Chapter 4 gives details about how the three safety-related computer-aided tools are developed and how they are related to one another.

The final chapter describes the contributions made by this thesis. Limitations of the work are considered and possibilities for future work are discussed.

2 HAZOP

Within the process industry, attention has been paid to hazard identification from the early stage of design throughout the whole life cycle of the plant with the aim of protecting workers, the public and the environment. Many techniques have been used for the hazard identification process, such as “What if”, “Interaction matrix”, “Zonal analysis”, “Checklists”, “Fault Modes and Effect Analysis (FMEA)”, “Event Tree Analysis (ETA)”, “HAZOP” etc (Redmill, et al., 1999). Among them, HAZOP has been widely recognized as the best and most rigorous hazard identification method throughout the world. It applies a procedure called "guideword-consequence tracing" on each equipment item in a plant to comprehensively check what fault(s) occurring in the item would cause a deviation from its normal operating conditions, such as an increase in pressure or a decrease in temperature and deduce the consequence(s) of each deviation.

2.1 THE BASIC METHODOLOGY

The basic process of a conventional HAZOP study starts with the P&ID of a process plant. Once the P&ID has been defined, the following steps apply.

1. Divide the whole plant into sections with nodes that connect them.
2. Give a full description to the first section.
3. Give a specific design intention to the first section.
4. Generate the first deviation.
5. Identify the cause(s) of that deviation (the cause of this deviation might be another deviation, so the root cause(s) need to be identified).
6. Evaluate the consequence(s) (the consequence of this deviation might be another deviation, so the final consequence(s) need to be identified).
7. Consider safeguards (protection).

8. Give recommendations for actions.

9. Generate the HAZOP report.

10. Repeat 4-9 for the next deviation generated until all the meaningful deviations have been considered.

11. Repeat 2-10 for the next section identified until all the sections have been analysed.

12. Finalise the HAZOP report.

According to Crawley et al. (2000), there are two ways of generating deviations, “parameter-first” or “guideword-first”. Parameter is a variable that describes an attribute of an item, such as “Flow”, ”Pressure”, ”Temperature” and ”Level” of a vessel. A guideword is a qualitative description of the value change of a variable, such as “More”, “Less”, “No” and “Reverse”. A parameter can be combined with a guideword to form a particular and meaningful deviation, such as “More flow” and “Less pressure”. The “parameter-first” approach considers in turn all causes of a particular deviation for a parameter and a guideword combination. When that is completed, the same parameter is considered with another guideword to see if a meaningful deviation can be generated. This continues until all the guidewords have been tried. Then it moves on to another parameter and repeats this generating process again until all the parameters have been considered. The “guideword-first” approach is similar but begins with a guideword.

Step 4, 5, 6 establish the fault propagation path from the root cause to the final consequence.

Figure 2.1 shows the manual HAZOP process from step 2 to step 10 as a flow chart for analysing a complete section. More description of the HAZOP study technique can be found in Crawley et al. (2000) and Kletz (1999).

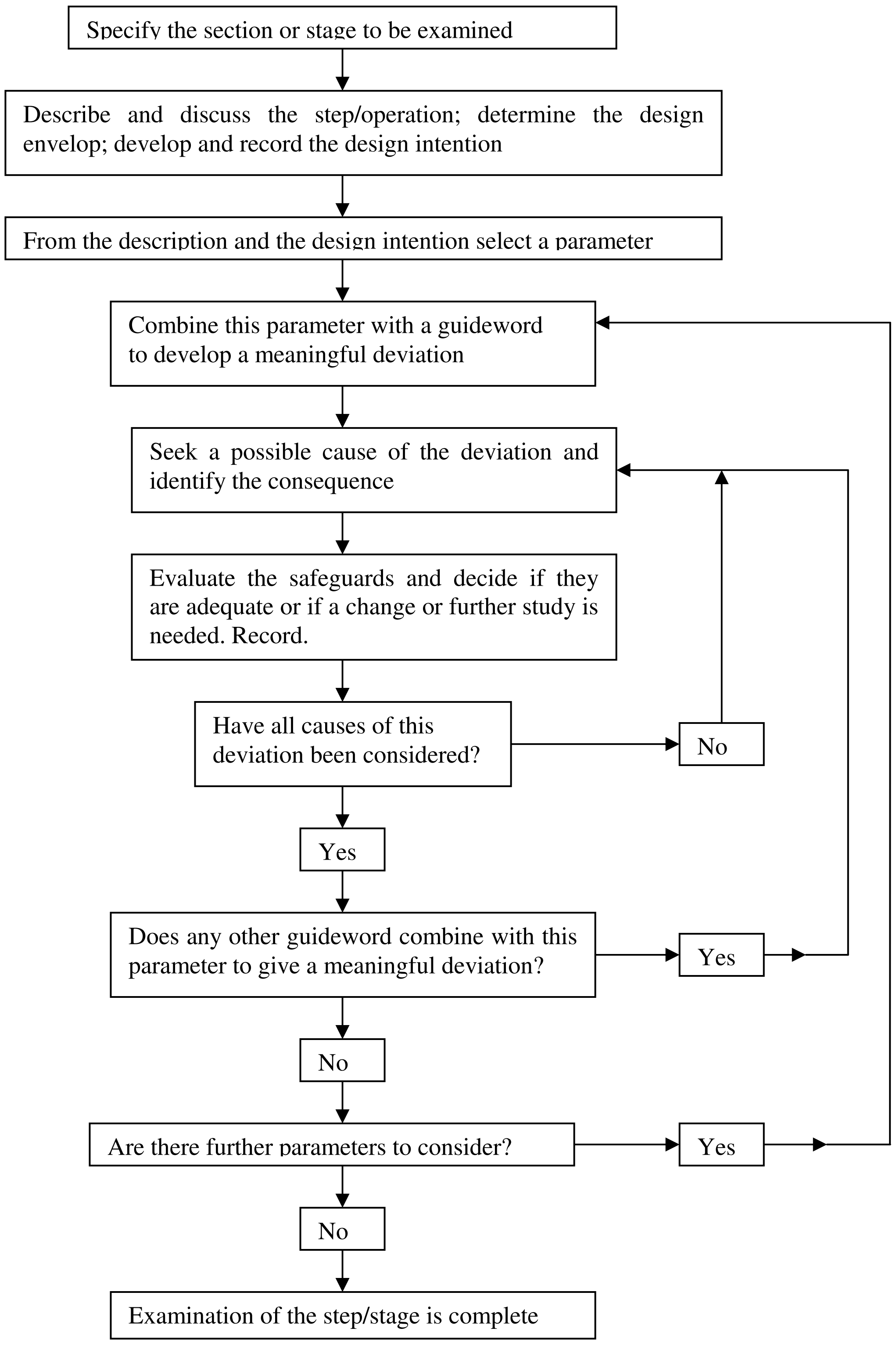

Specify the section or stage to be examined
Describe and discuss the step/operation; determine the design envelop; develop and record the design intention
From the description and the design intention select a parameter
Combine this parameter with a guideword to develop a meaningful deviation
Seek a possible cause of the deviation and identify the consequence
Evaluate the safeguards and decide if they are adequate or if a change or further study is needed. Record.
Have all causes of this deviation been considered?
No
Yes
Does any other guideword combine with this parameter to give a meaningful deviation?
Yes
No
Are there further parameters to consider?
Yes
No
Examination of the step/stage is complete

Figure 2.1 Manual HAZOP process-flow diagram for the HAZOP analysis of a section or stage of an operation-the parameter first approach (taken from Crawley et al. (2000))

2.2 HAZOP SUPPORT TOOLS

From the process illustrated in figure 2.1, it is obvious that a great deal of effort has to be put into the repetitive work of generating deviations, identifying fault paths and analysing one section of the plant to another. Therefore, a computer-aided system could be of great help.

In response to the demand of computer-aid, HAZOP support tools have been developed and made commercially available. These HAZOP support tools make the traditional HAZOP analysis easier, although they do not perform automated HAZOP analysis. Two examples are HAZOP+ 2008 and HAZOP Manager 6.0.

HAZOP+ 2008 developed by Isograph Ltd. provides specifically designed data sheets for HAZOP study. Users can choose the plant sections, edit their deviations, causes and consequences on the data sheets. It offers an editable study grid for the users to easily structure and to record their HAZOP analysis results (Isograph, 2009).

HAZOP manager 6.0 developed by Lihou technical & software service is another HAZOP support tool. Similarly, it focus on supporting the HAZOP meeting results recording, giving easy access to historical information such as previously identified problems, producing formatted HAZOP reports etc. (Lihou technical & software service, 2009).

This kind of HAZOP tools only provides basic support for HAZOP analysis. The knowledge-based approach which can produce HAZOP analysis automatically will be discussed next.

2.3 AUTOMATED HAZOP

In automated HAZOP system, all the steps in figure 2.1, except the first two, are hidden from the user and automatically done by the computer thus removing some of the repetitive work. All the user needs to provide is the plant input and review the output. Based on the results, the user might like to modify the P&ID and re-run the system, and then compare the new results with the previous set.

Before introducing the HAZID system in detail, this section considers several HAZOP automation systems that are built for research purpose. They are surveyed here because they all use qualitative modelling and reasoning in its automation process just as HAZOP study is a qualitative analysis method. The similarities and differences between HAZID system and these other systems can be seen.

CHECKOP (McCoy et al., 2006) is a prototype system developed by Loughborough University in 2003 for checking operating procedures for batch plants. Instead of signed-directed graphs (SDG), modelling in CHECKOP is based on states. CHECKOP models include not only tangible equipment information as that in HAZID models but also dynamic information such as state changes, operation, event sequence, time and assumptions associated with each step of an operation. CHECKOP applies guidewords systematically to an operating procedure to look at variations in the operating instructions of the plant. For example, CHECKOP infers what will be the consequence if a certain instruction is not executed by the application of guideword “No”.

In 2008, CHECKOP was extended and improved based on collaboration between Loughborough University and Hazid technologies Ltd. as a result of another EngD project. The details can be obtained in Palmer (2008).

Hazard Support Tool (**HAST**) is a knowledge-based system to support HAZOP automation in Palermo University (Bartolozzi et al., 2000). It was originally developed for HAZOP analysis of continuous plant but later extended to carry out automatic HAZOP analysis of batch and semi-continuous plant.

Like CHECKOP, HAST has a knowledge base that contains qualitative models of equipment units of process plants. The models in the library include models designed for performing HAZOP analysis for batch plants. A plant is split into very elementary units such as valves, pumps and procedure phases. HAST models are classified as “cause model”, “HAZOP model”, and “consequence model”. A model is a collection of mini-fault trees, which are used to trace a root cause.

For batch plant, HAST analyses different models of the same equipment item existing for each phase. A phase in this tool is a period when an action has to be carried out or a specific objective has to be reached. A predetermined stage or a certain value for a variable must be reached at the end of each phase (Bartolozzi et al., 2000). There are common phase models and specific phase models. Each model inherits features from its parent model. A taxonomy editor is used to implement this inheritance feature. Model information excludes any intermediate mini trees that are only necessary to propagate the deviation. HAST provides a facility to automatically insert available monitoring and protection devices by retrieving their information from the database to include in the HAZOP result.

HAZOPExpert is another expert system designed to automate HAZOP analysis, developed in Purdue University. The HAZOPExpert architecture is described in Vaidhyanathan and Venkatasubramanian (1995). It is important to review this system because it helps develop an understanding of how people design and build up models used by the inference engine and what is behind that engine.

HAZOPExpert is similar to HAZID, although it uses a so called HDG model (HAZOP-Digraph Model) instead of SDG to model the equipment units and their process variables and their behaviours. The plant state is assumed to be steady state like that in HAZID. Both SDG and HDG are extended to cover the nodes of abnormal cause and the nodes of adverse consequences. Both are defined for each process unit model of a plant.

Both HDG and SDG are used to propagate process variable deviations and to find abnormal cause and adverse consequences. They all have conditional arcs in which conditions are attached to some SDG or HDG arcs to deal with conditional propagation causal relationships. For example, the increasing flow rate at the inlet port of a tank will result in the increase of temperature of the fluid in the tank only if the temperature of the fluid at the inlet port is higher than the temperature of the fluid in the tank. On the other hand, the increasing flow rate at inlet port of a tank will result in the decrease of temperature of the fluid in the tank only if the temperature of the fluid at the inlet port is lower than the temperature of the fluid in the tank.

HAZOPExpert models are stored in a library organized in a class hierarchy using an object-oriented framework. Models represent two kinds of information: one is the generic knowledge represented by generic cause and effect models, the other is the process specific knowledge relating to a specific chemical plant, such as the normal physical state of the material, and whether the material is corrosive, flammable, volatile or toxic. HAZID models also represent these two kinds of knowledge but in a different way. The HAZID SDG model represents generic knowledge of an equipment item and the fault, deviation propagation information within and outside of the model unit. The process specific knowledge like the properties of material is supplied (in HAZID) by external software through Fluid Model System (FMS), which is a system that performs fault path validation by using predicates and functions (McCoy et al., 1999 c).

HAZOPExpert has a process-materials-cause procedure attached to HDG arcs to identify root causes of faults like “Leak in the tank” or “tank rupture” or “breakage due to presence of a corrosive process material”. HAZID performs this in another way. It applies the Run Time Condition (RTC) queries attached to SDG arcs, faults and consequences to check on the properties of the process materials using predictions and functions defined in the Fluid Model System as described in McCoy et al. (1999 c).

HAZOPExpert has a graphical representation of models since it uses the real-time expert system shell G2 to do the reasoning work. The graphic tool is provided by this shell.

The above are HAZOP automation systems built for research purpose. The next section focuses on HAZID, which was originally built for research purpose then developed into a commercial system.

2.4 HAZID

As mentioned before, HAZID, a knowledge-based system which automates the process of HAZOP studies is currently the most advanced commercially available auto HAZOP tool. The development of HAZID is given in Rushton et al. (1998) and McCoy et al. (1999a, 1999b, 1999c).

HAZID takes the information from P&IDs as input. It builds a plant model based on the P&ID plant description and a library of equipment models. Deviations are applied to all the equipment items in the plant model. Then it deduces the cause consequence path based on a Signed Direct Graph (SDG) of the plant built from the models. SDG is a network of nodes connected by arcs. Nodes are variables such as "flow", "temperature" or "pressure" in system. Arcs are influences between nodes, of one variable on another. Sign identifies the type of influence attached to an arc, negative or positive. For example, "more flow" will result in "less pressure" if the sign attached to the arc is negative or "more temperature" will result in "more flow" if the sign attached to the arc is positive etc. The variable "signal" can only have "no" as its keyword to form a deviation "no signal". Details about SDG are described in Chung (1993).

Figure 2.2 is the workflow of HAZID. Dashed area A is where the knowledge base is created and formulated. Dashed area B1 is where P&IDs are imported from the database and sections being selected and analysed as well as results being presented in HMeeting, a presentation tool developed in PHP. Dashed area B2 is where the modified P&IDs are again imported from the database and again sections being selected and analysed, and new results being presented in HMeeting. Dashed area C is where the results from B1 and B2 are compared so that the effects of the design change can be easily seen.

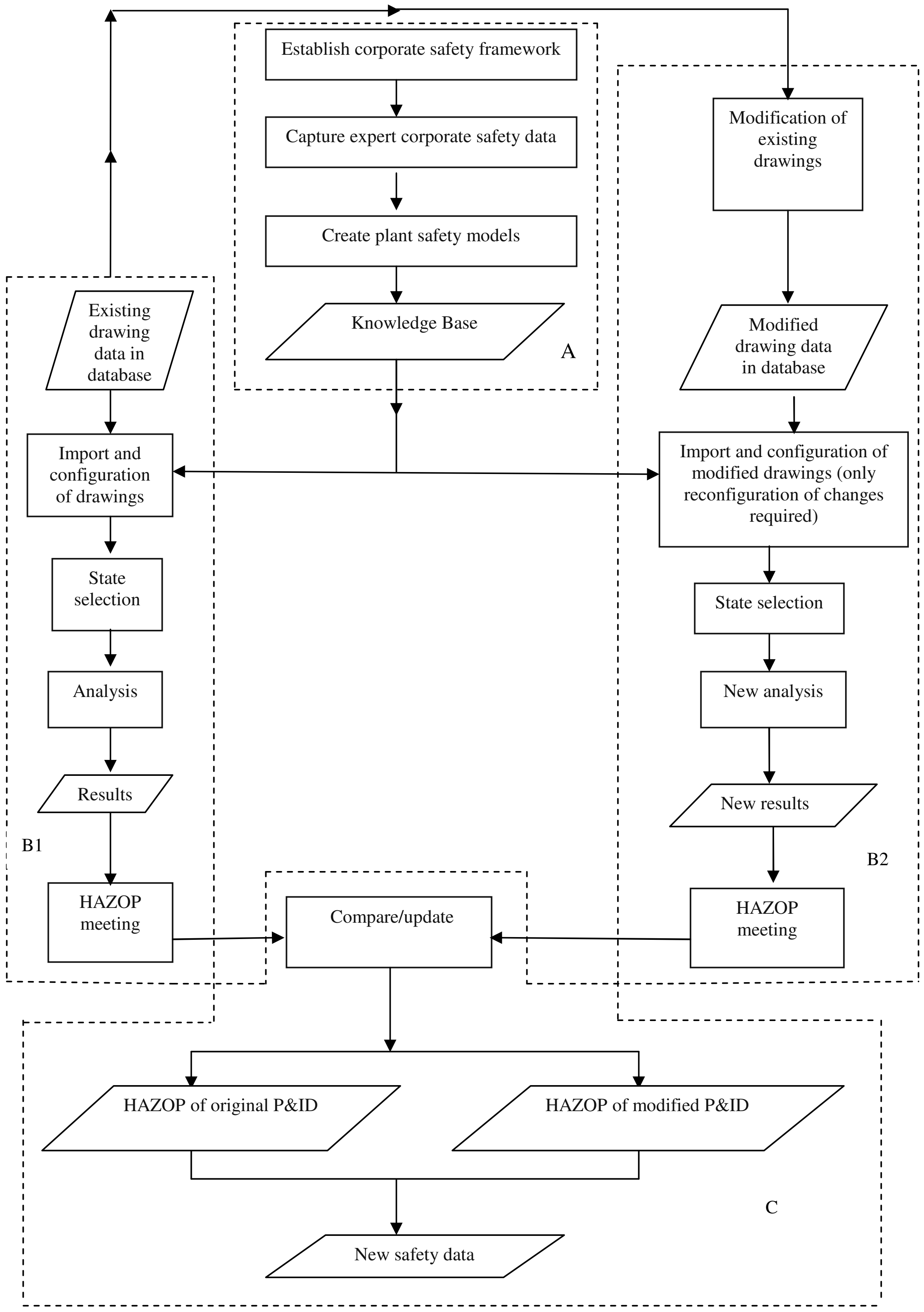

Establish corporate safety framework
Capture expert corporate safety data
Create plant safety models
Knowledge Base
A
Modification of existing drawings
Modified drawing data in database
Existing drawing data in database
Import and configuration of drawings
State selection
Analysis
Results
B1
HAZOP meeting
Import and configuration of modified drawings (only reconfiguration of changes required)
State selection
New analysis
New results
B2
HAZOP meeting
Compare/update
HAZOP of original P&ID
HAZOP of modified P&ID
C
New safety data

Figure 2.2 Workflow of HAZID system

From the above description, it is clear that the main function of the HAZID is to automate the HAZOP analysis and produce the results. However, it lacks the facility to verify the correctness of the knowledge base. For example, the models in the library may not behave as the developer intended if not tested before being used. Also, safety issues related to maintenance work are not considered. Furthermore, there is no support for the design of safe process control.

As mentioned in chapter1, the research project reported in this thesis has built four tools that integrate with HAZID to enhance its usability and acceptability by addressing the above identified problems. Chapter 3 and 4 give full details.

2.5 SUMMARY

In this chapter, the basic HAZOP methodology is described, showing that its repetitious nature could be automated using state-of-the-art computer-aided technology. Computer support tools for HAZOP are commercially available. However, they are not knowledge-based system that can automate the HAZOP analysis process.

Several HAZOP automation tools for research purposes have been reviewed, CHECKOP, HAST and HAZOPExpert. A common feature of the above systems is that they are all knowledge-based systems, and they all require models to represent equipment units and simulate their behaviour using qualitative reasoning approach since conventional HAZOP analysis is performed qualitatively. Two of these systems are extended to deal with batch plants.

HAZID is currently the most advanced and only commercialised HAZOP automation tool. It has been introduced in terms of its workflow, modelling, reasoning methods, showing the need to verify the knowledge base and extend the system to support other safety-related applications.

3 MODELLING IN COMPUTER-AIDED HAZOP STUDY

3.1 MODELLING TECHNOLOGY

When a computer system is used to perform a task, the system must have all the necessary information to carry out that task and organize the information in an effective way that helps carry out the task successfully in terms of accuracy, speed and completeness etc.

Modelling is one of many ways to meet the above requirements. A comprehensive review of modelling techniques was produced as a coursework for the EngD program. It is included in this thesis as appendix E. This chapter provides only a short description version and introduces a novel tool that helps with the verification of models.

3.1.1 THE MODELLING GOAL

Although modelling is about knowledge representation and organization, the objective of modelling for various tasks could be very different. When the task is to help identify hazards in the process industry, the modelling goal should be able to describe the consequences of all possible causes and deviations and be able to provide the effect of preventive actions for each entry of a given HAZOP result table (Nemeth et al., 2005).

The first question is what makes a good model? Features that make a good model include:

- Accuracy. A good model should be able to represent the information as accurately as possible.
- Completeness. A good model should capture the information as completely as possible. For example, the functionality must be considered at the model design stage, in both normal and abnormal states (Bartolozzi et al., 2000). In other words, a good

model should be able to simulate all the behaviour of the object to be modelled. For example, when modelling fault path propagation in a process plant, the model should contain the information of propagating deviations with both directions from upstream to downstream and from downstream to upstream.

- Conciseness. A good model should not contain information that is irrelevant to fulfil the application objective (Palmer & Chung, 1997). As a result, the size of a good model could be kept as small as possible.
- Generality. A good model must have high levels of flexibility and generality that can be applied on any objects to be modelled with different configurations. For example, in the case of automating HAZOP study, a good model must be able to represent a plant item of a certain class with different states.
- Reusability. A good model should be as independent as possible from plant specific details so that it can be used widely in different plants.
- Simplicity. A good model should be as simple as possible.
- Ease. The maintenance of a good model should be as easy as possible.
- Balance. A good balance among expressive power of the model, complexity of building models, and computational cost of driving simulations must be properly maintained (McCoy et al., 2006).

In terms of using and managing the model, computer-aided modelling systems must have facilities to validate the models built (Palmer & Chung, 2000).

In a nutshell, a good model should make the modelling effort as minimal as possible for fast and easy deployment of any application (Venkatasubramanian, et al., 2003a).

3.1.2 Qualitative Modelling and Quantitative Modelling

Current modelling methodology can be broadly classified as qualitative and quantitative.

Qualitative modelling is a non-numerical description of a physical system. In Palmer and Chung (1997), qualitative models are defined as abstract representations of a plant that are able to simulate behaviour of the plant by emphasizing a causal explanation. The SDG-based modelling and state-based modelling used in HAZID and CHECKOP are examples of qualitative modelling.

Quantitative modelling, on the other hand, could be defined as a numerical description of a physical system. Models of this kind includes “Black-box models”, “First-principles models”, and “Frequency response models”. “Black-box models” includes general input-output models and state-space models (Venkatasubramanian et al., 2003 a).

However, the modelling technology used in an application does not have to be either qualitative or quantitative. A qualitative modelling approach could be slightly or partly quantitative when the precise values of some variables are demanded for some good reason. For example, when operation is being modelled, an action that is taken “too early” or “too late” will be the deviation of action time. But in some applications how early is “too early” and how late is “too late” may need to be defined. On the other hand, it is possible to have qualitative modelling to be added into some part of a quantitative modelling whenever necessary. In the quantitative modelling review by Venkatasubramanian et al. (2003a), a method of computing residual is used to capture the failure in a system. It examines what the expected behaviour should be and what the real-time value or actual behaviour is. The value of residual is obtained by the discrepancy between these two values. When the value of residual is zero, it means no failure happens in the system. But if serious failure happens or the underlying system is changed or the real-time system is completely or partly modified without updating the modelling system, the value of residual will reach a very critical level. A qualitative value of "too high" can indicate the critical difference and a value of "Yes" or "No" can indicate if there is a difference.

When designing models, which modelling approach is used depends on the objectives of the system the models are designed for. In the example above, if the modelling objective is to tell if there is fault and the extent of the problem then a numerical value of residual is needed. On the other hand, if the modelling objective is to tell whether a change has occurred then an indication of an increase of the residual or decrease of the residual is sufficient.

Qualitative models use qualitative functions to express the relationship between inputs and outputs of each unit of a physical process. Quantitative models, on the other hand, use mathematical functions to express the relationships between inputs and outputs of a system to describe a physical process (Venkatasubramanian, et al., 2003a).

Emphasis in a qualitative system is how to obtain comprehensible models that are able to intuitively explain the system behaviour just as the mental model of a human expert is capable of (Bratko & Šuc, 2003). Advantages of qualitative modelling can be briefly summarised as:

- The behaviours of qualitative models can be induced even if no mathematical models are available since qualitative models do not require precise numerical values of process variables (Venkatasubramanian et al., 2003b).
- The format of a qualitative model can either be a qualitative causal model or an

abstraction hierarchy (Venkatasubramanian et al., 2003b).

- Explanations for physical behaviour in a qualitative model are more causal and intuitive, therefore providing better insight into the working mechanism of the physical object being modelled (Bratko & Šuc, 2003).
- The highly abstract level of qualitative models makes it possible to represent more than one behaviour at a given variable value because it does not require a precise value, "more" or "less" or "none" is usually sufficient. On the other hand, quantitative models represent only one behaviour at a given variable value because it requires a precise numerical value (McCoy, et al., 1999a).
- Qualitative modelling can detect subtle qualitative dependences and trace their effect. For example, more temperature is more no matter how much more it is, and it will be regarded as a deviation which may propagate to cause a significant consequence. Quantitative modelling in this case may ignore the change of the temperature when the change is very small and consider it as noise (Bratko & Šuc, 2003). It is also easier to check if the model can perform the desired behaviour qualitatively rather than quantitatively (Schaich et al., 2001).

However, the main problems with qualitative models are that:

- they may contain ambiguities;
- they may cause the generation of spurious solutions.

These disadvantages may be overcome by combining qualitative and quantitative modelling, where qualitative modelling is the main modelling method and quantitative modelling is applied where necessary.

3.1.3 Model Representation in HAZOP Automation

A model represents two kinds of information: description of the object being modelled and representation of the behaviour of the object being modelled. Some modelling approaches represent the behaviour by using parameters, variables, key words etc. (Nemeth et al., 2005).

The development of qualitative model representation for HAZOP analysis has gone through from representing continuous plants, which assumes that each equipment item is kept in a steady state all the time, to representing batch plant operation, which includes not only the dynamic change of states over time, but also the event or action sequence, time and operator's action.

First, in order to represent the information accurately and completely, a model has to be based on a fundamental understanding of the physical system to be modelled. Then this understanding will be expressed in different ways according to the modelling approach used (Venkatasubramanian, et al., 2003b).

In the HAZID system, model information is represented in various slots for each equipment type. There are inPort slots, outPort slots, unitPort slots that describe internal ports information and comp_connection slots that describe flow information.

The behaviour of models in HAZID is done through fault propagation. There are many ways to describe fault propagation, such as functional equations, program rules, logical expressions, truth tables, fault trees, event trees, and reliability block diagrams, influence graphs, SDGs and bond graphs (Palmer & Chung, 2000).

Fault path propagation in HAZID is described using four types of SDG arcs. They are

- "Deviation" to "Deviation"
- "Fault" to "Deviation"
- "Deviation" to "Consequence"
- "Fault" to "Consequence"

These arcs represent the plant behaviour in a qualitative way.

As mentioned in section 2.3, in Hazard Support Tool (HAST) (Bartolozzi et al., 2000) the behaviour of an equipment unit is represented using three types of models: cause model, HAZOP model and consequence model. Among them, "cause model" represents the root cause of fault, "HAZOP model" represents the fault propagation, and "consequence model" represents the effect of the root cause. The connection between consequence model and the cause model is the deviation with the effect of the consequence.

However, knowledge representation ambiguities can happen in a qualitative model built using SDGs. When qualitative models are combined to form a system, the result may contain multiple paths that lead to ambiguities (Palmer & Chung, 2000).

Venkatasubramanian et al. (2003b) states that ambiguities of knowledge representation can only be completely eliminated by the use of actual quantitative values. However, obtaining actual quantitative values for the purpose of solving ambiguities would largely reduce the advantage of using qualitative reasoning. Some researchers have proposed various ways to solve this problem.

One way to eliminate ambiguities is by choosing the shortest path when and only when the shortest path does have a correct influence (Palmer & Chung, 2000). Another way is by adding constraint representation (McCoy et al., 2006). In the Equipment Model Builder (EMB) tool presented in Palmer and Chung (2000), a modular method is used to remove ambiguities in qualitative SDG models. It identifies the header/divider combinations and the recycle loops, then specifies a module for each of them, then substitutes the module for each of them into the plant description.

3.1.4 HOW ARE MODELS ORGANIZED?

As mentioned before, knowledge organization is an important part of modelling. A good modelling structure can help the system to reason effectively and easily.

In the HAZID, a Unit Model Library (UML) is built and models in the library are organized in a hierarchical way that supports the inheritance of features between models. Each model has one and only one parent model and it inherits all the features of its parent model but can have its own unique feature(s) that distinguish it from its sibling model(s). A model may be a parent model for other model(s). When a new model is created, it will be added into this library as a child of an existing model. The structure of the library is like a family tree. In this way the size of the library can be kept as small as possible and features of each model kept clear. A brief description of the hierarchical structure of HAZID models is given in paper 1, "A model test bed to verify the correctness of safety-related behavioural knowledge in a system for automated hazard identification", An et al. (2007), in appendix A. A detailed description of UML organization is given by McCoy et al. (1999b).

3.2 MODEL TEST BED

3.2.1 INTRODUCTION

A seven step modelling procedure is proposed by Nemeth et al. (2005):

“(1) Model goal-set definition (modelling problem specification).

(2) Model conceptualization (identifying controlling factors).

(3) Modelling data: needs and sources.

(4) Model building and model analysis.

(5) Model verification.

(6) Model solution.

(7) Model calibration and validation.”

These seven steps suggest that it would be very helpful if we could have computer tools designed for model definition, model construction and model verification. Most research in modelling technology combines model definition, model building and model verification into a single tool. However, HAZID’s model builder does not perform the model verification function and no model verification tool is available in general. In this chapter, a model test bed which is developed for model verification for HAZID is described.

The model test bed is integrated with MBuilder, HAZID’s model builder, to test the correctness of models built in the Unit Model Library. It allows the behaviour of the model under test to be compared to its expected behaviour, as specified by a domain expert. This section establishes the significance of such a testing tool by comparing it with the manual approach. The tool’s methodology, architecture and data structure are described. The presentation of test results is also discussed by exploring different result formats so that the user can consider the model from different points of view.

The existing MBuilder only helps engineers create models. However, a model might have wrong or missing behaviour or some extra behaviour that is not expected by its builder. An extra behaviour, which may be right or wrong, is one that is not considered by the process engineer when constructing the model but is revealed by HAZID. Therefore, it is important to identify all the above to ensure that a model behaves exactly as intended by its builder.

3.2.2 MODEL TESTING METHODOLOGY

3.2.2.1 Unit Models and Model Library

In this HAZOP emulation system, a plant is represented as a network consisting of a series of interconnected units. Each type of unit is described as a model built using the SDG representation and is stored in a library named Unit Model Library (UML). A plant description file is derived from a given P&ID. For example, figure 3.1 shows that "test_unit" is an instance of the "'centrifugal pump'" unit model.

```
//"test_unit" is an instance of the "'centrifugal pump'" unit model
instance('test_unit' isa 'centrifugal pump', [
                    inPorts info [       //there are three in ports for this model,
                              'ignoreIn', // one is called 'ignoreIn',
                              'in' ,        // one is  called 'in',
                              'sealIn'      // one is called 'sealIn'.
                    ],
                    outPorts info [          //there are four out ports for this model,
                              'ignoreOut',   //one is called 'ignoreOut',
                              'out' ,        //one is called 'out',
                              'drainOut' ,   //one is called 'drainOut',
                              'ventOut' ,    //one is called 'ventOut',
                              'sealOut'      //one is called 'sealOut'.
                    ],
                    unitPorts info [      //there is no internal port for this model
                    ],
                    propLinks info [   //fault path propagation information
                       % faults
                              //fault 'loss of drive' will cause 'less pressure' at port 'out',
                              arc([fault,['loss of drive',use_is_not_standby]],-1,['out', 'pressure']),
                              //fault 'loss of drive' will cause 'more pressure' at port 'in',
                              arc([fault,['loss of drive',use_is_not_standby]],1,['in', 'pressure']),
                              //fault 'composition' at port 'in' will cause 'composition' at port 'out'.
```

```
                        arc(['in', 'composition'],1,['out', 'composition']),
                        % propagation
                        //deviation 'noFlow' at port 'in' will cause 'noFlow' at port 'out',
                        arc(['in', 'noFlow'],2,['out', 'noFlow']),
                         //deviation 'noFlow' at port 'out' will cause 'noFlow' at port 'in',
                        arc(['out', 'noFlow'],2,['in', 'noFlow']),
                         //deviation 'more/less temperature' at port 'in' will cause 'more/less
                        //temperature' at port 'out',
                        arc(['in', 'temp'],1,['out', 'temp']),
                         //deviation 'more contamination' at port 'in' will cause 'more
                        //contamination ' at port 'out',
                        arc(['in', 'contamination'],2,['out', 'contamination']),
                        //deviation 'more solid' at port 'in' will cause 'more solid' at port 'out'.
                        arc(['in', 'solid'],2,['out', 'solid'])
                                  ]
                             ]
                  ).
```

This model states that the “test_unit” is a centrifugal pump, it has 3 in ports as listed in the slot “inPorts info”, and has 5 out ports as listed in the slot “outPorts info”. It does not have any internal port as none is listed in the slot “unitPorts info”. The “propLinks info” slot stores a list of arcs that define the mini-SDG related to a centrifugal pump. “-1”,“1” and “2” are influences. The sign “-1” indicates a reversed influence. It means that a higher input will result in a lower output and vice versa. The sign “1” indicates a direct influence. It means that a higher input will result in a higher output and similarly a lower input will result in a lower output. The sign “2” means that a higher input will result in a higher output but does not indicate an effect of lower input. For example,“ arc(['in', 'solid'], 2, ['out', 'solid'])” means an increase of solid at the in port “in” will cause an increase of solid at the out port “out”, but a decrease of solid at the in port does not mean that there is going to be a decrease of solid at the out port.

As mentioned in section 3.1.4**,** models for unit classes are organized into a hierarchical structure in which a child model inherits all the characters of the parent model but it can have its own characteristics that distinguish itself from other sibling models. For instance, in table

3.1, “loss of drive” in the “Rotary pump” model will cause “revFlow” at port “out”, this is inherited by both child models “centrifugal pump” and “double seal centrifugal pump”. On the other hand “leaks to seal” is a cause of “morePressure” only through the child model “double seal centrifugal pump” not through “centrifugal pump”. In addition, each child model unit may have its child unit as well.

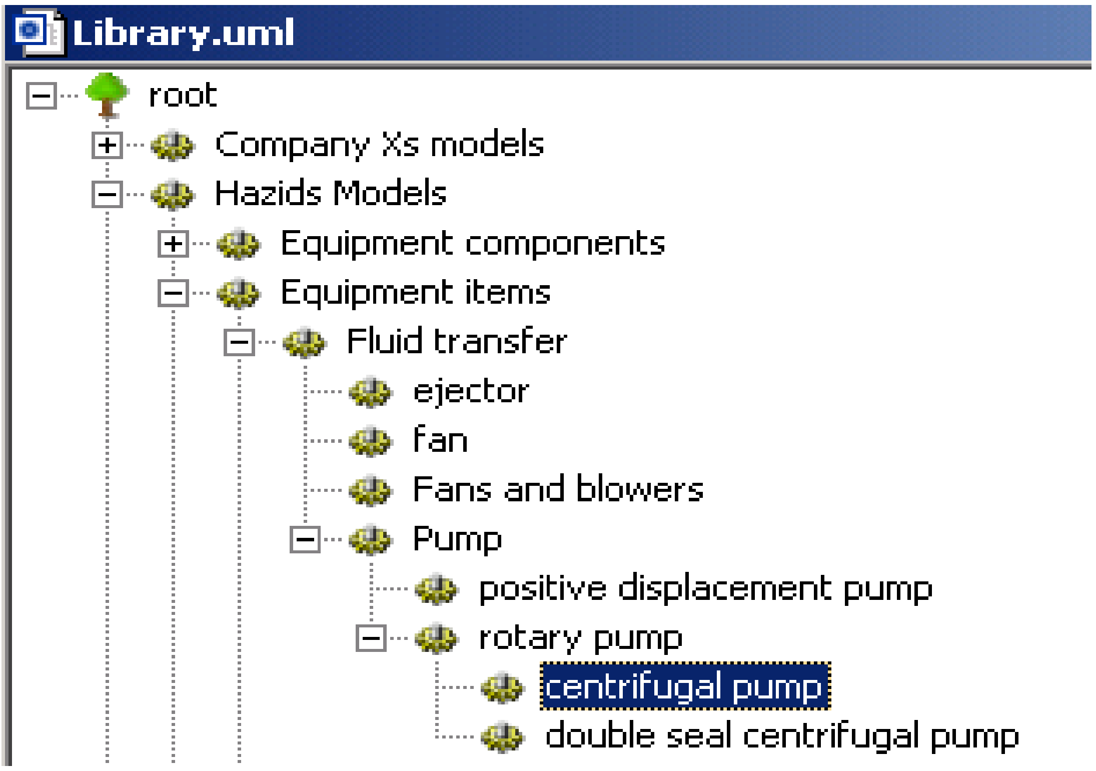


Figure 3.1 Part of the Unit Model Library (UML)

	Parent model	**Child models**	
	Rotary pump	Centrifugal pump	Double seal centrifugal pump
Inherited arc	“loss of drive” causes “revFlow” at port “out”		
Specific arc for seal centrifugal pump			“leaks to seal” causes “morePressure” at port “seal1out”

Table 3.1 An example of inheritance in the Unit Model Library (UML)

The use of inheritance to build the unit model library keeps the unit model as simple as possible, avoids duplication, and eases the task to build new models, as only the specific feature(s) needs to be defined for any child models.

The unit model structure is described in detail by McCoy et al. (1999b). The unit-based approach is widely used, for example, by Vaidhyanathan and Venkatasubramanian (1995). Another model based approach applying qualitative physics in process safety is described by Catino and Ungar (1995).

3.2.2.2 The Use Case Analysis

The user of the model test bed will be a trained user of the MBuilder, with the knowledge of constructing unit models and is responsible for developing models for HAZID for their organization. This user may be an experienced process engineer. During model development the user will test individual models in order to ensure that the model under construction does not have any unintended behaviour or does not lack any intended behaviour. The intended behaviour of each model needs to be stated explicitly in order to verify the actual behaviour of a test. The user may test a model after making a change to it so that there will be a result report to show any differences between the expected and the actual results. The user may then either go back to make further modifications to rectify any problems in the model or sign off the model as acceptable. The user may apply different conditions when testing a model and the expected differences in the results will need to be stated. The user may test more than one model at a time, so a summary report for all tested models and an individual report for each model will need to be generated.

3.2.2.3 The Interface

Since the model test bed is designed to be used by engineers responsible for developing HAZID models, the interface for initiating tests is closely integrated with MBuilder. A facility for selecting which model is to be tested is shown in figure 3.2.

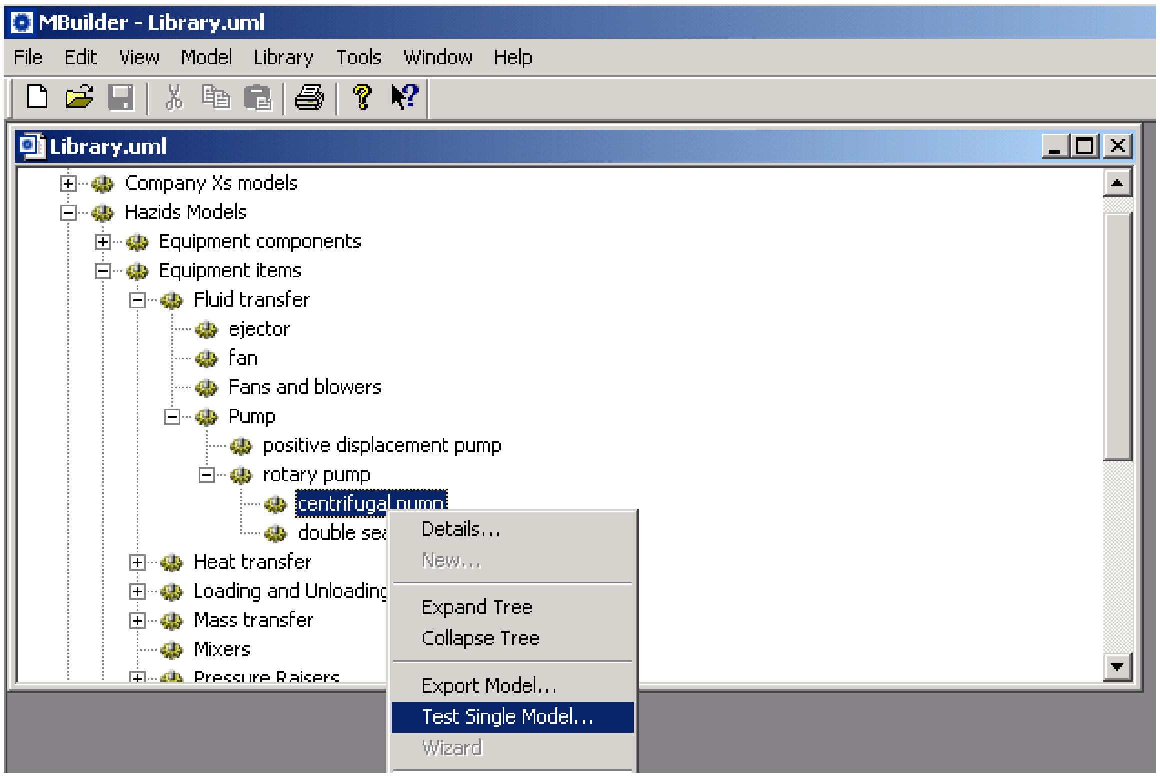
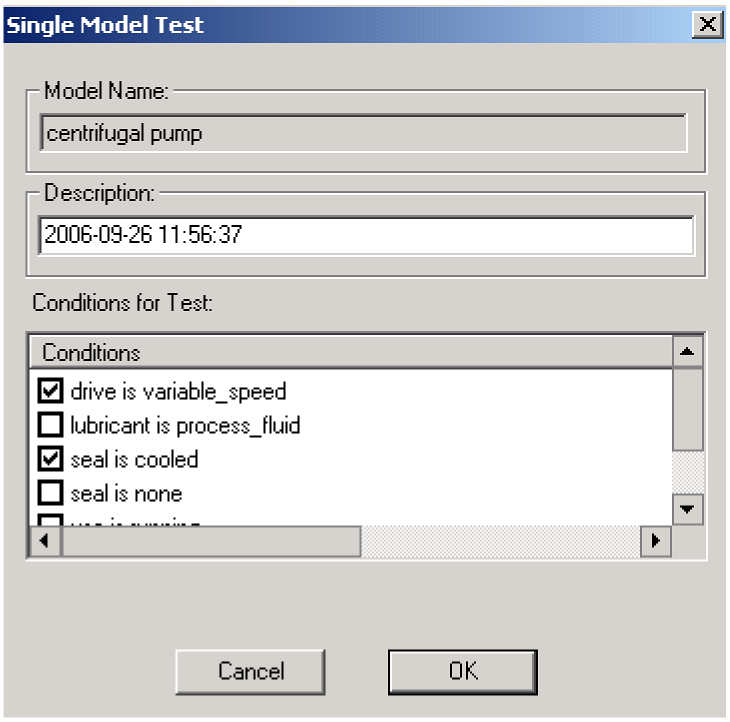


Figure 3.2 Interface to select a single model to be tested

For a single model to be tested, a dialog box is provided (as shown in figure 3.3) to allow the user to set up the test and give any relevant description with time and date provided as default information. The applicable conditions are drawn from those defined on the "Link Conditions" elsewhere in MBuilder. These conditions are operating conditions that define which options are available to customize the unit model by using attributes. For example, when testing the model "centrifugal pump", there are condition options (such as "drive is variable_speed", "lubricant is process_fluid", "seal is cooled", "seal is none" etc., as shown in figure 3.3) that could be selected by the user, The model test bed will combine these selected conditions and apply them to the HAZOP analysis.


Single Model Test

Model Name: centrifugal pump

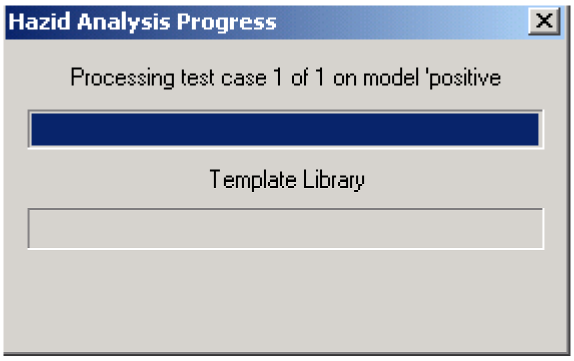
Description: 2006-09-26 11:56:37

Conditions for Test:

Conditions
☑ drive is variable_speed
☐ lubricant is process_fluid
☑ seal is cooled
☐ seal is none

Figure 3.3 Interface to apply link conditions to a model to be tested

To test multiple models, the user selects an option from the "Tools" menu in MBuilder. For both single and multiple model tests, a feedback is provided to show the progress of a single or a batched test as shown in figure 3.4.


Hazid Analysis Progress

Processing test case 1 of 1 on model 'positive

Template Library

Figure 3.4 Progress display during model testing

A message to inform the user that the results are available for viewing in HAZID's HMeeting tool will appear when the test is finished.

3.2.2.4 Function and Data

To start the new analysis on the model, the application will call HAZOP analysis automatically.

To test the model, the application will call HAZOP automatically to analyse all of the defined deviations for each model. For example, to test the model 'centrifugal pump', the application will examine all deviations automatically although only 7 deviations are relevant to this model (See appendix F for a full list of the deviations). All that means, it will test a fault caused within the model, which could result in deviations in any upstream or downstream port.

The model test bed follows the same way that other HAZID applications use to open/connect to a database. The results of the test are stored in an ORACLE database with a series of tables. These results will later be compared to the intended results for the given model and will be available for later review, audit or comparison.

Multiple tests on the same model are also recorded with the time and the results are generated so that it is possible to refer to earlier results as well as the most recently generated ones.

3.2.2.5 Connect to OPC (Off-Page-Connector)

An OPC is a small unit model that shows continuation of a process plant drawing onto another page. Sometimes it is called sheet connector. Each time when a model is tested, a test plant file is created automatically and stored as a temporary file in the user's temporary directory. The test plant consists of a single instance of the model to be tested, with each of its external ports connected to an OPC instance.

For each "inPort", the application will add an OPC, and connect to the port automatically. In the example of "centrifugal pump" in figure 3.1, each "inPort" will be connected to an OPC.

For each "outPort", the application will also add an OPC, and connect it to the port automatically, In the example of "centrifugal pump", each "outPort" will be connected to an OPC.

In this way, the model under test is connected to deviations caused by faults in the OPCs and link any deviations caused by faults in the test model to consequences in the OPCs.

3.2.2.6 Single Model Test and Multiple Model Test

As mentioned earlier, the user can test one single model or more than one model in a batch run. As demonstrated in figure 3.5, a single model test means to test one model in a test run using one test case. A multiple test means testing more than one model in a test run using one model test case, in other words, by using the test case to generate a plant file for each model. The test results are shown as "Fault Comparison", "Consequence Comparison", "Fault Path Propagation" and "Deviation Oriented" tables. A detailed discussion of these four types of comparisons is in section 3.2.3.

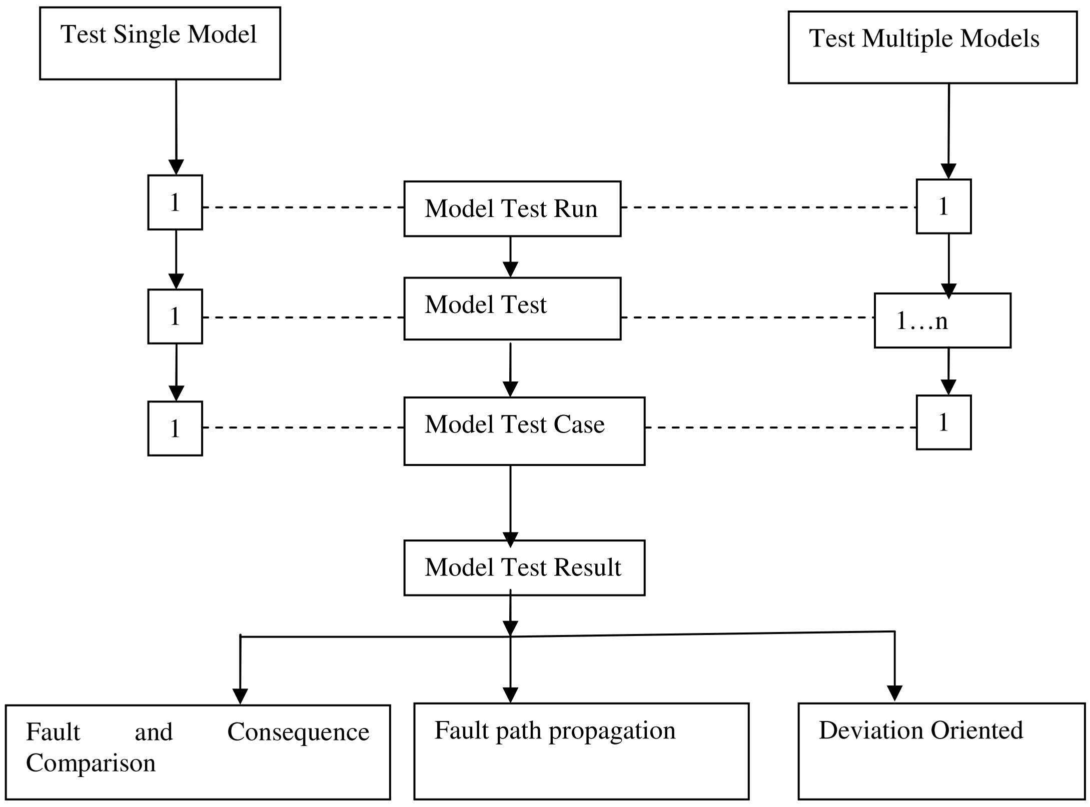


Figure 3.5 Architecture of the model test bed

3.2.2.7 The Fault Path and Deviation

A fault path is an acyclic path that links the fault and the variable under consideration within a given SDG. Deviations consist of keywords such as “More”, “Less”, “No” and “Reverse”, and parameters such as “Flow”, “Pressure”, “Temperature” and “Level”. A fault path that propagates through a unit model under test will fall into one of the four types shown in figure 3.6, where “F”, “D” and “C” stand for “Fault”, “Deviation” and “Consequence” respectively. In case A, the fault happens in an OPC and causes a series of deviations that lead to a consequence in the test unit. In case B, the fault happens within the test unit while the consequence happens in some OPC. In case C, both the fault and consequence happen within the test unit. In case D, both the fault and consequence happen outside of the test unit and only deviations propagate through the test unit. To complete the test, all these four types are considered in the model test bed and the appropriate results generated.

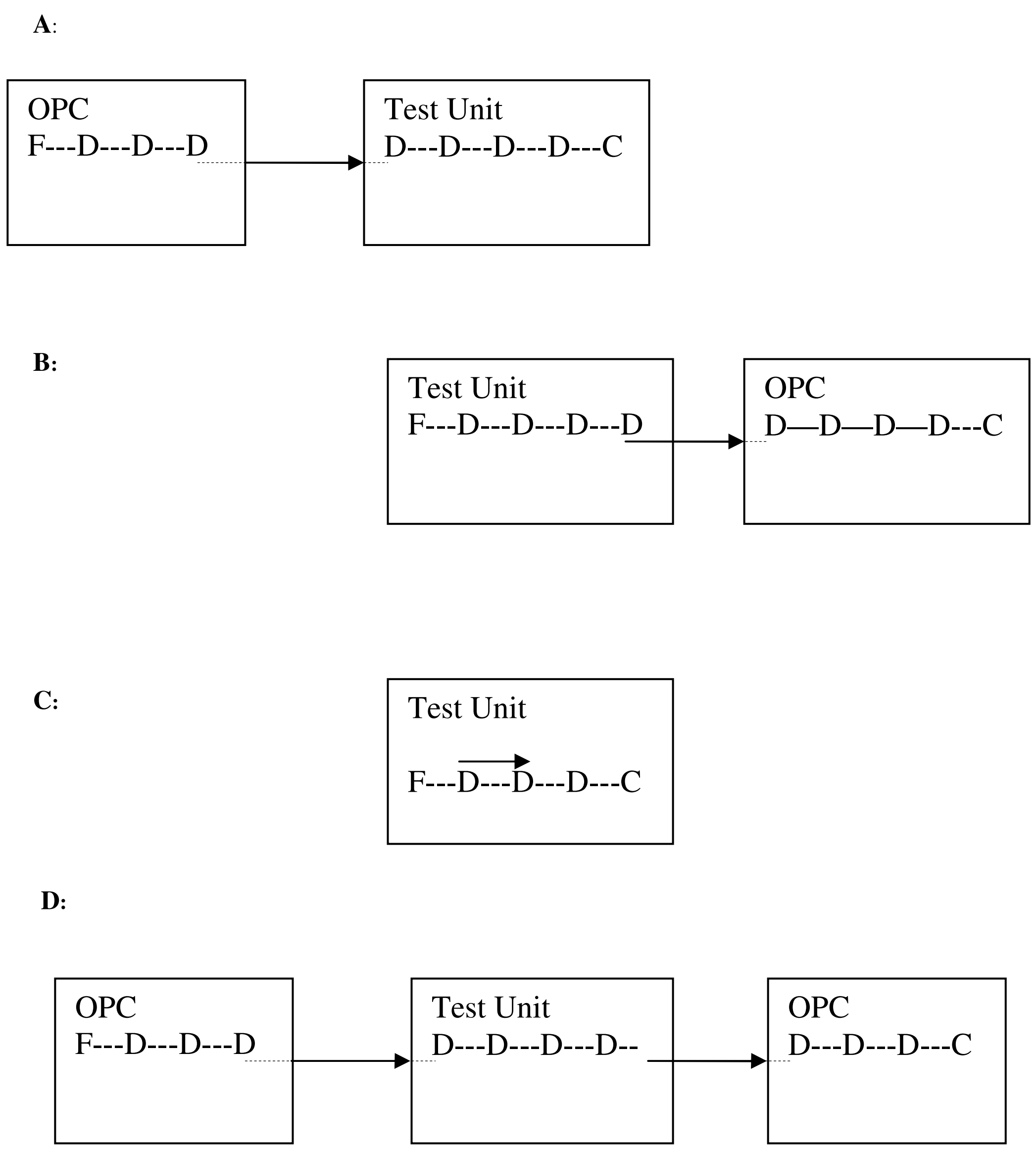


Figure 3.6 Four types of fault paths propagating through a test unit model

3.2.3 MODEL TEST RESULT

The results of model testing are presented in four different tables showing the different kinds of comparisons with expected results. Tables 3.2, 3.3, 3.4 and 3.5 are the test results generated for the "centrifugal pump" model.

3.2.3.1 Expected Results

The results show how the actual behaviour of the model differs from the expected behaviour, as specified by the engineer.

3.2.3.2 Comparing Deviations Caused by Fault(s)

Table 3.2, shows "Faults" that give rise to "Deviations" which propagate to the OPCs and therefore elsewhere in the plant. The expected results are compared with the test bed generated results. The outcome of each deviation and fault pair can be "MATCH", "MISSING", or "EXTRA". A "MATCH" means that the expected result is found in the test result. "MISSING" means that the expected result is not found in the test result. "EXTRA" means that a test result is not found in the expected result.

Expected Description	**Expected Port**	**Expected Keyword**	**Actual Description**	**Actual Port**	**Actual Keyword**	**Status**
loss of drive	out	revFlow	Loss of drive	Out	revflow	MATCH
loss of drive	in	morePressure				MISSING
loss of drive	out	lessFlow				MISSING
loss of drive	out	lessPressure				MISSING
switched on in error	in	lessPressure				MISSING
switched on in error	out	morePressure				MISSING
switched on in error	out	moreTemp				MISSING
			contamination from sheet connector	in	contamination	EXTRA
			lessComposition from sheet connector	in	lessComposition	EXTRA
			lessPressure from sheet connector and less flow	in	lessPressure	EXTRA

Table 3.2 Fault comparison table

3.2.3.3 Comparing Consequence(s) Caused by Deviations

Table 3.3 shows part of a table of consequences caused by deviations. It shows how "Deviations" in the OPCs can cause "Consequences" in the unit model under test. This is compared with the expected deviations and consequences, Again the outcome of a comparison can be either "MATCH", "MISSING" or "EXTRA".

Expected Description	**Expected Port**	**Expected Keyword**	**Actual Description**	**Actual Port**	**Actual Keyword**	**Status**
possible pump or seal damage	out	revFlow	Possible pump or seal damage	out	revFlow	MATCH
cavitation	in	moreGas				MISSING
cavitation	in	moreVapour				MISSING
churning	out	noFlow				MISSING
dry running	in	noFlow				MISSING
			cavitation	out	moreGas	EXTRA
			cavitation	out	moreTemp	EXTRA
			cavitation	out	moreVapour	EXTRA
			churning	in	noFlow	EXTRA

Table 3.3 Consequence comparison table

3.2.3.4 Fault Path Propagation

Table 3.4 demonstrates how a specific fault propagates through a test unit model and reaches its consequence. All deviation(s) along the path are also shown.

Fault					
Item	Descriptor	Probability	Condition		
test_unit	loss of drive	0	“use_is_not_standby”		
Deviations					
Item	Port	Keyword			
test_unit	out	revFlow			
test_unit	in	revFlow			
inopc_in	out	revFlow			
Conseque -nce					
Item	Descriptor	Hazard	Operability	Rank	Condition
inopc_in	Reverse flow and contamination to inlet sheet connector	FALSE	FALSE	5	not(fault_is_this(“reverseflow at inlet sheet connector”))||”fault_unit_is_ not_this_one”)

Table 3.4 Path propagation display

3.2.3.5 Deviation Propagation

The deviation propagation is demonstrated as a “Deviation Oriented” table. It checks for deviations that happen in each port and identifies their effects on all other ports within a test unit model and gives faults that cause that deviation on that port as well as consequence(s) that can be caused by it.

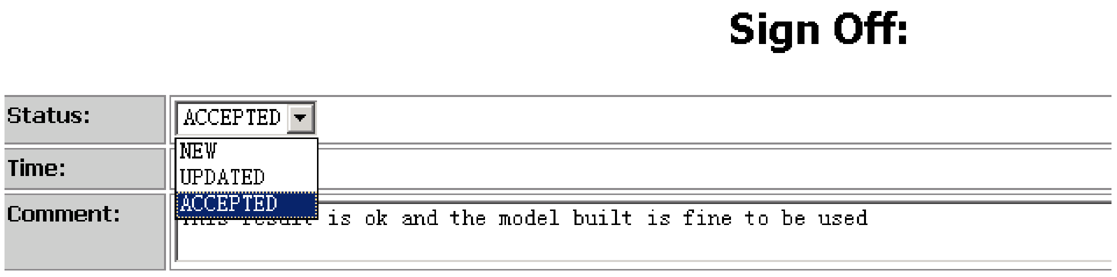
Table 3.5 shows part of the deviation propagation table of the “centrifugal pump” model. The whole table shows all the deviation effects propagating through each port of the test unit. For example, when fault “loss of drive” causes “reverse flow FR” at port “in”, it causes “reverse flow FR” at port “out”, and causes consequence “possible pump or seal damage”. “0” means there is no effect caused by the deviation “reverse flow FR” in port “drainOut” , “ventOut”, “IgnoreIn” and “IgnoreOut”. “F0” means “No Flow”; “C+” means “More Composition”; “Cont+” means “More Contamination”. Please see appendix F for a full list of deviations.

Path_id	**Deviation**	**Affected port**	**Fault**	**Effects on other ports**						**Consequence**
				drainOut	IgnoreIn	IgnoreOut	in	out	ventOut	
0000000000-0000006888	C+	in	propagated from connection	0	0	0	C+	C+	0	propagated from unit
0000000000-0000006890	C-	in	propagated from connection	0	0	0	C-	C-	0	propagated from unit
0000000000-0000006892	Cont+	in	propagated from connection	0	0	0	Cont+	Cont+	0	propagated from unit
0000000000-0000006792	F0	in	propagated from connection	0	0	0	F0	F0	0	dry running, propagated from unit
0000000000-0000006806	F0	in	propagated from connection	0	0	0	F0	F0	0	churning, propagated from unit
0000000000-0000006800	FR	out	propagated from connection	0	0	0	FR	FR	0	possible pump or seal damage, propagated from unit
0000000000-0000006804	FR	in	loss of drive	0	0	0	FR	FR	0	possible pump or seal damage, propagated from unit

Table 3.5 Deviation oriented table

3.2.3.6 Sign off Status

A facility is provided for recording the test status of each model in chronological order. The user can sign off a model as acceptable after reviewing the test results or record that the results are not yet acceptable. This is shown in figure 3.7.



Sign Off:

Status:	ACCEPTED
Time:	
Comment:	This result is ok and the model built is fine to be used

Figure 3.7 Sign off status

3.2.4 CONCLUSIONS

The correctness of a model built by the process engineer is critical to the result of the HAZOP study, because errors in a model may result in incorrect fault, consequence, deviations as well as propagations in the HAZOP result, therefore distracting HAZOP study members' attention from really critical hazards. A model test bed is developed to allow engineers to verify model correctness according to their expected behaviour.

Figure 3.8 gives the overall workflow of the model testing process. After a new model is built or an old model is modified, a model test file with all the necessary connections to OPCs is created. Then HAZID is called to do a HAZOP analysis automatically to achieve the test result. The test result is then automatically compared and displayed with the expected results using the HMeeting tool. If all the results are as expected by its builder (that means there is NO 'missing' or 'wrong' or 'extra and wrong' behaviours in a model under test), then the process ends. Otherwise, the engineer will go back to modify the model and test it again until the model under test would behave exactly as intended (that means there are only ‘matching’ or ‘extra but right’ behaviours in a model under test). In addition, the engineer can choose to test one model at a time or to test multiple models at one go. Different kinds of comparisons are done and shown in different table formats.

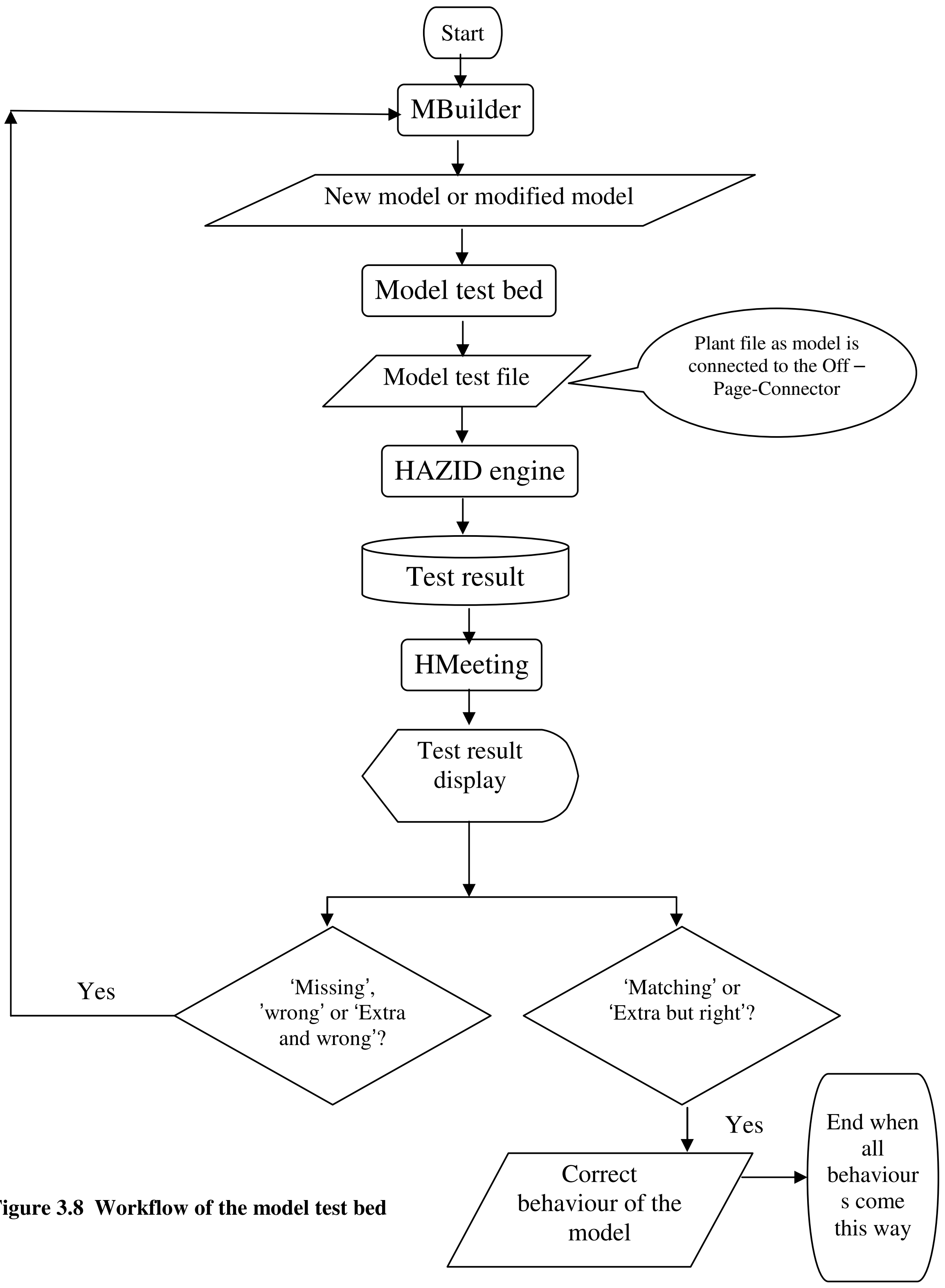

Start
MBuilder
New model or modified model
Model test bed
Model test file
Plant file as model is connected to the Off – Page-Connector
HAZID engine
Test result
HMeeting
Test result display
Yes
‘Missing’, ’wrong’ or ‘Extra and wrong’?
‘Matching’ or ‘Extra but right’?
Yes
Correct behaviour of the model
End when all behaviours come this way

Figure 3.8 Workflow of the model test bed

3.3 SUMMARY

This chapter discussed modelling in computer-aided HAZOP study. The qualities that make a good model are "accuracy", "completeness", "conciseness", "generality", "independence", "simplicity", "flexibility". Differences between qualitative modelling and quantitative modelling are also discussed and the advantage and disadvantage of qualitative modelling are highlighted, therefore, justifying why qualitative reasoning is widely used. This chapter also discussed model information representation and organization, particularly presenting some ways that researchers had used to solve ambiguities. A model test bed developed as part of this project is illustrated in detail.

4 SAFETY-RELATED APPLICATIONS

This chapter introduces and describes in full detail three safety-related computer-aided applications: “Safe Isolation Tool”, “Instrument Checker” and “Auto Cause-Effect System”. It details how the three safety-related computer-aided tools are developed and how they are related to one another. Section 4.1 illustrates the “Safe Isolation Tool”. Section 4.2 explains the “Instrument Checker” and “Auto Cause-Effect System”. Section 4.3 describes how cause-effect analysis can be utilised in hazard identification for safe isolation when combined with the safe isolation tool.

4.1 ISOLATION FOR SAFE MAINTENANCE

4.1.1 INTRODUCTION

The safety of hazardous processing plants is of paramount importance as an accident could cause major damage to property and/or injury to people. Well-maintained equipment in the process plant can give smooth running of the plant and increase the plant productivity and lifetime.

However, the maintenance work of process plant is often dangerous as it requires appropriate isolation of the equipment items being maintained because the maintenance task is a set of intrusive activities that could introduce the risk of releasing hazardous substances. Impropriate isolation for maintenance work could result in significant consequences. Any release of hazardous material can cause damage to the whole plant or even take human life. For example, a pool, jet or flash fire, or a vapour-cloud explosion could become the consequences should a flammable substance be released during the maintenance process. Furthermore, long range and greater hazardous threats to people and the environment could become the consequences should the released material be toxic (Health and Safety Executive, 2006).

Unsafe isolation in process plant could not only cause release of dangerous materials, but also lead to pipe-work failure and deviations from normal operations, etc (Hale, et al., 1998).

It is reported that 30% of accidents are maintenance-related and 50% of them release harmful substances (Wallace & Merritt, 2003). Therefore, a comprehensive identification of potential hazards caused by maintenance work is necessary before carrying out the actual maintenance work.

This section describes a computer-aided tool that considers the safety issues for isolation process of maintenance work in the process plant. It is integrated in the HAZID system. Currently, no other commercial tool available for this purpose in the market can be found. Section 4.1.1 gives the objectives of the tool and a brief analysis of safe isolation, then section 4.1.2 explains the methodology for identifying the isolation boundary and how the algorithm is tested. Section 4.1.3 illustrates how HAZOP analysis is applied to identify hazards after the boundary is identified and selected, then section 4.1.4 gives the overall workflow of the tool. The whole section ends with a summary of the overall methodology.

4.1.1.1 Objectives

Objectives of the tool are to define an isolation boundary and to identify potential hazards related to the isolation task. The isolation tool will help to mitigate the likelihood of harmful release by automatically identifying the isolation boundary and examining the hazards that might occur due to the isolation activity before the maintenance work commences. It can be used as a preventive and mitigative risk reduction tool. Furthermore, this tool has to ensure the isolation boundary is complete (e.g., no valve that must be closed is missing) and correct (eg. no valve that need not be closed is included).

4.1.1.2 Analysis of Safe Isolation

According to the Health and Safety Executive (2006), process isolation stages consist of "Hazard identification", "Risk assessment and selection of isolation scheme", "Planning and preparing of equipment", "installation of isolation", "Draining, venting, purging and flushing", "Testing and monitoring effectiveness of the isolation", "Carrying out the intrusive activity", "Reinstatement of plant". Also, final isolation methods include "Positive isolation", which is a complete separation of the to be isolated items from other parts of the plant; "Proved isolation", which is the "valved isolation" where the effectiveness of valve closures will be confirmed before the maintenance work; "Non-proved isolation", which is also the "valved isolation" where the effectiveness of valve closures will NOT be confirmed before the maintenance work.

Currently this isolation tool helps with the first stage which is the "Hazard identification" and helps with the "valved isolation" with either proved or non-proved isolation. The confirmation of valve closures is not within the scope of this tool. Also, it does not deal with isolation for maintaining instruments.

Results of the isolation tool can be used to support:

- Maintaining sufficient understanding of potential hazards associated with isolation by providing records on hazard analysis with different isolation procedures in which the closing order of the isolable valves is different.
- Establishing integrity of isolation arrangements of the whole process plant.
- Designing rigid isolation procedures.
- Controlling the isolation activities.
- Developing and maintaining a "library" of standard isolation schemes.
- Reinstatement of the plant by providing an isolation record.
- Linking to chemical industry maintenance database.

4.1.2 Identifying the Isolation Boundary

An isolation boundary is a collection of isolation points. When equipment items are to be maintained in a process plant, a process engineer will analyse the P&ID of that plant, and identify the valves that must be closed in order to isolate the equipment items so that they are safe to work on. For example, consider figure 4.1 in page 52, if the centrifugal pump "P-0101A" is to be isolated, then 5 valves, "V015", "V002", "V014", "V013" and "V001", will need to be closed to isolate the pump. This process of identification and the analysis of the potential hazards caused by closing these valves are automated by the tool and is the subject of the rest of section 4.1.

4.1.2.1 The Plant File

As mentioned in chapter 3, the plant file, which is a text file generated from the P&ID, describes the equipment items in the plant, which includes their connections, flow directions and other attributes. The plant file is used as input into the isolation tool.

All isolable valves are specified at the design stage and should be indicated and the position of manually operated valves should be effectively secured in the P&IDs. This is reflected in the attribute “canBeIsolationValve” of each valve where the value of it must be “yes” or “no” in the plant file.

4.1.2.2 The Tracing Procedure

The procedure for identifying the isolation boundary is to trace upstream and downstream from the equipment items to be isolated to find the valves which must be closed. During tracing, the tool will look only for the first valve that is isolable in each line branching out from the equipment items.

When a Pressure Relief Valve (PRV) is met, the algorithm will compare the direction of the tracing and the opening direction of the PRV, if they are in the same direction, then the tracing procedure will carry on and the PRV will be identified as within the boundary but must be isolated, otherwise, the PRV will be identified as on the boundary and the tracing procedure at this branching line ends at this point.

When searching from the equipment items to be isolated, if the propagation passes through an OPC onto another P&ID then it will continue on the next P&ID until an isolable valve is found. If there is no continuation from that OPC, the OPC will be treated as being on the boundary and the user will be warned that an isolation point has not been found for that particular branch. The OPC ID and the pipe connected to it will be identified in the warning.

When multiple items are to be isolated, the same procedure will be applied to all the items. The valves that need to be closed for all the items will be combined together and any duplicates will be removed.

4.1.2.3 Testing

Two plants, a hydrocarbon separation unit (Lawley, 1974) and Benzene plant (Wells & Seagrave, 1976), are used to test the algorithm. The algorithm correctly identifies the isolation boundaries when given different equipment items as input for the two plants and this is verified by the process engineers in the sponsor company. In order to help the user visualise a boundary, the isolation tool automatically highlights the valves to be closed on the Smart Plant P&ID system. Figure 4.1 shows the output for isolating pump "P-0101A" for the hydrocarbon separation unit. All the isolable valves are highlighted in red. The item to be isolated, "P-0101A", is highlighted in bright blue (the valves filled with black colour are valves that have already been closed). Please refer to appendix G for notation explanation.

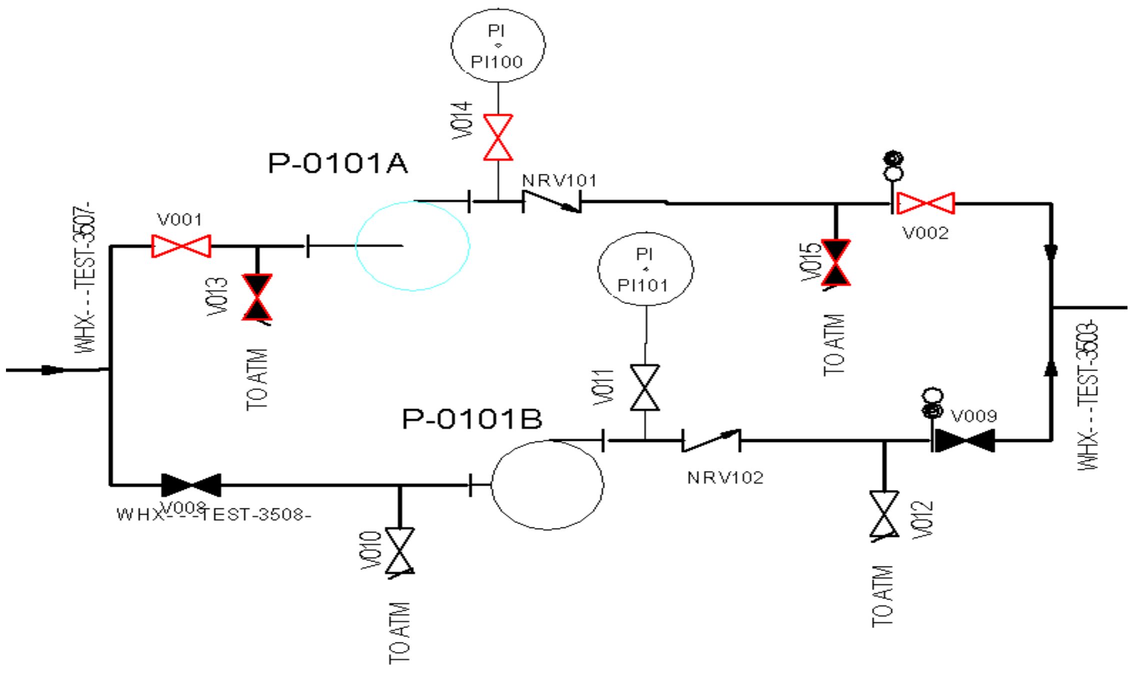
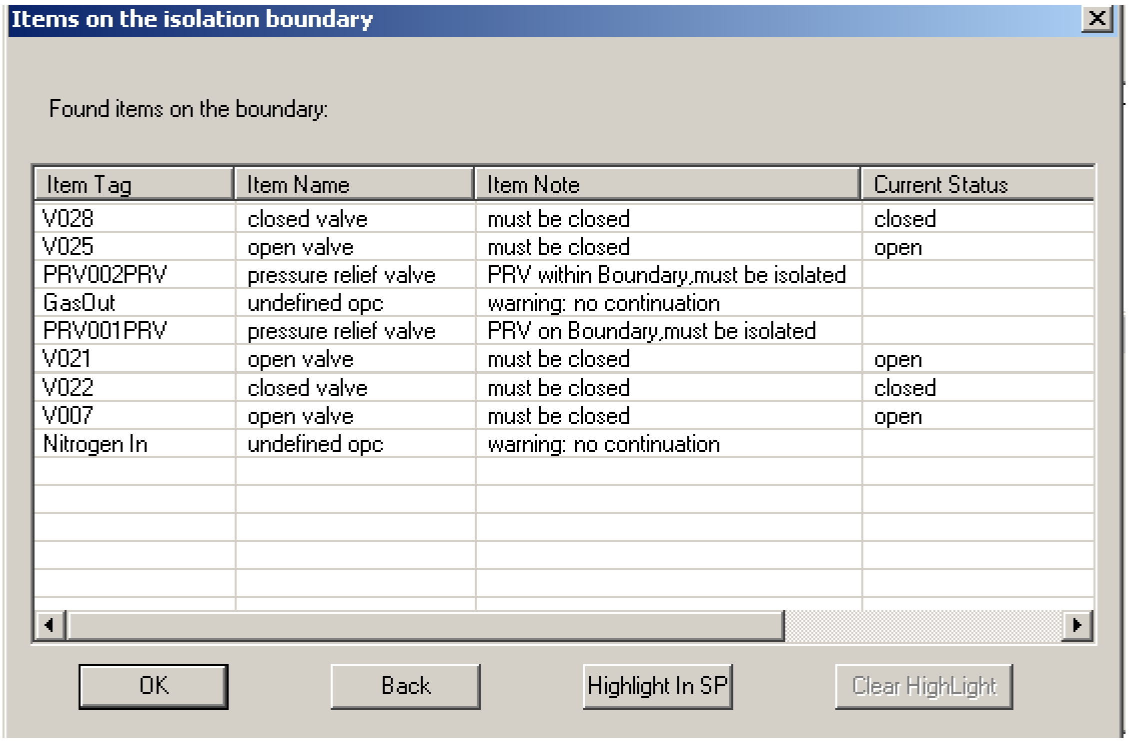


Figure 4.1 The highlighted isolation boundary for isolating centrifugal pump "P-0101A"

A more sophisticated example is to isolate the liquid-liquid-gas separator in the hydrocarbon separation unit plant. Figure 4.2 shows a table of equipment items on the boundary with their current status and related notes. The tool also displays a table with all the items within the boundary. Figure 4.3 shows these items highlighted in Smart Plant P&ID. This example shows that several different types of equipment items are on the boundary. The first type is the “must be closed” valve, such as “V028”, “V025”,”V021”, “V022” and “V007”, which are highlighted in Smart Plant P&ID drawing in red. The second type is “Pressure Relief Valve (PRV)”, such as “PRV002PRV”, “PRV001PRV”. The third type is the “Off-Page-Connector (OPC)”. The second and third types are highlighted in Smart Plant P&ID drawing in purple. The items to be isolated are highlighted in bright blue.


Items on the isolation boundary

Found items on the boundary:

Item Tag	Item Name	Item Note	Current Status
V028	closed valve	must be closed	closed
V025	open valve	must be closed	open
PRV002PRV	pressure relief valve	PRV within Boundary,must be isolated	
GasOut	undefined opc	warning: no continuation	
PRV001PRV	pressure relief valve	PRV on Boundary,must be isolated	
V021	open valve	must be closed	open
V022	closed valve	must be closed	closed
V007	open valve	must be closed	open
Nitrogen In	undefined opc	warning: no continuation	

Figure 4.2 Items on the isolation boundary for isolating separator “T-0100”

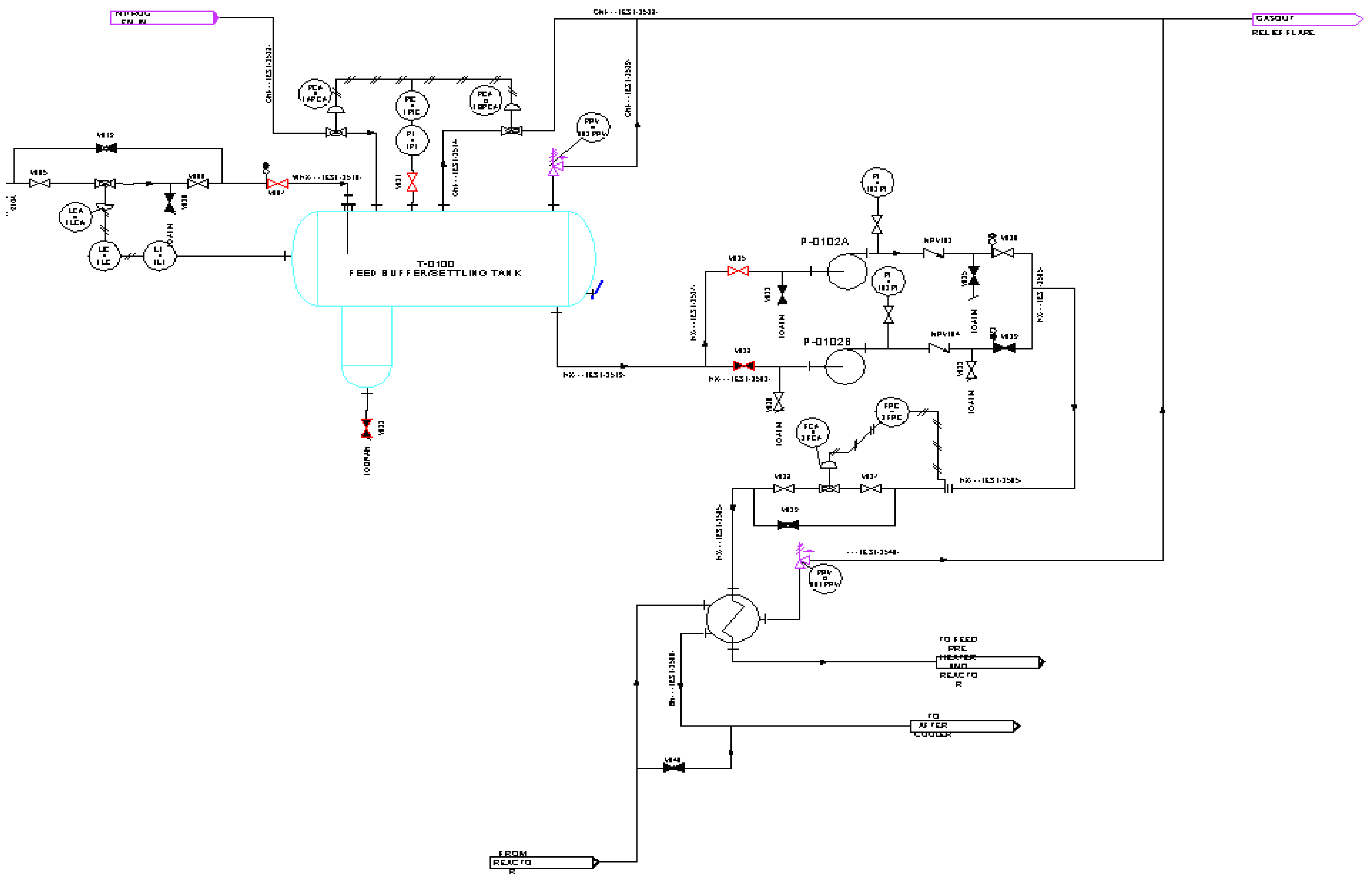


Figure 4.3 The highlighted isolation boundary for isolating separator "T-0100"

4.1.3 IDENTIFYING HAZARDS AFTER THE ISOLATION BOUNDARY IS DEFINED

Once the boundary is identified a HAZOP analysis will be carried out to identify the hazards related to the isolation procedure. The HAZID engine is called automatically for this purpose.

After the items on the boundary are identified, the user is asked to review the items and specify the order in which they should be closed. Consider the example shown in figure 4.1. If the first item selected to be closed is “V001” then a HAZOP analysis is carried out by applying the deviation 'no flow' to “V001”. The isolation tool does this by invoking the HAZID engine to carry out an analysis to identify the hazards that might be introduced by this deviation. If hazards are identified then they are reported. If no hazard is identified then the tool will go on to consider the deviation 'no flow' for the second boundary item to be closed (e.g. “V002”) and with “V001” being closed. This process goes on until the closing of all the valves on the boundary has been considered. In this way, there will be many HAZOP results as the valves are being closed in sequence. The HAZOP analysis carried out in HAZID is produced quickly and the successive analysis occupy little user time. Table 4.1 illustrates this hazard identification process. Presentation of the results is shown in HMeeting, as mentioned, a tool to view HAZOP analysis results in HAZID. The presentation of the results highlights the differences of the results, so that the engineer can focus on the new hazards that might occur by closing another valve. Please note, after the valves being closed and before the actual maintenance work (or works for any other purposes) commences, there is draining work needed to be carried out in order to complete the isolation process. For example, in figure 4.1, after valve ‘V001’, ‘V014’ and ‘V012’ being closed, valve ‘V013’ and ‘V015’ must be open to drain the flow that already present in the pipe and then close them to complete the isolation.

Conditions	Actions	Result
"no flow" passing through V_{001} (this should be the original HAZOP result with only "no flow" being considered)	Doing HAZOP analysis	HAZOP result_1
"no flow" passing through V_{002} + V_{001} closed	Doing HAZOP analysis	HAZOP result_2
....	…………………..	………………….
"no flow" passing through V_N + V_{N-1} closed + V_{N-2} closed …+V_{001} closed	Doing HAZOP analysis	HAZOP result $_n$

Table 4.1 The hazard identification process

Analysis of HAZOP results when scenarios of wrong order are analysed

Take the example of figure 4.1, after the isolation boundary is defined, five valves are identified that must be closed to isolate the pump "P-0101A". If the user chooses to close "V001" first, the HAZOP result will show no hazard occurred; but if the user chooses to close "V002" first, there will be "reverse flow" from down stream to upstream, the HAZOP result will show this deviation and its consequences. So the ideal closing sequence would be "V001", "V014", "V002", and then open "V013" and "V015" to drain the pipe, then close both "V013" and "V015" in either sequence.

4.1.4 THE OVERALL WORKFLOW

Figure 4.4 shows the overall workflow of the Isolation tool. It starts with loading the original plant file as input, and then a list of frames is generated in which each frame describes the information of an equipment item. After allowing the user to specify the item to be isolated, the tool will run its algorithm to look for the items including valves, OPCs without continuations, and PRVs in the isolation boundary and then display them. Then it will move onto the second stage which is to identify the hazards after the boundary is defined. It starts by allowing the user to specify the sequence for closing the valves on the boundary. Then it generates a series of input files for the HAZID engine with each valve closed in each stage of the sequence as described in Table 4.1. The HAZID engine is then called to do a HAZOP analysis with only “no flow” being considered for each input file. The results are then automatically compared and any differences are highlighted to show any hazards related to closing any of the valves.

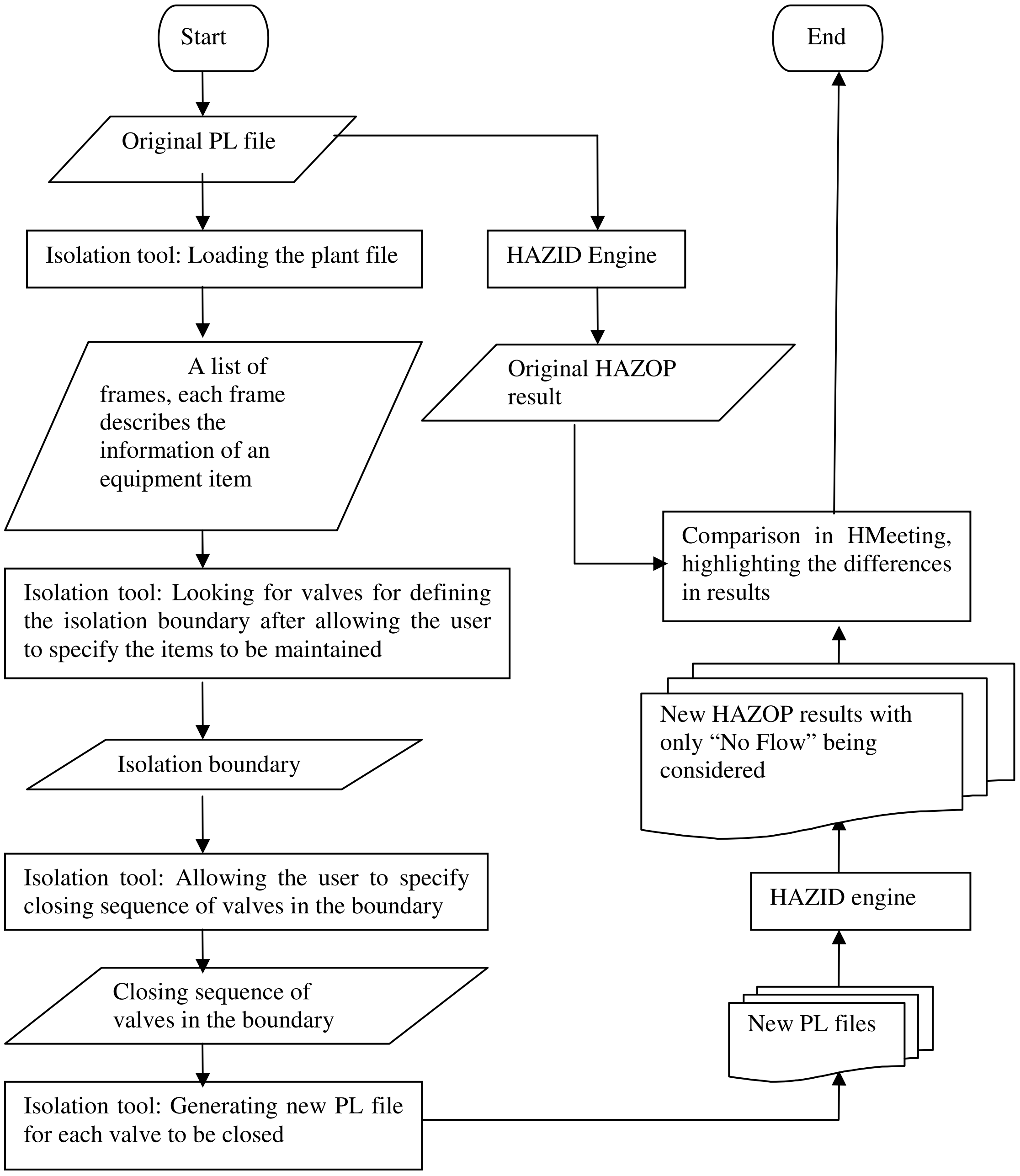

Start
End
Original PL file
Isolation tool: Loading the plant file
HAZID Engine
A list of frames, each frame describes the information of an equipment item
Original HAZOP result
Comparison in HMeeting, highlighting the differences in results
Isolation tool: Looking for valves for defining the isolation boundary after allowing the user to specify the items to be maintained
New HAZOP results with only "No Flow" being considered
Isolation boundary
HAZID engine
Isolation tool: Allowing the user to specify closing sequence of valves in the boundary
Closing sequence of valves in the boundary
New PL files
Isolation tool: Generating new PL file for each valve to be closed

Figure 4.4 Workflow of the isolation tool

4.1.5 Further work

The isolation tool can be further developed to identify Double Block and Bleed (DBB) isolation or Double seals in a single valve body with a bleed in between or Single Block and Bleed (SBB) isolation. It should be able to highlight and be able to differentiate them in the P&ID drawing. For example, using SBB as a final isolation for maintaining living plant that contains hazardous substance is considered to be highly risking. If in such situation a SBB is identified and highlighted, it would draw the engineer's attention and might result in the modification of isolation design to achieve higher safety level.

4.1.6 Conclusions

Carrying out isolation safely is of paramount importance to avoid releasing hazardous material into the atmosphere and to prevent isolation related accidents. A thorough hazard analysis ahead of the actual work is more likely to give a successful and safe isolation implementation.

Although process plant P&IDs are already available electronically for process engineers to identify potential hazards, no computer-aided tool was used to help with defining an isolation boundary for the isolation process and calling the hazard identification analysis automatically from the isolation safety point of view.

This section describes a novel computer-aided tool to help with safe isolation for maintenance work or works for other purposes. In order to define the isolation boundary, the tool searches upstream and downstream of the item to be isolated to find the isolable valves which must be closed. After the boundary is defined, HAZOP analysis with deviation "no flow" is applied to identify hazards that may occur when the valves are closed in a specified order. The results are displayed with differences of each HAZOP analysis being highlighted.

4.2 AUTOMATED CAUSE-EFFECT ANALYSIS

The design of the control of a process plant is important to ensure the safe running of the plant. Although P&IDs are available electronically to control engineers, many tasks associated with Process Control Engineering (PCE) are still carried out manually. One of these tasks is the cause-effect analysis of the control system. The result of the analysis is the cause-effect diagrams.

As part of this research project, a computer-aided system has been developed to generate cause-effect diagrams automatically given a set of P&IDs. This helps to reduce the effort required with the labour intensive analysis of PCE information by the control engineers. It is achieved by encoding knowledge related to PCE in rules so that they can be applied automatically to a given set of P&IDs to produce the corresponding cause-effect diagrams.

This section describes in detail the components of the system, the rules and the reasoning process. Distinctiveness of the cause-effect system is that it applies a general purpose rule engine and it is integrated with HAZID and its results cover all the instrumentation and their control of the entire plant. The cause-effect system takes the P&ID input from HAZID and displays the results in a format that comply with the ISO standard 10418 (International Organization for Standardization (ISO), 2003). ISO 10418: 2003 is Petroleum and Natural Gas International standardization that describes the basic concepts related to the analysis and design of a process safety system. It provides objectives, requirements and guidelines for technique in analysis, design and testing of a process safety system.

The rest of this section is organized as follows. Section 4.2.1 gives the overall introduction, then section 4.2.2 introduces the Instrument Checker. Section 4.2.3 illustrates how the output from the Instrument Checker is transformed into the format required by the inference engine and section 4.2.4 demonstrates the reasoning rules and reasoning process. Following that, section 4.2.5 explains how the result table is generated in required format. Section 4.2.6 gives a comparison with a closely similar system reported in Drath et al. (2006), then section 4.3 further evaluates the system by applying it to safety analysis of the isolation work. By the way of a summary, section 4.4 takes into account some possible future enhancements and section 4.5 draws the conclusion.

4.2.1 INTRODUCTION

Safety analysis in control design of a process plant is required to help with the identification of unfavourable outcomes that may present a safety risk and to help with the design of protective measures to avoid or to mitigate against such unfavourable events (ISO, 2003).

Safety Analysis Function Evaluation chart (SAFE) is one of the established cause-effect analysis techniques stated in the ISO standard 10418 (ISO, 2003). It can be applied to achieve the above objectives and ensure that the procedures and devices provided for safeguarding the process components form an integrated system covering the entire process plant. The SAFE chart is referred to as cause-effect table as it provides information about process events and process responses.

Manual generation of a cause-effect table given a P&ID of a process plant is labour intensive, time consuming, repetitive and error-prone. With P&IDs available in electronic format there is the potential of developing a computer-aided tool that can take the P&ID information as input and produce the cause-effect analysis result automatically.

This section introduces a computer-aided cause-effect system that links the information of the process control loops with the cause-effect reasoning rules to produce process event and response results. It highlights the novelty of the cause-effect system and describes its components and the reasoning process.

Features of the cause-effect system include:

- Capture of process control information using an Instrument Checker.
- Converting and manipulating of the description of the process control information.
- Linkage of the description and cause-effect reasoning rules.
- Generating of neutral XML inference results.
- Parsing XML results and displaying it in required format.

Two case studies are used to illustrate the working of the system. The first is a very small part of a P&ID of a much larger plant just to illustrate the working of the system. The second is the interlock system described in Drath et al. (2006). The system and results described by Drath et al. (2006) are compared with the current system. This section ends with a summary of the overall methodology.

4.2.1.1 Objective

The objective of the cause-effect system is to automate the generation of cause-effect tables from the P&ID input information hence reduce the labour intensive analysis effort required of the control engineers.

4.2.1.2 Related Work Review

Several computer-aided tools of this kind have been developed, such as a knowledge-based system described in Drath et al. (2006) that illustrates the auto-generation of cause-effect table through a standardized plant description model called **C**omputer-**A**ided **E**ngineering e**X**change (CAEX) and on rule-based algorithms. It uses the specification and implementation of interlocks as an example. "Interlocks are pieces of control code that ensure the safety of a plant" (Drath et al., 2006). A detailed comparison of that system with this one reported here is given in section 4.2.6.

A prototype software described in Yim et al. (2006) provides a way to identify control loops and indicators and connectivity between instruments and process items by making queries to its Prolog inference engine. The plant representation in Yim et al. (2006) is achieved by writing the plant topology information in XML according to the CAEX schema. Thambirajah et al. (2009) further advances this representation by generating a process connectivity matrix from the CAEX&XML description that lends itself to automated, exhaustive searching for paths and links.

The cause-effect system reported in this section uses an Instrument Checker to identify all the instruments and their links with related process items. Details of the Instrument Checker can be found in section 4.2.2. Differences with that reported in Yim et al. (2006) are also given in there.

There is other commercial tool that generates cause-effect diagram manually as a cause-effect configuration tool that provides an easy way of filling in the cause-effect table although it is not knowledge-based tool.

4.2.1.3 Components of Cause-Effect System

The cause-effect system consists of an Instrument Checker, a general purpose knowledge-based rule engine and an output tool that generates the cause-effect table that can be easily displayed in Microsoft Excel. Output from Instrument Checker is converted into input of the rule engine. Output from the rule engine is converted into a format that automatically generates the Excel cause-effect table in a format compliant with ISO10418 (ISO, 2003). Each component is an object-oriented design with a primary function.

Two programming technologies were used to implement the cause-effect system: C Language Integrated Production System (CLIPS) is used to execute the functions of the reasoning engine and C++ is used for all the remaining parts of the system.

Designing the system in this way brings together the advantages of the two different types of programming language. The CLIPS exploits the rule-based nature of the linking information in the control loop and the C++ allows an efficient parser and graphical application and enables the neutral XML results to be displayed in the required format. The workflow of the overall cause-effect system is shown in figure 4.5.

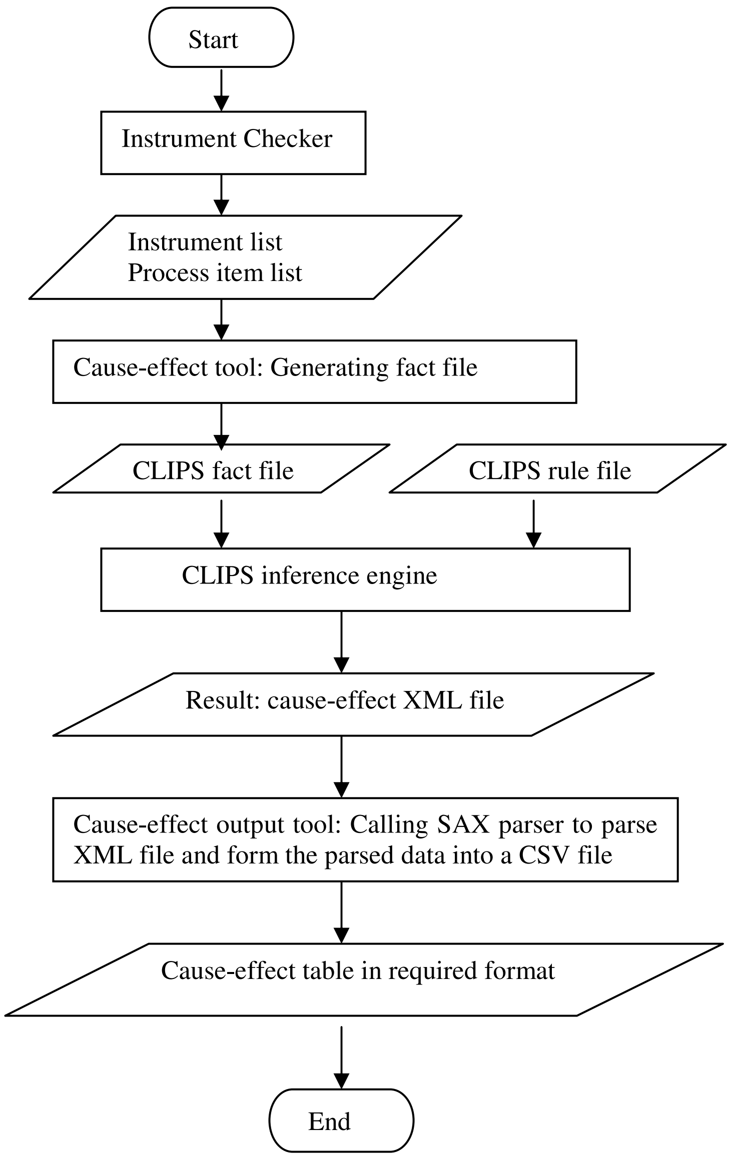

Start
Instrument Checker
Instrument list
Process item list
Cause-effect tool: Generating fact file
CLIPS fact file
CLIPS rule file
CLIPS inference engine
Result: cause-effect XML file
Cause-effect output tool: Calling SAX parser to parse XML file and form the parsed data into a CSV file
Cause-effect table in required format
End

Figure 4.5 Workflow for cause-effect analysis

4.2.2 INSTRUMENT CHECKER

Given a P&ID, the Instrument Checker identifies the instrument loops and their connections with the process items. The purpose is for the engineers to consider whether the instrument and its loops provide safeguards for deviations in the HAZOP procedure and also to prepare the data to be analysed by the inference engine in the cause-effect system. Therefore the output of this tool is used as input by both HAZID and the cause-effect system.

The checker first identifies all the instruments in the process plant. For each instrument, it traces the upstream and downstream connections of each branch line until a process item is found. Therefore, information about which process items are connected to which instruments is collected. Given a process item the Instrument Checker can list which instruments are connected to it, and given an instrument, it can also list which process items are connected to it.

Consider the P&ID shown in figure 4.6, which is a very small part taken from a much larger plant. The tool identifies the following instruments:

- Two high level alarms – “ZEH-59010” and “ZLH-59010”.
- Two low level alarms – “ZEL-59010” and “ZLL-59010”.
- One control valve – “FCV-59010”.

The tool then traces upstream and downstream to find the process item(s) attached to these instruments. In this case they are all connected to the same pipe with tag “test1001PU34-PU”.

Figure 4.7 shows the results. The loop number “59010” indicates that they are all in the same instrument loop. Figure 4.7 also shows the related deviations of each instruments (“L+”

means “high level”; “L-“ means “low level”; “L0” means “no level”) and the response, which can be either an indicator, an alarm or a control.

Figure 4.8 shows all the instruments (“ZEH-59010”, “ZLH-59010”, “ZEL-59010”, “ZLL-59010” and “FCV-59010”) are attached to the same process item (“test1001PU34-PU”).

Compare to the prototype software reported in Yim et al. (2006), while this cause-effect system uses plant file to represent the plant, their system uses a CAEX XML file to describe it and get the result only when the user asks. The Instrument Checker displays all the connections and items in one go without being asked, users have an option to choose to show it or not. An additional button can be added, for instance, at the “process item” list to allow the user to check the connectivity between two specified elements as well.

In summary, the instrument checker reports the connectivity between instrumentation and process items as well as the control loops, i.e. it identifies how equipment items in a process plant are linked together physically and/or through electronic control signals. This information is used as input by the cause-effect analysis inference engine to draw the conclusions.

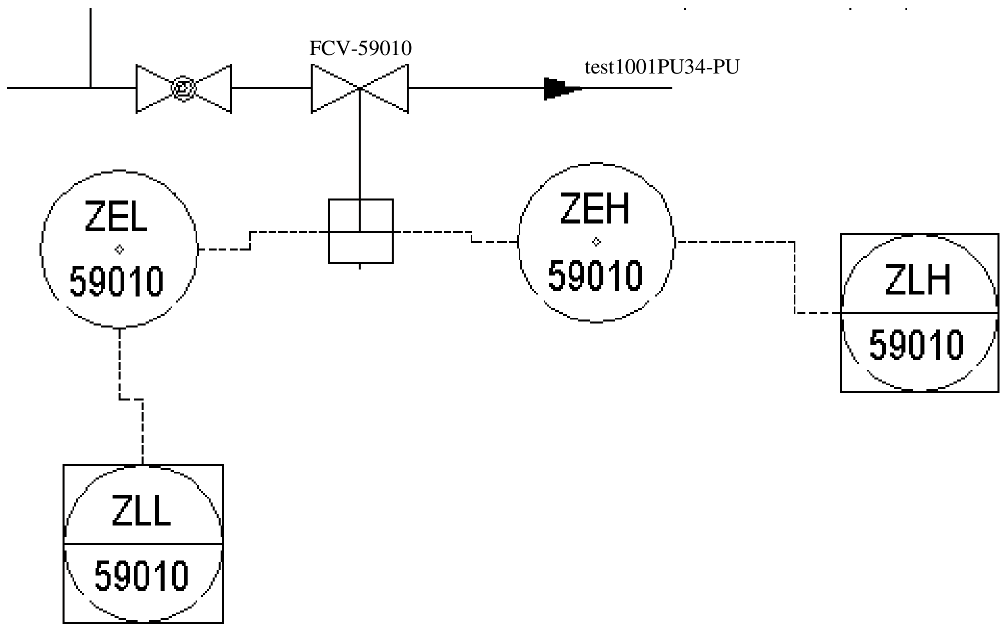
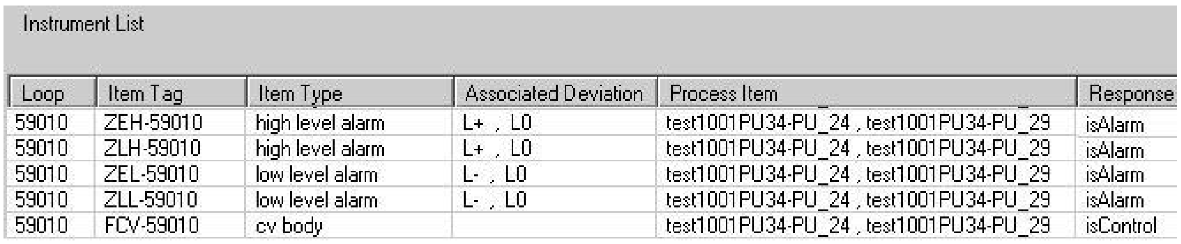


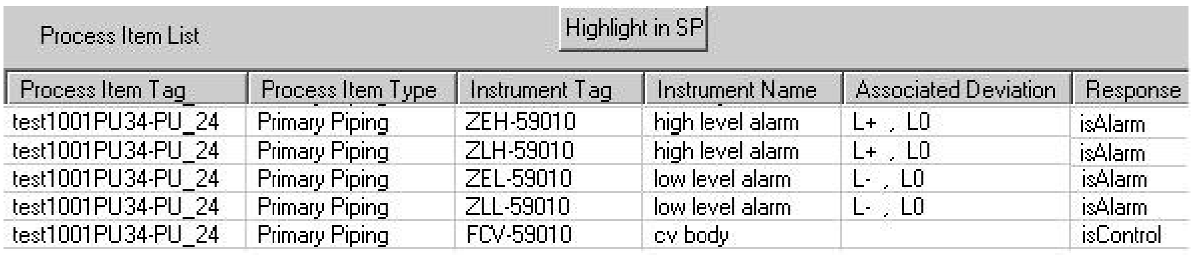
Figure 4.6 A simple instrument loop



Instrument List

Loop	Item Tag	Item Type	Associated Deviation	Process Item	Response
59010	ZEH-59010	high level alarm	L+ , L0	test1001PU34-PU_24 , test1001PU34-PU_29	isAlarm
59010	ZLH-59010	high level alarm	L+ , L0	test1001PU34-PU_24 , test1001PU34-PU_29	isAlarm
59010	ZEL-59010	low level alarm	L- , L0	test1001PU34-PU_24 , test1001PU34-PU_29	isAlarm
59010	ZLL-59010	low level alarm	L- , L0	test1001PU34-PU_24 , test1001PU34-PU_29	isAlarm
59010	FCV-59010	cv body		test1001PU34-PU_24 , test1001PU34-PU_29	isControl

Figure 4.7 Instrument list



Process Item List

Highlight in SP

Process Item Tag	Process Item Type	Instrument Tag	Instrument Name	Associated Deviation	Response
test1001PU34-PU_24	Primary Piping	ZEH-59010	high level alarm	L+ , L0	isAlarm
test1001PU34-PU_24	Primary Piping	ZLH-59010	high level alarm	L+ , L0	isAlarm
test1001PU34-PU_24	Primary Piping	ZEL-59010	low level alarm	L- , L0	isAlarm
test1001PU34-PU_24	Primary Piping	ZLL-59010	low level alarm	L- , L0	isAlarm
test1001PU34-PU_24	Primary Piping	FCV-59010	cv body		isControl

Figure 4.8 Process item list

4.2.3 GENERATING INPUT FOR THE REASONING ENGINE

In order to analyse the process events and the corresponding process responses, a rule-based system is built for this purpose. The rule-base captures the expert's knowledge and carries out the inference to produce the result. CLIPS is chosen as the development tool as it supports rule-based, object-oriented and procedure programming methods (Giarratano & Riley 1994; Riley, 2008).

A rule-based system in CLIPS consists of three basic components: a set of facts, a set of rules and the inference engine that controls the overall execution by matching the rules against the facts to infer new information (Giarratano & Riley 1994). Therefore, the first step for building our expert system is to generate the CLIPS fact file.

Bearing in mind that the output of Instrument Checker has already prepared all the necessary data of process items and their connections with the instruments, therefore it is easy to use that data in the reasoning process. However, information is still represented in different ways due to differing data representations used in the two languages. A C++ function is developed to convert data that represents the equipment items and their connectivity into a format that can be read by the CLIPS reasoning engine.

According to Yim et al. (2006), a physical link (or path) is an equipment item like a pipe carrying a flow of mass or energy while a control link (or path) is a cable connecting an equipment item like a valve to a controller carrying an electronic signal. They are defined as "flow-connection" and "signal-connection" respectively in the system reported in this section. Among them, "flow-connection" is directional while "signal-connection" is bi-directional.

From the output of the Instrument Checker, the tool defines the CLIPS fact file according to these rules:

The structure of each fact is:

```
([type],[tag],[item name]);[comment]
```

All the process items are classified as “equipment”, e.g.

```
(equipment pipe test1001PU34-PU 1-in-2-out); 1 in 2 out
```

This means equipment “test1001PU34-PU” is a “1-in-2-out” pipe.

All the instruments are classified according to its item name, e.g.

```
(device high-level-alarm ZEH-59010);high level alarm
(device high-level-alarm ZLH-59010);high level alarm
(device low-level-alarm ZEL-59010);low level alarm
(device low-level-alarm ZLL-59010);low level alarm
(device cv-actuator NOTAG_Instrument_0_0.106_0.505);cv actuator
```

Each control instrument is classified as “equipment controlDevice”, e.g.

```
(equipment controlDevice FCV-59010 cv-body);cv body
```

For all the pipes, the CLIPS fact file defines the “flow-connection”, e.g.

```
(flow-connection test1001PU34-PU out test1001PU34-P_4)
(flow-connection test1001PU34-P_4 out NOTAG_PipingComp_0_0.222_0.469)
(flow-connection NOTAG_PipingComp_0_0.222_0.469 out test1001PU34-P_2)
(flow-connection test1001PU34-P_2 out NOTAG_PipingComp_0_0.214_0.469)
```

The above flow-connection definitions indicate how pipe “test1001PU34-PU” is connected to other pipes and equipment items to deliver the flows.

If there is flow connection from A to B, and there is flow connection from B to C, then the CLIPS asserts an “in-line” fact as follow:

“(in-line A to C)” indicates that there is flow connection from A to C.

For all the signal lines, the CLIPS fact file defines the “signal-connection”, e.g.

```
(signal-connection ZLH-59010   NOTAG_SignalRun_33)
(signal-connection NOTAG_SignalRun_33   ZEH-59010)
(signal-connection ZEH-59010   NOTAG_SignalRun_39)
(signal-connection NOTAG_SignalRun_39   NOTAG_Instrument_0_0.106_0.505)

(signal-connection ZLL-59010   NOTAG_SignalRun_24)
(signal-connection NOTAG_SignalRun_24   ZEL-59010)
(signal-connection ZEL-59010   NOTAG_SignalRun_43)
(signal-connection NOTAG_SignalRun_43   NOTAG_Instrument_0_0.106_0.505)
(signal-connection NOTAG_Instrument_0_0.106_0.505   FCV-59010)
```

From the above signal-connection definition, it is shown that “ZEH-59010”, “ZLH-59010”, “ZEL-59010”, “ZLL-59010” and “FCV-59010” are connected by signal through a cv actuator “NOTAG_Instrument_0_0.106_0.505”.

The template of each processItem-instruments-connection is:

```
(deftemplate processItem-instruments-connection
          (slot processComponent)
            (multislot deviceIdent))
e.g.

(processItem-instruments-connection
  (processComponent test1001PU34-PU)
  (deviceIdent ZEH-59010 ZLH-59010 ZEL-59010 ZLL-59010 FCV-59010))
```

The above example means instruments with the tags as “ZEH-59010”, “ZLH-59010”, “ZEL-59010”, “ZLL-59010” and “FCV-59010” are connected to one process item with the tag as “test1001PU34-PU”.

In a nutshell, all the equipment items and their connections are explicitly represented in the fact file and all the physical paths such as flow-connections between start and end point and control paths such as signal-connections are clearly identified.

4.2.4 Reasoning Rules and Reasoning Process

4.2.4.1 Developing the rules

The reasoning rules are defined according to the ISO10418 (ISO, 2003). The principle of defining a rule is to make sure that it is as generic and as reusable as possible.

Here is an example of a descriptive rule:

IF there is a level sensor;
AND there is a level vessel and it has (at least) one input;
AND there is (at least) a control valve that is able to close the input of the level vessel;
AND the level sensor can detect the level in the level vessel and raise an alarm;
THEN close the control valve(s) if the level sensor raises a maximum alarm.

The equivalent rule in CLIPS format is:

```
(defrule levelVessel-highLevel-close-inputValve
  (device level-indicator | high-level-alarm ?HLA)
  (equipment pipe | majorProcessItem  ?VESSEL-TAG ?VESSEL-NAME)
  (equipment  controlDevice  |  controlDevicePump  ?INPUT-CONTROL-DEVICE-
TAG ?INPUT-CONTROL-DEVICE-NAME)
  (or(signal-connection ?HLA ?VESSEL-TAG)
    (signal-connection ?HLA ?INPUT-CONTROL-DEVICE-TAG))
  (in-line ?INPUT-CONTROL-DEVICE-TAG to ?VESSEL-TAG)
=>
(print-result-levelAlarmHigh  ?VESSEL-TAG  ?VESSEL-NAME  ?HLA  ?INPUT-
CONTROL-DEVICE-TAG ?INPUT-CONTROL-DEVICE-NAME)
)
```

The above rule in CLIPS means:

The name of the rule is “levelVessel-highLevel-close-inputValve”;

If there is a level sensor device which is either a level indicator or a high level alarm that is connected to a level vessel, which can be a “liquid-liquid-gas-separator” as a major equipment item, or a primary pipe, or any other container in other forms;

AND

If there is an input control device such as a control valve or a control pump;

AND

If the input control device is connected to the level vessel;

AND

(If there is signal connection between the level sensor and the input control device

Or

If there is signal connection between the level sensor and the level vessel);

AND

If the level sensor senses an overflow in the level vessel, then it will close the input control valve;

then (shown as "=>")

call the function "print-result-levelAlarmHigh" to print the result.

4.2.4.2 The Reasoning Process

Rules become activated whenever all the patterns at the left hand of the rule are matched by the facts. When a rule is fired, the action(s) specified at its right hand will be taken.

Taking the example of the above rule, apply it to the simple instrument loop at figure 4.6. Here we have a primary pipe "test1001PU34-PU", a high level alarm "ZEH-59010", and a control valve "FCV-59010".

There is signal connection between the high level alarm and the control valve. We have connection rules to assert "(signal-connection ZEH-59010 FCV-59010)" since there are signal connection facts below.

```
(signal-connection ZEH-59010   NOTAG_SignalRun_39)
(signal-connection NOTAG_SignalRun_39   NOTAG_Instrument_0_0.106_0.505)
(signal-connection NOTAG_Instrument_0_0.106_0.505   FCV-59010)
```

There is also flow connection from the control valve to the pipe "(flow-connection FCV-59010 out test1001PU34-PU)" to indicate that this is an input control device,

Remember that if all the patterns at the left hand side of a rule are satisfied by the facts provided in the CLIPS fact file, the rule will be triggered to perform the action(s) at its right hand side.

Therefore, rule "levelVessel-highLevel-close-inputValve" becomes activated as all the patterns at the left hand side of the rule are matched by the facts. The results are written into the result file in the XML format as follows:

```
<cause_effect>
<cause_comment processItemTag='test1001PU34-PU'>Primary-Piping high level
</cause_comment>
<cause instrumentTag='ZEH-59010'>level alarm high</cause>
<effect controlInstrumentTag='FCV-59010'>close input control device: cv-body
</effect>
</cause_effect>
```

The CLIPS output can be in any format. XML as a neutral scripting language is chosen because

- it enables the easy sharing of data across different systems.
- it can be easily understood by both humans and computers.
- it can be readily generated from the CLIPS inference engine.

4.2.5 GENERATING THE EXCEL CAUSE-EFFECT DIAGRAMS

After calling the CLIPS engine, the tool generates the results in the XML format as described above. Once the XML results are gained, the Simple API for XML (SAX) parser in C++ function is called to parse the XML output and format the parsed data into a Comma-Separated Values (CSV) text file, which is an implementation of delimited text file. Once the CSV text file is generated, the engineer can open it with Microsoft Excel, and the cause-effect result will be presented in the format specified in ISO10418 (ISO, 2003).

4.2.5.1 Parsing the XML results

The purpose of the parser is to read and deconstruct the XML output file containing the cause-effect analysis results.

Parsing the XML output means to read the XML document, to identify key words and symbols and to transform the information into a data structure which can be processed and manipulated to display. The SAX parser is applied and integrated into the cause-effect system for the above objectives.

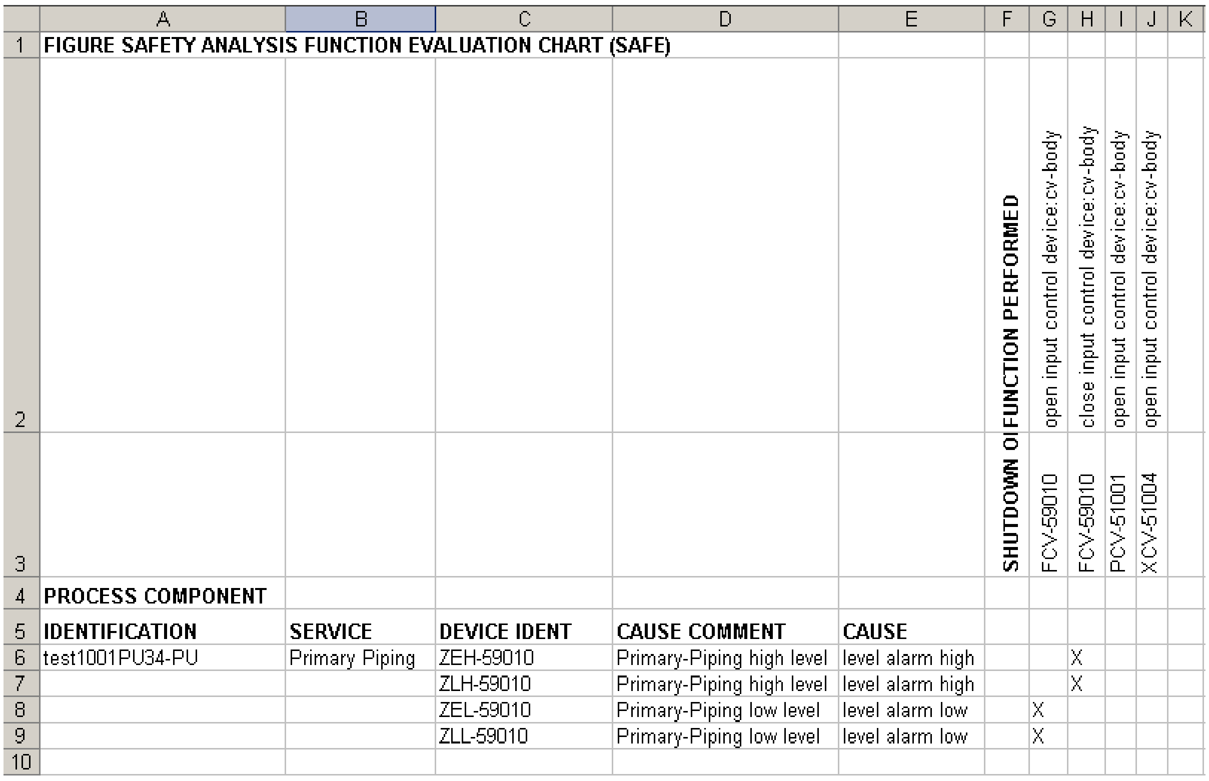
Functions provided by the SAX parser are listed as follows:

- Return the list of cause-effects in the plant.
- Return the list of cause-comment with the associated process item ID/Tag.
- Return the list of causes with the associated instrument/device ID/Tag.
- Return the list of corresponding effects with the associated control device ID/Tag.

4.2.5.2 Displaying Result Table

Part of the result is shown in figure 4.9. The cause-effect table in Excel shows that process component “test1001PU34-PU” has four instrument devices (“ZEH-59010”, “ZLH-59010”, “ZEL-59010”, “ZLL-59010”) attached to it. If the high level alarm “ZEH-59010” or “ZLH-59010”goes off, then the input control valve “FCV-59010” must be shut down. If the low level alarm “ZEL-59010” or “ZLL-59010” goes off, then the input control valve “FCV-59010” must be open. The interconnecting cross marks are placed in the cells indicating the cause-effect link between the process component, sensor instrument and control device. The “Function Performed” column describes the action that must be taken on the control device.

The automated cause-effect analysis significantly reduces the effort required from an engineer. The main task for them is to check the results and confirm whether the suggested actions are correct.



	A	B	C	D	E	F	G	H	I	J	K
1	FIGURE SAFETY ANALYSIS FUNCTION EVALUATION CHART (SAFE)										
2						SHUTDOWN OI FUNCTION PERFORMED	open input control device:cv-body	close input control device:cv-body	open input control device:cv-body	open input control device:cv-body	
3							FCV-59010	FCV-59010	PCV-51001	XCV-51004	
4	PROCESS COMPONENT										
5	IDENTIFICATION	SERVICE	DEVICE IDENT	CAUSE COMMENT	CAUSE						
6	test1001PU34-PU	Primary Piping	ZEH-59010	Primary-Piping high level	level alarm high			X			
7			ZLH-59010	Primary-Piping high level	level alarm high			X			
8			ZEL-59010	Primary-Piping low level	level alarm low		X				
9			ZLL-59010	Primary-Piping low level	level alarm low		X				
10											

Figure 4.9 Part of the cause-effect table in Excel

4.2.6 Comparison with Related Work

The knowledge-based system reported in Drath et al. (2006) is designed to implement the automatic generation of the cause-effect table using the specification and implementation of interlocks as an example. In order to make a comparison, we have produced and extended a P&ID from that example and applied our cause-effect analysis tool on it as shown in figure 4.10.

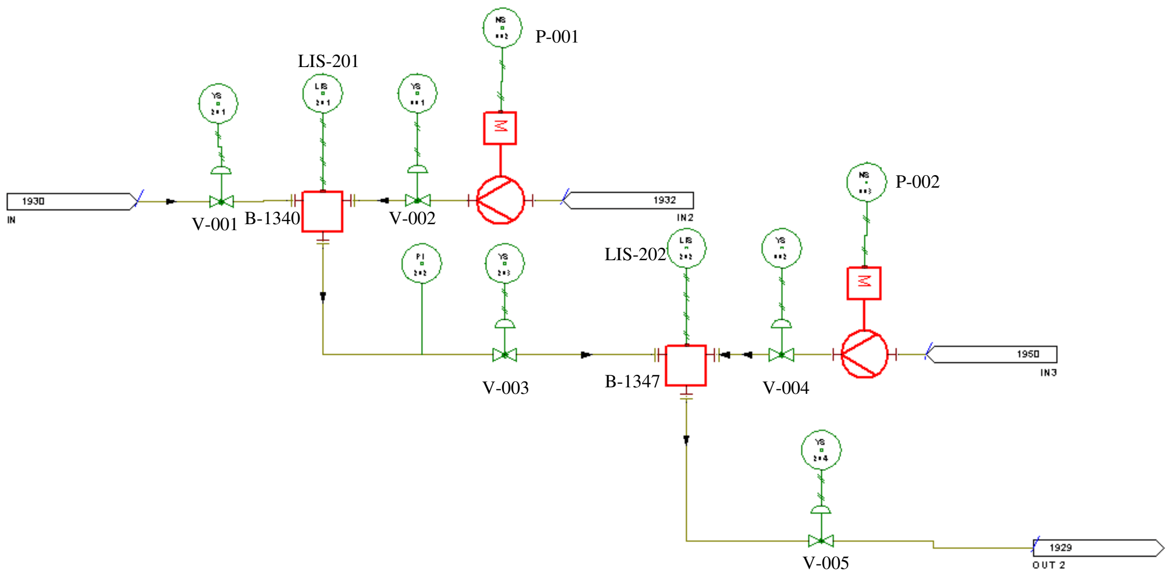


Figure 4.10 P&ID example extended from Drath et al. (2006)

In this P&ID, “B-1340” and “B-1347” are two level vessels called “Blanketed Fixed Roof”; “V-001” and “V-002” are two input control valves of “B-1340”; “V-003” is an output control valve of “B-1340” and an input control valve of “B-1347”; “V-004” is an input control valve of “B-1347”; “V-005” is an output control valve of “B-1347”. “LIS-201” and “LIS-202” are the level sensors of “B-1340” and “B-1347” respectively. “P-001” and “P-002” are two generic centrifugal pumps.

Cause and effect analysis rules that are applicable to this plant can be simply stated:

IF

the level in a vessel has reached its maximum

THEN

close its input device.

IF

the level in a vessel has reached its minimum

THEN

close its output device.

IF

a valve is closed

THEN

stop the in-line control pump to protect the pump.

IF

a pump is started

THEN

ensure the in-line valve is opened to protect the pump.

IF

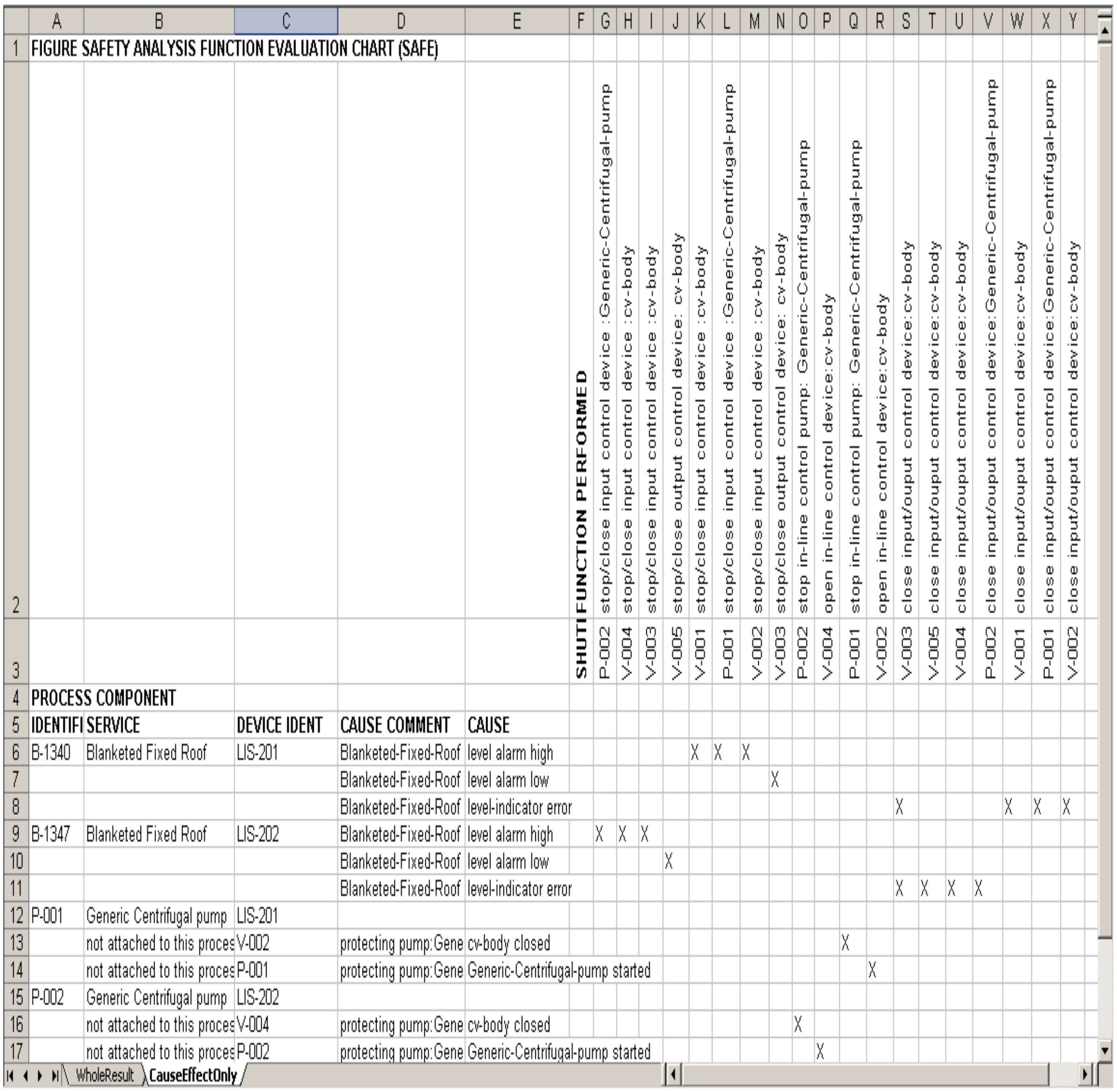
an error occurred with a level sensor for a vessel

THEN

close the input and output control valves for that vessel.

The result of the analysis is shown in figure 4.11. Comparing with the result reported in Drath et al. (2006), the result reported here has two additional features:

- The result table provides a comprehensive list of process components and their attached devices. The user has the option of viewing only components that have cause-effect links that apply to them.
- The table provides more detailed classification of function performed. For example, “V-003” is an output control valve for “B-1340” and an input control valve for “B-1347”. Therefore, any cause-effect control that applies to “V-003” is appropriately linked to the correct vessels.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	FIGURE SAFETY ANALYSIS FUNCTION EVALUATION CHART (SAFE)																								
2						FUNCTION PERFORMED	stop/close input control device :Generic-Centrifugal-pump	stop/close input control device :cv-body	stop/close input control device :cv-body	stop/close output control device: cv-body	stop/close input control device :cv-body	stop/close input control device :Generic-Centrifugal-pump	stop/close input control device :cv-body	stop/close output control device: cv-body	stop in-line control pump: Generic-Centrifugal-pump	open in-line control device:cv-body	stop in-line control pump: Generic-Centrifugal-pump	open in-line control device:cv-body	close input/ouput control device:cv-body	close input/ouput control device:cv-body	close input/ouput control device:cv-body	close input/ouput control device:Generic-Centrifugal-pump	close input/ouput control device:cv-body	close input/ouput control device:Generic-Centrifugal-pump	close input/ouput control device:cv-body
3						SHUTI	P-002	V-004	V-003	V-005	V-001	P-001	V-002	V-003	P-002	V-004	P-001	V-002	V-003	V-005	V-004	P-002	V-001	P-001	V-002
4	PROCESS COMPONENT																								
5	IDENTIFI	SERVICE	DEVICE IDENT	CAUSE COMMENT	CAUSE																				
6	B-1340	Blanketed Fixed Roof	LIS-201	Blanketed-Fixed-Roof	level alarm high						X	X	X												
7				Blanketed-Fixed-Roof	level alarm low									X											
8				Blanketed-Fixed-Roof	level-indicator error														X				X	X	X
9	B-1347	Blanketed Fixed Roof	LIS-202	Blanketed-Fixed-Roof	level alarm high		X	X	X																
10				Blanketed-Fixed-Roof	level alarm low					X															
11				Blanketed-Fixed-Roof	level-indicator error														X	X	X	X			
12	P-001	Generic Centrifugal pump	LIS-201																						
13		not attached to this proces	V-002	protecting pump:Gene	cv-body closed												X								
14		not attached to this proces	P-001	protecting pump:Gene	Generic-Centrifugal-pump started													X							
15	P-002	Generic Centrifugal pump	LIS-202																						
16		not attached to this proces	V-004	protecting pump:Gene	cv-body closed										X										
17		not attached to this proces	P-002	protecting pump:Gene	Generic-Centrifugal-pump started											X									

Figure 4.11 Cause-effect table in Excel for example P&ID

4.3 APPLYING CAUSE-EFFECT ANALYSIS TO THE ITEM TO BE ISOLATED

Section 4.1 and 4.2 have introduced the “Isolation Tool” and the cause-effect analysis system in detail. While the isolation boundary tool derives part of its information from the P&ID and is valuable for planning maintenance, the automatic generation of the cause-effect tables provides a valuable HAZOP aid to help assess safeguards when considering consequences and risks.

The cause-effect table has a role to highlight sensors inside the isolated plant and to highlight controls outside the boundary and vice versa. HAZID can then detect cause and consequence scenarios around the boundary. This section illustrates how the cause-effect analysis tool can be used to support the hazard identification before carrying out the isolation work.

Take the sample plant shown in figure 4.6 in page 67, the aim is to isolate the pipe “test1001PU34-PU”. The isolation tool is used to identify the isolation boundary. Figure 4.12 and figure 4.13 show the result. Figure 4.12 highlights 9 valves in red colour that must be closed in order to isolate this pipe. Figure 4.13 shows the items on the boundary including the OPCs without continuation and items within the isolation boundary.

Figure 4.14 shows the cause-effect analysis for isolating pipe “test1001PU34-PU”. In this analysis table, “test1001PU34-PU_24”, “test1001PU34-PU_31”, “test1001PU34-PU_32”, “test1001PU34-PU_5” and “test1001PU34-PU_7” are different parts of pipe “test1001PU34-PU”. If isolating the pipe causes any of the following alarms to go off then the control valve “FCV-59010” must be open: “ZEL-59010”, “ZLL-59010”, “FQ-59010”, “FAL-59010”, “FI-

59010”, “PI-59010” or “PAL-59010”. On the other hand if the high level alarm “ZEH-59010” or “ZLH-59010” goes off then the input control valve FCV-59010 must be closed.

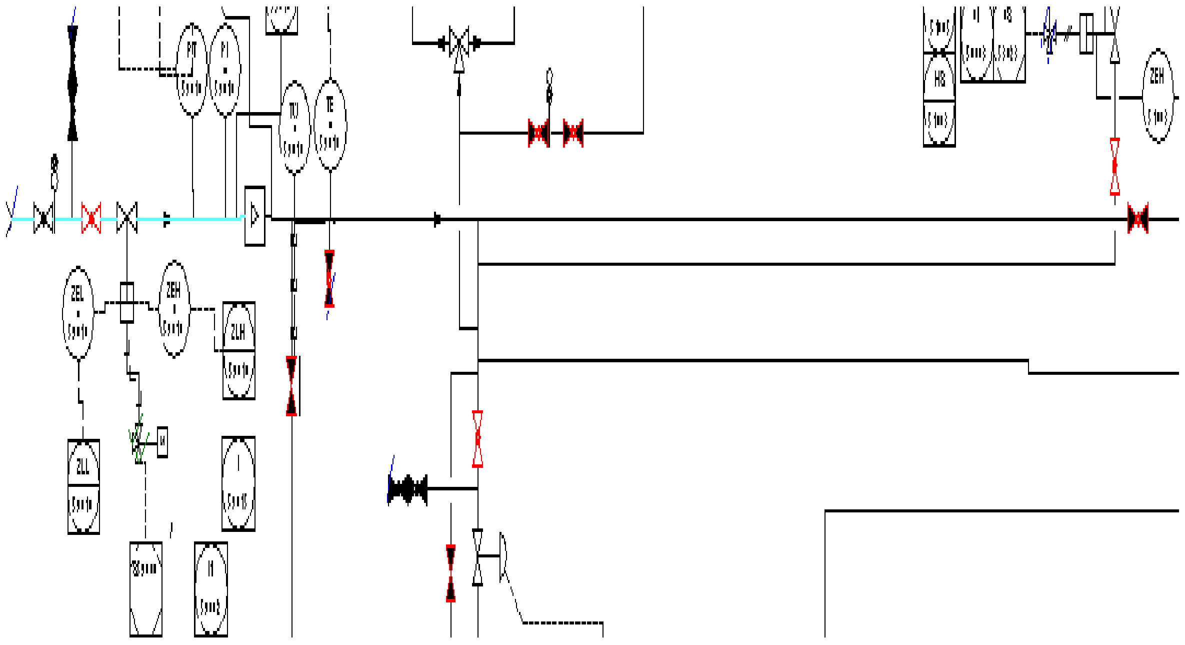
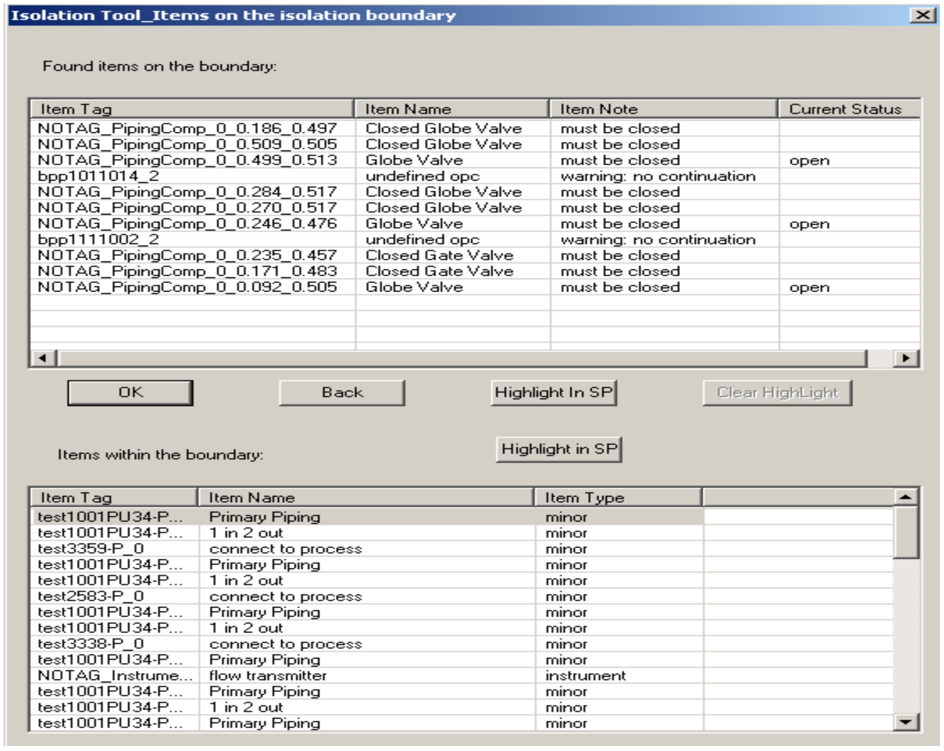


Figure 4.12 The highlighted isolation boundary for isolating a primary pipe



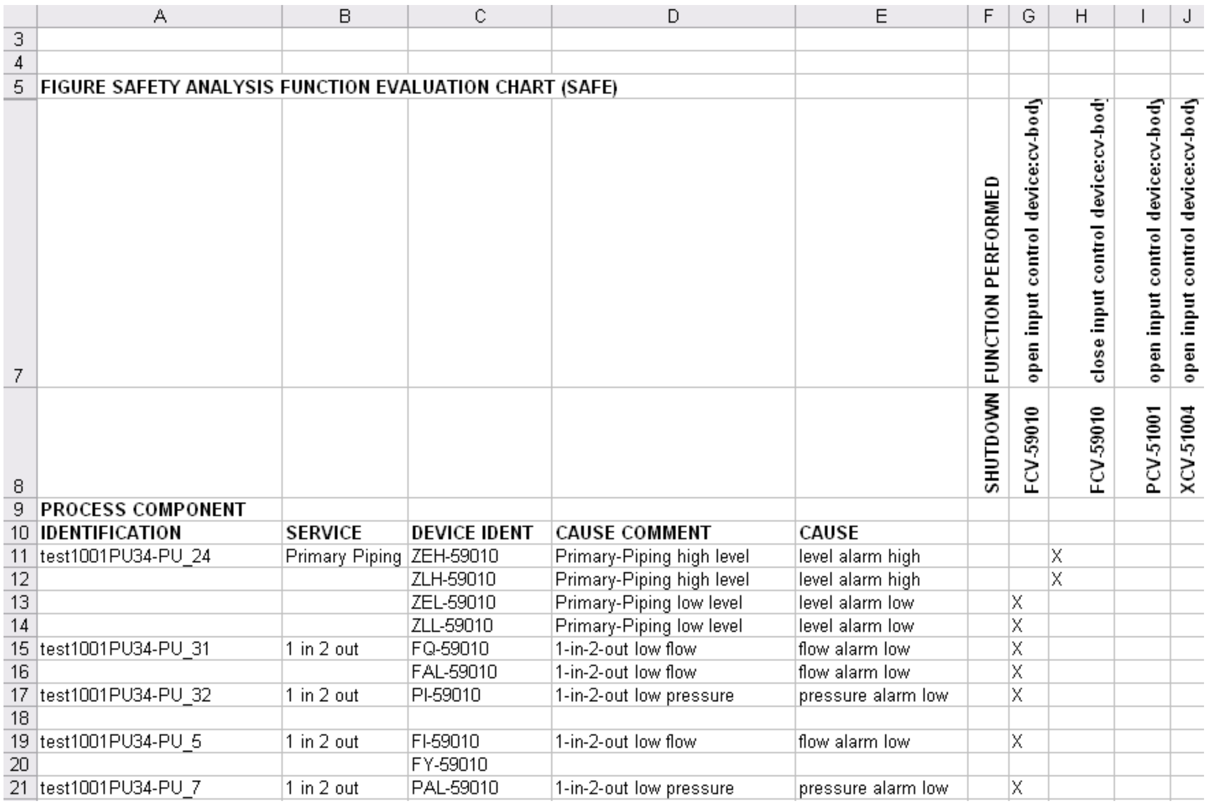
Found items on the boundary:

Item Tag	Item Name	Item Note	Current Status
NOTAG_PipingComp_0_0.186_0.497	Closed Globe Valve	must be closed	
NOTAG_PipingComp_0_0.509_0.505	Closed Globe Valve	must be closed	
NOTAG_PipingComp_0_0.499_0.513	Globe Valve	must be closed	open
bpp1011014_2	undefined opc	warning: no continuation	
NOTAG_PipingComp_0_0.284_0.517	Closed Globe Valve	must be closed	
NOTAG_PipingComp_0_0.270_0.517	Closed Globe Valve	must be closed	
NOTAG_PipingComp_0_0.246_0.476	Globe Valve	must be closed	open
bpp1111002_2	undefined opc	warning: no continuation	
NOTAG_PipingComp_0_0.235_0.457	Closed Gate Valve	must be closed	
NOTAG_PipingComp_0_0.171_0.483	Closed Gate Valve	must be closed	
NOTAG_PipingComp_0_0.092_0.505	Globe Valve	must be closed	open

Items within the boundary:

Item Tag	Item Name	Item Type	
test1001PU34-P...	Primary Piping	minor	
test1001PU34-P...	1 in 2 out	minor	
test3359-P_0	connect to process	minor	
test1001PU34-P...	Primary Piping	minor	
test1001PU34-P...	1 in 2 out	minor	
test2583-P_0	connect to process	minor	
test1001PU34-P...	Primary Piping	minor	
test1001PU34-P...	1 in 2 out	minor	
test3338-P_0	connect to process	minor	
test1001PU34-P...	Primary Piping	minor	
NOTAG_Instrume...	flow transmitter	instrument	
test1001PU34-P...	Primary Piping	minor	
test1001PU34-P...	1 in 2 out	minor	
test1001PU34-P...	Primary Piping	minor	

Figure 4.13 Items on and within the isolation boundary for isolating primary pipe "test1001PU34-PU"



	A	B	C	D	E	F	G	H	I	J
3										
4										
5	FIGURE SAFETY ANALYSIS FUNCTION EVALUATION CHART (SAFE)									
7						SHUTDOWN FUNCTION PERFORMED	open input control device:cv-body	close input control device:cv-bod	open input control device:cv-body	open input control device:cv-body
8							FCV-59010	FCV-59010	PCV-51001	XCV-51004
9	PROCESS COMPONENT									
10	IDENTIFICATION	SERVICE	DEVICE IDENT	CAUSE COMMENT	CAUSE					
11	test1001PU34-PU_24	Primary Piping	ZEH-59010	Primary-Piping high level	level alarm high			X		
12			ZLH-59010	Primary-Piping high level	level alarm high			X		
13			ZEL-59010	Primary-Piping low level	level alarm low		X			
14			ZLL-59010	Primary-Piping low level	level alarm low		X			
15	test1001PU34-PU_31	1 in 2 out	FQ-59010	1-in-2-out low flow	flow alarm low		X			
16			FAL-59010	1-in-2-out low flow	flow alarm low		X			
17	test1001PU34-PU_32	1 in 2 out	PI-59010	1-in-2-out low pressure	pressure alarm low		X			
18										
19	test1001PU34-PU_5	1 in 2 out	FI-59010	1-in-2-out low flow	flow alarm low		X			
20			FY-59010							
21	test1001PU34-PU_7	1 in 2 out	PAL-59010	1-in-2-out low pressure	pressure alarm low		X			

Figure 4.14 Cause-effect table for the item to be isolated

4.4 FURTHER ENHANCEMENT

Limitations in the current system will need to be addressed by further development. One limitation is the lack of an interface that can call the CLIPS reasoning process directly from the C++ environment and the other is that the current rule-base is incomplete and for a new or modified plant more rules might have to be developed to adopt the new or updated situation of the plant. Therefore, an interface to link the C++ and CLIPS could be considered. Another limitation is that only a small rule-base has been developed. The current rule-base can be enhanced by considering more plants, adding more rules and generalising the rules to make them more reusable. Furthermore, there is a potential to expand the reasoning rules to carry out configuration checks for process plants.

4.5 CONCLUSIONS

Carrying out safety analysis is important to prevent adverse consequences from happening and to be of assistance to the control design of protective methods in a process plant. Control and sensor devices and their working procedures can be presented in a SAFE chart to help with the analysis process. A SAFE chart is also called cause-effect table as it reflects information about process events and their corresponding safety guards.

With the electronically available P&IDs, a computer tool for automatically generating cause-effect tables is developed to reduce the effort required for the labour-intensive analysis process. The cause-effect analysis tool is introduced in this chapter with details of its components, working principles and data processing methods. Two case studies are used to exemplify the working of the system. The highlights of the system are that it provides a comprehensive list of process components and their attached devices that covers all the plant no matter whether there is a cause-effect applied to it or not and it offers two options in the result presentation as the user can choose to show only results with the cause-effect link or the complete results.

The cause-effect analysis tool can be used in conjunction with isolation tool to identify control design on and within the isolation boundary, therefore, giving the engineer a better insight of the picture of the isolation area.

5 FINDINGS & IMPLICATIONS

This chapter begins by highlighting the original contributions made by this thesis. Limitations of the work will be considered and possibilities of future work will be discussed.

5.1 CONTRIBUTIONS

As a result of this EngD project, the main contributions are the development of the three tools and one automatic reasoning system described above, how they can be integrated with an existing knowledge-based system, HAZID, and with one another. The "Model Test Bed", "Instrument Checker" and "Isolation Tool" are novel tools in the process industry for plant safety. The cause-effect system is a new automatic reasoning system to infer the process events and process responses in the control design. Drath et al. (2006) reported similar work and their results are compared with this new system. The new system provides more complete and detailed results covering the whole process plant under analysis.

This thesis represents an initial attempt to improve the usability of an existing knowledge-based system, HAZID, for HAZOP automation and integrate it with other novel safety-related tools. This research investigates, for the first time in the process safety area, extending the use of a well-developed hazard identification system, HAZID, together with a tool for the identification of isolation boundary. The research also explores, for the first time in the process safety area, applying a cause-effect analysis system in conjunction with the isolation tool to examine the safety of the control design within and on an identified isolation boundary. The research also represents an initial attempt at extending the application of P&IDs to isolation safety and control design of a process plant.

5.2 LIMITATIONS AND RECOMMENDATIONS FOR FURTHER WORK

The main limitation of these four tools is the input. Currently, all of them take only SmartPlant P&IDs as input. The plant representation can be changed to be compatible with the CAEX scheme. As mentioned in section 4.2.1.2**,** CAEX stands for **C**omputer-**A**ided **E**ngineering e**X**change. It is a neutral data transfer language for data exchange between different applications, for example from P&ID to Process Control Engineering (PCE) tool and vice versa. It is now available in XML format. When this is applied then HAZID and its accompanying tools can be used with other main P&ID packages and this would greatly increase the usability of the system.

The isolation tool can only help with the “hazard identification” step in the isolation process defined in the Health and Safety Executive (2006). Further development can cover the other steps in the process, for example, it can be developed to cover the “Risk assessment and selection of isolation scheme”.

The cause-effect system lacks an interface that can call the CLIPS reasoning process directly from the C++ environment and the other is that the current rule base is incomplete and more rules will need to be developed.

The overall limitation of tools is the lack of experiment from the real customers, which is also a limitation of the HAZID system. This is because some process engineers prefer trusting the manual HAZOP results to the computer generated results. So the tools can only be used as an aid to HAZOP study, control safety analysis and isolation safety, they are by no means a full replacement of manual work.

Furthermore, it is suggested that the tools can be further developed to have some learning capabilities by analysing the previously generated results. For example, by analysing the HAZOP results from the isolation tool considering difference closing sequence of the valves on the boundary, it might be able to suggest which sequence is better for the next analysis.

5.3 CONCLUSION

This thesis addresses three tools and one reasoning system that have been developed and integrated into an existing knowledge-based system HAZID to improve its usability and acceptability, covering the issues of knowledge representation testing, process plant isolation safety, process control safety.

To achieve the general and specific modelling goals, qualitative modelling dominates auto HAZOP analysis while quantitative modelling can be applied wherever necessary. Models built have to be verified and a Model Test Bed (MTB) is developed to reveal the qualitative behaviours of the model under test and to compare it with the intentional design, allowing further modification to gain confidence on the correctness of models built.

Two safety-related applications have been illustrated in detail in this thesis. One is the isolation tool which relates to isolation safety. The other is a cause-effect system which relates to control design safety. They all take P&IDs as input and produce safety-related results. They can be used together to help with hazard identification in the isolation area. An instrument checker is embedded in the cause-effect system to provide input to the inference engine and also reveal an insight of the control loops of a process plant to the process control engineer.

All above tools and system are tested and verified by the process engineers in the sponsor company.

The thesis represents, in the process safety area, the first attempt to use an isolation tool together with a well-developed auto HAZOP system for the hazard identification of isolation process and the first attempt to apply control safety analysis to isolation area, extending the use of P&IDs to isolation safety and control design of a process plant.

Further development of the tools is discussed. A limitation in the input can lead to the use of CAEX scheme, allowing neutral data transfer between different P&ID package. The isolation tool can be further developed to cover risk assessment and selection of an isolation scheme. It can also obtain some learning capabilities by analysing the results previously gained. The rule base in the cause-effect system can be further enhanced by applying it to more plants, detecting more control patterns.

6 REFERENCES

Bartolozzi, V., Castiglione, L., Picciotto A., and Galluzzo, M., 2000. Qualitative models of equipment units and their use in automatic HAZOP analysis. *Reliability Engineering & System Safety,* 70(1), pp.49-57.

Bratko, I., Suc, D., 2003. Learning qualitative models. *AI Magazine,* 24(4), pp.107-119.

Catino, C.A., Ungar,L.H.., 1995. Process Systems Engineering, Model-based approach to automated hazard identification of chemical plants. *AICHE (American Institute of Chemical Engineers') Journal*, 41(1), pp.97-109.

Chung, P.W.H., 1993. Qualitative analysis of process plant behaviour. *Proceedings of the Sixth International Conference,* Edinburgh, Scotland, 1-4 Jun, 1993, pp.277-283.

Crawley, F., Preston, M., and Tyler, B., 2000. *HAZOP: Guide to best practice _ guidelines to best practice for the process and chemical industries.* Rugby, Warwickshire, UK: Institution of Chemical Engineers (IChemE).

Drath, R., Fay, A., and Schmidberger, T., 2006. Computer-aided Design and Implementation of Interlock Control Code. *Computer-Aided Control Systems Design, 2006 IEEE International Symposium*. Ladenburg, Germany, 4-6 Oct, 2006, pp.2653-2658.

Giarratano, J., Riley, G., 1994. *Expert Systems, Principles and Programming.* 2nd ed. Boston: PWS Publishing Company.

Hale, A.R., Heming, B.H.J., Smit, K., Rodenburg, F.G.T., and Van Leeuwen, N.D., 1998. Evaluating safety in the management of maintenance activities in the chemical process industry. *Safety Science,* 28(1), pp.21-44.

Health and Safety Executive, 2006. *The safe isolation of plant and equipment.* London: HSE.

Hazid Technology Ltd., 2009. *Hazid Technology* [online], available from http://www.hazid.com [Accessed 10, Oct, 2009]

International Organization for Standardization, 2003. *ISO10418 Petroleum and natural gas industries — Offshore production installations — Basic surface process safety systems.* Switzerland: HIS.

Isograph Ltd., 2009. *HAZOP+ 2008* [online], available from http://www.isograph-software.com/hazover.htm?gclid=COnD5pSGpp0CFZQA4wodBG8J3A [Accessed 02, Oct, 09].

Kletz,T., 1999. *HAZOP AND HAZAN, Identify and assessing process industry hazards.* 4th ed. Institution of Chemical Engineers.

Lawley, H.G., 1974. Operability studies and hazard analysis. *Chemical Engineering Progress,* 70(4), pp.45–56.

Lihou technical & software service, 2009, *HAZOP Manager V6.0* [online], available from http://www.lihoutech.com/index1.html?gclid=CJGAsaGIpp0CFaBb4wod60JJ3Q [Accessed 09, Oct, 09].

McCoy, S.A., Zhou, D., and Chung, P.W.H., 2006. State-based modelling in hazard identification. *Applied Intelligence*, 24(3), pp.263-279.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Jefferson, M.L., 1999a. HAZID, A Computer Aid for Hazard Identification. 1. The STOPHAZ Package and The SPPS Code: An Overview, The Issues and The Structure. *Process Safety and Environmental Protection,* 77(B6), pp.317-327.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Chung, P.W.H., 1999b. HAZID, a computer aid for hazard identification. 2. Unit model system. *Process Safety and Environmental Protection,* 77(B6), pp.328-334.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Jefferson, M.L., 1999c. HAZID, a computer aid for hazard identification. 3. The fluid model and consequence evaluation systems. *Process Safety and Environmental Protection,* 77(B6), pp.335-353.

Nemeth, E., Cameron, I.T., and Hangos, K.M., 2005. Diagnostic goal driven modelling and simulation of multiscale process systems. *Computers and Chemical Engineering*, 29, pp.783-796.

Palmer, C., 2008. *Compute-Aided Hazard Identification of Batch Operations*. EngD. Loughborough, UK: Loughborough University.

Palmer, C., Chung, P. W.H., 2000. Creating Signed Directed Graph Models for Process Plants. *Ind. Eng. Chem. Res*, 39, pp.2548-2558.

Palmer, C., Chung, P. W.H., 1997. Constructing Qualitative Models. *Proceedings of the 1997 Jubilee Research Event, IChemE*, pp.725-728.

Palmer, C., Chung, P.W.H., 2001. Verification of Process Plant Models. *Proceedings of the 2001 AAAI Symposium on Model-based validation of intelligence*, Stanford, pp.37-41, AAAI press.

Redmill,F., Chudleigh, M., and Catmur, J., 1999. *System Safety: HAZOP and Software HAZOP.* West Sussex, England: John Wiley & Sons Ltd.

Riley, G., 2008. *CLIPS, A tool for building expert systems*. [online] USA: SourceForge. Available at http://clipsrules.sourceforge.net/WhatIsCLIPS.html [Accessed 15,Sep,2009].

Rushton, A.G., Chung, P.W.H., and Lees, F.P., 1998. HAZID, A computer aid for hazard identification: An Overview. *Proceedings of the Loss Prevention and Safety Promotion, the Process Industries 9th international symposium*, 2, pp.503-512.

Schaich, D., Becker, R., and King, R., 2001. Qualitative modelling for automatic identification of mathematical models of chemical reaction systems. *Control Engineering Practice,* 9(12), pp.1373-1381.

Thambirajah, J., Benabbas, L., Bauer, M., and Thornhill, N.F., 2009. Cause-and-effect analysis in chemical processes utilizing XML, plant connectivity and quantitative process history. *Computers & Chemical Engineering*, 33(2), pp.503-512.

Vaidhyanathan, R., Venkatasubramanian, V., 1995. Digraph-based models for automated HAZOP analysis. *Reliability Engineering and System Safety,* 50, pp.33-49.

Venkatasubramanian, V., Rengaswamy, R., Yin, K., and Kavuri, S.N., 2003a. A review of process fault detection and diagnosis Part I: Quantitative model-based methods. *Computers & Chemical Engineering,* 27(3), pp.293-311.

Venkatasubramanian, V., Rengaswamy, R., and Kavuri, S.N., 2003b. A review of process fault detection and diagnosis Part II: Qualitative models and search strategies. *Computers & Chemical Engineering,* 27(3), pp.313-326.

Wallace, S.J., Merritt, C.W., 2003. Know when to say "when": a review of safety incidents involving maintenance issues. *Process Safety Progress,* 22(4), pp.212-219.

Wells, G. L., Seagrave, C. J., 1976. *Flowsheeting for Safety - A Guide for Chemical Engineers on Safety Measures to Consider during the Design of Chemical Plant.* Institution of Chemical Engineers.

Yim, S. Y., Ananthakumar, H. G., Benabbas, L., Horch, A., Drath, R., and Thornhill, N. F., 2006. Using process topology in plant-wide control loop performance assessment. *Computers & Chemical Engineering*, 31(2), pp.86–99.

APPENDIX A PAPER 1

A Test Bed for verifying the correctness of safety-related behavioural knowledge in equipment model for use in automated hazard identification

An, H., McCoy, S.A., Chung, P.W.H., McDonald, J., and Madden, J., 2007, A model test bed to verify the correctness of safety-related behavioural knowledge in a system for automated hazard identification. *Proceedings of the 17th Advances in Risk and Reliability Technology Symposium (AR2TS) (Ed L.Bartlett)*, Loughborough University, UK, 17-19 Apr, 2007, pp. 204-218.

Abstract:

HAZID is a state-of-the-art software system used to identify hazards by emulating the conventional HAZOP study used in the Chemical Engineering domain. Unit models are used in the system to model the behaviour of various equipment items within a plant.

The Model Test Bed is an effective tool integrated with the current Model Builder to test the correctness of models built in the Unit Model Library. It allows the behaviour of the model under test to be compared to its expected behaviour, as specified by a chemical engineer domain expert. This paper establishes the significance of such testing by comparing the manual model test with the auto test, and illustrates the solution in terms of the methodology used, the architecture of the tool and the data structure. The presentation of testing results is also discussed by exploring different formats to present the same results in order for a chemical engineer to consider the model from different points of view. Further improvements of the tool are discussed.

Keyword: Process hazard identification, HAZOP, HAZID, Knowledge testing

Introduction

This paper describes a tool that considers the test of Unit Library Models in terms of correctness. It commences by introducing the HAZOP study, HAZID system and the Unit Library Models, then leading to the introduction of Model Test Bed. A number of development issues are addressed and different ways of result presentation are illustrated. Further developments are discussed.

HAZOP and HAZID

The HAZard and OPerability (HAZOP) study has been widely recognized as an effective hazard identification method throughout the process industry. It applies a procedure called "guideword-consequence tracing" on each equipment item in a plant to comprehensively check what faults occurring in the item would cause a deviation from its normal operating conditions, such as an increase in pressure or a decrease in temperature as explained by Madden et al. (2005) [1].

However, because of this comprehensive nature, it is not easy to apply HAZOP in practice. First, it is time consuming, because it requires people from different disciplines to meet together in many meetings. Second, it demands rich and complex data to be recorded and then analyzed repeatedly. If all these practical difficulties could be overcome, HAZOP would be even more powerful than it is now. HAZID is a state-of-the-art knowledge-based system developed by Hazid Technology Ltd. and Loughborough University to provide facilities to automate the HAZOP process. It is based on a Signed Direct Graph (SDG) theory that describes the qualitative change of variables. SDG is network of nodes connected by arcs. Nodes are variables such as “flow”, “temperature” or “pressure” in the system. Arcs are influences between nodes, of one variable on another. Sign identifies type of influences attached to an arc, negative or positive. For example, “more flow” will result in “less pressure” if the sign attached to the arc is negative or “more temperature” will result in “more flow” if the sign attached to the arc is positive etc. Details about SDG are described by Chung (1993) [2].

Unit Models and Model Library

In this HAZOP emulation system, a plant is represented as a network consisted of a series of interconnected units and is placed into a library named Unit Model Library (UML). Each unit is described as a model built on the theory of SDG. For example, the “'centrifugal pump'” in figure 1 is one of the unit models. It is described as an instance in the plant file:

```
instance('test_unit' isa 'centrifugal pump', [
                inports info [
                        'ignoreIn',
                        'in' ,
                        'sealIn'
                ],
                outports info [
                        'ignoreOut',
                        'out' ,
                        'drainOut' ,
                        'ventOut' ,
                        'sealOut'
                ],
                unitports info [
                ],
                propLinks info [
                   % faults
                        arc([fault,['loss of drive',use_is_not_standby]],-1,['out', 'pressure']),
                        arc([fault,['loss of drive',use_is_not_standby]],1,['in', 'pressure']),
                        arc(['in', 'composition'],1,['out', 'composition']),
                        % propagation
                        arc(['in', 'noFlow'],2,['out', 'noFlow']),
                        arc(['out', 'noFlow'],2,['in', 'noFlow']),
                        arc(['in', 'temp'],1,['out', 'temp']),
                        arc(['in', 'contamination'],2,['out', 'contamination']),
                        arc(['in', 'solid'],2,['out', 'solid'])
                                      ]
                            ]
              ).
```

This model says that the test_unit is a centrifugal pump, it has 3 inports as listed in the slot "inports info", and has 5 outports as listed in the slot "outports info". It does not have any internal port as none listed in the slot "unitPorts info". The "propLinks info" slot stores a list of arcs that define the mini-SDG related to a centrifugal pump. "-1","1" and "2" are influences. Among them, "-1" is a reversed influence, it means "Higher input implies lower output and lower input implies higher output" ,"1" is a direct influence and it means "Higher input implies higher output and lower input implies lower output" while "2" only means "Higher input implies higher output". For example," arc(['in', 'solid'],2,['out', 'solid'])" means a solid increase at the in port "in" will cause a solid increase at the out port "out" but a solid decrease at the in port "in" does not mean that there is going to be a solid decrease at the out port "out".

The units described above are organized into a hierarchical structure in which the child model unit inherits all the characters of its parent model while contains its new characters that distinguish itself from other child unit models of its parents. For instance, in table 1, "loss of drive" in parent model "Rotary pump" will cause "revFlow" at port "out", this is inherited by both child models "centrifugal pump" and "double seal centrifugal pump", but "leaks to seal" is a cause of "morePressure" only through the child model "double seal centrifugal pump" not through "centrifugal pump". In addition, each child model unit may have its child unit as well.

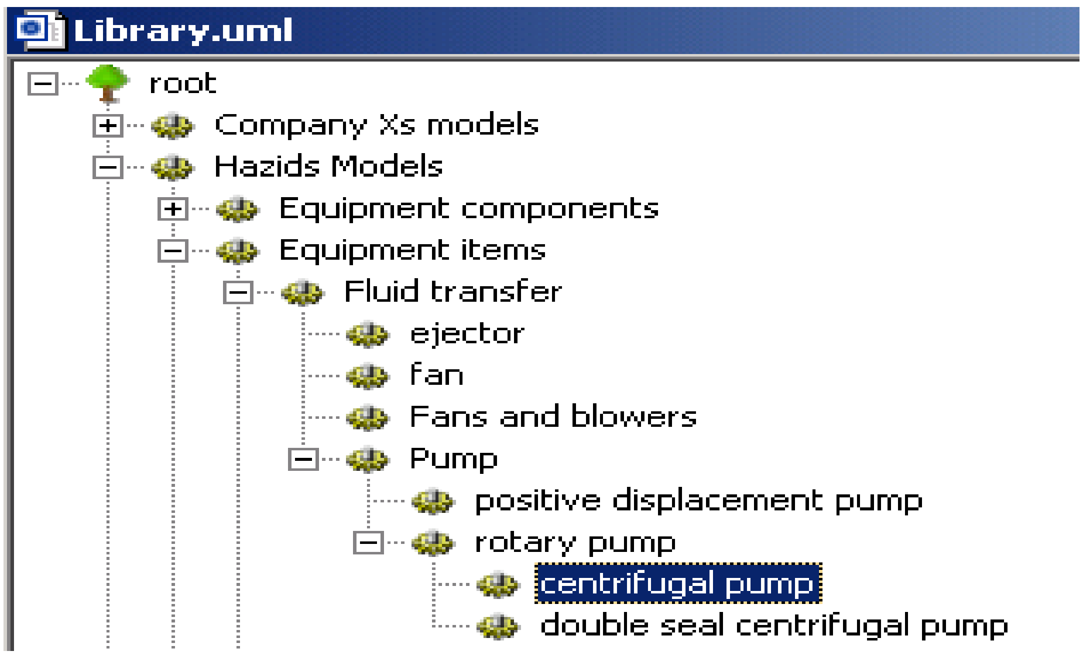


Figure 1: Part of the Unit Model Library (UML)

	Parent model	**Child models**	
	Rotary pump	Centrifugal pump	Double seal centrifugal pump
Inherited arc	"loss of drive" causes "revFlow" at port "out"		
Specific arc for seal centrifugal pump			"leaks to seal" causes "morePressure" at port "seal1out"

Table 1 an example of inheritance in Unit Model Library (UML)

The use of inheritance to build the unit model library keeps the unit model as concise as possible and increases speed to build a new model, as only the unique feature(s) of it have to be defined.

The unit model structure is described in detail by McCoy et al. (1999) [3]. The unit-based approach is widely used, for example, by Vaidhyanathan and Venkatasubramanian (1995) [4]. Other model based approach applying qualitative physics in process safety is described by Catino and Ungar (1995) [5].

Model Testing Methodology

The correctness of a model built by the chemical engineer is critical to the result of HAZOP study, because errors in a model may result in incorrect faults, consequences, deviations as well as wrong propagations in the HAZOP result, therefore distracting HAZOP study members' attention from really critical hazards. Definition of verifying process plant models is described by Claire and Chung, 2001[6]. A tool is demanded to be able to allow the chemical engineer to verify the model's correctness and then come back to modify the model until it is the one in his /her own mind.

Traditionally, this work are done manually by constructing a test drawing using the model to be tested, adding the Off-Page-Connector(OPC) to each inlet and each outlet of the model, analyzing using HAZID.

Manually generated test results are not presented in a format which allows the user to quickly check that they agree with his/her expectation. The user must browse through the HAZOP-style results system to see the actual results, and must refer to the Model Builder software to see if the results are what were expected. Therefore, there is a need to present both actual and expected result of a test more effectively on screen. The interface developed to solve this problem is one of the subjects of the rest of this paper.

Automated Model Test Bed

The Model Test Bed is an effective tool integrated with the current HAZID Model Builder (MBuilder) to test the correctness of models built in the Unit Model Library. It allows the behaviour of the model under test to be compared to its expected behaviour, as specified by the chemical engineer domain expert.

The use case

The user of the Model Test Bed will be a trained user of the MBuilder, with the knowledge of constructing unit models and is responsible for the development of the HAZID knowledge base for their organization. This user may also have experience as a process engineer. During development of models, the user will test individual model in order to ensure that the model he/she is building does not have any unintended behaviours or does not lack of any intended behaviours, which means the same model may be tested more than once. There must be a record of the intended behaviour of each model under test in order to verify the actual behaviour of the model during each test. The user may test a model after making a change to it, so that there will be a result report to show him/her any differences between the expected and the actual results, then he/she would either go back to make further modifications to rectify the model or sign off the model as acceptable. The user may apply different conditions when testing a model and the result may be different for each test case of the same model. Also, the user may sometimes test more than one model at one go, or perhaps the whole unit model library, so that there will be a summary report for all tested models and an individual report for each model**.**

The interface

To test the model in MBuilder which already includes GUI, the unit model testing application includes graphical user interface(s) and facilities for recording test results. Also, since the Model Test Bed is designed to be used by the engineer in charge of developing new models, the interface for initiating tests on individual models is closely integrated with the Model Builder. A facility for selecting which model is going to be tested is shown in figure 2.

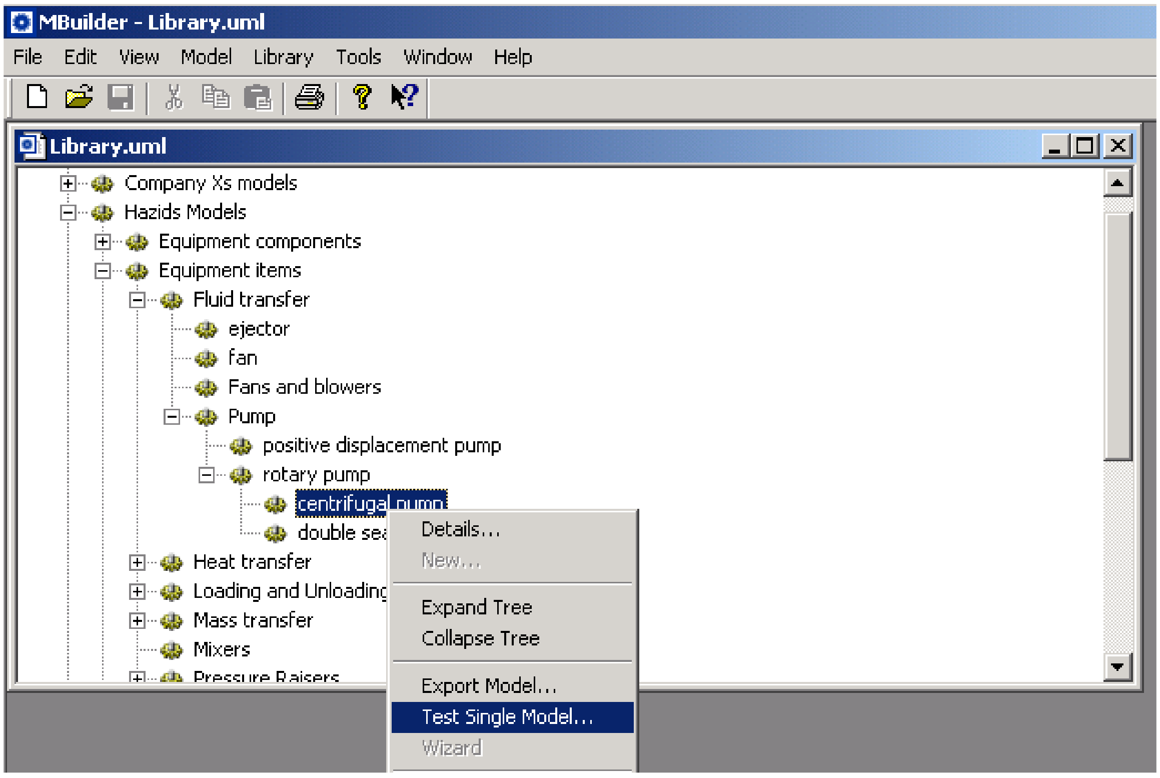
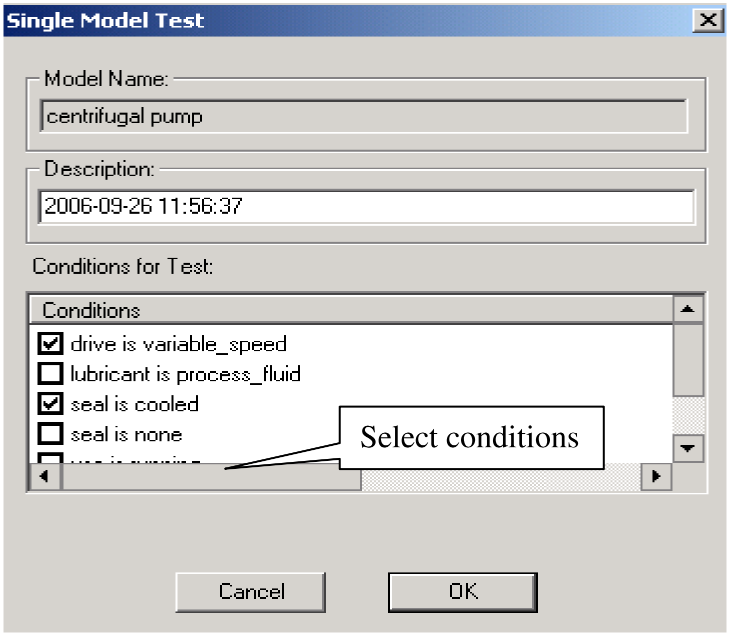


Figure 2. Interface to select a single model to be tested

For a single model to be tested, a dialog is provided (as shown in figure 3) to allow the user to set up the test and give a description with current system time and date as the default description, so that each test case is easily identified. The applicable conditions are drawn from those defined on the "Link Conditions" (explained later at "operating conditions") elsewhere in the MBuilder.


Single Model Test

Model Name: centrifugal pump

Description: 2006-09-26 11:56:37

Conditions for Test:

Conditions
☑ drive is variable_speed
☐ lubricant is process_fluid
☑ seal is cooled
☐ seal is none

Figure 3. Interface to apply link conditions to a model to be tested

To test multiple models, the user selects an option from the "Tools" menu in the MBuilder.

For both single and multiple model tests, a feedback is provided to show the progress of a single or a batched test as shown by figure 4.

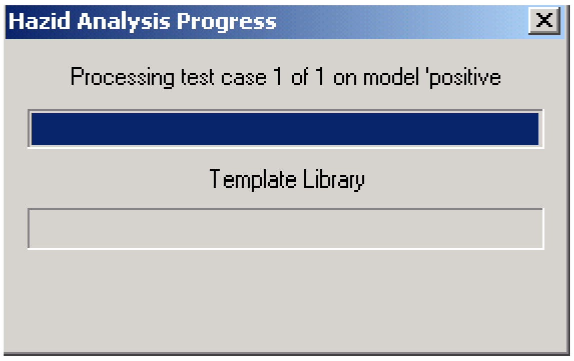


Figure 4: Progress display during model testing

A dialog to inform the user that the results are available in model test page of HMeeting (this is the PHP script online interface to view results) appears when the test is finished.

Function and data

To start the new analysis on the model, the application will call 'HAZID' automatically.

To complete the test, the application tests each of the defined deviations on each model. For example, to test the model 'centrifugal pump', the application will exam all the deviations automatically although it ends up that only 7 deviations are relevant to this model. The application is able to test the deviation in reverse direction, eg: from model to upstream port such as a "pump drive fails" causes "reverse flow" at its out port, then "reverse flow" at its in port and "reverse flow" at the out port of an OPC , eventually "causes reverse flow and contamination" at inlet sheet connector as its consequence. All that means, it will test fault caused within the model, which could result in deviations in any upstream or downstream port.

Model Test Bed follows the same "Method" that other HAZID applications use to open/connect to a database. The results of the analysis of test plants are stored safely into ORACLE database with a series of tables –these results will later be compared to the intended results for the given model and will be available for later review, audit or comparison.

Multiple tests on the same model are also recorded intact, and indexed on their time of production (among other factors), so that it is possible to refer to earlier results as well as the most recently generated ones.

Connect to OPC (Off-Page-Connector)

Each time a model is tested, a test plant file is created automatically and stored as a temporary file in user's temporary directory. The test plant consists of a single instance of the model to be tested, with each of its external ports connected to an OPC instance.

For each "import", the application will add an OPC, and connect to the port automatically. In the example of "centrifugal pump" in figure 1, each in port will be connected to its OPC.

For each "outPort", the application will add an OPC, and connect to the port automatically, In the example of "centrifugal pump", each out port will be connected to its OPC.

By this way, the model under test is connected to deviations caused by faults in the OPCs and links any deviations caused by faults in the test model to consequences in the OPCs.

Single Model Test and Multiple Model Test

As mentioned early, the user can test one single model or more than one model in a batch run. As demonstrated in figure 5, a single model test means to test one model at a model test run, to have one model to be tested at one model test case. A multiple test means to test more than one model at a model test run, to have at least two models to be tested at one model test case. Either of them will have four forms of test result presentation, which are "Fault Comparison", "Consequence Comparison", "Fault Path Propagation" and "Deviation oriented" , these four formats of result presentation will be discussed later.

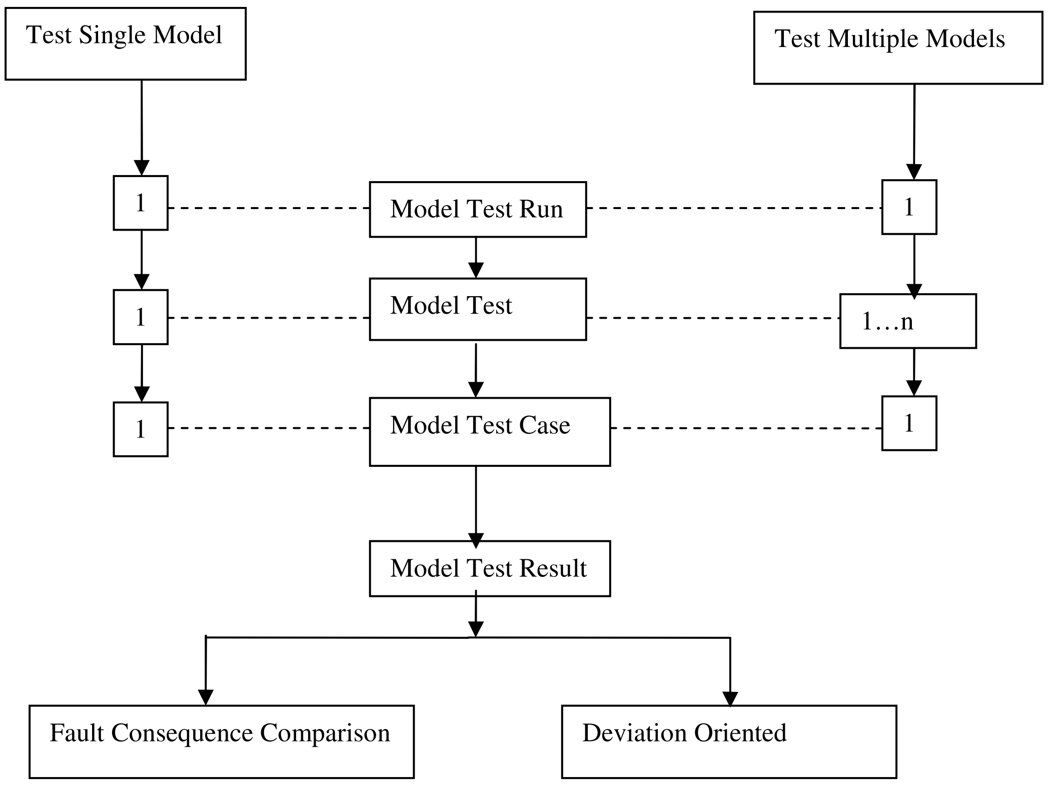


Figure 5. Architecture of Model Test Bed

Operating conditions

The operating conditions define which options are available to customize the unit model by using attributes, for example, when testing the model “centrifugal pump”, there are 7 options that could be selected by the user, the model test bed will combine these selected conditions and apply them to the HAZID analysis to get the test result.

The fault path and Deviation

A fault path is an acyclic path within the Signed Direct Graph (SDG) between the fault and the variable under consideration, which the fault can cause deviations or consequences. Deviations consist of keywords such as “More”, “Less”, ”No”, ”Reverse” etc and parameters such as “Flow”, ”Pressure”, ”Temperature” ,”Level” etc.

Any fault path propagating through a unit model under test will fall into one of the four types shown in figure 6. In case A, the fault happens in an OPC and causes a series of deviations that lead to a consequence in the test unit. In case B, the fault happens within the test unit while the consequence happens in some OPC. In case C, both the fault and consequence happen within the test unit. In case D, both the fault and consequence happen outside of the test unit and only deviations propagate through the test unit. To complete the test, all these four types are considered in the Model Test Bed and the test results show each of them whenever it happens.

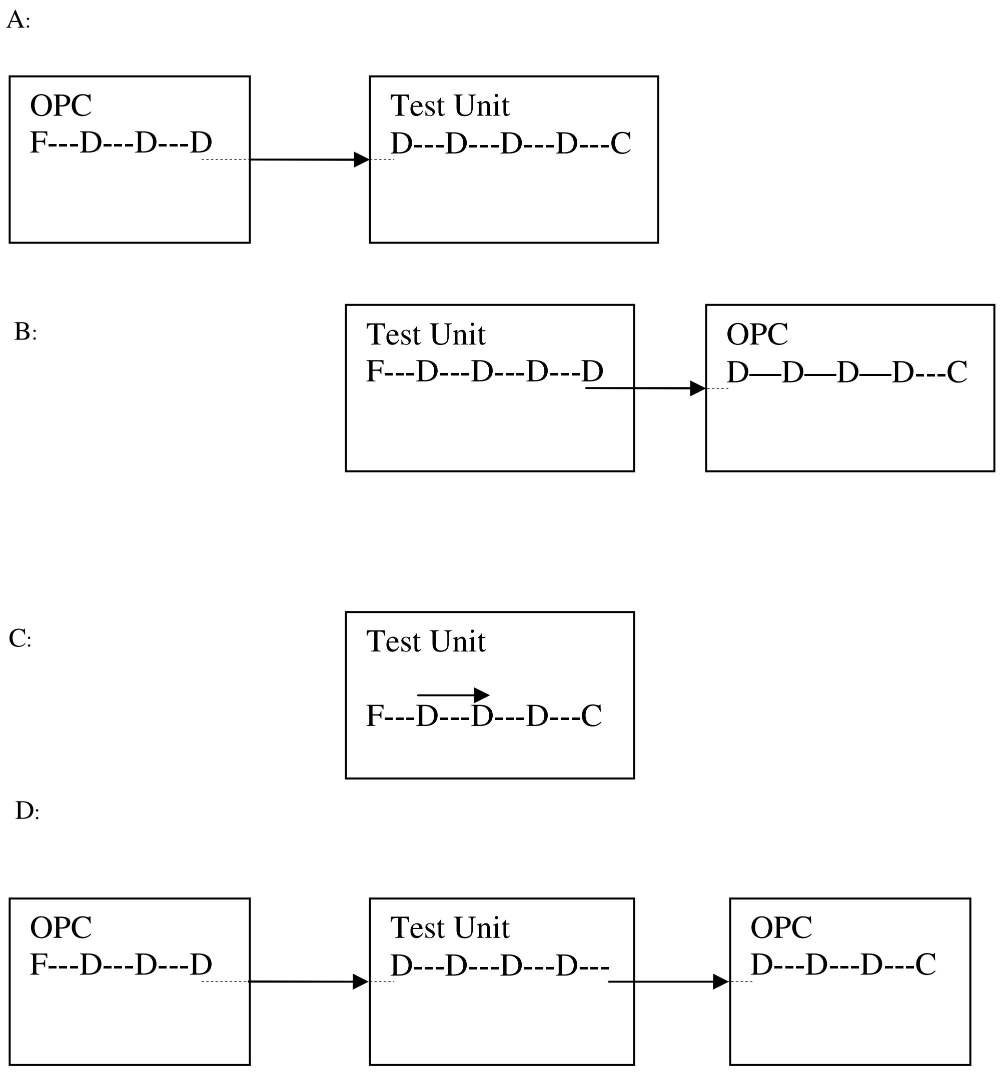


Figure 6. Four types of fault paths propagating through a test unit model

Model Test Result

The results of model test are presented in four different formats so that the chemical engineer can consider the model from different points of view. Table 2, 3 , 4, 5 are different result presentations of testing “centrifugal pump” model.

Expected Results

The test bed provides a display to show how the actual behaviour of the model differs from the intended behaviour, as specified by the engineer in the MBuilder. The expected results of the test runs on specified unit models are listed in the result table as a section listing the faults or consequences (“Expected Description”) in the model and the directly associated deviations (“Expected Keyword”).This is shown in left 3 columns of table 2 and 3.

Deviation caused by Fault(s)

As demonstrated in table 2, one section of the display shows a table of “Fault → Deviation” results, showing how initial faults can give rise to deviations which propagate to the OPCs and therefore elsewhere in the plant. This is compared with the expected faults and deviations, giving a status as “MATCH” if they are the same, “MISSING” if there is no actual fault found and “EXTRA” if an actual fault is found but it is not expected by the engineer.

Deviation causes Consequence(s)

As demonstrated in table 3, one section of the display shows a table of “Deviation → Consequence” results, showing how deviations in the OPCs can cause consequences in the unit under test. This is compared with the expected deviations and consequences, giving a status as “MATCH” if they are the same, “MISSING” if there is no actual consequence found and “EXTRA” if there is actual consequence found but it is not expected by the engineer.

Fault Path Propagation

Table 4 demonstrates how a specific fault path propagates through a test unit model, its fault and consequence as well as deviation(s) along the path between the fault and consequence.

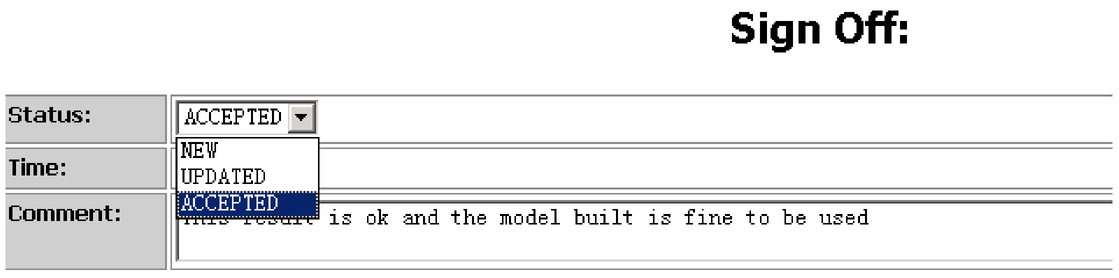
Deviation Propagation

The deviation propagation is demonstrated as a “Deviation Oriented” table. It checks deviations happened in each port and observes its effects on all other ports within a test unit model and gives fault that causes that deviation on that port as well as consequence(s) caused by it ultimately.

Table 5 demonstrates part of the deviation propagation table of the “centrifugal pump” model. The whole table shows all the deviation effects propagating through each port of the test unit, from “more Composition C+” to “more Vapour V+”. For example, when fault “loss of drive” causes “reverse flow FR” at port “out”, it causes “reverse flow FR” at port “in”, and causes consequence “possible pump or seal damage”. “0” means there is no effect caused by the deviation “reverse flow FR” in port “drainOut”, “ignoreIn” and “ignoreOut” and “ventOut”.

Sign off status

A facility has been provided for recording the test status of each model tested through time. User can sign off a model as acceptable after reviewing the test results or record that the results are not yet acceptable, this is a so called sign-off status that can be accompanied by a textual comment from the user. This is shown in figure 7.



Sign Off:

Status:	ACCEPTED
Time:	
Comment:	This result is ok and the model built is fine to be used

Figure 7: Sign off status

Further Improvement

Although the current model test bed is sufficient to be used as a tool to verify the correctness of the models built by the engineer, further improvement can be made to enhance its usability and acceptability.

- ***Run Time Conditions (RTC)***

Run Time Conditions (RTC) are tests based on fluid properties or details of the plant equipment, which are evaluated when HAZID is analysing the plant for hazards. For example, a consequence of “fire or explosion” may have an attached RTC to check that the fluid in the plant is flammable — if the RTC fails, this consequence will not be reported.

The Model Test Bed should be able to test model with RTCs, so that users will know that the model performs correctly whether the RTC fails or succeeds.

- ***Operating conditions for multiple Model Test***

The operating conditions set mentioned early in the single model test will be considered at Multiple Model Test.

- ***Interface with external applications***

We have proven that exporting result to external application is feasible (eg. Excel). Further work will focus on building up an interface to implement this feature.

- ***Graphic result display***

Displaying results graphically and animatedly will be explored as soon as possible as it will enhance the tool’s result interface greatly.

Summary

This paper has given a description of the Model Test Bed, a tool used to verify the correctness of unit model built by the chemical engineer in HAZID. It briefly introduces HAZOP, HAZID and Unit Model Library to lead the significance of Model Test Bed. Then it illustrates the methodology used to build this tool and different ways of testing results presentation. Several figures and tables are used to help the illustration. Further developments are discussed and are now undertaking.

References

1. Madden, J., Brugha, D.J. and McCoy, S.A., 2005. Complete, consistent Hazops with less time, cost and pain. *Proceedings of the World Congress in Chemical Engineering.*

2. Chung, P.W.H., 1993. Qualitative analysis of process plant behaviour. *Proceedings of the Sixth International Conference,* Edinburgh, Scotland, 1-4 Jun, 1993, pp.277-283.

3. McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Chung, P.W.H., 1999b. HAZID, a computer aid for hazard identification. 2. Unit model system. *Process Safety and Environmental Protection,* 77(B6), pp.328-334.

4. Vaidhyanathan, R., Venkatasubramanian, V., 1995. Digraph-based models for automated HAZOP analysis. *Reliability Engineering and System Safety,* 50, pp.33-49.

5. Catino, C.A., Ungar, L.H.., 1995. Process Systems Engineering, Model-based approach to automated hazard identification of chemical plants. *AICHE (American Institute of Chemical Engineers') Journal*, 41(1), pp.97-109.

6. Palmer, C., Chung, P.W.H., 2001. Verification of Process Plant Models. *Proceedings of the 2001 AAAI Symposium on Model-based validation of intelligence*, Stanford, pp.37-41, AAAI press.

Acknowledgement

The author H. An is an EngD Research Engineer funded by the Engineering and Physical Sciences Research Council (EPSRC) and Hazid Technologies and she is very grateful to both of them.

Address

Correspondence relating to this paper should be addressed to Professor P.W.H.Chung, Department of Computer Science, Loughborough University, Loughborough LE11,3TU,UK.

Other authors are working at Hazid Technologies Ltd., SGCS Business Park, Beeston, Nottingham NG9 2ND,UK

Fault Comparison Table

Expected Description	**Expected Port**	**Expected Keyword**	**Actual Description**	**Actual Port**	**Actual Keyword**	**Status**
loss of drive	out	revFlow	loss of drive	out	revFlow	MATCH
loss of drive	out	revFlow	loss of drive	out	revFlow	MATCH
loss of drive	in	morePressure				MISSING
loss of drive	out	lessFlow				MISSING
loss of drive	out	lessPressure				MISSING
switched on in error	in	lessPressure				MISSING
switched on in error	in	lessPressure				MISSING
switched on in error	out	morePressure				MISSING
switched on in error	out	morePressure				MISSING
switched on in error	out	moreTemp				MISSING
switched on in error	out	moreTemp				MISSING
			contamination from sheet connector	in	contamination	EXTRA
			lessComposition from sheet connector	in	lessComposition	EXTRA
			lessPressure from sheet connector and less flow	in	lessPressure	EXTRA
			lessPressure from sheet connector and less flow	out	lessPressure	EXTRA

Table 2 Fault comparison table

Consequence Comparison Table

Expected Description	**Expected Port**	**Expected Keyword**	**Actual Description**	**Actual Port**	**Actual Keyword**	**Status**
possible pump or seal damage	out	revFlow	possible pump or seal damage	out	revFlow	MATCH
possible pump or seal damage	out	revFlow	possible pump or seal damage	out	revFlow	MATCH
cavitation	in	moreGas				MISSING
cavitation	in	moreVapour				MISSING
churning	out	noFlow				MISSING
dry running	in	noFlow				MISSING
			cavitation	out	moreGas	EXTRA
			cavitation	out	moreTemp	EXTRA
			cavitation	out	moreVapour	EXTRA
			churning	in	noFlow	EXTRA

Table 3 Consequence comparison table

Path Propagation Display

Fault					
Item	Descriptor	Probability	Condition		
test_unit	loss of drive	0	“use_is_not_standby”		
Deviations					
Item	Port	Keyword			
test_unit	out	revFlow			
test_unit	in	revFlow			
inopc_in	out	revFlow			
Conseque-nces					
Item	Descriptor	Hazard	Operability	Ran	Condition
inopc_in	Reverse flow and contamination to inlet sheet connector	FALSE	FALSE	5	not(fault_is_this(“reverseflow at inlet sheet connector”))‖”fault_unit_is_not_this_one”)

Table 4 Path Propagation Display

UML Model Test Case Results_Dev Table

Path_id	Deviation	Affected _port	Fault	Effects on other ports						Consequence
				drainOut	Ignore In	Ignore Out	in	out	Vent Out	
0000000000-0000006888	C+	in	propagated from connection	0	0	0	C+	C+	0	propagated from unit
0000000000-0000006890	C-	in	propagated from connection	0	0	0	C-	C-	0	propagated from unit
0000000000-0000006892	Cont+	in	propagated from connection	0	0	0	Cont+	Cont+	0	propagated from unit
0000000000-0000006792	F0	in	propagated from connection	0	0	0	F0	F0	0	dry running, propagated from unit
0000000000-0000006806	F0	out	propagated from connection	0	0	0	F0	F0	0	churning, propagated from unit
0000000000-0000006800	FR	in	propagated from connection	0	0	0	FR	FR	0	possible pump or seal damage, propagated from unit
0000000000-0000006804	FR	out	loss of drive	0	0	0	FR	FR	0	possible pump or seal damage, propagated from unit

Table 5. Deviation Oriented Table

APPENDIX B PAPER 2

A Computer Tool to Support Safe Isolation for Maintenance

An, H., Chung, P.W.H., McDonald, J., and Madden, J., 2008. A Computer Tool to Support Safe Isolation for Maintenance. *Proceedings of the 2nd World Conference on Safety of Oil and Gas Industry*, Texas, USA, 28-29 Oct 2008, pp.406-415.

Abstract

Unsafe maintenance in process plant can cause release of dangerous materials, pipe-work failure and deviations from normal operations, etc (Hale, et al., 1998). It is reported that 30% of accidents are maintenance-related and 50% of them release harmful substances (Wallace & Merritt, 2003). Therefore, it is important that systematic hazard identification is carried out and precaution is taken before maintenance work commences.

A computer-aided tool is developed as part of the HAZID system (a knowledge-based software system used to identify hazards by emulating conventional HAZOP study) to help the task of identifying hazards related to maintenance work. This tool focuses on safe isolation. It serves two functions. One is to suggest an isolation boundary for maintenance. Given specific equipment items to be maintained, the system will analyze the Process and Instrumentation Diagram (P&ID) to identify the boundary that needs to be closed off for safe maintenance. The other function is to identify the potential hazards related to the isolation tasks. This paper describes in detail how this tool is developed and a case study is used to illustrate how it works.

1. Introduction

The safety of hazardous processing plants is of paramount importance as an accident could cause major damage to properties and/or injury to people. Well-maintained equipment in the process plant can give smooth running of the plant and increase the plant productivity and life time. However, the maintenance work of process plant is often dangerous as it requires appropriate isolation of the equipment items being maintained. Any release of hazardous material can cause damage to the whole plant or even take human life. Therefore, a comprehensive identification of potential hazards caused by maintenance work is necessary before carrying out the actual maintenance work.

This paper describes a computer-aided tool that considers the safety issues for maintenance work in the process plant. It serves two main functions, one is to define an isolation boundary, the other is to identify potential hazards related to the isolation task. The tool is newly developed and integrated with the HAZID system, a computer system that helps designers and operators of process plants to identify potential design and operation problems given a process plant design.

The paper commences by describing the methodology for identifying the isolation boundary and how the algorithm is tested. It then describes how HAZard and OPerability study (HAZOP) analysis is applied to identify hazards after the boundary is identified and selected.

The details of HAZOP study technique can be found at Crawley et al. (2000) and at Kletz (1999). The overall workflow of the tool is then given. The paper ends with a summary of the overall methodology.

2. Identifying the isolation boundary

When equipment items are to be maintained in a process plant, a process engineer will analyse the P&ID of that plant, and identify the valves that must be closed in order to isolate the equipment items so that they are safe to work on. For example, consider figure 1, if the centrifugal pump "P-0101A" is to be maintained, then 5 valves, "V015", "V002", "V014", "V013" and "V001", will need to be closed to isolate the pump. This process of identification and the analysis of the potential hazards caused by closing these valves are automated by a computer tool and is the subject of the rest of this paper.

2.1 Introducing the plant file

The plant file, which is a text file generated from the P&ID, describes the equipment items in the plant, which includes their connections, flow directions and other attributes. The plant file is used as input into the isolation tool. Each valve in the plant file has an attribute called "canBeIsolationValve". This is used to indicate whether the valve can be in general used for isolation purpose. The value of the attribute is either "yes" or "no".

2.2 The tracing procedure

The procedure for identifying the isolation boundary is to trace upstream and downstream from the equipment items to be maintained to find the valves which must be closed. During tracing, the tool will only look for the first valve that is isolable in each line branching out from the equipment items.

When a Pressure Relief Valve (PRV) is met, the algorithm will compare the direction of the tracing and the opening direction of the PRV, if they are in the same direction, then the tracing procedure will carry on and the PRV will be identified as within the boundary but must be isolated, otherwise, the PRV will be identified as on the boundary and the tracing procedure at this branching line ends at this point.

When searching from the equipment items to be maintained, if the propagation passes through a sheet connector onto another P&ID then it will continue on the next P&ID until an isolable valve is found. If there is no continuation from the sheet connector, the sheet connector will be treated as being on the boundary and the user will be warned that an isolation point has not been found for that particular branch. The sheet connector ID or tag number and the pipe connected to the sheet connector will be identified in the warning.

When multiple items are to be maintained, the same procedure will be applied to all the items. The valves that need to be isolated for all the items will be combined together and any duplicates are removed.

2.3 Testing

Two plants, a hydrocarbon separation unit (Lawley, 1974) and Benzene (Wells and Seagrave, 1976), are used to test the algorithm. The algorithm correctly identifies the isolation boundaries when given different maintenance items as input for the two plants. In order to help the user to visualize a boundary the isolation tool automatically highlights the valves to be closed on the Smart Plant P&ID CAD system. Figure 1 shows the output for isolating pump "P-0101A" for the hydrocarbon separation unit. All the isolable valves are highlighted in red. The item to be maintained, "P-0101A", is highlighted in bright blue.

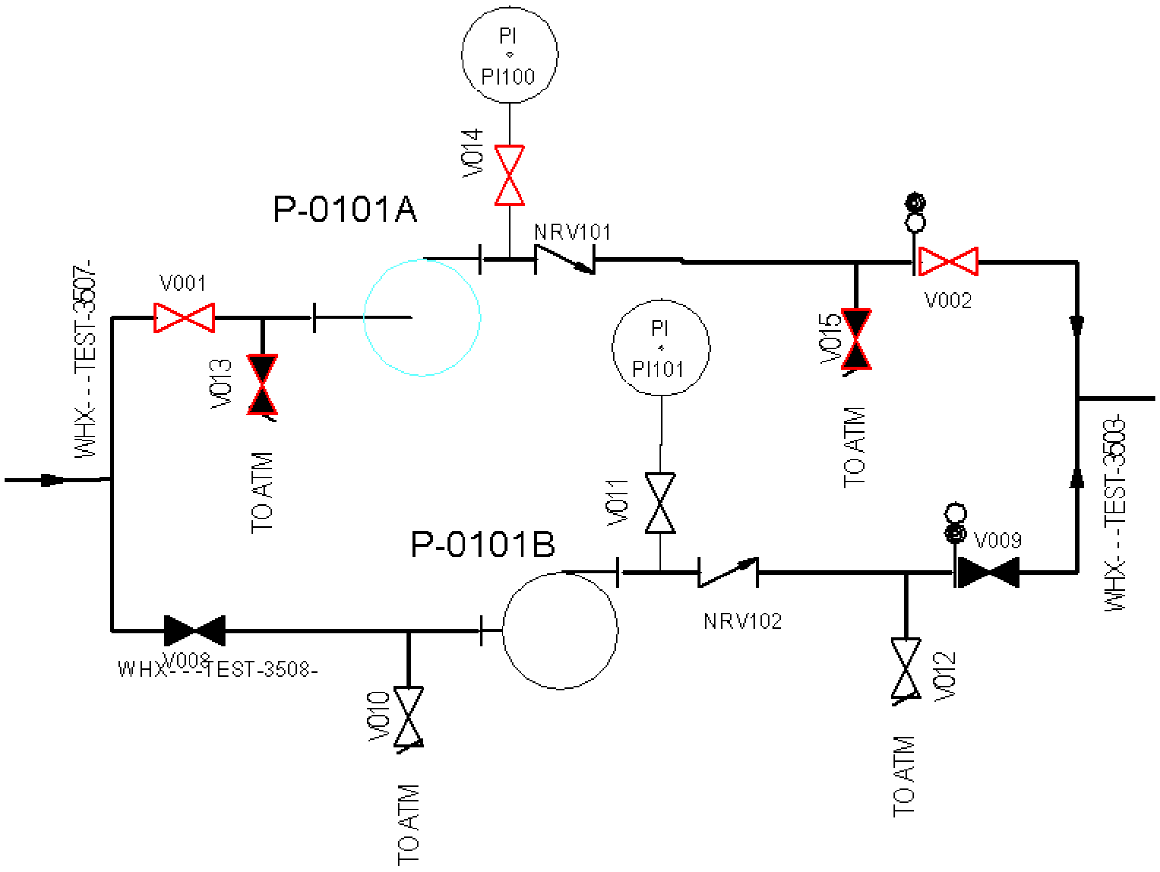
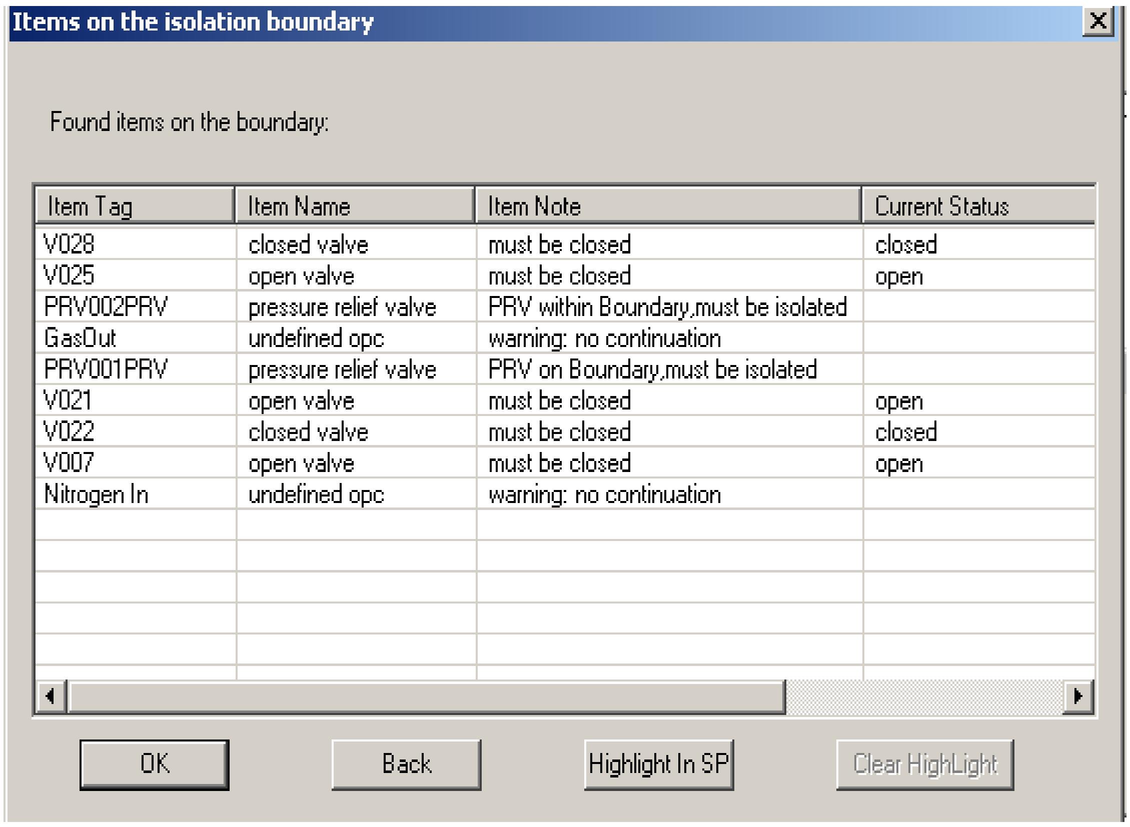


Figure 1. The highlighted isolation boundary for maintaining centrifugal pump "P-0101A"

A more sophisticated example is to maintain the liquid-liquid-gas separator in the hydrocarbon separation unit plant. Figure 2 shows a table of equipment items on the boundary with their current status and related notes. The tool also displays a table with all the items within the boundary. Figure 3 shows these items highlighted in Smart Plant P&ID. This example shows that several different types of equipment items are on the boundary. The first type is the "must be closed" valve, such as "V028", "V025","V021", "V022", "V007", which are highlighted in Smart Plant P&ID drawing in red. The second type is "Pressure Relief Valve (PRV)", such as "PRV002PRV", "PRV001PRV". The third type is the "Off-Page-Connector (OPC)". The second and third types are highlighted in Smart Plant P&ID drawing in purple. The items to be maintained are highlighted in bright blue.



Found items on the boundary:

Item Tag	Item Name	Item Note	Current Status
V028	closed valve	must be closed	closed
V025	open valve	must be closed	open
PRV002PRV	pressure relief valve	PRV within Boundary,must be isolated	
GasOut	undefined opc	warning: no continuation	
PRV001PRV	pressure relief valve	PRV on Boundary,must be isolated	
V021	open valve	must be closed	open
V022	closed valve	must be closed	closed
V007	open valve	must be closed	open
Nitrogen In	undefined opc	warning: no continuation	

Figure 2. Items on the isolation boundary for maintaining separator “T-0100”

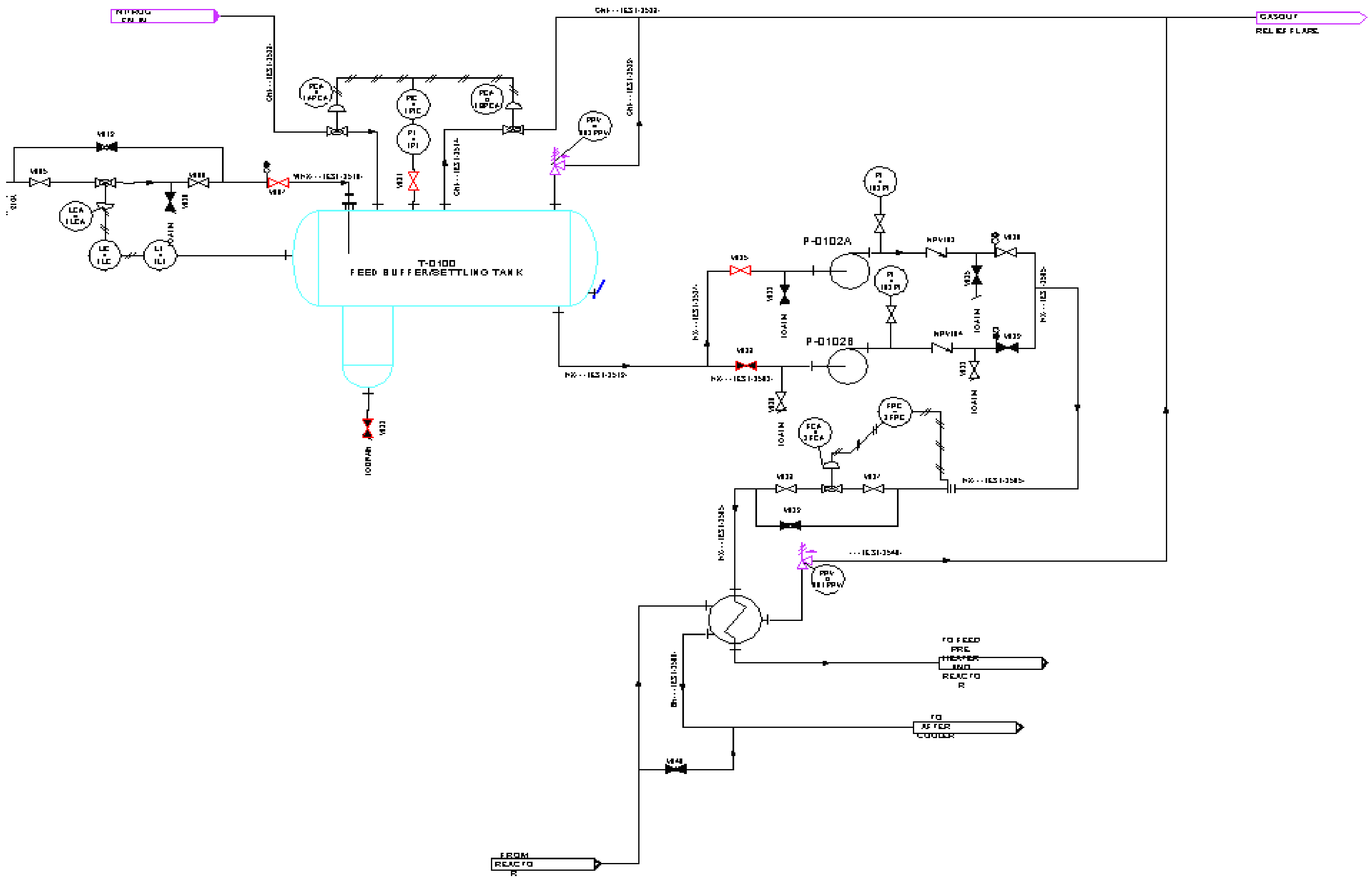


Figure 3. The highlighted isolation boundary for maintaining separator “T-0100”

3. Identifying hazard after the isolation boundary is defined

Once the boundary is identified a HAZOP analysis has to be carried out to identify the hazards related to the isolation procedure. HAZID, a knowledge-based system which automates the process of HAZOP studies, is used for this purpose. HAZID system is developed by Hazid Technologies Ltd. and is currently the most advanced commercially available knowledge-based HAZOP tool. (Hazid Technology. Ltd, 2007). The development of HAZID is given by Rushton et al. (1998) and by McCoy et al. (1999a, 1999b, 1999c). There are other HAZOP automation systems for research purpose that can be found at Bartolozzi et al. (2000) and at Vaidhyanathan and Venkatasubramanian (1995).

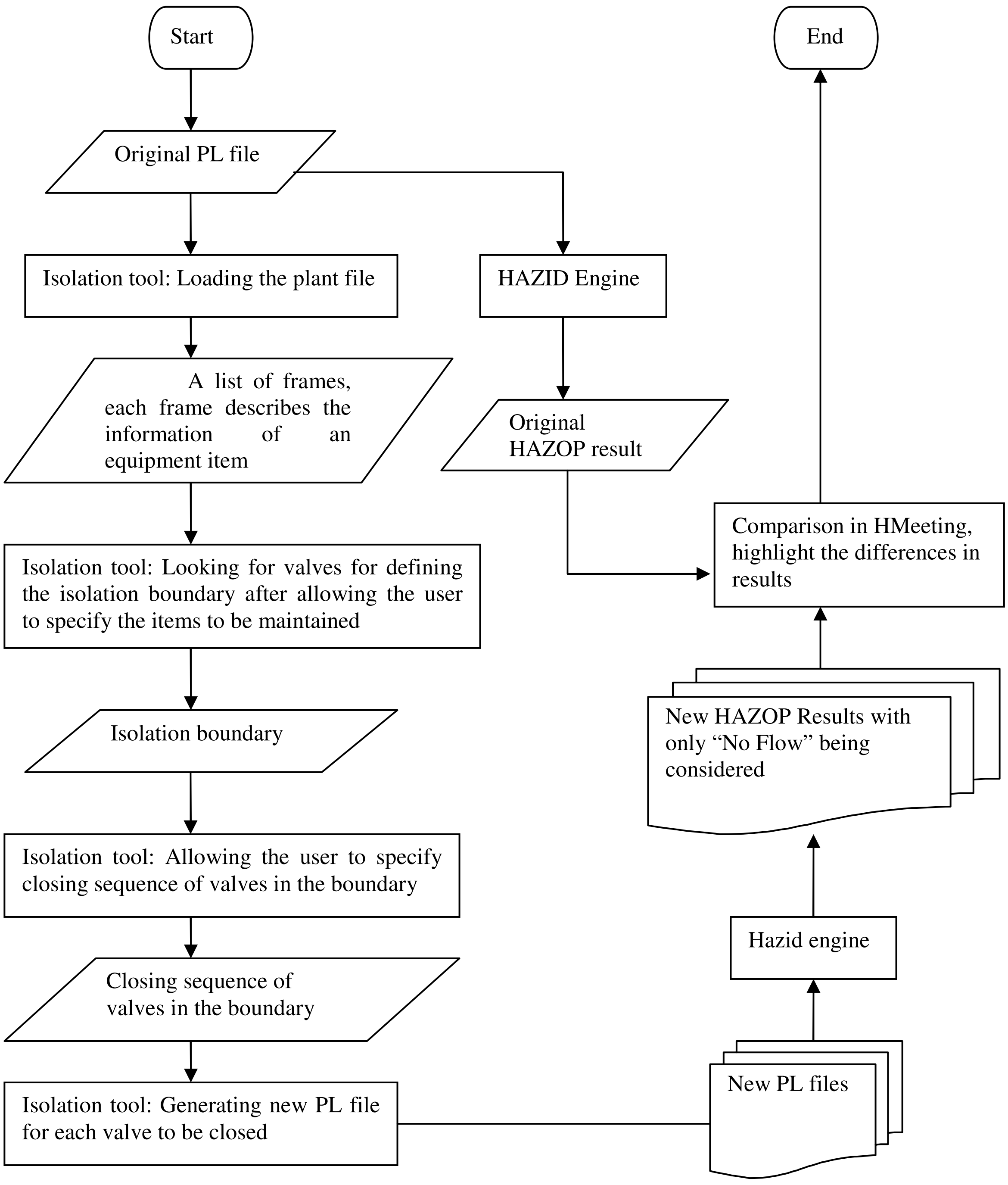
After the items on the boundary are identified, the user is asked to review the items and specify the order in which they should be closed. Consider the example shown in figure 1, if the first item selected to be closed is “V001” then a HAZOP analysis is carried out by applying the deviation 'no flow' to V001. This is done by the isolation tool by invoking HAZID to carry out an analysis to identify the hazards that might be introduced by this deviation. If hazards are identified then they are reported. If no hazard is identified then the tool will go on to consider the deviation 'no flow' for the second boundary item to be closed (e.g. “V002”) and with “V001” being closed. This process goes on until the closing of all the valves on the boundary have been considered. In this way, there will be many HAZOP results as the valves are being closed in sequence. The HAZOP analysis carried out in HAZID are produced quickly and the successive analysis occupy little user time. Table 1 illustrates this hazard identification process. Presentation of the results is shown in HMeeting, a tool to view HAZOP analysis results in HAZID. The presentation of the results will highlight the differences of the results, so that the engineer can focus on the hazards that might happen by closing a valve.

Conditions	**Actions**	**Result**
“no flow” passing through V001 (this should be the original HAZOP result with only “no flow” being considered)	Doing HAZOP analysis	HAZOP result 1
“no flow” passing through V002 + V001 closed	Doing HAZOP analysis	HAZOP result 2
“no flow” passing through V003 + V002 closed +V001 closed	Doing HAZOP analysis	HAZOP result 3
....	…………………..	………………….
“no flow” passing through VN + VN-1 closed + VN-2 closed …+V001 closed	Doing HAZOP analysis	HAZOP result N

Table 1. The hazard identification process

4. The Overall Workflow

Figure 4 shows the overall workflow of the isolation tool. It starts with loading the original plant file as input, and then a list of frames is generated in which each frame describes the information of an equipment item. After allowing the user to specify the item to be maintained, the tool will run its algorithm to look for the items including valves, OPCs without continuations, and PRVs in the isolation boundary and then display them. Then it will move onto the second stage which is to identify the hazards after the boundary is defined. It starts by allowing the user to specify the sequence for closing the valves on the boundary. Then it generates a series of input files for HAZID with each valve is closed in each stage of the sequence as described in table1. HAZID is then called to do a HAZOP analysis with only "no flow" being considered for each input file. The results are then automatically compared and any differences are highlighted to show any hazards related to closing any of the valves.



Start
End
Original PL file
Isolation tool: Loading the plant file
HAZID Engine
A list of frames, each frame describes the information of an equipment item
Original HAZOP result
Comparison in HMeeting, highlight the differences in results
Isolation tool: Looking for valves for defining the isolation boundary after allowing the user to specify the items to be maintained
New HAZOP Results with only "No Flow" being considered
Isolation boundary
Isolation tool: Allowing the user to specify closing sequence of valves in the boundary
Hazid engine
Closing sequence of valves in the boundary
New PL files
Isolation tool: Generating new PL file for each valve to be closed

Figure 4. Working flow of the Isolation Tool

5. Further development

Although the current tool is sufficient to carry out hazard identification before commencing the real maintenance tasks, further improvement is ongoing to enhance its usability and acceptability, for example, it is being considered to extend the current tool to do a maintenance analysis by keeping the history of maintenance activity, setting a time condition when multiple items are to be maintained, etc.

6. Conclusion

Carrying out maintenance safely is of paramount importance to avoid releasing hazardous material into the atmosphere and to prevent maintenance related accidents. A thorough hazard analysis ahead of the actual work is more likely to give a successful and safe maintenance implementation.

Although process plant P&IDs are already available electronically for process engineers to identify potential hazards, no computer-aided tool was used to help with defining an isolation boundary for the maintenance work and calling the hazard identification analysis automatically from the maintenance safety point of view.

This paper describes a novel computer-aided tool to help with safe isolation for maintenance work. In order to define the isolation boundary, the tool searches upstream and downstream of the item to be maintained to find the isolable valves which must be closed . After the boundary is defined, HAZOP analysis with deviation “no flow” is applied to identify hazards that may happen when the valves are closed in a specified order.

Acknowledgements

The research reported in this paper is supported by an EngD (Engineering Doctorate) studentship funded by the Engineering and Physical Sciences Research Council (EPSRC) and Hazid Technologies Ltd. The studentship is administrated by the CICE at Loughborough University.

References

Bartolozzi, V., Castiglione, L., Picciotto A., and Galluzzo, M., 2000. Qualitative models of equipment units and their use in automatic HAZOP analysis. *Reliability Engineering & System Safety,* 70(1), pp.49-57.

Crawley,F., Preston, M., and Tyler, B., 2000. *HAZOP: Guide to best practice _ guidelines to best practice for the process and chemical industries.* Rugby, Warwickshire, UK: Institution of Chemical Engineers (IChemE).

Hale, A.R., Heming, B.H.J., Smit, K., Rodenburg, F.G.T., and Van Leeuwen, N.D., 1998. Evaluating safety in the management of maintenance activities in the chemical process industry. *Safety Science,* 28(1), pp.21-44.

Hazid Technology Ltd., 2008. *Hazid Technology* [online], available from http://www.hazid.com [Accessed 10, May, 2008]

Kletz,T., 1999. *HAZOP AND HAZAN, Identify and assessing process industry hazards.* 4th ed. Institution of Chemical Engineers.

Lawley, H.G., 1974. Operability studies and hazard analysis. *Chemical Engineering Progress,* 70(4), pp.45–56.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Jefferson, M.L., 1999a. HAZID, A Computer Aid for Hazard Identification. 1. The STOPHAZ Package and The SPPS Code: An Overview, The Issues and The Structure. *Process Safety and Environmental Protection,* 77(B6), pp. 317-327.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Chung, P.W.H., 1999b. HAZID, a computer aid for hazard identification. 2. Unit model system. *Process Safety and Environmental Protection,* 77(B6), pp. 328-334.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Jefferson, M.L., 1999c. HAZID, a computer aid for hazard identification. 3. The fluid model and consequence evaluation systems. *Process Safety and Environmental Protection,* 77(B6), pp.335-353.

Rushton, A.G., Chung, P.W.H., and Lees, F.P., 1998. HAZID, A computer aid for hazard identification: An Overview. *Proceedings of the Loss Prevention and Safety Promotion, the Process Industries 9th international symposium*, 2, pp.503-512.

Vaidhyanathan, R., Venkatasubramanian, V., 1995. Digraph-based models for automated HAZOP analysis. *Reliability Engineering and System Safety,* 50, pp.33-49.

Wallace, S.J., Merritt, C.W., 2003. Know when to say "when": a review of safety incidents involving maintenance issues. *Process Safety Progress,* 22(4), pp.212-219.

Wells, G. L., Seagrave, C. J., 1976. *Flowsheeting for Safety - A Guide for Chemical Engineers on Safety Measures to Consider during the Design of Chemical Plant.* Institution of Chemical Engineers.

APPENDIX C PAPER 3

Automated Cause-Effect Analysis for Process Plants

An, H., Chung, P.W.H., McDonald, J., and Madden, J., 2009. Automated Cause-Effect Analysis for Process Plants. *Proceedings of the ninth International Conference on Chemical & Process Engineering (ICheaP-9)*. Rome, Italy, 10-13 May 2009, pp.1281-1287.

This paper was then selected to publish with its full-length in Italian Association of Chemical Engineering (AIDIC) Conference Series Vol.9, Reed Business International. The publication of the Volume is expected by the end of January 2010.

Abstract

Cause-effect analysis for process plants is one of the tasks associated with Process Control Engineering (PCE). With the availability of electronic Piping and Instrumentation Diagrams (P&IDs), a computer-aided tool is developed to carry out the analysis automatically by encoding knowledge related to PCE in rules so that they can be applied to a given set of P&IDs to produce the corresponding cause-effect diagrams. This paper describes how this is achieved. A rule-based system and an instrument checker are developed. They are used to generate the results and the results are displayed in a format that complies with ISO 10418 (ISO, 2003).

1. Introduction

Safety Analysis Function Evaluation chart (SAFE), or cause-effect table, is one of the established cause and effect analysis techniques stated in ISO 10418 (ISO, 2003) that can be applied to identify unfavourable safety-related outcome and the design of protective measures. A computer-aided tool that can produce the cause-effect analysis result automatically is developed and integrated with Intergraph's Engineering Enterprise Suite through Smart Plant Process Safety (SPPS). SPPS is a knowledge-based system that automates the process of HAZard and Operability study (HAZOP). It is developed by Hazid Technologies Ltd , UK. The cause-effect tool consists of an Instrument Checker, a general purpose knowledge-based rule engine and a tool that outputs the results for displaying using Microsoft Excel. The layout of the table complies with ISO 10418 (ISO, 2003).

This paper describes the above components of the system. An example is used to illustrate the working of the system and a comparison between results of this tool and that of the tool described by Drath et al. (2006) is given.

2. Instrument Checker

Given a P&ID, the Instrument Checker is a tool that identifies the instrument loops and their connections with the process items. The output of this tool is used as input to the rule-based system.

The tool first identifies all the instruments in the process plant. For each instrument, it traces the upstream and downstream connections of each branch line until a process item is found. Given a process item, a list of instruments that are connected to it is kept. Similarly, given an instrument, a list of process items that are connected to it is kept.

Consider the P&ID shown in figure 1, which is a very small part taken from a much larger plant. The following instruments are identified:

- two high level alarms – "ZEH-59010" and "ZLH-59010";

- two low level alarms – "ZEL-59010" and "ZLL-59010";
- one control valve – "FCV-59010".

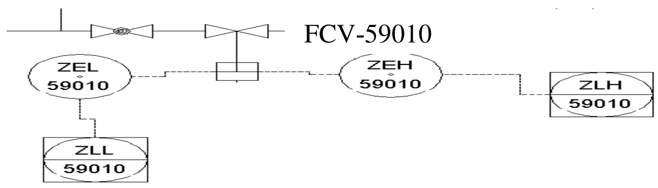
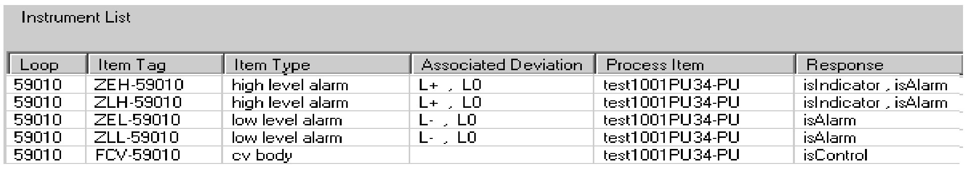


Figure 1. A simple instrument loop

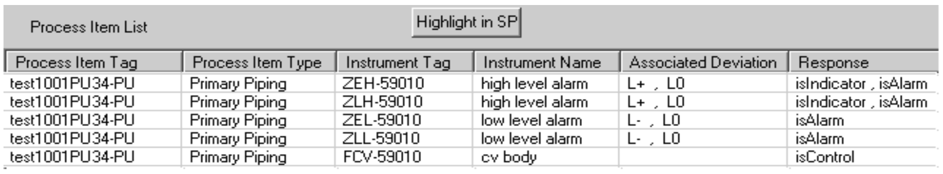
Figure 2 shows the instruments, their connections with process items, related deviations and responses. Figure 3 shows all the instruments in the loop are attached to the same process item "test1001PU34-PU".



Instrument List

Loop	Item Tag	Item Type	Associated Deviation	Process Item	Response
59010	ZEH-59010	high level alarm	L+ , L0	test1001PU34-PU	isIndicator , isAlarm
59010	ZLH-59010	high level alarm	L+ , L0	test1001PU34-PU	isIndicator , isAlarm
59010	ZEL-59010	low level alarm	L- , L0	test1001PU34-PU	isAlarm
59010	ZLL-59010	low level alarm	L- , L0	test1001PU34-PU	isAlarm
59010	FCV-59010	cv body		test1001PU34-PU	isControl

Figure 2. Instrument List



Process Item List

Highlight in SP

Process Item Tag	Process Item Type	Instrument Tag	Instrument Name	Associated Deviation	Response
test1001PU34-PU	Primary Piping	ZEH-59010	high level alarm	L+ , L0	isIndicator , isAlarm
test1001PU34-PU	Primary Piping	ZLH-59010	high level alarm	L+ , L0	isIndicator , isAlarm
test1001PU34-PU	Primary Piping	ZEL-59010	low level alarm	L- , L0	isAlarm
test1001PU34-PU	Primary Piping	ZLL-59010	low level alarm	L- , L0	isAlarm
test1001PU34-PU	Primary Piping	FCV-59010	cv body		isControl

Figure 3. Process Item List

3. Rule-based System

A rule-based system is built to analyse process events and the corresponding process responses. CLIPS (C Language Integrated Production System) is chosen as the development tool as it supports rule-based, object-oriented and procedure programming methods (Riley, 2008). A rule-based system in CLIPS consists of three components: a set of facts, a set of rules and the inference engine that controls the overall execution by matching the rules against the facts to infer new information (Giarratano & Riley, 1994).

3.1 Facts about process items, instruments and connectivity

The output from the Instrument Check is converted into CLIPS facts as input for the rule-based system. Here are some example facts:

(equipment pipe test1001PU34-PU 1-in-2-out)
(device high-level-alarm ZEH-59010)
(flow-connection FCV-59010 out test1001PU34-PU)
(signal-connection ZEH-59010 FCV-59010)

The first fact states that “test1001PU34-PU” is a “1-in-2-out” pipe of the class “equipment”. The second fact states that “ZEH-59010” is a high level alarm of the class “device”. The third fact states that the out flow of “FCV-59010” is connected to “test1001PU34-PU”. The fourth fact states that there is a signal connection between the high level alarm “ZEH-59010” and the control valve “FCV-59010”.

3.2 The reasoning rules

The reasoning rules are extracted from ISO 10418 (ISO, 2003). The following is an example rule and the rule is coded in CLIPS format in the system.

IF
there is a level indicator or a high level alarm
AND
there is a vessel
AND
there is a control device which is either a control valve or a control pump
AND
the control device is connected to the vessel
AND
there is a signal connection between the level indicator or alarm to the control device or the vessel
THEN
conclude that the control device will be triggered when the level of the vessel reaches a pre-defined high level.

3.3 The reasoning process

A rule is activated when all the conditions specified are satisfied by the facts contained in the system. When a rule is fired the action(s) specified will be taken. Normally the action is to call a function to write some output in the result file in XML format. Part of the output is shown in figure 4.

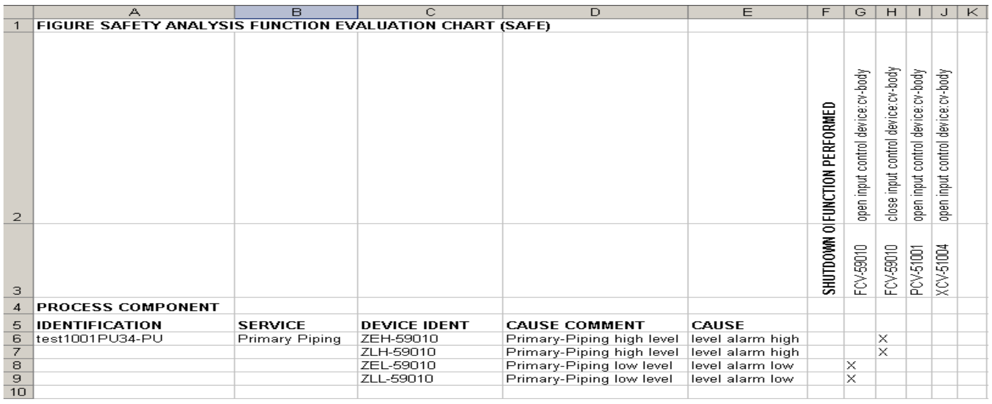
```
<cause_effect>
<cause_comment processItemTag='test1001PU34-PU'>Primary-Piping high level</cause_comment>
<cause instrumentTag='ZEH-59010'>level alarm high</cause>
<effect controlInstrumentTag='FCV-59010'>close input control device></effect>
</cause_effect>
```

Figure 4 Output in XML format.

4. Displaying the Cause-Effect Result table

After the rule-based system in CLIPS engine has generated the results in the XML format a parser is called to parse the XML result and convert it into a Comma-Separated Values (CSV) text file. An engineer can open the CSV file with Excel, and the cause-effect table will be presented in the format specified in ISO 10418 (ISO, 2003).

Part of the Cause-Effect table is shown in figure 5. A cross is placed in a cell to indicate the cause and effect link between a process component, sensor instrument and control device. Figure 5 shows that process component “test1001PU34-PU” has four instrument devices attached to it (“ZEH-59010”, “ZLH-59010”, “ZEL-59010”, “ZLL-59010”). If the high level alarm “ZEH-59010” or “ZLH-59010” goes off then the input control valve “FCV-59010” will be closed. If the low level alarm “ZEL-59010” or “ZLL-59010” goes off then the input control valve “FCV-59010” will be open.



	A	B	C	D	E	F	G	H	I	J	K
1	FIGURE SAFETY ANALYSIS FUNCTION EVALUATION CHART (SAFE)										
2						SHUTDOWN OF FUNCTION PERFORMED	open input control device:cv-body	close input control device:cv-body	open input control device:cv-body	open input control device:cv-body	
3							FCV-59010	FCV-59010	PCV-51001	XCV-51004	
4	PROCESS COMPONENT										
5	IDENTIFICATION	SERVICE	DEVICE IDENT	CAUSE COMMENT	CAUSE						
6	test1001PU34-PU	Primary Piping	ZEH-59010	Primary-Piping high level	level alarm high			X			
7			ZLH-59010	Primary-Piping high level	level alarm high			X			
8			ZEL-59010	Primary-Piping low level	level alarm low		X				
9			ZLL-59010	Primary-Piping low level	level alarm low		X				
10											

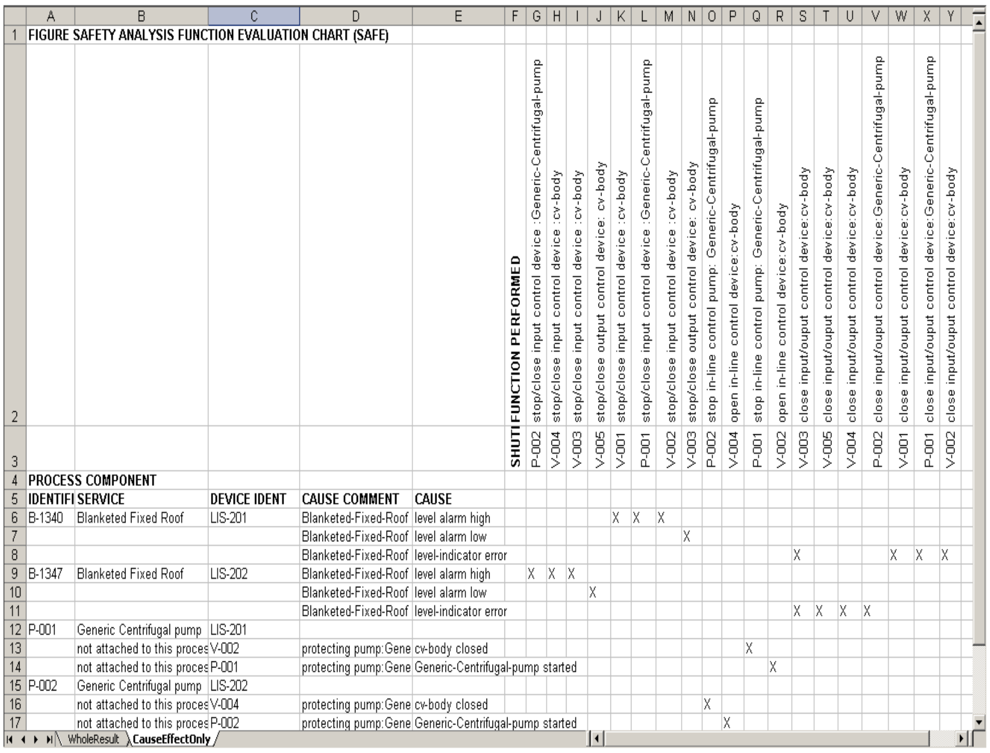
Figure 5 Part of the result table

5. Comparison with related work

The knowledge-based system reported in Drath et al. (2006) is also designed and implemented to automate the generation of Cause-Effect Table. Their paper uses an interlock example to illustrate their work. In order to carry out a comparison, the authors have produced and extended a P&ID from their example and applied the cause effect analysis tool reported here to that plant. The result is shown in figure 6.

Comparing with the result reported in Drath et al. (2006), the result produced by our system has two additional features:

- The result table provides a comprehensive list of process components and their attached devices. The user also has the option of viewing only components that have cause and effect links that apply to them.
- The table provides more detailed classification of function performed.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	FIGURE SAFETY ANALYSIS FUNCTION EVALUATION CHART (SAFE)																								
2						SHUTI FUNCTION PERFORMED	stop/close input control device :Generic-Centrifugal-pump	stop/close input control device :cv-body	stop/close input control device :cv-body	stop/close output control device: cv-body	stop/close input control device :cv-body	stop/close input control device :Generic-Centrifugal-pump	stop/close input control device :cv-body	stop/close output control device: cv-body	stop in-line control pump: Generic-Centrifugal-pump	open in-line control device:cv-body	stop in-line control pump: Generic-Centrifugal-pump	open in-line control device:cv-body	close input/ouput control device:cv-body	close input/ouput control device:cv-body	close input/ouput control device:cv-body	close input/ouput control device:Generic-Centrifugal-pump	close input/ouput control device:cv-body	close input/ouput control device:Generic-Centrifugal-pump	close input/ouput control device:cv-body
3							P-002	V-004	V-003	V-005	V-001	P-001	V-002	V-003	P-002	V-004	P-001	V-002	V-003	V-005	V-004	P-002	V-001	P-001	V-002
4	PROCESS COMPONENT																								
5	IDENTIFI	SERVICE	DEVICE IDENT	CAUSE COMMENT	CAUSE																				
6	B-1340	Blanketed Fixed Roof	LIS-201	Blanketed-Fixed-Roof	level alarm high						X	X	X												
7				Blanketed-Fixed-Roof	level alarm low									X											
8				Blanketed-Fixed-Roof	level-indicator error														X				X	X	X
9	B-1347	Blanketed Fixed Roof	LIS-202	Blanketed-Fixed-Roof	level alarm high		X	X	X																
10				Blanketed-Fixed-Roof	level alarm low					X															
11				Blanketed-Fixed-Roof	level-indicator error														X	X	X	X			
12	P-001	Generic Centrifugal pump	LIS-201																						
13		not attached to this proces	V-002	protecting pump:Gene	cv-body closed												X								
14		not attached to this proces	P-001	protecting pump:Gene	Generic-Centrifugal-pump started													X							
15	P-002	Generic Centrifugal pump	LIS-202																						
16		not attached to this proces	V-004	protecting pump:Gene	cv-body closed										X										
17		not attached to this proces	P-002	protecting pump:Gene	Generic-Centrifugal-pump started											X									

Figure 6 Cause-Effect Table in Excel for an extended P&ID based on Drath et al. (2006)

6. Conclusion

Carrying out safety analysis is important to prevent accidents and help in the design of control and protective systems for process plants. Control and sensor devices and their related control actions can be presented in a SAFT chart to help with the analysis process. A SAFT chart is also called a cause-effect table as it reflects information about process events and their corresponding safe guards.

An automated cause-effect analysis system is introduced in this paper. Its components, working principles and data processing methods are described. The system consists of an Instrument Checker which prepares the data for analysis by identifying all the instruments and their attached process items. A general purpose rule engine is used to build the knowledge-based system. The output from the rule engine is converted into cause-effect table in Excel and the layout of the table is compatible with ISO 10418 (ISO, 2003).

Acknowledgements

The research reported in this paper is supported by an EngD (Engineering Doctorate) studentship funded by the Engineering and Physical Sciences Research Council (EPSRC) and Hazid Technologies Ltd. The studentship is administrated by the CICE at Loughborough University.

Reference

Drath, R., Fay, A., and Schmidberger, T., 2006. Computer-aided Design and Implementation of Interlock Control Code. *Computer-Aided Control Systems Design, 2006 IEEE International Symposium*. Ladenburg, Germany, 4-6 Oct, 2006, pp.2653-2658.

Giarratano, J., Riley, G., 1994. *Expert Systems, Principles and Programming*. 2nd ed. Boston: PWS Publishing Company.

Hazid Technology .LTD, 2007, *Hazid Technology*. [online] Nottingham, UK. Available at http://www.hazid.com [Accessed 10, Sep, 2008]

Riley, G., 2008, *CLIPS, A tool for building expert systems*. [online] USA: SourceForge. Available at http://clipsrules.sourceforge.net/WhatIsCLIPS.html [Accessed 15,Sep,2008]

ISO (International Organization for Standardization), 2003. *ISO 10418 Petroleum and natural gas industries — Offshore production installations — Basic surface process safety systems*. Switzerland: HIS.

APPENDIX D PAPER 4

Computer-aided identification of isolation boundary for safe maintenance and Cause-Effect analysis for assessing safeguards

An, H., Chung, P.W.H., McDonald, J., and Madden, J., 2009. Computer-aided identification of isolation boundary for safe maintenance and Cause-Effect analysis for assessing safeguards. *Int. J. Process Systems Engineering*, Vol. 1, No. 1, pp.29–45.

Abstract: Systematic hazard identification is required before maintenance work commences as unsafe maintenance in process plant can cause release of dangerous materials, pipe-work failure and deviations from normal operations, etc (Hale, et al., 1998). Cause and effect analysis is also required to identify unfavourable safety-related outcome and for the design of protective measures.

With the availability of electronic Piping and Instrumentation Diagrams (P&IDs), two computer-aided tools are developed as part of the Smart Plant Process Safety (SPPS) system. One is to help the task of identifying hazards related to maintenance work and the other is to carry out cause and effect analysis automatically.

This paper highlights the main functions of these two tools and describes how they are developed. It also illustrates how the cause effect analysis tool can be used to support the hazard identification before carrying out the maintenance work.

Keywords: Process systems engineering; process control engineering; maintenance safety; safe isolation; cause and effect analysis; safety analysis function evaluation chart; hazard identification; hazard and operability study.

Biographical notes: Hong An is a full-time Engineering Doctorate (EngD) student at Loughborough University, UK. Her research investigates the development of computer-aided safety applications and how they can be integrated together. This research is funded by an Overseas Research Student Scholarship (ORS), an EngD studentship from Engineering and Physical Sciences Research Council (EPSRC) and Hazid Technologies Ltd.

Paul Chung is a Professor of Computer Science and Director of the Research School of Informatics at Loughborough University. One of his main research areas is computer-aided safety engineering and he has published extensively in the area.

Joe McDonald is an Applications Consultant for Hazid Technologies and Industrial Supervisor for Hong An. His main focus of work is in developing novel process safety software that is relevant to the industry.

Jim Madden spent many years in plant engineering with major UK companies, holding technical and management positions in plant design, construction, maintenance and plant management. He was a pioneer developer of computerised pipe drafting before moving on to pioneer the 3-D plant design program, PDMS. After leading development of automated 3-D plant layout, he then pioneered the development and successful commercialisation of the first integrated process design tool, Zyqad. His current main activity is with Hazid, an innovative tool for identifying process hazards and making the hazard information the basis for integration of process safety information and management.

1. Introduction

The safety of hazardous processing plants is of paramount importance. An accident could cause major damage to properties and/or injury to people. Well-maintained equipment in the process plant can give smooth running of the plant and increase the plant productivity and lifetime. However, the maintenance work of a process plant can be dangerous if proper isolation of the equipment items being maintained is not carried out beforehand. Therefore, a comprehensive identification of potential hazards is necessary before carrying out the actual maintenance work.

Safety analysis of the control design of a process plant is another hazard identification application. Safety Analysis Function Evaluation chart (SAFE) is one of the established cause and effect analysis techniques stated in ISO 10418 (ISO, 2003). The SAFE chart is also referred to as cause-effect table as it provides information about process events and control responses.

Manual generation of a cause-effect table of a process plant is labour intensive, time consuming, repetitive and error-prone. With P&IDs available in electronic format there is the potential of developing a computer-aided tool that can analyse the P&ID information and produce the cause-effect table automatically.

Two novel computer-aided tools are developed as a joint project between Hazid Technologies Ltd. and Loughborough University. The first serves two main functions related to isolation for maintenance: identifies the isolation boundary, and identifies the potential hazards related to the isolation task. The other is a cause-effect analysis tool that automates the generation of cause-effect tables. Both of these tools are integrated with Intergraph's Engineering Enterprise Suite through the SmartPlant Process Safety (SPPS) system (Hazid Technology. Ltd, 2008). SPPS is a knowledge-based system that automates the process of HAZOP study. The development of SPPS is given in Rushton et al. (1998) and McCoy et al. (1999a, 1999b, 1999c).

This paper describes these two tools in detail. Case studies are also used to illustrate the working of these systems. Finally, it describes how these two tools can be used in conjunction to help with the identification of hazards for the item(s) to be maintained.

2. A Computer Tool to Support Safe Isolation for Maintenance

It is reported that 30% of accidents are maintenance-related and 50% of them release harmful substances (Wallace & Merritt, 2003). A thorough hazard analysis ahead of the actual work is more likely to give a successful and safe maintenance implementation.

Although process plant P&IDs are already available electronically for process engineers, little attention has been given to the development of computer-aided tool to help with defining an isolation boundary for maintenance work and identifying automatically hazards related to the isolation and maintenance tasks.

A novel computer-aided tool that considers the safety issues for maintenance work is described in this section. Section 2.1 and 2.2 describes the methodology for identifying the isolation boundary and demonstrates how the algorithm is tested by applying two case studies. Section 2.3 illustrates how HAZOP analysis is applied to identify hazards after the boundary is identified and selected. The details of HAZOP study technique can be found in Crawley et al. (2000) and in Kletz (1999). Section 2.4 gives the overall workflow of the tool.

2.1 Identify the Isolation Boundary

When equipment items are to be maintained in a process plant, a process engineer has to analyse the P&ID of that plant, and identify the valves that must be closed in order to isolate the equipment items that are to be maintained. Consider figure1, if pump “P-0101A” is to be maintained then 5 valves, “V015”, “V002”, “V014”, “V013” and “V001”, will need to be closed to isolate the pump.

The plant file, as mentioned before, is a text file generated automatically from the P&ID. It describes the equipment items in the plant, which includes their connections, flow directions and other attributes. It is used as input into the isolation tool. The procedure for identifying an isolation boundary is to trace upstream and downstream from the equipment items to be maintained to find the valves that must be closed. During tracing, the tool will only look for the first valve that is isolable in each line branching out from the equipment items.

When a Pressure Relief Valve (PRV) is met, the algorithm will compare the direction of the trace and the opening direction of the PRV, if they are in the same direction, then the tracing procedure will carry on and the PRV will be identified as within the boundary but must be isolated, otherwise, the PRV will be identified as on the boundary and the tracing procedure at this branching line ends at this point.

When searching from the equipment items to be maintained, if the propagation passes through a sheet connector onto another P&ID then it will continue on the next P&ID until an isolable valve is found. If there is no continuation from the sheet connector, the sheet connector will be treated as being on the boundary and the user will be warned that an isolation point has not been found for that particular branch. The sheet connector ID such as tag number will be identified in the warning.

When multiple items are to be maintained, the same procedure is applied to all the items. The valves that need to be isolated for all the items are combined together and any duplicates removed.

2.2 Case Studies

Two plants, a hydrocarbon separation unit (Lawley, 1974) and a benzene plant (Wells & Seagrave, 1976), have been used to test the system. When given different maintenance items as input for the two plants the tool correctly identifies the isolation boundaries.

Figure 1 shows the output for isolating pump “P-0101A” for the hydrocarbon separation unit. All the valves that need to be closed are highlighted in red. The item to be maintained, “P-0101A”, is highlighted in bright blue.

Another example is to maintain the liquid-liquid-gas separator in a hydrocarbon separation plant. Figure 2 shows a table of equipment items on the boundary with their current states and related notes. The tool also displays a table with all the items within the boundary. Figure 3 shows these items highlighted in Smart Plant P&ID. This example shows that several different types of equipment items are on the boundary. The first type is the “must be closed” valve, such as “V028”, “V025”, “V021”, “V022”, “V007”, which are highlighted in Smart Plant P&ID drawing in red. The second type is Pressure Relief Valve (PRV), such as “PRV002PRV”, “PRV001PRV”. The third type is the Off-Page -Connector (OPC). The second and third types are highlighted in Smart Plant P&ID drawing in purple. The items to be maintained are highlighted in bright blue.

2.3 Identifying Hazard Associated with Isolating the Identified Boundary

Once the boundary is identified, a HAZOP analysis has to be carried out to identify the hazards related to the isolation procedure. As mentioned, SPPS, a knowledge-based system which automates the process of HAZOP studies, is used for this purpose. The SPPS system is developed by Hazid Technologies Ltd. and is currently the most advanced commercially available knowledge-based HAZOP tool. (Hazid Technology. Ltd, 2008).

After the items on the boundary are identified, the user is asked to review the items and specify the order in which they should be closed. Consider the example shown in figure 1, if the first item selected to be closed is “V001” then a HAZOP analysis is carried out by SPPS by applying the deviation 'no flow' to V001, i.e. the effect of closing the valve. Any hazards identified are reported. If no hazard is identified then the tool will go on to consider the deviation 'no flow' for the next boundary item to be closed, with “V001” being closed. This process goes on until the closing of all the valves on the boundary have been considered. Table 1 illustrates this hazard identification process. The HAZOP analysis carried out by SPPS are very quick and the user does not have to wait long for the results. The presentation of the results will highlight the differences after each valve has been closed so that the engineer can focus on the hazards that might occur due to the closing of the valve.

2.4 The Overall Workflow

Figure 4 shows the overall workflow of the Isolation Tool. It starts with loading the original plant file as input. After allowing the user to specify the item(s) to be maintained, the tool will run its algorithm to look for the items – including valves, OPCs without continuations, and PRVs – on the isolation boundary and then display them. The second stage is to identify the hazards after the boundary is defined. It allows the user to specify the sequence for closing the valves on the boundary. The tool then generates a series of input files for SPPS with each valve is closed in each step of the sequence. SPPS is then called to do a HAZOP analysis with only “no flow” being considered for each input file. The results are then automatically compared and any differences are highlighted to show any hazards related to closing any of the valves.

3. Automated Cause-Effect Analysis

Control and sensor devices and their protective functions can be presented in a SAFE chart for the purpose of safety analysis. SAFT chart is also called cause-effect Table.

There are existing commercial tools that provide interfaces for filling in cause-effect tables, but they do not provide any knowledge-base help. With P&IDs available electronically, automatic generation of cause-effect tables is being investigated so as to ease the labour-intensive analysis process. Drath et al. (2006) describes a research prototype that generates cause-effect tables automatically by applying a rule-base to plants that are described in a standard format called **C**omputer **A**ided **E**ngineering e**X**change (CAEX).

This section gives details of another automatic cause-effect analysis system which takes the same input as the isolation tool described in the previous section. Section 3.1, 3.2 and 3.3 describe the components and the reasoning process of the cause-effect system in detail. Two case studies are used to illustrate the working of the system. The first is a very small part of a P&ID of a much larger plant just to illustrate the working of the system. The second is the interlock system described in Drath et al. (2006). Section 3.4 shows the result table. Section 3.5 compares the system and results described by Drath et al. (2006) with the current system.

3.1 Components of the Cause-Effect System

The cause-effect system consists of an Instrument Checker, a general purpose knowledge-based rule engine and an output tool that generates the cause-effect table that can be easily displayed in Microsoft Excel. Output from Instrument Checker is converted into input for the rule engine. Output from the rule engine is converted into cause-effect tables in a format complies with 10418 (ISO 2003). The working flow of the overall cause-effect system is shown in figure 5.

3.2 Instrument Checker

Given a P&ID, the Instrument Checker is a tool that identifies the instrument loops and their connections with the process items.

The tool first identifies all the instruments in the process plant. For each instrument, it traces the upstream and downstream connections of each branch line until a process item is found. Therefore, information about which process items are connected to which instruments is collected. Given a process item the Instrument Checker lists which instruments are connected to it. On the other hand, given an instrument it will also list which process items are connected it.

Consider the P&ID shown in figure 6, which is a small part taken from a much larger plant. The tool identifies the following instruments:

- two high level alarms – ZEH-59010 and ZLH-59010;
- two low level alarms – ZEL-59010 and ZLL-59010;
- one control valve – FCV-59010.

The tool then traces upstream and downstream to find the process item(s) that are attached to these instruments. The result is shown in figure 7. In this case they are all connected to the same pipe with tag “test1001PU34-PU”. The loop number “59010” indicates that they are all in the same instrument loop. The responses to the deviations are also shown – “L+” means “high level”, “L-” means “low level", "L0" means "no level". Figure 8 shows process item "test1001PU34-PU" is connected to instruments "ZEH-59010", "ZLH-59010", "ZEL-59010", and "ZLL-59010".

3.3 Rule-Based Expert System

A rule-based system, which captures expert knowledge, is built and used to analyse the process events and the corresponding process responses. CLIPS (C Language Integrated Production System) is chosen as the development tool. It is a powerful hybrid tool that supports rule-based, object-oriented and procedural programming (Giarratano & Riley 1994; Riley, 2008). A rule-based system in CLIPS consists of three basic components: a set of facts, a set of rules and the CLIPS inference engine that controls the overall execution by matching the rules against the facts to infer new information.

3.3.1 Converting information about process and instrumentation items into CLIPS facts

As the inference engine requires the information about the process and instrumentation items to be in CLIPS format, the tool takes the output of Instrument Checker and converts it to the required format. For example, ZEH-59010 is a high level alarm and is represented as:

(device high-level-alarm ZEH-59010)

The out flow of pipe test1001PU34-PU is connected to test1001PU34-P_4 and is represented as:

(flow-connection test1001PU34-PU out test1001PU34-P_4)

If there is a flow connection from "A" to "B", and there is a flow connection from "B" to "C", then the system will automatically infer that "A" and "C" are in the same line and a new fact like the following is asserted into the system:

(in-line A to C)

Signal connections are treated in a similar way.

3.3.2 Developing the reasoning rules

The reasoning rules are extracted from ISO 10418 (ISO, 2003). The following is an example rule.

IF
there is a level indicator or a high level alarm
AND
there is a vessel
AND
there is a control valve or a control pump
AND
the control device is connected to the vessel
AND
there is a signal connection between the level indicator or alarm to the control device and the vessel
THEN
conclude that the control device will be triggered when the level of the vessel reaches a high level.

The rule in CLIPS format is shown below.

```
(defrule levelVessel-highLevel-close-inputValve
  (device level-indicator | high-level-alarm ?HLA)
  (equipment vessel  ?VESSEL-TAG ?VESSEL-NAME)
  (equipment controlDevice | controlDevicePump ?INPUT-CONTROL-DEVICE-TAG ?INPUT-CONTROL-
DEVICE-NAME)
  (in-line ?INPUT-CONTROL-DEVICE-TAG to ?VESSEL-TAG)
  (or(signal-connection ?HLA ?VESSEL-TAG)
    (signal-connection ?HLA ?INPUT-CONTROL-DEVICE-TAG))
=>
(print-result-levelAlarmHigh    ?VESSEL-TAG    ?VESSEL-NAME    ?HLA    ?INPUT-CONTROL-DEVICE-
TAG ?INPUT-CONTROL-DEVICE-NAME))
```

3.3.3 The reasoning process

A rule is activated when all the conditions specified are satisfied by the facts contained in the system. When a rule is fired the action(s) specified will be taken. Normally the action is to call a function to write some output in the result file in XML format. Part of the output is shown below.

```
<cause_effect>
<cause_comment processItemTag='test1001PU34-PU'>Primary-Piping high level</cause_comment>
<cause instrumentTag='ZEH-59010'>level alarm high</cause>
<effect controlInstrumentTag='FCV-59010'>close input control device></effect>
</cause_effect>
```

3.4 Generating the Cause-Effect Result Table

After the rule-based system in CLIPS has generated the results in the XML format, a parser is called to parse the XML result and convert it into a Comma-Separated Values (CSV) text file. An engineer can open the CSV file with Excel, and the cause-effect table will be presented in the format specified in ISO 10418 (ISO, 2003).

Part of the cause-effect table is shown in figure 9. A cross is placed in a cell to indicate the cause and effect link between a process component, sensor instrument and control device. Figure 9 shows that process component “test1001PU34-PU” has four instrument devices attached to it – “ZEH-59010”, “ZLH-59010”, “ZEL-59010” and “ZLL-59010”. If the high level alarm “ZEH-59010” or “ZLH-59010” goes off then the input control valve “FCV-59010” will be closed. If the low level alarm “ZEL-59010” or “ZLL-59010” goes off then the input control valve “FCV-59010” will be opened.

3.5 Comparison with Related Work

The knowledge-based system reported in Drath et al. (2006) is also designed and implemented to automate the generation of Cause-Effect Table. Their paper uses an interlock example to illustrate their work. In order to carry out a comparison, the authors have produced and extended a P&ID from their example and applied the cause effect analysis tool reported here to that example.

The P&ID is shown in figure 10. “B-1340” and “B-1347” are two level vessels called “Blanketed Fixed Roof”; “V-001” and “V-002” are two input control valves for “B-1340”; “V-003” is an output control valve for “B-1340” and an input control valve for “B-1347”; “V-004” is an input control valve for “B-1347”; “V-005” is an output control valve for “B-1347”. “LIS-201” and “LIS-202” are level sensors for “B-1340” and “B-1347” respectively.

Cause and effect analysis rules that are applicable to this plant can be simply stated:

IF

the level in a vessel has reached its maximum

THEN

close its input device.

IF

the level in a vessel has reached its minimum

THEN

close its output device.

IF

a valve is closed

THEN

stop the in-line control pump to protect the pump.

IF

a pump is started

THEN

ensure the in-line valve is opened to protect the pump.

IF

an error occurred with a level sensor for a vessel

THEN

close the input and output control valves for that vessel.

The result of the analysis is shown in figure 11. Comparing with the result reported in Drath et al. (2006), the result reported here has two additional features:

- The result table provides a comprehensive list of process components and their attached devices. The user has the option of viewing only components that have cause and effect links that apply to them.
- The table provides more detailed classification of function performed. For example, “V-003” is an output control valve for “B-1340” and an input control valve for “B-1347”. Therefore, any cause-effect control that applies to “V-003” is appropriately linked to both vessels.

4. Applying cause and effect analysis to the item to be maintained

The above sections have introduced the isolation for maintenance and cause-effect analysis tools in some details. This section shows how the two tools can be used together for the purpose of safe maintenance.

Take the sample plant shown in figure 6, the aim is to maintain the pipe “test1001PU34-PU”. The isolation tool is used to identify the isolation boundary. Figure 12 and 13 show the result. Figure 12 highlights 9 valves in red color that must be closed in order to maintain this pipe. Figure 13 shows the items on the boundary including the OPCs without continuation and items within the isolation boundary.

Figure 14 shows the cause effect analysis for maintaining pipe test1001PU34-PU. In this analysis table, “test1001PU34-PU_24”, “test1001PU34-PU_31”, “test1001PU34-PU_32”, “test1001PU34-PU_5” and “test1001PU34-PU_7” are different parts of pipe “test1001PU34-PU”. If isolating the pipe for maintenance causes any of the following alarms to go off then the control valve “FCV-59010” must be open: “ZEL-59010”, “ZLL-59010”, “FQ-59010”, “FAL-59010”, “FI-59010”, “PI-59010” or “PAL-59010”. On the other hand if the high level alarm “ZEH-59010” or “ZLH-59010” goes off then the input control valve “FCV-59010” must be closed.

5. Conclusion

Identification of potential hazards that might be caused by maintenance work is a very important task. Given a P&ID, engineers have to define the isolation boundary manually. The computer-aided tool described in this paper simplifies this task. When the item to be maintained is specified, the tool automatically traces upstream and downstream to find the isolable valves and highlight them in SmartPlant P&ID. After the isolation boundary is defined, the task of hazard identification is done by carrying out a sequence of HAZOP analysis using SPPS to consider the consequences of closing the valves.

The casue and effect analysis tool incorporates an Instrument Checker and a rule-based system. Given a set of P&IDs the tool can carry out the analysis automatically and produce the corresponding cause-effect diagrams. There is a potential to expand the reasoning rules to carry out configuration checks for process plants.

Both the isolation tool and cause effect analysis tool make use of electronic P&IDs that are increasingly being made available by CAD systems thus extending intelligent CAD applications in the area of process safety.

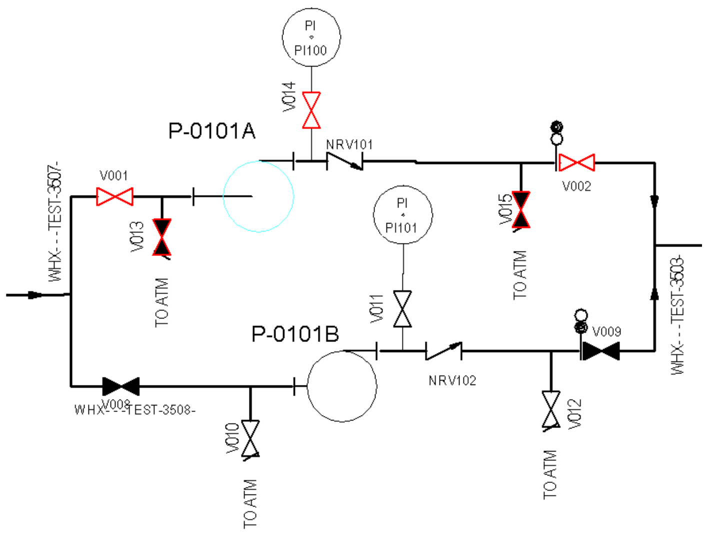
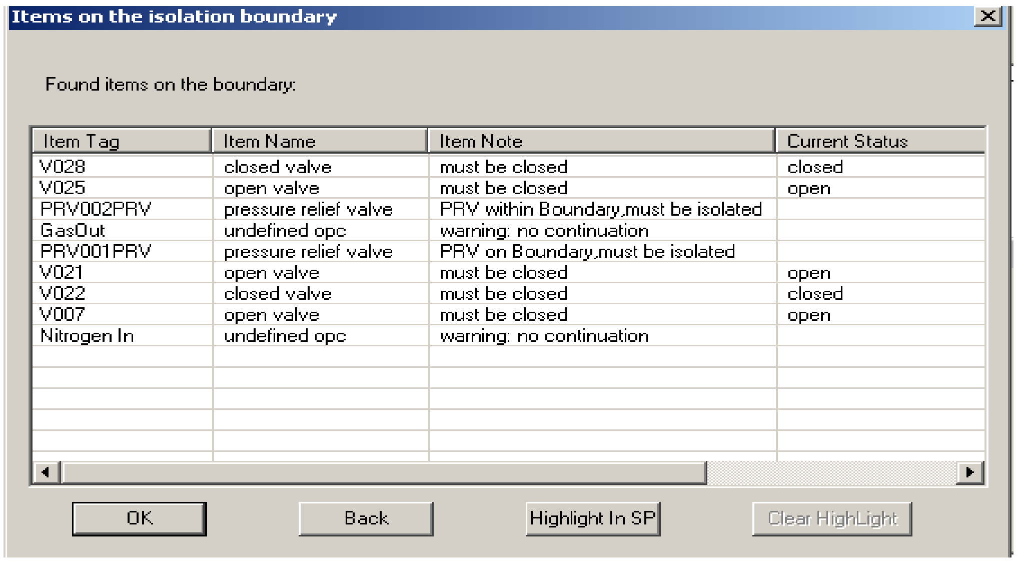


Figure 1. The highlighted isolation boundary for maintaining centrifugal pump “P0101A”


Items on the isolation boundary

Found items on the boundary:

Item Tag	Item Name	Item Note	Current Status
V028	closed valve	must be closed	closed
V025	open valve	must be closed	open
PRV002PRV	pressure relief valve	PRV within Boundary,must be isolated	
GasOut	undefined opc	warning: no continuation	
PRV001PRV	pressure relief valve	PRV on Boundary,must be isolated	
V021	open valve	must be closed	open
V022	closed valve	must be closed	closed
V007	open valve	must be closed	open
Nitrogen In	undefined opc	warning: no continuation	

Figure 2. Items on the isolation boundary for maintaining separator “T0100”

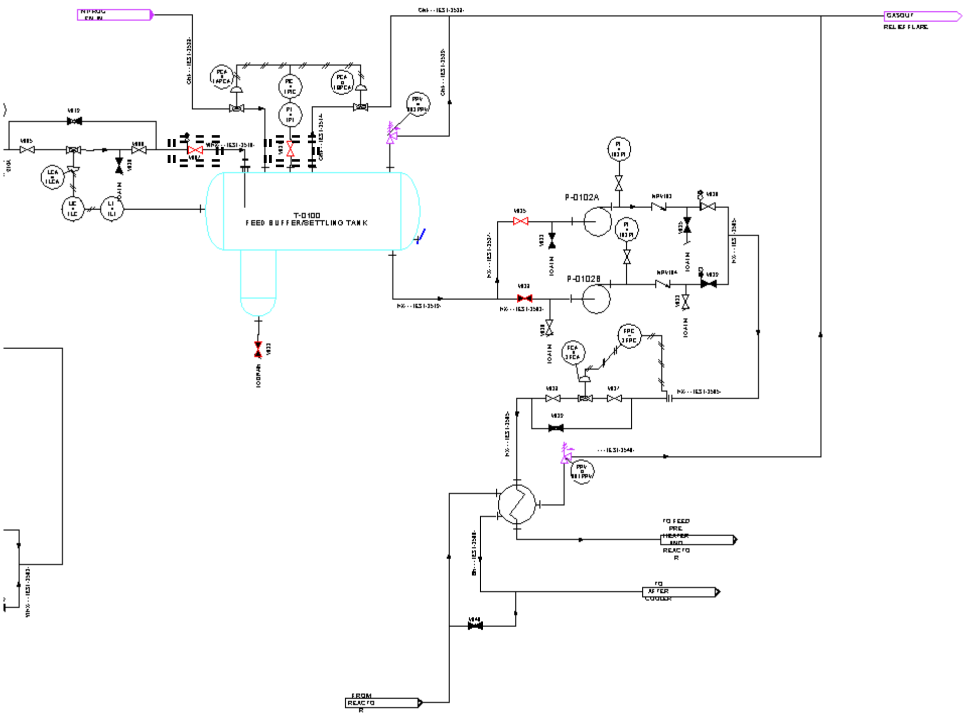


Figure 3. The highlighted isolation boundary for maintaining separator “T-0100”

Conditions	**Actions**	**Result**
“no flow” passing through V_{001} (this should be the original HAZOP result with only “no flow” being considered)	Doing HAZOP analysis	HAZOP result_1
“no flow” passing through V_{002} + V_{001} closed	Doing HAZOP analysis	HAZOP result_2
....	…………………..	………………….
“no flow” passing through V_N + V_{N-1} closed + V_{N-2} closed …+V_{001} closed	Doing HAZOP analysis	HAZOP result $_n$

Table 1. The hazard identification process

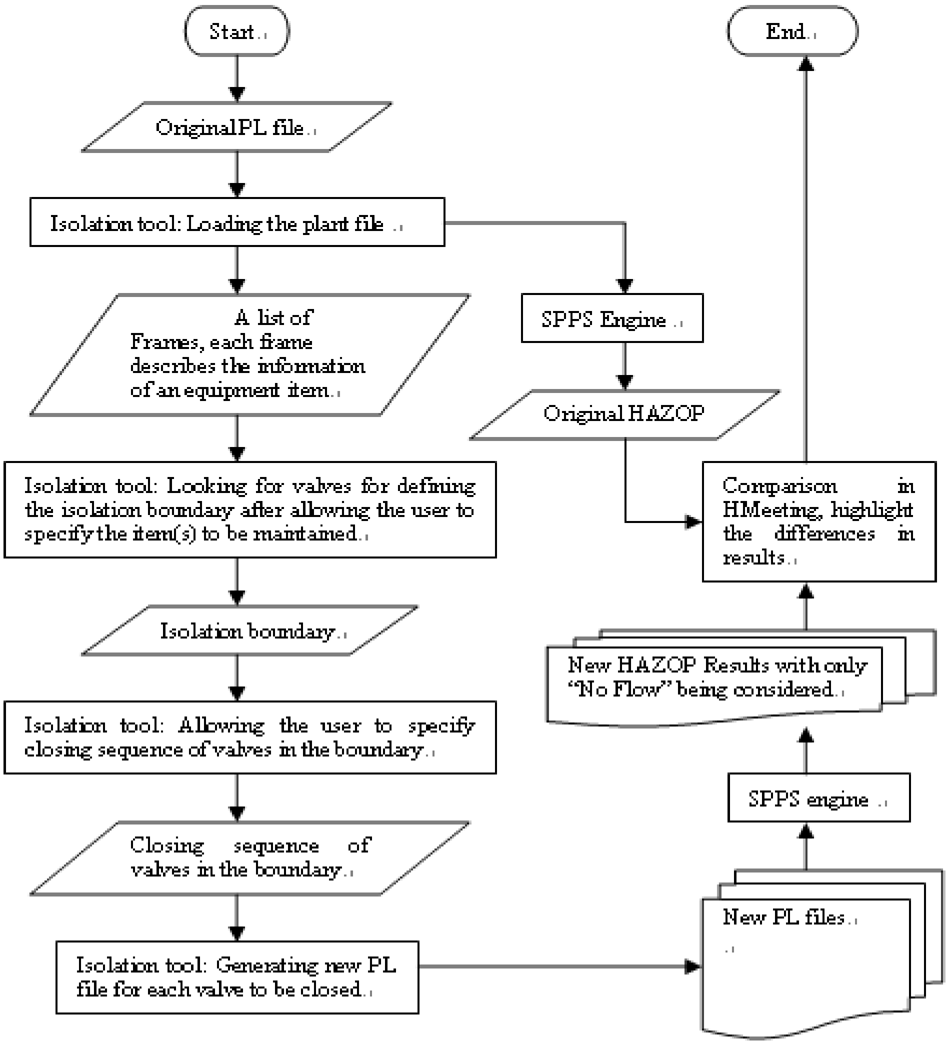

Start
Original PL file
Isolation tool: Loading the plant file
A list of Frames, each frame describes the information of an equipment item.
Isolation tool: Looking for valves for defining the isolation boundary after allowing the user to specify the item(s) to be maintained.
Isolation boundary.
Isolation tool: Allowing the user to specify closing sequence of valves in the boundary.
Closing sequence of valves in the boundary.
Isolation tool: Generating new PL file for each valve to be closed.
SPPS Engine
Original HAZOP
Comparison in HMeeting, highlight the differences in results.
New HAZOP Results with only “No Flow” being considered.
SPPS engine
New PL files.
End

Figure 4. Workflow of the Isolation Tool

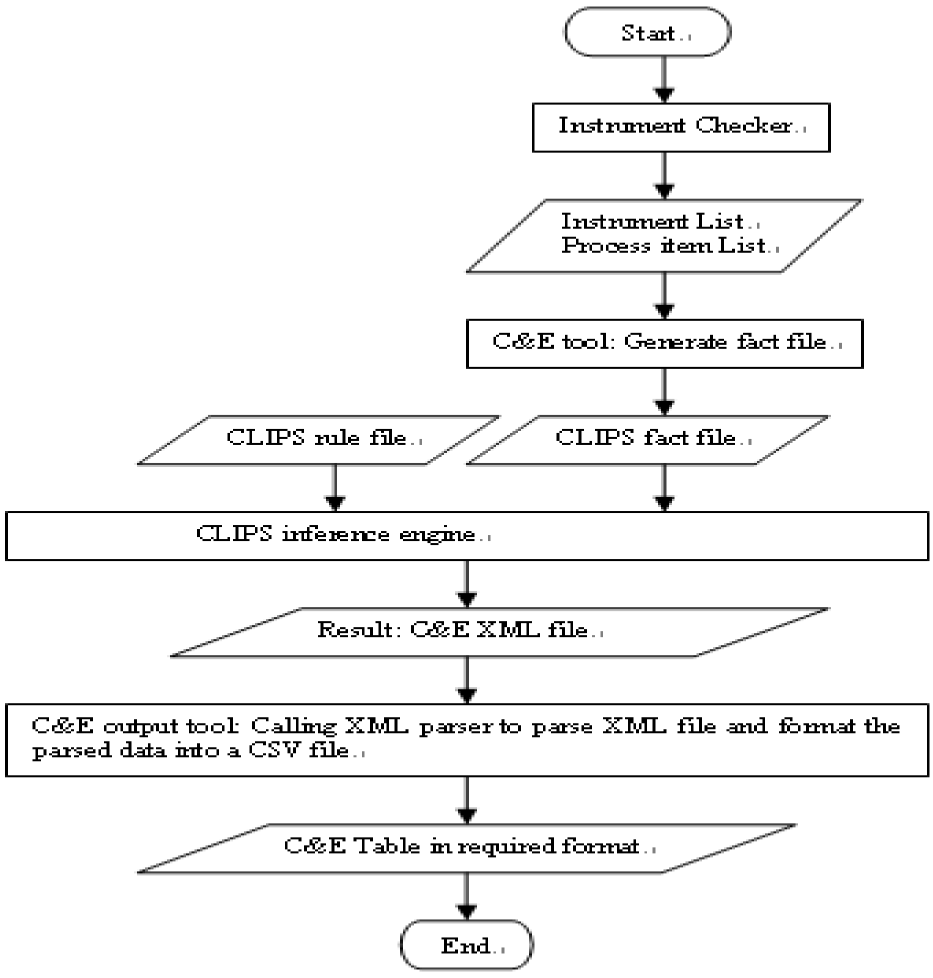

Start
Instrument Checker
Instrument List
Process item List
C&E tool: Generate fact file
CLIPS rule file
CLIPS fact file
CLIPS inference engine
Result: C&E XML file
C&E output tool: Calling XML parser to parse XML file and format the parsed data into a CSV file
C&E Table in required format
End

Figure 5. Workflow for Cause-Effect Analysis

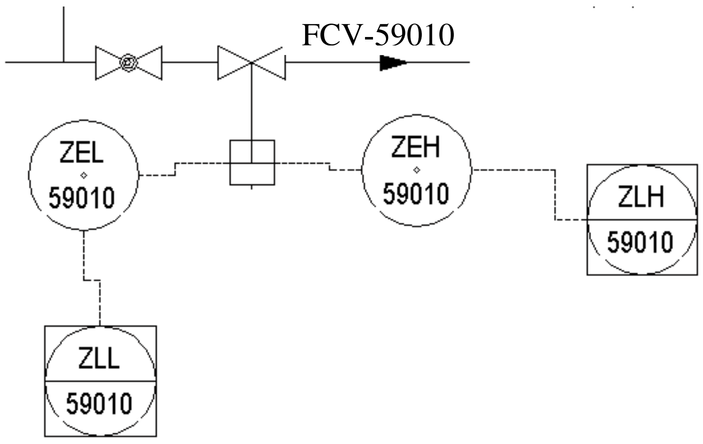
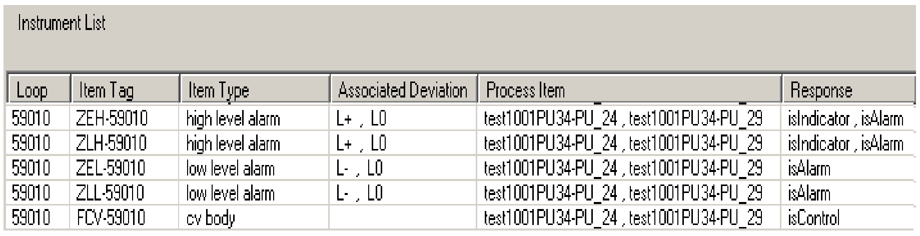


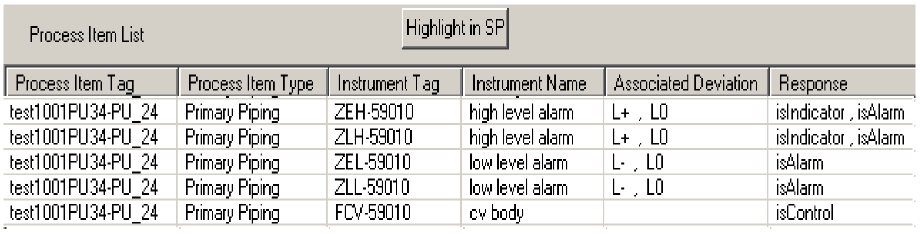
Figure 6. A simple instrument loop



Instrument List

Loop	Item Tag	Item Type	Associated Deviation	Process Item	Response
59010	ZEH-59010	high level alarm	L+ , L0	test1001PU34-PU_24 , test1001PU34-PU_29	isIndicator , isAlarm
59010	ZLH-59010	high level alarm	L+ , L0	test1001PU34-PU_24 , test1001PU34-PU_29	isIndicator , isAlarm
59010	ZEL-59010	low level alarm	L- , L0	test1001PU34-PU_24 , test1001PU34-PU_29	isAlarm
59010	ZLL-59010	low level alarm	L- , L0	test1001PU34-PU_24 , test1001PU34-PU_29	isAlarm
59010	FCV-59010	cv body		test1001PU34-PU_24 , test1001PU34-PU_29	isControl

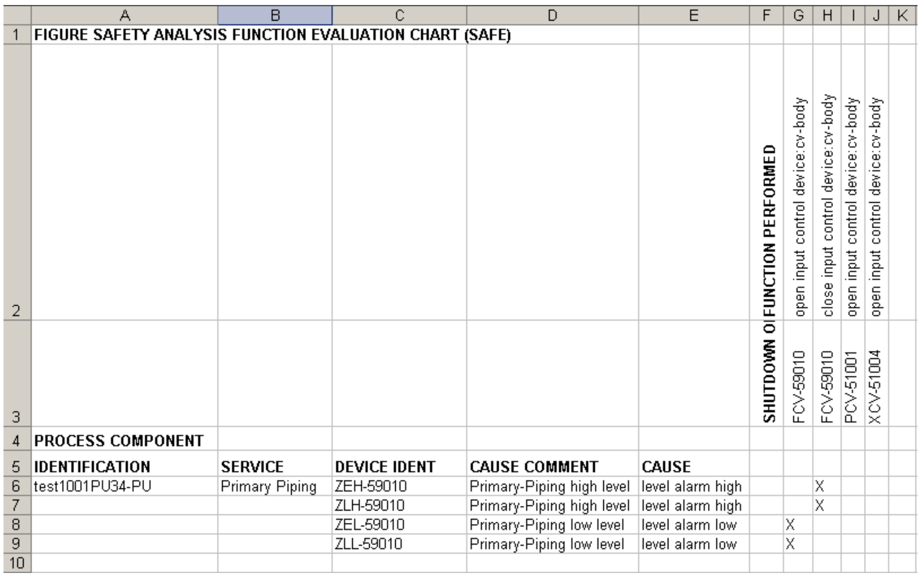
Figure 7. Instrument List



Process Item List

Process Item Tag	Process Item Type	Instrument Tag	Instrument Name	Associated Deviation	Response
test1001PU34-PU_24	Primary Piping	ZEH-59010	high level alarm	L+ , L0	isIndicator , isAlarm
test1001PU34-PU_24	Primary Piping	ZLH-59010	high level alarm	L+ , L0	isIndicator , isAlarm
test1001PU34-PU_24	Primary Piping	ZEL-59010	low level alarm	L- , L0	isAlarm
test1001PU34-PU_24	Primary Piping	ZLL-59010	low level alarm	L- , L0	isAlarm
test1001PU34-PU_24	Primary Piping	FCV-59010	cv body		isControl

Figure 8. Process Item List



	A	B	C	D	E	F	G	H	I	J	K
1	FIGURE SAFETY ANALYSIS FUNCTION EVALUATION CHART (SAFE)										
2						SHUTDOWN OI FUNCTION PERFORMED	open input control device:cv-body	close input control device:cv-body	open input control device:cv-body	open input control device:cv-body	
3							FCV-59010	FCV-59010	PCV-51001	XCV-51004	
4	PROCESS COMPONENT										
5	IDENTIFICATION	SERVICE	DEVICE IDENT	CAUSE COMMENT	CAUSE						
6	test1001PU34-PU	Primary Piping	ZEH-59010	Primary-Piping high level	level alarm high			X			
7			ZLH-59010	Primary-Piping high level	level alarm high			X			
8			ZEL-59010	Primary-Piping low level	level alarm low		X				
9			ZLL-59010	Primary-Piping low level	level alarm low		X				
10											

Figure 9. Part of the Cause-Effect Table in Excel

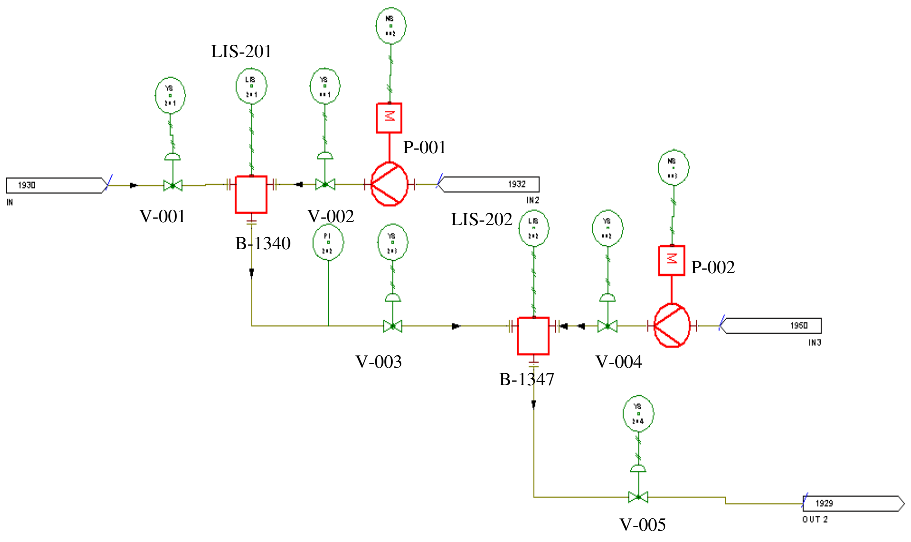
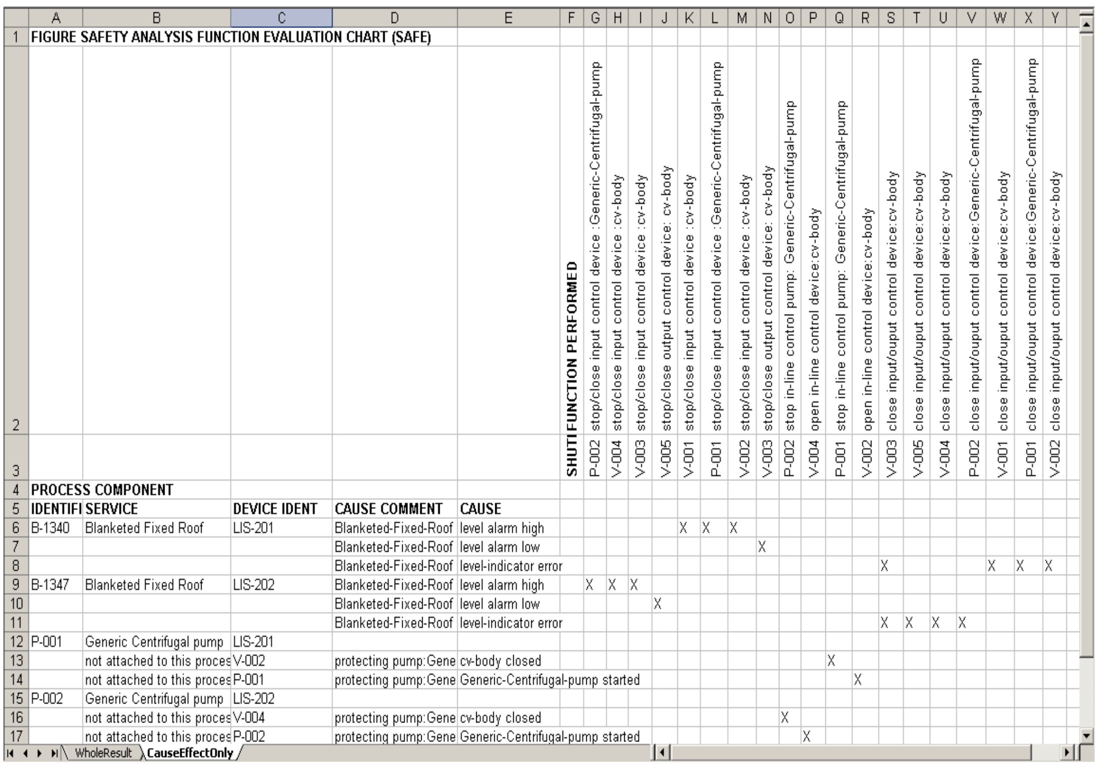


Figure 10. P&ID example extended from Drath et al. (2006)



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	FIGURE SAFETY ANALYSIS FUNCTION EVALUATION CHART (SAFE)																								
2						FUNCTION PERFORMED	stop/close input control device :Generic-Centrifugal-pump	stop/close input control device :cv-body	stop/close input control device :cv-body	stop/close output control device: cv-body	stop/close input control device :cv-body	stop/close input control device :Generic-Centrifugal-pump	stop/close input control device :cv-body	stop/close output control device: cv-body	stop in-line control pump: Generic-Centrifugal-pump	open in-line control device:cv-body	stop in-line control pump: Generic-Centrifugal-pump	open in-line control device:cv-body	close input/ouput control device:cv-body	close input/ouput control device:cv-body	close input/ouput control device:cv-body	close input/ouput control device:Generic-Centrifugal-pump	close input/ouput control device:cv-body	close input/ouput control device:Generic-Centrifugal-pump	close input/ouput control device:cv-body
3						SHUTI	P-002	V-004	V-003	V-005	V-001	P-001	V-002	V-003	P-002	V-004	P-001	V-002	V-003	V-005	V-004	P-002	V-001	P-001	V-002
4	PROCESS COMPONENT																								
5	IDENTIFI	SERVICE	DEVICE IDENT	CAUSE COMMENT	CAUSE																				
6	B-1340	Blanketed Fixed Roof	LIS-201	Blanketed-Fixed-Roof	level alarm high						X	X	X												
7				Blanketed-Fixed-Roof	level alarm low									X											
8				Blanketed-Fixed-Roof	level-indicator error														X				X	X	X
9	B-1347	Blanketed Fixed Roof	LIS-202	Blanketed-Fixed-Roof	level alarm high		X	X	X																
10				Blanketed-Fixed-Roof	level alarm low					X															
11				Blanketed-Fixed-Roof	level-indicator error														X	X	X	X			
12	P-001	Generic Centrifugal pump	LIS-201																						
13		not attached to this proces	V-002	protecting pump:Gene	cv-body closed												X								
14		not attached to this proces	P-001	protecting pump:Gene	Generic-Centrifugal-pump started													X							
15	P-002	Generic Centrifugal pump	LIS-202																						
16		not attached to this proces	V-004	protecting pump:Gene	cv-body closed										X										
17		not attached to this proces	P-002	protecting pump:Gene	Generic-Centrifugal-pump started											X									

Figure 11. Cause-Effect Table in Excel for P&ID example extended from Drath et al. (2006).

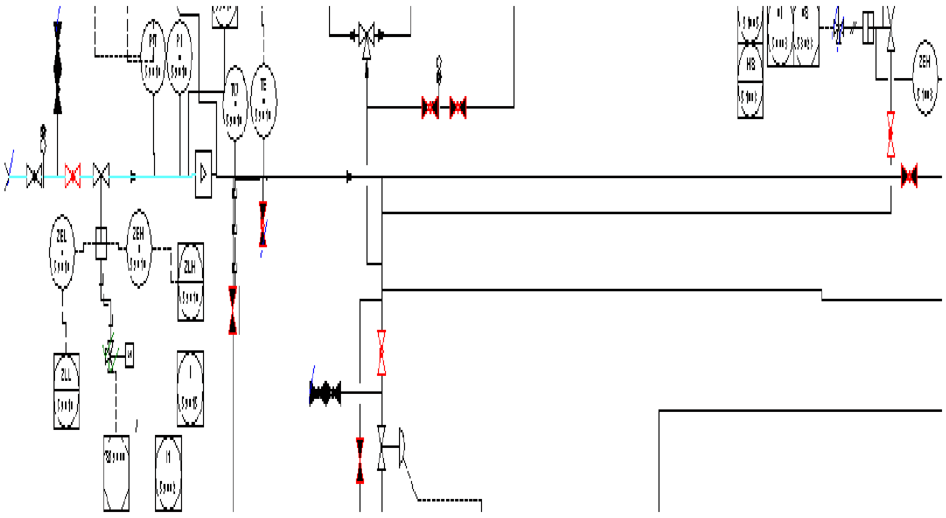
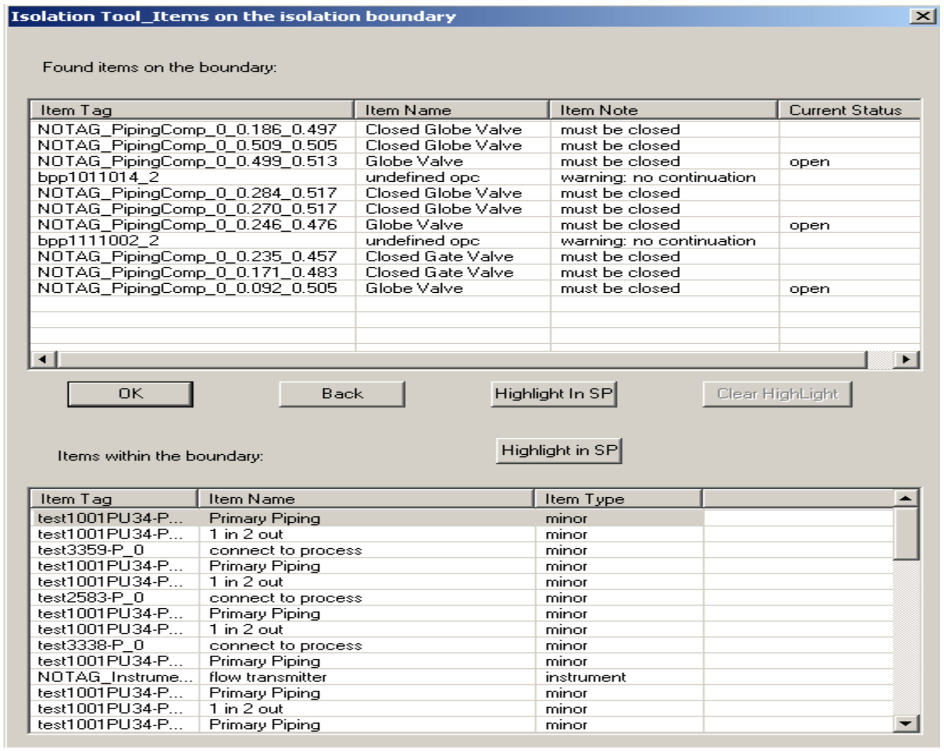


Figure 12. The highlighted isolation boundary for maintaining primary pipe "test1001PU34-PU"



Isolation Tool_Items on the isolation boundary

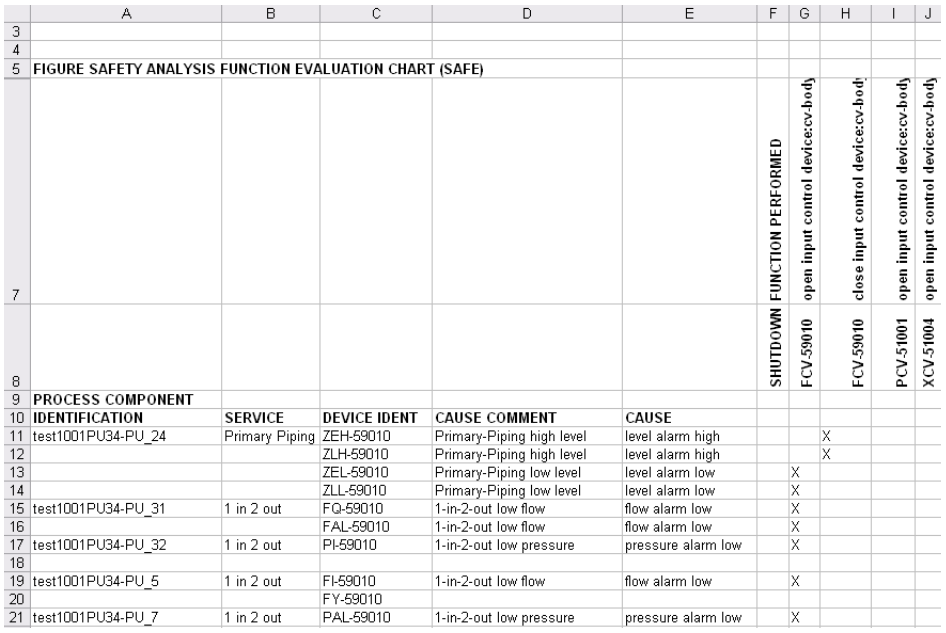
Found items on the boundary:

Item Tag	Item Name	Item Note	Current Status
NOTAG_PipingComp_0_0.186_0.497	Closed Globe Valve	must be closed	
NOTAG_PipingComp_0_0.509_0.505	Closed Globe Valve	must be closed	
NOTAG_PipingComp_0_0.499_0.513	Globe Valve	must be closed	open
bpp1011014_2	undefined opc	warning: no continuation	
NOTAG_PipingComp_0_0.284_0.517	Closed Globe Valve	must be closed	
NOTAG_PipingComp_0_0.270_0.517	Closed Globe Valve	must be closed	
NOTAG_PipingComp_0_0.246_0.476	Globe Valve	must be closed	open
bpp1111002_2	undefined opc	warning: no continuation	
NOTAG_PipingComp_0_0.235_0.457	Closed Gate Valve	must be closed	
NOTAG_PipingComp_0_0.171_0.483	Closed Gate Valve	must be closed	
NOTAG_PipingComp_0_0.092_0.505	Globe Valve	must be closed	open

Items within the boundary:

Item Tag	Item Name	Item Type	
test1001PU34-P...	Primary Piping	minor	
test1001PU34-P...	1 in 2 out	minor	
test3359-P_0	connect to process	minor	
test1001PU34-P...	Primary Piping	minor	
test1001PU34-P...	1 in 2 out	minor	
test2583-P_0	connect to process	minor	
test1001PU34-P...	Primary Piping	minor	
test1001PU34-P...	1 in 2 out	minor	
test3338-P_0	connect to process	minor	
test1001PU34-P...	Primary Piping	minor	
NOTAG_Instrume...	flow transmitter	instrument	
test1001PU34-P...	Primary Piping	minor	
test1001PU34-P...	1 in 2 out	minor	
test1001PU34-P...	Primary Piping	minor	

Figure 13. Items on and within the isolation boundary for maintaining primary pipe "test1001PU34-PU"



	A	B	C	D	E	F	G	H	I	J
3										
4										
5	**FIGURE SAFETY ANALYSIS FUNCTION EVALUATION CHART (SAFE)**									
7						**SHUTDOWN FUNCTION PERFORMED**	**open input control device:cv-body**	**close input control device:cv-body**	**open input control device:cv-body**	**open input control device:cv-body**
8							**FCV-59010**	**FCV-59010**	**PCV-51001**	**XCV-51004**
9	**PROCESS COMPONENT**									
10	**IDENTIFICATION**	**SERVICE**	**DEVICE IDENT**	**CAUSE COMMENT**	**CAUSE**					
11	test1001PU34-PU_24	Primary Piping	ZEH-59010	Primary-Piping high level	level alarm high			X		
12			ZLH-59010	Primary-Piping high level	level alarm high			X		
13			ZEL-59010	Primary-Piping low level	level alarm low		X			
14			ZLL-59010	Primary-Piping low level	level alarm low		X			
15	test1001PU34-PU_31	1 in 2 out	FQ-59010	1-in-2-out low flow	flow alarm low		X			
16			FAL-59010	1-in-2-out low flow	flow alarm low		X			
17	test1001PU34-PU_32	1 in 2 out	PI-59010	1-in-2-out low pressure	pressure alarm low		X			
18										
19	test1001PU34-PU_5	1 in 2 out	FI-59010	1-in-2-out low flow	flow alarm low		X			
20			FY-59010							
21	test1001PU34-PU_7	1 in 2 out	PAL-59010	1-in-2-out low pressure	pressure alarm low		X			

Figure 14. Cause Effect Analysis for the item to be maintained

References

Crawley, F., Preston, M., and Tyler, B., 2000. *HAZOP: Guide to best practice _ guidelines to best practice for the process and chemical industries.* Rugby, Warwickshire, UK: Institution of Chemical Engineers (IChemE).

Drath, R., Fay, A., and Schmidberger, T., 2006. Computer-aided Design and Implementation of Interlock Control Code, *Computer-Aided Control Systems Design, 2006 IEEE International Symposium*. Ladenburg, Germany, 4-6 Oct, 2006. pp.2653-2658.

Giarratano, J., Riley, G., 1994, *Expert Systems, Principles and Programming.* 2nd ed. Boston: PWS Publishing Company.

Hale, A.R., Heming, B.H.J., Smit, K., Rodenburg, F.G.T., and Van Leeuwen, N.D., 1998. Evaluating safety in the management of maintenance activities in the chemical process industry. *Safety Science,* 28(1), pp. 21-44.

Hazid Technology Ltd., 2008. *Hazid Technology* [online], available from http://www.hazid.com [Accessed 15, Feb, 2009]

International Organization for Standardization, 2003. *ISO10418 Petroleum and natural gas industries — Offshore production installations — Basic surface process safety systems.* Switzerland: HIS.

Kletz,T., 1999. *HAZOP AND HAZAN, Identify and assessing process industry hazards.* 4th ed. Institution of Chemical Engineers.

Lawley, H.G., 1974. Operability studies and hazard analysis. *Chemical Engineering Progress,* 70(4), pp.45–56.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Jefferson, M.L., 1999a. HAZID, A Computer Aid for Hazard Identification. 1. The STOPHAZ Package and The SPPS Code: An Overview, The Issues and The Structure. *Process Safety and Environmental Protection,* 77(B6), pp. 317-327.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Chung, P.W.H., 1999b. HAZID, a computer aid for hazard identification. 2. Unit model system. *Process Safety and Environmental Protection,* 77(B6), pp. 328-334.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Jefferson, M.L., 1999c. HAZID, a computer aid for hazard identification. 3. The fluid model and consequence evaluation systems. *Process Safety and Environmental Protection,* 77(B6), pp. 335-353.

Riley, G., 2008, *CLIPS, A tool for building expert systems*. [online] USA: SourceForge. Available at http://clipsrules.sourceforge.net/WhatIsCLIPS.html[Accessed 15,Sep,2008]

Rushton, A.G., Chung, P.W.H., and Lees, F.P., 1998. HAZID, A computer aid for hazard identification: An Overview. *Proceedings of the Loss Prevention and Safety Promotion, the Process Industries 9th international symposium*, 2, pp.503-512.

Wallace, S.J., Merritt, C.W., 2003. Know when to say "when": a review of safety incidents involving maintenance issues. *Process Safety Progress,* 22(4), pp. 212-219.

Wells, G. L., Seagrave, C. J., 1976. *Flowsheeting for Safety - A Guide for Chemical Engineers on Safety Measures to Consider during the Design of Chemical Plant*. Institution of Chemical Engineers.

APPENDIX E A LITERATURE REVIEW ON MODELLING AND MODELLING TOOLS FOR HAZOP

CONTENTS

1 INTRODUCTION

Modelling is all about knowledge representation, knowledge organization and knowledge application, which deals with the information and behaviour of the object to be modelled, and the reasoning strategy that determines how the model is going to be applied.

The aim of this literature review is to present and analyse the current development and technology used in modelling. While focusing on qualitative modelling in hazard identification used in HAZard and OPerability (HAZOP) study in the chemical engineering domain, it also reviews modelling in other domains which may be relevant. The purpose is to adapt good ideas or concepts from other modelling methodology to enhance the modelling power used in automation of HAZOP study.

Inference engine in a conventional expert system uses a set of if-then-else rules in the knowledge base to mimic the reasoning strategy that a human expert would use. This approach causes a problem that is when encountering new conditions, no rules were set up to deal with it because this reasoning does not have a fundamental understanding of the physical system and its activities. So a reasoning approach is demanded to build up models that can provide representation of the information of the physical system and can simulate the behaviours of the physical system as exactly how it works. Two reasoning approaches are proposed, they are quantitative and qualitative reasoning. However, quantitative reasoning requires precise numerical values for each attribute/variable of the physical system to be modelled. This makes the modelling very complex because large volume of unnecessary values has to be obtained. To model a physical system easier and effective, a reasoning approach which is able to capture the causal structure of the system in a more profound way than the conventional expert system but not as rigid as that required in the quantitative reasoning is needed. This leads to the development of the qualitative modelling (Venkatasubramanian et al. 2003b).

Research in modelling can provide more accurate and complete understanding of modelling strategy, and may lead to innovative development in the modelling field. A review of modelling and modelling tools is necessary to get basic understanding of what have been done in this field and how they have been achieved. Although quantitative modelling is inevitably mentioned, this review is concentrating on qualitative modelling technologies. This is because HAZOP analysis is primarily based on qualitative reasoning rather than quantitative reasoning. The modelling approach used to automate HAZOP study is essentially qualitative.

While the review focuses on recent publications, three publications are dated back in the 1990s (Chung 1993, Palmer & Chung 1997, Vaidhyanathan & Venkatasubramanian 1995). This is because the first one describes the Signed Directed Graph (SDG), which is a commonly used method of qualitative modelling in automated HAZOP system and describes its application in an expert system, QUalitative Effects ENgine (QUEEN). The second one gives a definition to qualitative models and describes the basic knowledge and tools creating qualitative models in automated HAZOP system and the third describes a prototype HAZOP automation system and how models are utilized in the HAZOP analysis engine.

This literature review is organized thematically. It begins by introducing the background to the project, leading to discussion of modelling technology.

Section 3 describes modelling in computer-aided HAZOP systems. It proposes the modelling goal and compares two different modelling approaches. It also describes the knowledge representation and knowledge organization in modelling system.

Section 4 surveys a number of modelling tools. The tools are classified into three categories: model definition tools, model building tools and model testing tools. Each category is discussed in terms of aim and objectives, requirements and development methodology.

Section 5 describes model application. Various modelling systems are presented and compared in this section. Interesting points from reviewing these applications are highlighted.

Section 6 concludes the literature review.

2 BACKGROUND INFORMATION

2.1 HAZARD IDENTIFICATION AND HAZOP STUDY

Hazard Identification is very important in any industry which has major potential hazards that may cause harm to the environment and people when hardware fails or mistakes are made. Industries like pharmaceuticals, food, oil & gas and nuclear use a number of established methods for hazard identification and analysis. These include "check list", "index methods", "What if? Analysis", "Preliminary Hazard Analysis (PHA)", "Coarse hazard studies", "HAZard and OPerability studies (HAZOP)", "sneak analysis", "Failure Modes Effect and Criticality Analysis (FMECA)", "Fault and Event tree analysis", and "human Error analysis" (McCoy, 1999a). Within the process industry HAZOP study is widely used and well recognized because of its systematic, exploratory and exhaustive nature. It is used for both in the design of a new plant and in the revision of existing plants (Bartolozzi et al., 2000). The methodology considers each equipment item in a plant against a guide word list (combination of keywords like "more or "less", with variables like "temperature", "pressure", "level"), by exploring any possible deviations and their effects, such as an increase in pressure or a decrease in temperature, that could happen in each equipment item during the operation procedure of a process plant. By tracing the propagation of each deviation, the causes and consequences are identified.

2.2 COMPUTER-AIDED HAZOP STUDY

A process plant could have thousands of equipment items, e.g. an oil refinery. This causes a serious workload problem as it is very time consuming and tedious to consider all equipment items in a big plant. Typically it requires people from different disciplines to work together in many meetings, over several weeks to consider deviations, analyse propagations, identify causes and consequences and record the HAZOP study result.

The motivation behind the development of computer-aided HAZOP study was to automate the laborious part of HAZOP study and to capture the expertise in a knowledge base so that HAZOP study can be carried out more speedly and effectively. A review of the development of computer-aided HAZOP study is given by Rushton et al. (1998) and by McCoy et al. (1999a, 1999b, 1999c). HAZID, developed by Hazid Technologies Ltd, is currently the most advanced commercially available knowledge-based HAZOP tool available (ref to company web site).

3 MODELLING IN COMPUTER AIDED HAZOP STUDY

When a computer system is used to perform a task, it is required that the system designed for aid can have all the necessary information to carry out that task and can organize these information in an effective way that can help carry out the task successfully in terms of accuracy, speed and completeness etc.

Modelling is one of the many ways to meet the above requirement.

3.1 MODELLING GOAL

Although modelling is all about knowledge representation and organization, the objective of modelling for various tasks could be very different. When the task is to help identify hazard in process industry, the modelling goal in general should be that the model can represent all the information required to identify hazard and can organize this information in a way that can make the performance of hazard identification analysis easier, more effective and more consistent than the performance of the analysis by human experts.

More specifically, it is said in Nemeth et al. (2005) that the model for HAZOP automation should be able to:

- Describe the consequences of all possible causes and all possible deviations between the causes and the consequences.
- Provide the effect of preventive actions for each entry of a given HAZOP result table.

So the first question we should answer is what makes a good model?

- It is no doubt that a good model should represent the information as accurately as possible.
- A good model should include the information as complete as possible. For example, the functionality must be considered at the model design stage, in both normal and abnormal states (Bartolozzi et al.,2000).
- A good model should be able to simulate all the behaviours of the object to be modelled. For example, when modelling fault path propagation in a process plant, the model should contain the information of propagating deviations with both directions, from upstream equipment unit to downstream equipment unit and from downstream equipment unit to upstream equipment unit.
- On the other hand, a good model should be able to ignore or skip the information that is irrelevant to fulfil the application objectives of building model (Palmer & Chung 1997).
- The size of a good model should be as small as possible.
- A good model should be as generic as possible.
- In terms of reusability, a good model should be as independent as possible from the environment of its application. So that it can be used widely under various circumstances/environment.
- A good model should be constructed as simply as possible.
- The maintenance of a good model should be as easy as possible.

- The application of a good model in real-time should be as simple as possible. In other words, the complexity level of model application must be as low as possible.
- A good model must have high levels of flexibility and generality that can be applied on any objects to be modelled with different configurations. For example, in the case of automating HAZOP study, a good model must be able to perform analysis on any plant with different states.
- A computer-aided modelling system must have facilities to validate models built (Palmer & Chung, 2000).
- A good balance among expressive power of the model, complexity of building models, and computational cost of driving simulations must be properly maintained (McCoy et al., 2006).

In a nutshell, a good model should make the modelling effort as minimal as possible for fast and easy deployment of any application (Venkatasubramanian et al., 2003a).

In order to have a good model as described above, let us take a look at the current modelling technology.

3.2 MODELS FOR VARIOUS MODELING GOALS

For various modelling goals, people have developed many kinds of models. In general, there are linear models, non-linear models. For models designed to be applied for specific purpose, for example, for HAZOP analysis, they are classified by Bartolozzi et al. (2000) in his Hazard Support System (HAST) system as: cause models, HAZOP models, and consequence models and even mental models which is very unique and is for the HAZOP study members/analysts at the stage of thinking about deviations, main variables that could be used in the plant. In HAZID system, we have equipment unit models. In HAZOPExpert system described by Vaidhyanathan and Venkatasubramanian (1995), there are HAZOP-Digraph (HDG) models. Nemeth et al. (2005) talks about process models.

All of these models are designed for their own specific modelling goal, so there must be similarity and difference between models designed for similar objective. We have introduction of these models in section 3.4 and section 5 when we discuss model application.

3.3 QUALITATIVE MODELLING AND QUANTITATIVE MODELLING

Current modelling methodology can be broadly classified as qualitative modelling and quantitative modelling.

In the glossary of the Holtzapple and Reece (2003), qualitative modelling is defined as “a non-numerical description of a physical system”. In Palmer and Chung (1997), qualitative models are defined as abstract representations of a plant that are able to simulate/emulate behaviour of the plant by emphasizing a causal explanation. We can have SDG-based modelling and state-based modelling used in HAZID system as examples of qualitative modelling.

Quantitative modelling, on the other hand and for the sake of the simplicity, could be defined as “a numerical description of a physical system”. Model of this kind includes “Black-box models”, “First-principles models”, and “Frequency response models”. “Black-box models” include general “input-output models” and “state-space models” etc. (Venkatasubramanian et al., 2003a).

However, modelling technology used in application does not have to be absolutely qualitative or quantitative. Qualitative modelling approach could be slightly or partly quantitative when the accurate values of some variables are demanded for some important purpose. For example, when operation/action is being modelled, action that is taken “too early” or “too late” will be the deviation of action time. But we have to define how early is too early and how late is too late. Therefore, an accurate measurement of the action time is demanded.

On the other hand, it is possible to have qualitative modelling to be added into some part of a quantitative modelling when necessary. In the quantitative modelling review by Venkatasubramanian et al. (2003a), a method of computing residual is used to capture the failure in a system. It examines what the expected behaviour should be and what the real time value or actual behaviour is. The value of residual is obtained by the discrepancy between these two values. When the value of residual is zero, it means no failure happens in the system. But if serious failure happens or underlying system is changed or the real time system is completely or partly modified without modifying the modelling system, the value of residual will reach to a very critical level. Residual is also named as symptom in Nemeth et al. (2005).

When designing models for a specific system, which modelling approach is to be dominantly used depends on the purpose of the whole system. In the example above, if the modelling objective is to tell if there is fault, then a numerical value of residual is needed. On the other hand, if the modelling objective is to tell the change of severity of failure, then an increase of the residual or decrease of the residual can indicate that the failure becomes serious or becomes less serious.

Let us take a look at the difference between quantitative and qualitative models and the advantages of qualitative modelling in application.

Difference between quantitative and qualitative models:

- Qualitative models use qualitative function to express the relationship between inputs and outputs of each unit of a physical process. Quantitative models, on the other hand, use mathematical functional relationships between inputs and outputs of the whole system to describe physics process (Venkatasubramanian & Rengaswamy, 2003).
- Emphasis in qualitative system is how to obtain comprehensible models that are able to intuitively explain the system behaviour just as the mental model of a human expert is intuitive essentially (Bratko & Šuc, 2003).

Advantages of qualitative modeling can be briefly described as:

- Behaviours of qualitative models can be induced even if no mathematical models can be developed since qualitative models do not require accurate information such as numerical values of process variables (Venkatasubramanian & Rengaswamy, 2003).
- The format of the qualitative model can either be a qualitative causal model or an abstraction hierarchy (Venkatasubramanian & Rengaswamy, 2003b).

- Explanations for physical behaviour in a qualitative model are more causal and intuitive, therefore providing better insight into the working mechanism of the physical object to be modelled (Bratko & Šuc, 2003). Its highly abstract level makes it possible to represent more than one behaviour at a giving variable value because it does not require precise value, "more" or "less" or "none" is usually sufficient. On the other hand, quantitative models represent only one behaviour at a giving variable value because it requires precise numerical value (McCoy, 1999a).
- Qualitative modelling can detect subtle qualitative dependences and trace its effect. For example, more temperature is more no matter how much more it is, and it will be regarded as a deviation which may propagate to cause a significant consequence. Quanalitative modelling in this case, may ignore the change of the temperature when the change is very little like a noise (Bratko & Šuc, 2003). Plus, it is easier to check if the model can perform the desired behaviour qualitatively than quantitatively (Schaich et al., 2001).

However, nothing is perfect. There are three main problems with qualitative model as listed below. Disadvantages of qualitative modelling can be briefly described as:

- It may cause generation of spurious solutions.
- It may cause generation of ambiguities in knowledge representation.
- Qualitative method defines static relations only, thus it is not sufficient to explain state that changes with time (Bratko & Šuc, 2003).

These disadvantages can be overcome by combining quantitative modelling with qualitative modelling. This is why current modelling applications normally have qualitative models dominated while quantitative modelling is also applied whenever necessary.

3.4 MODEL INFORMATION REPRESENTATION

Modelling includes two parts of work: description of the object being modelled and representation of the behaviour of the object being modelled. Some modelling approach represents the behaviour by using parameters, variables, key words etc. (Nemeth et al., 2005). This is used in the modelling work of HAZOP automation.

The development of model information representation in HAZOP analysis automation has gone through from representing continuously operating plants, which assumes that each equipment will keep in a steady state all the time, to representing discrete operation, which includes not only the dynamic change of state over the time, but also the event or action sequence, time and operator's action.

First, in order to represent the information accurately and completely, any models have to be built based on fundamental understanding of the physical system to be modelled. Then these understanding will be expressed in different way according to the modelling approach used (Venkatasubramanian et al., 2003 Part II).

Petri Net, which is a modelling language used to make mathematical representation in discrete distributed system, is explored by McCoy et al. (2006) to arrange action sequence. It is used in a hierarchical way in which actions/operations are decomposed into sub-actions in a sub-net, therefore allowing template to be built for repeatedly needed actions. However, this approach is finally proven not flexible enough to be used when sequence order or more action need to be added or equipment unit is modified.

The new model in McCoy et al. (2006) is enhanced by adding representation of event (operation event, fault event, hazard event) and representation of constraint.

In HAZID system, model information are represented in various slots for each equipment type.

There are inport slots, outPort slots, unitPort slots (describes internal ports information), must_connection slots (not used anymore), comp_connection slots(describes flow information).

There are lots of ways to describe fault propagation, such as functional equations, program rules, logical expressions, truth tables, fault trees, event trees, reliability block diagrams, influence graphs, SDGs and bond graphs (Palmer & Chung, 2000). Here we discuss the SDGs representation of fault path since it is the most widely used one in process plants.

In HAZID system, fault path propagation information is described by four types of SDG arcs . These four types are

- Deviation to Deviation
- Fault to Deviation
- Deviation to Consequence
- Fault to Consequence.

These arcs represent a dynamic modelling of the plant behaviours based on qualitative method, called Signed Directed Graph (SDG) qualitative method. SDG is network of nodes connected by arcs. Nodes are variables such as “flow”, “temperature”, “pressure” or “level” even “signal” in system. Arcs are influences between nodes, of one variable on another. The “sign” identifies type of influences attached to an arc, negative or positive. For example, “more flow” will result in “less pressure” if the sign attached to the arc is negative or “more temperature” will result in “more flow” if the sign attached to the arc is positive etc. The variable “signal” can only have “No” as its keyword. SDG are described in more detail by Chung (1993).

In old HAZID system, the fault is evaluated by fault frequency (1 to 5) at the level of UML rather than the level of model, a frequency table database is one of the further solutions. The consequence is evaluated by a set of rules (McCoy 1999c).

In a system named Hazard Support Tool (HAST) described by Bartolozzi et al. (2000), “cause model” of equipment unit is used to contain the propagation information that can trace back to the root cause of deviations. It is a use of the principle of fault tree to trace the prime cause of failure. Each cause model includes many mini-trees, the trees’ leaf contains the root cause of that deviation which is on the top of the tree. The root cause includes any operative faults or failures.

In HAST, fault propagation is represented in “HAZOP model”. HAZOP models are sub-sets of cause models, but not contain all the mini-trees of the cause model, instead, it contains only the mini trees that associate with output variable deviations and are relevant with the final result of the HAZOP study. Mini trees that are only for deviation propagation are not included in HAZOP models.

A “consequence model” is set up for finding the consequences of each deviation in HAST. Therefore, the behaviour of an equipment unit in HAST is represented by three models, which are cause model, HAZOP model and consequence model. The connection between consequence model and the cause model is the deviation with the effect of the consequence.

Knowledge representation in HAZOPExpert in Vaidhyanathan and Venkatasubramanian (1995) is discussed in section 5.3.

3.4.1 SOLVING AMBIGUITIES OF KNOWLEDGE REPRESENTATION

As mentioned earlier, a good model should represent information as accurately as possible. Otherwise it will cause serious incorrect application result when the model is applied. In other words, a good modelling approach should provide error free models to be used by the application system. However, incorrect knowledge representation does happen, knowledge representation ambiguity is one of them.

Here we discuss the knowledge representation ambiguities happened in qualitative model built by SDG modelling used in a process plant.-

The primary/root reason is the causal representation of knowledge allowed in the qualitative reasoning. Although causal knowledge representation is sufficient enough to modelling the behaviour of a physical system, it may cause knowledge representation ambiguities when qualitative models are combined to form a system as the multiple causal paths are combined/integrated (Palmer & Chung, 2000).

In Venkatasubramanian et al. (2003b), it is said that ambiguities of knowledge representation can only be completely eliminated by the use of actual quantitative values. However, obtaining actual quantitative values for the purpose of solving ambiguities would largely reduce the advantage of using qualitative reasoning. Some researchers have proposed various ways to solve this problem.

To solve problem of ambiguities in knowledge representation due to qualitative reasoning, quantitative information has to be brought and a worst case outcome strategy is proposed to ensure completeness of HAZOP analysis result although it may cause redundant outcome instead (Vaidhyanathan & Venkatasubramanian, 1995).

One way to eliminate ambiguities is by choosing the shortest path when and only when the shortest path does have a correct influence (Palmer & Chung, 2000). In Chung (1993), a spurious arc path is identified as a cause of knowledge representation ambiguities when two nodes are joined by more than one path(s). The ‘spurious paths’ can be generated when one path has a positive effect while other(s) have the negative effect. To eliminate it, the effect of shorter path or shortest path has been selected because of its “direct influence derived from the original mathematical model” (Chung 1993).

Another way is by adding constraint representation (McCoy et al., 2006).

In an Equipment Model Builder (EMB) presented in Palmer and Chung (2000), a modular method is used to remove ambiguities in qualitative model based on SDG modelling approach. These methods identifies the header/divider combinations and recycle loops, then specifies a module for each of them, then substitutes the module for each of them into the plant description file before going to HAZOP analysis.

3.5 HOW ARE MODELS ORGANIZED?

As mentioned before, knowledge organization is an important part of modelling. A good modelling structure can help the system to reason effectively and easily.

In HAZID System, A Unit Model Library (UML) is built and models in the library are organized in a hierarchical way that supports the inheritance of features between models. Each model has one and only one parent model and this model inherits all the features of its parent model while it owns its unique feature(s) that distinguish it from its sibling model(s). Each model may be a parent model of other model(s). When a new model is created, it will be added into this library as a child of an existing model. This kind of structure is like a family tree. By this way, the size of the library can be kept as small as possible and features of each model kept clear, and it makes model searching easy and quick. A detail description of UML organization is given by McCoy et al. (1999c).

More discussion in knowledge organization is in section 5.

3.6 TWO DIFFERENT QUALITATIVE MODELLING APPROACHES

SDG modelling and state-based modelling are two different qualitative modelling approaches. The following table shows their comparison. From table 1(please see the appendix)we can see that state-based modelling is an enhanced/augmented modelling approach from SDG modelling. It overcomes some shortcomings of SDG modelling by including modelling with change of plant state, change of time, operation order and change of operation sequence etc. And it can deal with situation when two or more actions have effects on each other by the use of flow modelling.

3.7 SUMMARY

In this section, we have discussed modelling in computer aided HAZOP study. We have proposed the basic elements to make a good model, which the key words could be “accuracy”, “completeness”, “conciseness”, “generality”, “independence”, ”simplicity”, “easiness”, “flexibility”, “validation” etc. We have also discussed the difference between qualitative modelling and quantitative modelling, highlighting the advantage and disadvantage of qualitative modelling, therefore, justifying why qualitative reasoning is widely used. We have also discussed model information representation and organization, particularly, presented some ways that researchers had used to solve ambiguities. We have made comparison between two qualitative modelling approaches, their similarity and differences are highlighted in table 1 in the appendix

4 MODELLING TOOLS IN COMPUTER SYSTEM

A seven step modelling procedure is proposed by Nemeth et al. (2005) as below:

"(1) Model goal-set definition (modelling problem specification).

(2) Model conceptualization (identifying controlling factors).

(3) Modelling data: needs and sources.

(4) Model building and model analysis.

(5) Model verification.

(6) Model solution.

(7) Model calibration and validation."

From these seven steps, we can see that it would be very helpful if we could have computer tools designed for model definition, model construction and model verification. Most research in modelling technology combine model definition, model building and model verification into one model creating tool. This review analyses them in separate sections.

4.1 MODEL DEFINITION TOOLS

At current stage, the model definition tools discussed here are for defining the models used for HAZOP automation.

4.1.1 AIM AND OBJECTIVES

Model definition tool is designed for knowledge acquisition. It is a computer aid for the third step of the seven modelling procedure listed above. 28 knowledge acquisition tools have been surveyed in Palmer and Chung 1997. The objective of the tool is summarized as:

- To acquire information necessary to define a model.
- To identify missing, duplicate or incorrect information.
- To group components into different entities.
- To rank information according to their importance level (priority).

4.1.2 GENERAL REQUIREMENTS OF MODEL DEFINITION TOOLS

Target user

Model definition tools should consider at least two types of user. One is the non-modelling Engineering Expert, the other is the Modeller. These two kinds of users might be working in different place all over the world.

Interface

The user interface should be online with internet security and authorization check so that both the Engineering Expert and the Modeller can access it.

The interface that is used to enter information should be in a direct and familiar but unambiguous way.

There should be an explanation facility to show the user what has been acquired and how the system organize them. In other words, the acquired knowledge representation has to be transparent to the user. This is a very important requirement for the long term success of an expert system (Vaidhyanathan & Venkatasubramanian, 1995).

Function and data verification

The acquired knowledge should be editable, removable by the user with some constrains. For example, when defining a model, if the status of that model is locked, this model is not allowed to be deleted unless it is unlocked by the user.

For the definition of a model, there should be different definition cases distinguished by their definition time so that the user can trace the history of definition of this model.

The definition tool should be able to deal with run time conditions.

The definition should be checked and modified by the Modeller and agreed with both Engineering Expert and Modeller before going to be built by the Model Construction Tools.

There should be an alert message to tell the user of any required information is missing or may be incorrect and prevent it finally be entered into the system.

The tool should be able to prevent, locate and remove any duplicated information.

4.1.3 METHODOLOGY

There are two main parts for defining a model. One is to define the basic model information such as model name, model description, model assumption and how the model is connected with other models such as port name, port type if the model is a tank or other equipment item with ports. The other part is to define the behaviour of the model. For example, in the HAZOP automation, it is required to define fault and deviation propagation to represent how the fault would cause consequences within and outside a equipment unit model.

Methods used to meet the above requirement could vary with different modelling applications. Here we present general methodology to implement a model definition tool.

- Internet authority and security technology will be used to restrict the access to this tool online to make it only available to authorized users.
- All users have an attribute as user role to identify if the user is a modeller or a non-modeller engineer.
- Before each model is going to be constructed, the modeller and the non-modeller engineer in the same organization will sign off the model definition as this model definition is agreed by both of them.

- Technology such as PHP or ASP can be used as a powerful tool to build up dynamic and interactive web pages for the user to enter definition information into the database online.
- For each model to be defined, the tool will generate a definition case when its behaviour changes and identify the change by the definition time of each definition case.
- By using JavaScript, user can add more information or delete typed or selected information without refreshing the whole page.
- Any missing or duplicated information will be identified before the content is submitted to the server.
- Any incorrect (eg. conflict) information will be prevented at the webpage design stage. For example, when selecting deviation symbols from the selection box, if the “B+” is selected, then the “B-“ will be removed from the selection list immediately. If “B+” is unselected, the “B-“ will be recovered into the selection list immediately to make both “B+” and “B-“ available to be selected by the user again.
- The defined behaviours of each model under different definition times will be retrieved from the database to make it editable and removable by the user.
- Recommended database includes Oracle database or mySQL database while Oracle is preferred for commercial purpose.

4.2 MODEL CONSTRUCTION TOOLS

This is the computer aid for the fourth step of the seven modelling procedure listed above, which is also the main task of modelling.

4.2.1 AIM AND OBJECTIVES

To construct models that can be utilized by the application.

4.2.2 REQUIREMENT ANALYSIS

It is by no means easy to build a model. Difficulty is discussed in Palmer and Chung 1997.

The difficulty is the building of model library. Two ways were adopted and assessed. One is such as the expert system, Qualitative Effects Engine (QUEEN) directly. The other is by software developers to write models in a specific format that can be used by application engineer to give information to an intermediary and then convert into a format that can be used by QUEEN.

However, some models built by above ways are too specific to be reused or validated. Experience gained in constructing lost, errors happen commonly. Subjective knowledge of important point preference makes it inconsistent, and hard to update, compare and detect error.

A graphical tool is a possible solution for above difficulties but again the priority of model parts is unknown to the expert/chemical engineer, and the graph is tend to be very large and complex whenever meeting a complex process units with many variables and faults waiting to be analysed.

When a modeller or a non-modelling engineer is constructing models, information of model definition has to be converted into the format that could be used in the model construction tool by the modeller or the non-modelling engineer‘s mind. It is very likely that some information defined would be forgotten. Therefore, it would be very helpful if there is a facility to link the model definition with the model construction to help the user recall definition information such as assumption of a model etc.

In current HAZID system, a compositional modelling approach is used to build a model and a plant. By breaking down the plant into several levels of component, instances of equipment item (unit) can be used to represent each component, and then be connected together by connections (McCoy et al., 2006).

4.2.3 VARIOUS MODEL CREATING TOOLS

A Equipment Model Builder (EMB) is built by Palmer (2000). This tool takes a plant description file supplied by the user using AutoCAD and a plant model library as its input. Outputs of the tool are a modified plant description file and a file of unit models occurring within the plant.

In AutoHAZID, an early computer-aided HAZOP tool, a Model Generation Tool (MGT) is used to create the model by asking questions and get answers, one or more templates will be added according to each answer. A number of building blocks are used: chambers, phases and ports. The term “chamber” refers to “a distinct region within a unit”. Whatever separates two chambers are referred as mechanical interface. Whatever separates two fluid phases are referred as phase interface. (vapour-liquid and liquid-liquid). Inlet, outlet, internal port. Phase includes gas, vapour and liquid. The MGT is described by McCoy et al. (1999c).

Coloured Petri Nets(CPN) are one of the most useful unifying modelling tools and has been used in diagnostic goal driven modelling and simulation of multiscale process systems by Nemeth et al. (2005).

4.3 MODEL TESTING TOOLS

Model testing can also be named as model verification. Verification is defined by Palmer and Chung (2001) as “ensuring that the internal structure of each model is , as far as possible, complete, correct and consistent”. The aim of model testing tool is to provide error free models to be used by application analysis of existing plant.

4.3.1 OBJECTIVE OF MODEL TESTING TOOL

- To present the actual input-output behaviour of the model to the modeller.
- To test out if there is missing information in order to ensure completeness.

- To test out if there is wrong information, the information that is mismatch with expectation in order to ensure correctness..
- To test out if there is duplicated or redundant/unnecessary information in order to ensure conciseness.
- To test out if there is conflict information in order to ensure consistence or no ambiguity.
- To sever as a first guard/step to achieve complete validation of the model because the test is performed before the model is going to be used.

4.3.2 REQUIREMENTS ANALYSIS

Target user

- It is expected that the user of any model testing tool should be a trained user of the model construction tool used for building the models under testing. The user is expected to have the knowledge of constructing models and is responsible for the development of the knowledge base of HAZOP automation system for his organization.
- It would be helpful if the user is also a process engineer.

The Interface

- It is required that the interface for initiating tests on models must be integrated with the interface of model construction tool for the convenience of use as the test tool will be used most frequently by the user who builds the model and as the user may go back to model construction tool to modify the model after reading the test result and then go to the testing tool to test the revised model again. This iterative procedure may repeat until the model behaves exactly as the modeller expected.

- The interface should provide feedback on progress of any model tests.

Function

- Model should either be tested individually or in a group to allow multiple model tests.
- There should be a facility to select which model or models to be tested.

- There should be a facility to allow the user to apply different conditions or combination of different conditions on the tests of the same model.
- The test result for each model at any different testing time should be stored and be accessed.
- There should be a facility to compare the intended model behaviour with the actual model behaviour as a result of test so that any missing or extra or wrong information can be identified and highlighted.
- There should be a facility to record the test status of each model tested through time. This facility should allow the user to sign off a model test result as acceptable (for example, the model can perform satisfactory behaviours as expected by the modeller) or as not yet acceptable with comments from the user.
- The test result should be a report in a format that can be exported to external applications such as MS Excel.

4.3.3 METHODOLOGY

By the use of a so called iterative refinement approach that the user/process engineer continues to check the correctness of the model by checking if the test result is matching with the expected result. If not, the user will go back to modify the model , then test and check again until the two sets (testing result and expected result) match.

Methods used to meet the above requirement could vary with different modelling applications. Here we present general methodology to implement a model test tool.

- For each model to be tested at each time, the tool will generate a test file to be analysed by the inference engine of the application system in the same way that a file generated from a true P&ID would be analysed.
- This test file will consist of a single instance of the model to be tested and the information of how this model is connected with other models. For example, when testing a model in UML, a plant file will be generated and it consists of an instance of that model, with each of its external ports connected to an Off-Page-Connector (OPC) instance so that the model under test can be connected to deviations caused by faults in the OPCs and can also communicate any deviations caused by faults in the test model to consequences in the OPCs.
- The intended behaviour (entered by the user with the model construction tool) of each model under different testing conditions will be retrieved from the database to make the comparison between actual and expected results.
- The connection with database will be the same way as the model construction tool. Recommended database includes Oracle database or mySQL database while Oracle is preferred for commercial purpose.

4.4 SUMMARY

In this section, we have analysed model definition tools, model construction tools and model testing tools in terms of their aim and objectives, target users, interfaces, data and function and methodologies etc. Some model construction tools are briefly introduced. The most interesting questions we have to answer are: How to make a model definition/construction/testing tool as effectively as possible to help with the model definition/construction/testing procedure? Is it worth or possible to integrate all these three tools into one? Or is it better to make them separate while provide some facilities to link them together? Further reading and research is needed to answer these questions.

5 MODEL APPLICATION

After above discussion about model and modelling, let us take a look at the model application.

As mentioned early, computerized support systems have been considered to perform the HAZOP analysis for HAZOP experts. Several systems have been proposed by different research groups around the world.

A prototype expert system named HAZOPExpert is proposed by Vaidhyanathan and Venkatasubramanian (1995). HAZID and CHECKOP are developed by Hazid Technologies Ltd. and Loughborough University. Hazard Support Tool (HAST) is developed by Bartolozzi et al. (2000) in Palermo University. Let us take a look at each of them.

5.1 HAZID AND CHECKOP

As mentioned earlier, HAZID system is a result of development of HAZOP automation by Hazid Technologies .Ltd and Loughborough University. It is now developed into a state-of-the-art knowledge-based Hazid Identification system. It is a successful application of qualitative modelling approach based on SDG.

CHECKOP is a prototype developed also by Loughborough University to be used in batch plants. It also uses qualitative modelling, but it enhances the model representation in order to be applied in batch plants. Table 2 (please see the appendix) shows the comparison of these two applications. From this table, we can see:

- The modelling approach used is different. Modelling in HAZID is based on SDG while modelling in CHECKOP is based on state.
- The way of defining a model in CHECKOP is enhanced by adding the knowledge representation of intangible information such as state change, operation/action and event/action sequence, time and assumptions associated with each step of operation.

- Because of the different modelling purpose, the deviation guidewords are different.
- HAZID can be purely performed based on qualitative models while CHECKOP has to combine qualitative models with quantitative reasoning by assigning some accurate numerical value to its variable. For example, we have to define "how late is too late" and "how early is too early" when defining wrong actions.
- Today's HAZID has integrated a model testing bed to verify the model before the model is utilized. In other words, HAZID can provide error free models to the HAZID analysis engine. CHECKOP did not consider this by the time it was developed.
- CHECKOP will need to generate as much as possible useful keyword/deviation of an operation instruction.

It is worthwhile to integrate HAZID with CHECKOP so that HAZID can benefit from the idea in CHECKOP to check out "startup" and "shutdown" procedures in process plants and CHECKOP can benefit from HAZID's simulation efficiency and its model verification facility.

5.2 HAZID, CHECKOP AND HAST

Hazard Support Tool (HAST) is a knowledge-based system to support HAZOP automation by Bartolozzi et al. (2000) in Palermo University.

HAST is originally an automation of the HAZOP analysis of continuous plant, it is then extended to carry out auto HAZOP analysis of batch or semi-continuous plant.

HAST also uses qualitative modelling in automatic HAZOP analysis, but in a different way. Table 3 (please see the appendix) shows the comparison between HAST with HAZID and CHECKOP. It can be seen:

- Building up a model library maybe a good way to store all the models since HAZID, CHECKOP and HAST all have it.
- Characteristic inheritance between models as a family tree produces a high level of generality, applicability and succinctness (the size of model library is kept small by this way) for HAZID, CHECKOP and HAST, which fulfils two of the modelling goals we proposed in section 3.1. UML in HAZID further fulfils this modelling goal by the use of template (McCoy et al., 1999c).
- Models built based on basic equipment units make them independent with the environment of application, which fulfils model reusability.
- The use of template is a good way of making model creating simply as many features could be imported from templates rather than creating these features for a particular model. Further more, it is also a good way of making maintenance easy as modifying a feature of many models could be achieved by modifying the feature in their template (McCoy et al., 1999c).
- The decomposition approach allowed by HAZID, CHECKOP and HAST produces a low level of complexity of model application. A suggestion is made by McCoy et al. (2006) to further decompose the unit model into smaller basic unit in order to cope with devices

such as level transmitter, breaking down it into a float gauge component and a transducer component and the connections between them.

- The consideration of functionality at the design stage in HAST in both normal and abnormal states is a good way to represent modelling information as complete as possible. This is why it can be used in both continuous plant and batch plant.
- Modelling fault path propagation using SDG arcs to connect fault, deviations and consequences is simpler and more intuitionistic than that using mini fault trees since it can not present direct connection between cause model, HAZOP model and consequence model to show a complete fault path.

5.3 HAZOPEXPERT AND ITS HDG MODEL

HAZOPExpert is an expert system designed to automate HAZOP analysis by Vaidhyanathan and Venkatasubramanian etc. in Purdue University in 1994 and further improved in 1995. Reviewing this system is because that it helps understand better how people design and build up models used by inference engine and what is behind that engine.

The old HAZOP automation system is in fact very similar to today's HAZID system although it uses a so called HDG model (HAZOP-Digraph Model) instead of SDG model to modelling the equipment units and their process variables and their behaviours. Both SDG and HDG are extended to cover nodes of abnormal cause and nodes of adverse consequences. Both are defined for each process unit model of a plant.

Both HDG and SDG are used to propagate process variable deviations and to find abnormal cause and adverse consequences. They all have conditional arcs in which conditions are attached to some SDG or HDG arcs to deal with conditional propagation causal relationship. For example, the increasing flow rate at inlet port of a tank will result in the increase of temperature of the fluid in the tank only if the temperature of the fluid at the inlet port is higher than the temperature of the fluid in the tank. On the other hand, the increasing flow rate at inlet port of a tank will result in the decrease of temperature of the fluid in the tank only if the temperature of the fluid at the inlet port is lower than the temperature of the fluid in the tank.

Table 4 (please see the appendix) shows the comparison between HAZOPExpert and HAZID. Some points gained from HAZOPExpert are interesting:

- Can the feature of graphic representation of model in HAZOPExpert be considered in current HAZID system?
- The SDG modelling which connects all the units by mini SDG arc simplifies the fault path propagation and makes it easier to be understood than HDG modelling.

5.4 SUMMARY

In this section, we have discussed four applications used for HAZOP automation, HAZID, CHECKOP, HAST and HAZOPExpert. A common feature of above systems is that they are all knowledge-based systems, and they all build up models to represent equipment units and simulate their behaviours. The most interesting point is that models built and used in these systems are all qualitative models since conventional HAZOP analysis is performed qualitatively. However, when these systems are extended to deal with batch plants for example, they all have to combine qualitative modelling with quantitative modelling in some extent although conventional HAZOP analysis of batch plant is also performed qualitatively. Another common feature is that they all develop their models in a context-independent manner so that their models can be used in wide variety of application. In addition, they all encounter ambiguity of knowledge representation, which is inherent in qualitative causal reasoning (Vaidhyanathan & Venkatasubramanian, 1995).

However, the way of model defining, model constructing, model verifying, model organization, model utilization is very different from each other in these systems.

HAZID and CHECKOP use SDG arcs to connect fault, deviations and consequences. HAST finds out the root cause by mini fault trees and finds out consequences by consequences tree. HAZOPExpert uses HDG model to represent equipment items and their behaviours.

HAZID establishes a fault path connected by SDG arcs to directly describe how a fault would cause consequence through deviations happen along the path. It is easier to understand than fault tree or HDG model.

6 CONCLUSION

This literature review has reviewed modelling and modelling tools used in computer-aided system. We have proposed the features that a good model should have. We have also discussed qualitative and quantitative modelling approaches. Modelling tools are discussed. Several modelling applications used in HAZOP automation are examined.

For modelling applications in HAZOP analysis automation, the core problems to solve for any applications are

- How to find the abnormal cause and root cause of a fault path.
- How to find the adverse consequences of a fault path.
- How to propagate the deviations between the cause and consequences.

The solution for above lies at the design stage of modelling. The inference engine for any application will reason in the way these models are designed by retrieving information from attributes of the models and relationship between models. So one point is that to design a good model, it is worth to consider how the inference engine is going to interact with that model to achieve the desired result.

It is interesting and worthwhile to investigate if there are any better methods for transforming quantitative data into qualitative data and if qualitative modelling language could be expressed more explicitly. It is possible to discover hidden variables by study deeply the principles of the existing variables. It is also possible to increase the model generality and decrease the size of models by studying deeply the model organization.

7 REFERENCES

Bartolozzi, V., Castiglione, L., Picciotto A., and Galluzzo, M., 2000. Qualitative models of equipment units and their use in automatic HAZOP analysis. *Reliability Engineering & System Safety,* 70(1), pp.49-57.

Bratko, I., Suc, D., 2003. Learning qualitative models. *AI Magazine,* 24(4), pp.107-119. © AAAI Press.

Chung, P.W.H., 1993. Qualitative analysis of process plant behaviour. *Proceedings of the Sixth International Conference,* Edinburgh, Scotland, 1-4 Jun, 1993, pp.277-283.

Holtzapple T Mark., Reece W. Dan, 2003, *Foundations of Engineering, 2/e,* available from http://highered.mcgraw-hill.com/sites/0072480823/student_view0/glossary.html, cited on 17:12, 19, Jan, 07.

McCoy, S.A., 1999 a, *Combining Qualitative and Quantitative Reasoning to support Hazard Identification by Computer.* PhD, Loughborough University.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Jefferson, M.L., 1999b. HAZID, A Computer Aid for Hazard Identification. 1. The STOPHAZ Package and The SPPS Code: An Overview, The Issues and The Structure. *Process Safety and Environmental Protection,* 77(B6), pp.317-327.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Chung, P.W.H., 1999c. HAZID, a computer aid for hazard identification. 2. Unit model system. *Process Safety and Environmental Protection,* 77(B6), pp.328-334.

McCoy, S.A., Wakeman, S.J., Larkin, F.D., and Jefferson, M.L., 1999d. HAZID, a computer aid for hazard identification. 3. The fluid model and consequence evaluation systems. *Process Safety and Environmental Protection,* 77(B6), pp.335-353.

McCoy, S.A., Zhou, D., and Chung, P.W.H., 2006. State-based modelling in hazard identification. *Applied Intelligence*, 24(3), pp.263-279.

Nemeth, E., Cameron, I.T., and Hangos, K.M., 2005. Diagnostic goal driven modelling and simulation of multiscale process systems. *Computers and Chemical Engineering*, 29, pp.783-796.

Palmer, C., Chung, P. W.H., 1997. Constructing Qualitative Models. *Proceedings of the 1997 Jubilee Research Event, IChemE*, pp.725-728.

Palmer, C., Chung, P. W.H., 2000. Creating Signed Directed Graph Models for Process Plants. *Ind. Eng. Chem. Res*, 39, pp.2548-2558.

Palmer, C., Chung, P.W.H., 2001. Verification of Process Plant Models. *Proceedings of the 2001 AAAI Symposium on Model-based validation of intelligence*, Stanford, pp.37-41, AAAI press.

Rushton, A.G., Chung, P.W.H., and Lees, F.P., 1998. HAZID, A computer aid for hazard identification: An Overview. *Proceedings of the Loss Prevention and Safety Promotion, the Process Industries 9th international symposium*, 2, pp.503-512.

Schaich, D., Becker, R., and King, R., 2001. Qualitative modelling for automatic identification of mathematical models of chemical reaction systems. *Control Engineering Practice,* 9(12), pp.1373-1381.

Vaidhyanathan, R., Venkatasubramanian, V., 1995. Digraph-based models for automated HAZOP analysis. *Reliability Engineering and System Safety,* 50, pp.33-49.

Venkatasubramanian, V., Rengaswamy, R., Yin, K., and Kavuri, S.N., 2003a. A review of process fault detection and diagnosis Part I: Quantitative model-based methods. *Computers & Chemical Engineering,* 27(3), pp. 293-311.

Venkatasubramanian, V., Rengaswamy, R., and Kavuri, S.N., 2003b. A review of process fault detection and diagnosis Part II: Qualitative models and search strategies. *Computers & Chemical Engineering,* 27(3), pp. 313-326.

8 LIST OF ABBREVIATIONS

CPN: Coloured Petri Nets (CPN).

FMS: Fluid Model System, which is a system that performs fault path validation by using predicates and functions.

HAST: Hazard Support Tool built by Bartolozzi et al. (2000) in Palermo University.

HAZID: A knowledge based HAZOP automation system by HAZID Technologies .Ltd and Loughborough University.

HAZOPExpert: A prototype computer aided HAZOP automation system by Vaidhyanathan and Venkatasubramanian in 1995.

HAZOP: hazard and operability (HAZOP) study used to identify hazards.

HDG: HAZOP-Digraph Model used in HAZOPExpert.

OPC: Off-Page-Connector.

MDT: Model Definition Tool.

MTB: Model Test Bed.

SDG: Signed Directed Graph.

UML: Unit Model Library used in HAZID system.

9. APPENDIX

Comparison between SDG modelling and state-based modelling

SDG-based modelling	**State-based modelling**
Used in continuously operating plants.	Used in discrete operating plants, can also be used in continuously operating plants.
Dealing with normal behavioural modes which assume the model is in good and healthy condition.	Support reasoning about “common mode failures” which would happen when for example, power supply is failed.
Assume no change with the state.	Consider time change, state change, action sequence and change of sequence.
deviation guidewords are “more, less, no etc” and parameters/variables are “flow”, “temperature”, “vapour” etc, which is not related to operations.	deviation guidewords are “no, early, late” and parameter is “action” and “termination”, which are related to operations.
Assumption is all the same.	Assumption is different for each step of operation.
	Models behaves in similar way as in SDG unless a failure mode happens.
	Use “action-synergy” (considering effect caused by incorporation of action1 and action2 or more actions) approach to flow modelling and modelling the effects.
Application includes: fault tree synthesis, HAZOP emulation, and fault diagnosis (Palmer & Chung, 2000).	HAZOP emulation in Batch plants.

Table 1. Comparison between SDG modelling and state-based modelling

Comparison between HAZID and CHECKOP

HAZID/conventional auto HAZOP tool for process plant	**CHECKOP/ prototype for batch plant**
Mature commercial product.	A prototype.
Used in continuously operating plants.	Used in batch plants or discrete operating plants.
Use SDG-based modelling.	Use State-based modelling.
	Flow modelling.
Representation information in the model includes only the tangible equipment information and fault path propagation information when fault or deviation happen in flow, temperature, vapour etc.	Representation information in the model includes not only the tangible equipment information, but the intangible information such as state change, operation/action and event/action sequence, time and assumptions associated with each step of operation.
Reads plant description file (PL file).	Reads not only plant description file but also operation instructions file.
PL file is used to build internal representation of the plant equipment, but not include the state because it assumes the state won't change. The state given is its normal state.	PL file is used to build internal representation of the plant equipment, including state because it considers the change of state. The first state given is "idle" state, in which valves are all closed, pumps are all off-line.
No operation instructions file.	operation instructions file is used to create a list of actions.
Use deviation guidewords to drive the engine to stimulate/generate fault path, guidewords are "more, less, no etc" and parameters/variables are "flow", "temperature", "vapour" etc.	Also use deviation guidewords to drive the engine but the guidewords is "No, early, late" and parameter is "action" and "termination", which are related to operations.
One output report is given in the format of tabular to summary of the cause, deviations and consequence etc and without suggested actions. The suggested actions are provided by the user at HMeeting, which is a PHP presentation of HAZID results.	One output report is given in the format of tabular to summary of the deviations/keyword, operation/action and consequence etc and without suggested actions because it is telling the effects of wrong action (no, too early, too late) Another output report (simulation trace report)is given which is to tell the changes of state as it is simulated by deviations.

Table 2. Comparison between HAZID and CHECKOP

Comparison between HAZID, CHECKOP and HAST

Comparison point	**CHECKOP and HAZID**	**HAST**
Model organization	Has model library to contain qualitative models of equipment units of process plants.	Has model library to contain qualitative models of equipment units of process plants. However, the models in the library include models designed for performing HAZOP analysis of the batch plants.
Model Independence	Models are totally independent with the environment that models are going to be applied.	Models are totally independent with the environment that models are going to be applied.
Plant decomposition	Plant is split into very elementary units such as valves, pumps etc.	Plant is split into very elementary units such as valves, pumps as well as procedure phases etc.
Model feature inheritance and implement tools	Each model inherits features from its parent model. Implemented by MBuilder.	Each model inherits features from its parent model. A taxonomy editor is used to implement this inherence feature.
Modelling Fault path propagation	Using mini SDG arcs to connect deviations, faults and consequences together.	Using mini fault trees to trace out root cause.
Model building tools	Models are created by MBuilder.	Models are created by a unit editor.
Model classification	Models are classified as “Unit Model” only.	Models are classified as “cause model”, “HAZOP model”, and “consequence model”. Particularly, it has a distinctive “mental model” for the HAZOP study members/analysts at the stage to think about deviations, main variables that could be used in the plant. Models of the same equipment unit for batch plant HAZOP analysis is differentiated by its phase. There are common phase models and specific phase models.
P&ID	SmartPlant to create P&ID.	A plant editor to create P&ID.
Model verification	Model testing/ model verification is carried out by Model testing tool before the model is being analyzed/utilized. Once going to the analysis process, the user can just wait for the HAZOP study	Do not verify the model before it is utilized and the HAZOP analysis is performed interactively for the user to test and confirm/sign off the result.

	output. That means the user does not interact with the system during the analysis.	
Suggested actions	No suggested actions are provided in the output of either HAZID or CHECKOP. HAZID complements this by allowing users to type their suggested actions at HMeeting, which is a php presentation of HAZID results.	No suggested actions are provided in the output. Users have to insert recommendations by themselves. However, it provides a facility to automatically insert available monitoring and protection devices by retrieving their information from the database to the HAZOP result.
application	HAZID is currently used only in continuously operating plants. CHECKOP is used in batch plants.	It can be used to do autoHAZOP on both process plants and batch plants.
Reducing output volume	Use a consequence evaluation system to reduce volume of output. .	Consider this in the HAZOP models by excluding any intermediate mini trees that only necessary to propagate the deviation.
Use of template	Use template to build models.	Does not use template to build models.
Reducing time and efforts	Time and effort is reduced by: Efficiency of the HAZID analysis , no interactive action during the analysis, the check/test of the model is performed before the analysis.	Time and effort is reduced by : • part of the validation of output is performed by HAST. • auto suggested information from the system by existing preventive and/or mitigatory interventions.

Table 3. Comparison between HAZID, CHECKOP and HAST

Comparison between HAZOPExpert and HAZID

HAZOPExpert	**HAZID**
Models are called HDG model (HAZOP-Digraph Model).	Models are called unit model based on SDG.
It has a model library to contain all models.	It has a model library to contain all models.
Models in the library are organized in a class hierarchy using the object-oriented framework of G2 . (Not sure if there is similar way of feature inheritance in the framework of G2 as that in HAZID).	Feature inheritance between models.
What kind of knowledge it represents: Generic knowledge represented by generic cause and effect models. Process specific knowledge, such as normal physical state of the material, and whether the material is corrosive, flammable, volatile or toxic.	HAZID SDG model represents generic knowledge of equipment item and the fault, deviation propagation information within and outside that model unit. The process specific knowledge like the properties of material is supplied by external software through Fluid Model System (FMS), which is a system that performs fault path validation by using predicates and functions. (McCoy et al., 1999 d)
The plant state is assumed to be steady state.	The plant state is assumed to be steady state.
It has a Graphical representation of models since it uses the real-time expert system shell G2 to do the reasoning work. The graphic tool is provided by this shell.	Does not have the feature of model graphic representation.
Pipe is also defined as a connection-class.	Does not have pipe-type models, because it requires that a unit must at least have a chamber to create a model for it (McCoy et al., 1999 c).
HAZOPExpert has a process-materials-cause procedure attached to HDG arcs to identify root cause of fault like “Leak in the tank or tank rupture or breakage due to presence of a corrosive process material”.	HAZID applies the runtime condition queries (RTCs) attached to SDG arcs, faults and consequences to check on the properties of the process materials using predictions and functions defined in the “Fluid Model System” as described in McCoy et al. (1999 d).

Table 4 Comparison between HAZOPExpert and HAZID

APPENDIX F A DEVIATION LIST

This deviation list is generated based on Crawley et al. (2000), "*HAZOP: Guide to best practice _ Guidelines to best practice for the process and chemical industries*".

Parameters	**Guidewords**	**Meaningful Deviations**
Boil up (2)	More	Boil+ (more boil up)
	Less	Boil- (less boil up)
Communication (6)	None	Comm0 (none communication)
	Part of	CommP (part of communication is achieved)
	More of	Comm+ (more communication)
	Less of	Comm- (less communication)
	Other	CommO (Substitute communication)
	As well as	CommA (additional communication)
Composition (4)	Part of	CompP (Part of composition is achieved)
	As well as	CompA (additional composition)
	More	Comp+ (more composition)
	Less	Comp- (less composition)
Contamination (1)	More	Cont+ (more contamination)
Conversion (2)	Higher	Conv+ (higher conversion)
	Lower	Conv- (lower conversion)
Flow (6)	More	F+ (more flow)
	Less	F- (less flow)
	No	F0 (no flow)
	Reverse	FR (reverse flow)
	Elsewhere	FE (flow goes elsewhere where it should not be)
	As well as	FA (flow is an additional activity which should not be)
Gas (1)	More	G+ (more gas)
Heat transfer (2)	More	HT+ (more heat transfer)
	Less	HT- (less heat transfer)
Lever (3)	Higher	L+ (high level)
	Lower	L- (low level)
	None	L0 (none level)
Liquid (1)	More	Liq+ (more liquid)
Maintain (3)	More	Main+ (more maintain)
	Less	Main- (less maintain)
	No	Main0 (no maintain)
MixingQuality (3)	Worse	M- (worse mixing)
	Better	M+ (better mixing)
	None	M0 (none mixing)

Operate (1)	No	O0 (no operation)
Phase (3)	Other	PO (substitute phase)
	Reverse	PR (reverse phase)
	As well as	PA (additional phase)
Pressure (3)	Higher	P+ (high pressure)
	Lower	P- (low pressure)
	Reverse	PR (reverse pressure)
Reaction (7)	Higher (rate of)	R+ (higher rate of reaction)
	Lower (rate of)	R- (lower rate of reaction)
	None	R0 (none reaction)
	Reverse	RR (reverse reaction)
	As well as	RA (additional reaction)
	Other than	RO (substitute reaction)
	Part of	RP(part of reaction is achieved)
Signal (1)	No	SN (no signal)
Speed (2)	faster	S+ (faster speed)
	slower	S- (slower speed)
Temperature (2)	Higher	T+ (high temperature)
	Lower	T- (low temperature)
Time (2)	Too early	The operation timing is too early
	Too late	The operation timing is too late
Vapour (1)	More	Vap+ (more vapour)
Viscosity (2)	More	Vis + (more viscosity)
	Less	Vis + (less viscosity)

APPENDIX G EXPLANATION OF NOTATIONS

‘P ‘ means ‘Pump’, eg., ‘P-0101A’ means ‘ a pump with the tag as P-0101A’;

‘V’ means ‘valve’, eg., ‘V001’ means ‘a valve with the tag as V001’;

‘PI’ means ‘Pressure Indicator’, eg., ‘PI100’ means ‘a pressure indicator with the tag as PI100’;

‘PRV’ means ‘Pressure Relief Valve’, eg., ‘PRV001PRV’ means ‘a pressure relief valve with the tag as PRV001PRV’;

‘ZEL’ and ‘ZLL’ mean ‘low level alarm, eg., ‘ZEL-59010’ means ‘a low level alarm with the tag as ZEL-59010’;

‘ZEH’ and ‘ZLH’ mean ‘high level alarm, eg., ‘ZEH-59010’ means ‘a high level alarm with the tag as ZEH-59010’;

‘FCV’ means ‘Control Valve’, eg., ‘FCV-59010’ means ‘a control valve with the tag as FCV-59010’;