DEVELOPMENT OF A COMPUTER-AIDED FAULT TREE SYNTHESIS METHODOLOGY FOR QUANTITATIVE RISK ANALYSIS IN THE CHEMICAL PROCESS INDUSTRY

A Dissertation

by

YANJUN WANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2004

Major Subject: Chemical Engineering

DEVELOPMENT OF A COMPUTER-AIDED FAULT TREE SYNTHESIS METHODOLOGY FOR QUANTITATIVE RISK ANALYSIS IN THE CHEMICAL PROCESS INDUSTRY

A Dissertation

by

YANJUN WANG

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Approved as to style and content by:

December 2004

Major Subject: Chemical Engineering

ABSTRACT

Development of a Computer-Aided Fault Tree Synthesis Methodology for Quantitative Risk Analysis in the Chemical Process Industry. (December 2004)

Yanjun Wang, B.En., Zhejiang University, Hangzhou, China

Chair of Advisory Committee: Dr. M. Sam Mannan

There has been growing public concern regarding the threat to people and environment from industrial activities, thus more rigorous regulations. The investigation of almost all the major accidents shows that we could have avoided those tragedies with effective risk analysis and safety management programs. High-quality risk analysis is absolutely necessary for sustainable development.

As a powerful and systematic tool, fault tree analysis (FTA) has been adapted to the particular need of chemical process quantitative risk analysis (CPQRA) and found great applications. However, the application of FTA in the chemical process industry (CPI) is limited. One major barrier is the manual synthesis of fault trees. It requires a thorough understanding of the process and is vulnerable to individual subjectivity. The quality of FTA can be highly subjective and variable.

The availability of a computer-based FTA methodology will greatly benefit the CPI. The primary objective of this research is to develop a computer-aided fault tree synthesis methodology for CPQRA. The central idea is to capture the cause-and-effect logic around each item of equipment directly into mini fault trees. Special fault tree models have been developed to manage special features. Fault trees created by this method are expected to be concise. A prototype computer program is provided to illustrate the methodology. Ideally, FTA can be standardized through a computer package that reads information contained in process block diagrams and provides automatic aids to assist engineers in generating and analyzing fault trees.

Another important issue with regard to QRA is the large uncertainty associated with available failure rate data. In the CPI, the ranges of failure rates observed could be quite wide. Traditional reliability studies using point values of failure rates may result in misleading conclusions. This dissertation discusses the uncertainty with failure rate data and proposes a procedure to deal with data uncertainty in determining safety integrity level (SIL) for a safety instrumented system (SIS). Efforts must be carried out to obtain more accurate values of those data that might actually impact the estimation of SIL. This procedure guides process hazard analysts toward a more accurate SIL estimation and avoids misleading results due to data uncertainty.

To

my parents
Wenkang Wang and Shuie Zhuo

my husband
Tao Yu

my sister
Fanjun Wang

And my son
Ray Yu

ACKNOWLEDGEMENTS

Looking back over my five years stay at Texas A&M, I have received so much help from so many people. This dissertation would not be possible without the contributions of many people.

I would like to express my deepest gratitude to my advisor, Dr. Sam Mannan. It has been my good fortune to have the opportunity of working with Dr. Mannan. He gave me the freedom to choose this project, which excited me during the past five years. I want to thank him not only for his financial support but also for his concern and support whenever I met difficulties. From him, I learned not only to be a PhD, but also a mature human being. His trust has been inspiring to me in both research work and personal life.

I am deeply grateful to Dr. Harry West. His contribution to this dissertation is unique. His broad knowledge and humorous stories have made working with him such a happy life. I really enjoy the time that I spent with him and appreciate all the help from him. I am thankful to Dr. Tom Teague as well for providing fundamental and insightful advice to my research and for serving on my committee. I want to acknowledge Dr. Jianer Chen for spending time reading my dissertation and for agreeing to serve on my doctoral committee.

Living and studying in such a unique and remarkable place as the Mary Kay O’Connor Process Safety Center (MKOPSC) made me feel warm just as at my home. All the staff and students of the MKOPSC deserve special appreciation. Dr. Rogers has been helping me with research and language problems. Donna Startz, Mary Cass, and

Shun Xu helped me in my daily work. Students bounce creative ideas about research work and help each other in personal life. Staying in this group has given me a feeling of family.

I want to thank the Department of Chemical Engineering at Texas A&M University for admitting me into the PhD program and for giving me an opportunity to study abroad and learn about a different culture.

I want to express my sincere appreciation to my family for their unconditional support in my graduate school career. I have been blessed to have Tao Yu as my husband. I want to thank him for his constant support and for working as a non-stopping machine during the writing of my dissertation. Special appreciation goes to my son, Ray Yu, for being such a good boy. He always calls me at the right time when I need rest and fun. I owe my parents, my parents-in-law, and my sister a debt of gratitude for their continuous encouragement and support. They have mailed us so much stuff that always reminds me of their love.

And last but not least, I am grateful to all my friends here. They have made my life here more joyful and happy.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

Page

Page

LIST OF FIGURES

Page

CHAPTER I

INTRODUCTION

1.1 Risk Analysis in the Chemical Process Industry

Risk is defined as a measure of human injury, environmental damage, or economic loss in terms of both the incident likelihood and the magnitude of the injury, damage, or loss (CCPS, 2000). Risk analysis involves the development of an overall estimation of risk by gathering and integrating information about scenarios, frequencies and consequences, and it is one major component of the whole risk management process of a particular enterprise. In the process of risk analysis, both qualitative and quantitative techniques can be used, as shown in Figure 1.1 (Krishna *et al.*, 2003).

This dissertation follows the style and format of the *Journal of Loss Prevention in the Process Industries*.

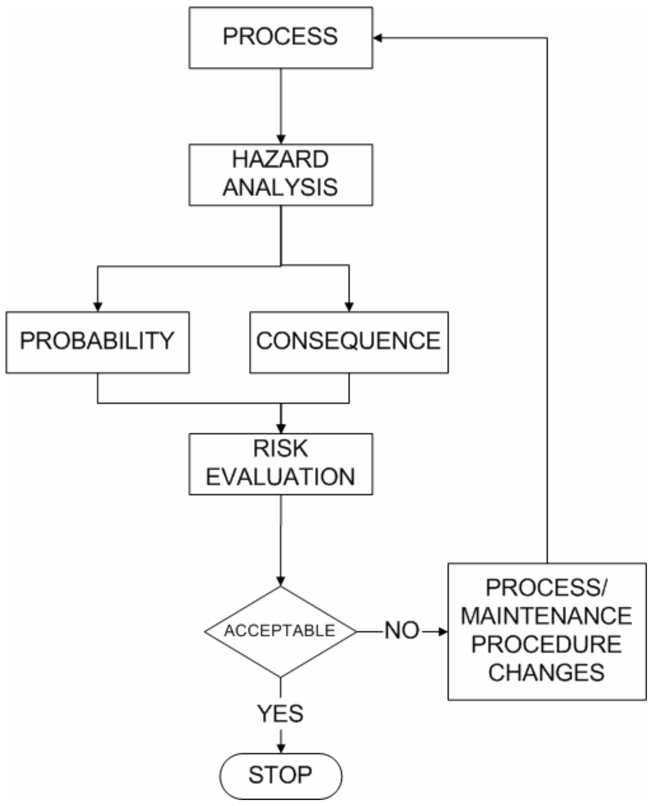


Figure 1.1 The process of risk analysis (Krishna *et al.*, 2003)

1.1.1 Risk Analysis – Why?

Along with the rapid progress of industrialization, the risk of incidents (such as fire, explosion, and chemical release) is increasing as well. It became increasingly recognized that there was a worldwide trend for losses due to accidents to rise more rapidly than gross national product (Lees, 1996). The results of a major industrial accident can be devastating, such as the Flixborough, England accident, which cost the lives of 28 people, the whole plant and many injuries (Crowl & Louvar, 2002); the

Bhopal, India accident, which killed more than 2000 civilians and injured 20,000 more (Crowl & Louvar, 2002); a massive explosion in Pasadena, Texas on Oct. 23, 1989, resulted in 23 fatalities, 314 injuries, and capital loss of over $715 million (Lees, 1996). These are extreme cases of major accidents in the process industry, but minor incidents are very common in the process industry, occurring on a day to day basis, resulting in many occupational injuries, illnesses, and costing the society billions of dollars every year.

Serious accidents receive a great deal of publicity, demanding the industry to remedy the situation. The investigation of almost all the major accidents shows that we could have avoided those tragedies with an effective risk analysis and safety management program. High-quality risk analysis is absolutely necessary for sustainable development. There has been growing public awareness and concern regarding the threat to people and to the environment from industrial activities, particularly those in which the process industry are engaged. This public concern has been translated to some extent into federal or state regulations. More than 50 federal regulations are dedicated or directly related to chemical process safety (Crowl & Louvar, 2002), among which are the Occupational Safety and Health Administration’s (OSHA) Process Safety Management (PSM) standard and the Environmental Protection Agency’s (EPA) Risk Management Program (RMP). In 1996, the Instrumentation, Systems, and Automation Society (ISA) approved ISA-S84.01 (ANSI/ISA-S84.01-1996). The American National Standards Institute (ANSI) adopted this standard in 1997, which requires that any US based instrumented systems developed after March 1997 must meet this standard.

Subsequently in 1998, the International Electrotechnical Commission issued IEC 61508 (IEC 61508, 1998). Safety instrumented systems designed thereafter must abide by this regulation with the exception of US installations that must follow ANSI/ISA-S84.01. This performance-based standard does not have prescriptive requirements, but it provides the overall risk-based safety life cycle model and guidelines for process industry to meet a desired safety integrity level (SIL).

1.1.2 Risk Analysis – How?

A principal theme of risk analysis is to ensure that we know what the hazards are before the system is allowed to operate. Risk analysis in the chemical process industry (CPI) can either be deterministic or probabilistic. The deterministic methods focus on consequence assessment (such as worst-case scenario analysis) while the probabilistic approaches consider both frequency and consequence. A variety of techniques have been used for risk analysis in the CPI including Safety Review, Checklist Analysis, Relative Ranking, “What-if” Analysis, Preliminary Hazard Analysis, Hazard and Operability Study (HAZOP), Failure Modes and Effects Analysis (FMEA), Fault Tree Analysis (FTA), Event Tree Analysis (ETA), Cause-Consequence Analysis (CCA), Human Reliability Analysis (HRA) (CCPS, 1992; Lees, 1996). Some of the techniques are more appropriate for “broad-brush” screening while others are more adept to detailed analysis. Some of techniques require special expertise such as FTA, ETA, CCA, and HRA. Brief overviews of each method are presented with their strengths and limitations hereafter except for FTA, which will be described in detail in Section 1.3.

Safety Review

Safety Review is probably the first hazard identification technique since the dawn of risk analysis. Safety Review involves a thorough and detailed examination of process design, plant conditions, operation practices, and maintenance activities in a particular facility; and is directed to find critical plant conditions or operating procedures that could potentially lead to an incident. It is normally used together with other hazard identification techniques such as Checklist Analysis and What-if Analysis.

Checklist Analysis

A Checklist Analysis examines a plant using a comprehensive list of questions, covering material properties, process design, and operation procedures. It is applicable to a project throughout its lifetime, and is often used to verify compliance with standards and practices. Checklist Analysis has been extensively used in CPI, and a large number of checklists have been published in the literature, a compilation of which can be found in Lees, 1996. The completeness of checklists depends on the knowledge of those who develop them. Many organizations use standard checklists throughout all stages of a project, however, there is often a tendency for them to be left on the shelf (Lees, 1996). Checklists are useful only if they are updated and improved regularly.

Relative Ranking

Relative Ranking are normally used to compare several design alternatives and identify the “best” option in the early stage of process design while opportunities for significant changes still exist. However, it can also be performed to an existing process to pinpoint the hazards of various aspects of process operation. A number of hazard indices have been developed for different purposes, among which the Dow Fire and Explosion Index (F&EI), the Mond Index and the Instantaneous Fractional Annual Loss (IFAL) Index are the most popular ones.

The Dow F&EI is the most widely used hazard index. It has gone through seven editions since it was first developed by the Dow Chemical Company in 1964. The analysis divides a plant into separate process units and assigns indices based on material properties, process conditions, areas of exposure, and other damage factors to derive the base maximum probable property damage (MPPD). Loss control credits are then applied to adjust MPPD to calculate actual MPPD.

The Mond Index is an extension of the Dow F&EI index. The hazard is assessed in a similar way to the Dow F&EI index but introduces additional considerations. Initial assessments of fire, explosion and toxicity are carried out for each process unit. Offsetting factors for prevention and protection measures are then assigned and combined with initial indices. Finally, an overall risk rating is derived from individual fire, explosion, and toxicity indices.

The IFAL index was originally developed for insurance assessment purpose by the Insurance Technical Bureau. It requires dividing the plant into blocks, and the contribution of each major item of process equipment is determined according to process factors, engineering factors, and management factors. Frequency and size of potential emissions and chance of ignition are used to determine damage.

Preliminary Hazard Analysis

Preliminary Hazard Analysis is required in the MIL-STD-882B System Safety Program. It is performed to identify hazards at an early stage of process design using resources from design criteria, equipment specifications, material specifications, and other sources of information. Preliminary Hazard Analysis is most often conducted in the conceptual design or R&D phase of a plant when little or no experience exists to predict or identify potential safety problems. It is useful to reduce or eliminate a potential hazard from the beginning of process design.

What-if Analysis

What-if Analysis is a typical inductive technique. It is probably one of the earliest hazard identification techniques. As its name indicates, the What-if Analysis requires a group of experienced people to review the plant by seeking to answer a series of questions starting with “what if”. There is little publication on What-if Analysis in the open literature. It is not inherently structured as other methods and is often used to complement a Checklist Analysis, which is called a What-if/Checklist Analysis.

Hazard and Operability Study

The HAZOP Study is usually applied when detailed process Piping and Instrumentation Diagrams (P&ID) are ready but not yet frozen. It focuses on specific points of the process or operation called “study nodes”. The basic concept is to take a full description of the process and to question every “study node” to discover what kinds of deviations from the intention of the design can occur and their potential causes, consequences and associated safeguards. This study is exhausted systematically by applying appropriate guidewords to each process parameter at each “study node”.

Failure Modes and Effects Analysis (FMEA)

FMEA is a systematic procedure in which each equipment failure mode is examined to determine its effects on the system and classify it according to severity and criticality. FMEA is an inductive method oriented toward equipment rather than process parameters. All of the failure modes for each item of equipment are tabulated with their effects, safeguards, and related actions listed. An FMEA is especially useful to identify single failure modes that lead to an incident directly, while it is not powerful to identify combinations of equipment failure and human errors as risk contributors.

Event Tree Analysis (ETA)

An event tree is an inductive reasoning process that starts with an initiating event followed by the binary success or failure of subsequent safeguards, human responses, and other safety measures to determine its possible outcomes. It is especially suitable to find possible outcomes of particular initial events and their respective probabilities with the data for initial events and subsequent protections and procedures.

Cause-Consequence Analysis

Cause-Consequence Analysis is actually a combination of ETA and FTA. A cause-consequence diagram is constructed by defining a critical event and then developing the causes and consequences of this event using a fault tree and an event tree, respectively. It is a graphical representation of the relationship between the incident outcomes and their primary causes. Although Cause-Consequence Analysis incorporates features from FTA and ETA, it is not commonly used since the cause-consequence diagram for a fairly simple process is detailed and somewhat cumbersome. It is mostly used when the logic model for the concerned event is simple enough for a graphical display, which limits its application in the CPI.

Human Reliability Analysis (HRA)

HRA is a relatively new area of hazard identification, however, it is getting increasing attention nowadays. A typical HRA identifies the factors that will affect the performance of operators and the consequences associated with human errors. It involves task analyses that consider factors such as job characteristics, personal skill, knowledge, and stress level. Although some other techniques like FTA, HAZOP study, *etc* can incorporate some consideration of human errors; a HRA is dedicated to analyze human factors in a process systematically. It is usually performed to supplement other hazard identification techniques.

1.1.3 Quantitative Risk Analysis

Risk analysis has been a work of art rather than a science. The evaluation and acceptance of risk depend to some extent on the expertise and perception of the analysts. On one hand, modern chemical plants have become very complex and highly integrated, processing large volumes of materials and operating at extremes of pressure and temperature to achieve optimal performance. Advanced information technology ties units together in large complex system with short time constants, allowing little or no time for the correction of mistakes or for counteracting effects due to unforeseen circumstances. All these factors have made it more difficult to analyze and assess the risk associated with a process facility. On the other hand, the development of industrialization, the concentration of people, dangerous chemicals, energy, information and other values is increasing, which may significantly increase the severities of incidents. Hence, there has recently been an increased consciousness both in academia and industry to develop advanced risk analysis methodology to predict and reduce risk in the process industry. Consequently, the application of quantitative risk analysis (QRA) has been widely spreading since being adapted to the particular needs of the process industry during the 1980s (Arendt, 1990).

QRA involves a numerical evaluation of incident consequences and frequencies by gathering data and information, and their combination into an overall measure of risk. It is actually an analysis strategy instead of a well-defined analysis methodology. QRA can be presented in many forms, all of which are trying to answer the following questions in a quantitative manner:

What can go wrong?

How often will it happen?

What are the consequences if it happens?

The very first step of a QRA is hazard identification. Any methods or combination of methods described above can be employed to identify existing hazards in a system. To answer “how often will it happen?”, we need to calculate the probability of each incident scenario. Historical data, FTA, and Markov Modeling may be applied to determine the probability. Historical data are primarily limited to the use of failure data of similar systems while FTA is based on the failure rate of components and processes. Markov modeling is an advanced technique, which is especially useful when there is a dynamic system and time dependent failures. However, many engineers do not feel comfortable with Markov modeling and its fundamental mathematical background. A variety of source models, dispersion models, and effect models may be used to answer "what are the consequences if it happens?". A more detailed description can be found in CCPS, 2000 and Crowl and Louvar, 2002.

1.2 Fault Tree Analysis

FTA was first developed in 1961 at Bell Telephone Laboratories for missile launch control reliability during the Polaris project, and has been extensively used in reliability studies in the nuclear and aerospace industry. As a powerful tool for risk assessment, FTA has long been adapted for application in the chemical process QRA to predict the likelihood of hazardous incidents and identify major risk contributors (CCPS, 2000).

1.2.1 Fundamentals

In contrast to ETA, FTA begins with the top event (incident) and continues by deductive reasoning through all the intermediate events to primary failures and initiating events. FTA postulates the occurrence of the undesired incidents (termed top event). All necessary and sufficient direct causes of the top event are identified and linked to the top event using Boolean logic (AND, OR). This procedure continues until system boundary, primary failures, human errors, or environmental conditions (termed basic events) are reached. This technique is especially powerful to enumerate combinations of equipment failures, human errors, and external conditions that will lead to the undesired top event. Minimum Cut Sets are the smallest combinations of basic events that will lead to the top event. With failure rate data for basic events, probability of the top event can be estimated. Figure 1.2 shows the standard symbols used in fault tree representation. Figure 1.3 is a sample fault tree for the top event “fire occurs in the storage tank”.

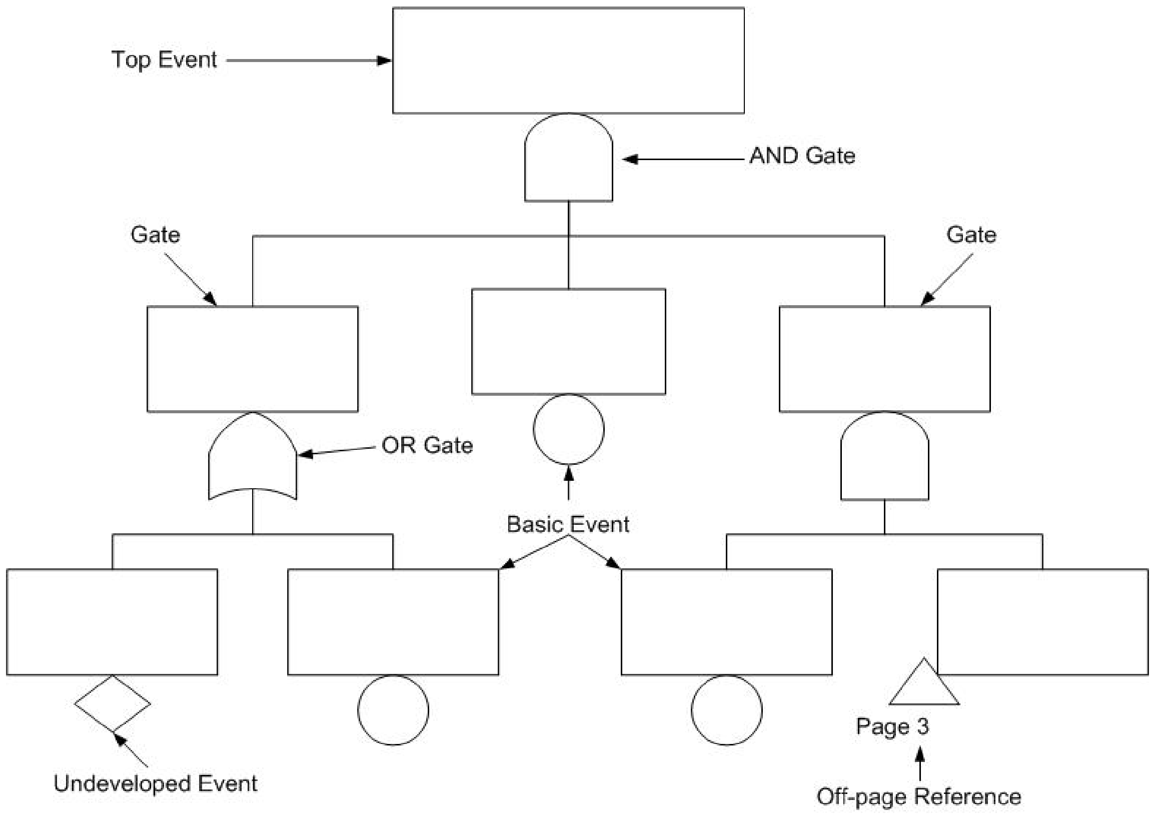


Figure 1.2 Standard fault tree symbols

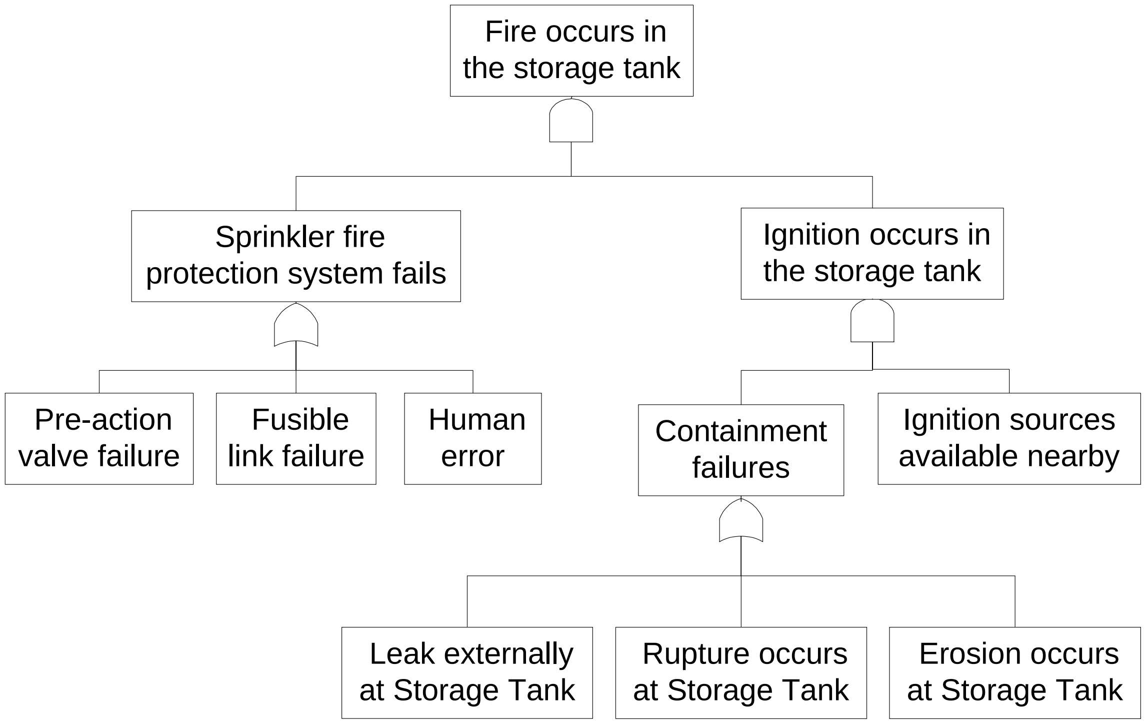

Fire occurs in the storage tank
Sprinkler fire protection system fails
Ignition occurs in the storage tank
Pre-action valve failure
Fusible link failure
Human error
Containment failures
Ignition sources available nearby
Leak externally at Storage Tank
Rupture occurs at Storage Tank
Erosion occurs at Storage Tank

Figure 1.3 A sample fault tree for "fire occurs in the storage tank"

Two principal logic gates, AND and OR, are used in fault tree. Mathematical fundamentals behind them are simple and easy to understand. Basic events are assumed to be statistically independent unless treated specially. Given this assumption, the probability of event A OR event B is calculated as P(A OR B)=P(A)+P(B)-P(A AND B), and the probability of event A AND event B is P(A AND B)=P(A)P(B). However, this assumption does not restrict FTA from considering dependent failures. Minimum Cut Set analysis can detect common cause failures, and a beta factor can be used for some dependent failures (Summers, 1998).

1.2.2 Fault Tree Construction

The applications of fault trees have been increasing gradually since fault tree technique was invented in the 1960s. However, the construction of fault tree models remains an art rather than a science. There is no standard taxonomy for fault tree construction. However, some general guidelines have been extracted to assist engineers to produce high-quality fault tree models. These rules have been examined and listed in the fault tree handbook as ground rules and procedural rules (NUREG-0492, 1998).

Ground Rule 1: Write the event statements precisely about what the fault is and when it occurs. Do not abbreviate words that might lead to ambiguous meaning.

Ground Rule 2: If a specific fault consists of a component failure, classify the event as a "state-of-component fault". Otherwise, classify it as a "state-of-system fault".

Procedural Rule 1 (No Miracles Rule): If the normal functioning of a component propagates a fault sequence, then it is assumed that the component functions normally. In other words, it is treated as a certain event. If the normal functioning of a component prevents the propagation of a fault sequence, AND logic is used to consider its failures.

Procedural Rule 2 (Complete-the-Gate Rule): All inputs to a particular intermediate gate should be completely defined before further analysis of any one of them is undertaken. Each child must be an immediate and direct cause of its parent. Each child must be either sufficient in case of an OR gate and necessary in case of an AND gate.

Procedural Rule 3 (No Gate-to-Gate Rule): All gate inputs should be properly defined, and no gate should directly feed into another gate.

1.2.3 Strengths and Limitations

The strengths of FTA are straightforward. FTA explicitly expresses how equipment failures, operator errors, and external factors lead to system failures in a graphical representation that is easy to understand. It identifies major risk contributors and combinations of primary failures and human errors that lead to an undesirable incident. In particular, it quantifies benefits associated with process safe guards and compares risk-reduction measures quantitatively in terms of safety. It is a structured methodology and well documented, ready to modify according to system changes.

However, despite all its advantages, there are some limitations in its application that needs to be mentioned. FTA is a “snap shot” of the system, and a component in a fault tree is categorized into two states: working or fail. Intermediate states or partial failures are not considered, thus it is not suitable for dynamic analysis.

1.3 Statement of the Problem

FTA is one of the best tools available for a comprehensive reliability study. Much can be gained by the application of FTA in the CPI. Unfortunately, the current application of FTA in the CPI is somehow limited (Andrew, 1980). Then what is the enemy within?

One major barrier with regard to its application in the CPI is fault tree construction. Processes, materials, equipment, and control mechanisms are much more diverse in the CPI than in the nuclear industry. The method does not itself assure that all failure modes have been considered. It requires specially trained and skilled practitioners who are familiar with the methodology and understand the process under analysis to ensure the completeness and correctness of the analysis. Fault trees for a reasonably complicated process will be enormous and needs overwhelming expert time, sometimes measured in years, to complete. Due to the human labor and time required, the cost is relatively high compared to other methodology. Subjective factors are extensively involved in fault tree construction. Fault trees developed by different individuals for the same process are usually different in structure, and may thus lead to different results.

In order to calculate the probability or failure distribution of top events, probabilities or failure distributions of all basic events are required. Dearth of failure rate data and the large uncertainty associated with the data is a considerable problem in the application of FTA. With data collection and exchange efforts from both the government agencies and industry, this problem is slowly being ameliorated.

While there are well-developed commercialized software packages for fault tree evaluation, no satisfactory methodology has been published for fault tree synthesis, especially when control loops are encountered. The availability of a computer-based analysis methodology that simplifies the system understanding process and takes a structured approach to develop fault trees would greatly increase the attractiveness of FTA techniques in the CPI.

1.4 Objectives

From the above description, it is apparent that fault tree construction is one major barrier with regard to its application in the CPI. To overcome this difficulty, a computerizable fault tree synthesis methodology is developed in this research. The basic idea is to capture the cause-and-effect logic between equipment behaviors, human responses, and environment factors around each item of equipment directly into mini fault trees. Special fault trees are developed to model the behaviors of special features such as control loops, trip systems, and bypasses.

The objectives of this research include:

- Development of fault propagation model for components
- Development of cause-and-effect model for special features in the CPI
- Development of a computerizable fault tree synthesis methodology
- Design of database structure to be used in this methodology

- Validation of the methodology by comparing the results with published examples

Prototype computer codes will be programmed to demonstrate the correctness and applicability of this methodology using published examples in the literature.

The ultimate goal of this research is to standardize quantitative risk analyses through a computer package and guide decision makers toward more formal and more cost-effective solutions for the allocation of resources to prevent incident and reduce risk.

1.5 Organization of the Dissertation

This dissertation introduces a framework to model fault propagation in process plants and develops a structured computerizable fault tree synthesis methodology for QRA in the CPI.

Chapter I introduces a broad overview of the importance of risk analysis and current practices in the process industry. The second part of Chapter I explains the fundamental, pros and cons of fault tree techniques. The difficulties faced in its application in the CPI are discussed followed by objectives of this work. Chapter II presents a detailed literature review of current research status of computer-aided fault tree synthesis. Preliminary work in this area has been categorized as digraph-based methods, mini fault tree based methods, rule based methods, loop based methods, and other methods based on the fundamental modeling techniques that have been used.

Applicability and limitations of each method have been discussed respectively. A description of the overall methodology is provided in Chapter III in detail, as well as the fault propagation model. Top event identification, mini fault tree generation, fault tree generation, tree rationalization, and tree simplification are discussed subsequently. The discussion of fault propagation model is divided into two parts. The first part is dedicated to component modeling while the second one focuses on special feature modeling. Database structure is presented in Chapter IV that has been used to represent component modeling and special feature modeling in the demonstration computer codes. The public database and the specific database are discussed respectively. Chapter V describes the interface and flow charts of the demonstration program. Algorithms of major functions are presented with diagrams. Chapter VI discusses two case studies. One is related to fire event identification and the other one is a cooling-down process of hot nitric acid. Results are compared and validated with published fault trees. Chapter VII addresses data uncertainty problems in risk analysis and proposes a practical and efficient procedure to deal with data uncertainty in determining Safety Integrity Level (SIL) for a Safety Instrumented System (SIS) and identify the inputs that may eventually lead to a change in the estimation of SIL.

CHAPTER II

BACKGROUND

2.1 Overview

While commercialized computer codes are available for the evaluation of fault trees, the synthesis of fault trees is a barrier with regard to its application in the process industry. Fault trees are generated manually in most cases. Although general guidelines are available for fault tree synthesis, learning how to create fault tress is akin to learning to ski: some people never quite make it (Evans, 1981).

The great diversity of equipment, hazardous materials, chemical processes, and control mechanisms in the CPI has made the task of generating fault tree models difficult. Subjective factors are involved extensively in the manual construction of fault trees, and different analysts will almost certainly generate different fault tree models for the same processes. Conventional manual construction of fault trees is a difficult and time-consuming task, which requires trained analysts who understand the methodology and the system under study as well. For instances, fault trees in the Wash 1400 report have taken many man years to complete. On the other hand, industrial development, public awareness, and more rigorous regulation are demanding manufacturers to prove that they have identified major existing hazards and adopt appropriate and adequate prevention and protection measures. As a result, many researchers have been working on computerized fault tree synthesis during the past two decades, and a variety of methods have been published. The state of research work on computer assisted fault tree construction has been reviewed and presented (Allen & Rao, 1980; Carpignano & Poucet, 1994.)

2.2 Preliminary Research

2.2.1 Digraph Based Methods

A digraph (directed graphs) is a graphical representation that explicitly describes the cause-and-effect relationship within and among equipment, processes, human interactions, and environment in a chemical plant. It consists of nodes and arcs labeled with a signed numerical value. A node represents a process variable, human interaction, an environment event, or a specific fault, and an arc indicates the influence of the first node on the second one. A number is assigned depending on the relative direction and magnitude of the deviation of the second variable related to the first. This number essentially indicates a qualitative relationship between process variables. A moderate deviation is represented by “1” while a large deviation is reflected by “10”. If a deviation of the first variable results in a deviation of the second variable in the same direction, the sign of the number is positive, otherwise, it is negative.

Lapp and Powers (1977, 1979) proposed their well-known algorithm based on digraphs. In their work, the modeling of a single component requires the identification of all process variables that are transferred through it and their relative gains. A digraph model is manually created for the system under consideration. Loops, particularly feedback and feedforward loops, are managed by introducing subtrees (called operators) whenever nodes on a loop are encountered. These operators have been refined further by Andrews and Brennan (1990), Lambert (1979), and Chang and Hwang (1992). Given the system digraph and a list of the feedback and feedforward loops, it is possible to construct fault trees. To address the problem of identifying all the loops in a digraph model, Chang *et al.* (1997) proposed a loop identification and classification algorithm for this methodology, which has yet to be tested against realistic processes. However, manual construction of system digraphs is tedious and laborious. Creating a satisfactory digraph representation of the system under study requires an effort equivalent to or sometimes greater than the effort required for developing fault tree models itself. No algorithm has been published for the automation of digraph generation for a chemical process. This impacts the significance of digraph-based algorithms. In addition, the digraph model for a chemical process is not unique in the sense that the analyst will have different taste regarding failure modes and effects of the components within a chemical process and the interaction among them. In particular, this methodology works based on a node-by-node algorithm. Nodes especially on any loops are treated numerous times. This is not only a waste of the computation time but also results in many repeated events which compromise the readability of the generated fault trees.

Kuo *et al.* (1997; 1998) developed prototype software called "IHAS (Integrated Hazard Analysis System)". FTA, ETA, and HAZOP Study can be performed with component digraphs, which are connected according to the P&ID automatically to obtain the system digraph. This system digraph is then utilized to locate loops and construct fault trees. This algorithm is essentially based on Lapp & Powers' work (Lapp & Powers, 1977). However, the digraph models of a component are context dependent (Carpignano & Poucet, 1994). They depend not only on the equipment itself but also on the particular process within which it is used.

Powers and Tompkins (1974) suggested generating fault trees from input/output models of components. These models incorporate the functional relationship between inputs and outputs of equipment. The effects of equipment failures have been taken into account as well. However, this approach has not yet been developed to be applicable to complex processes.

Bossche (1991a; 1991b) used the "relations" between process variables and component states to model fault propagation in the process industry. These "relations" are essentially similar to digraph. Signed numbers called "propagation factors" are used to indicate the magnitude and direction of influences between process variables. Component states and conditional variables are attached to each relation pattern. An intermediate causal tree is utilized first to model bi-directional flow and fault propagation, and then this causal tree is adjusted according to feedback and feedforward controls. Final fault trees are extracted from the intermediate causal tree. This methodology has been adapted for real-time fault diagnosis in the chemical process, whose capability is yet to be proved (Kocza & Bossche, 1997).

2.2.2 Mini Fault Tree Based Methods

Fussell (1973) developed a formal fault tree synthesis methodology for electrical systems, from which failure transfer functions were used to model failure modes. In his synthesis tree model, the system-independent component failure transfer functions were utilized together with the system schematic diagram and associated system boundary conditions to construct fault trees. This methodology has been applied to simple electrical systems. Though claimed in his paper that the method extends to all fault tree construction beyond the area of electrical systems, problems arise in the preparation of failure transfer functions and fault events.

Fussell’s algorithm was later modified and improved by Taylor to manage loops in the system (Taylor, 1982). For each item of equipment, a set of functional and failure modes is defined, equations are written that describe component performance under functional and failure states, which is reflected by equation bigraphs. Fault propagation between the components is indicated by the arrows between the bigraphs, from which a cause-and-effect graph is generated. Equation bigraphs together with the cause-and-effect graph are then used to draw signal flow graph. Bigraphs, signal flow graphs, and state transition tables are converted to decision tables. Instead of being just logical, the decision tables represent the qualitative gains between the input and output variables within a component. The decision tables are then translated into mini-trees stored in a component library. Similar to the idea suggested in the digraph methodology, special models are applied when control loops are encountered. This fault tree construction is on a component-by-component basis, which is a major advantage over the node-by-node methodology. However, the generation of component models is a procedure of eleven major steps, which involves the development of equations for all functional and failure states, equation bigraphs, cause-and-effect graphs, signal flow graphs, input/output event table, state transition tables, decision tables, and a mini fault tree library, which restricts the methodology from any realistic application.

Poucet (1990) developed an interactive computer program, which requires extensive computer and user information exchange. This method requires modular component models and transfer logic models. This algorithm is a two-step procedure: A “skeleton” fault tree is built first, and then expanded to incorporate component models. Modular component models are stored in a permanent database and are used to give a general description of the functionalities of components and the component failure modes and their effects. Transfer logic models, similar to decision tables, are used to trace fault propagation from one component to another according to the P&IDs. Transfer logic models contain the physical and behavioral descriptions of components in their various functioning or failure modes, expressed as logical expressions. As acknowledged by the authors, users have to intervene in the construction process to avoid the process going to irrelevant branches. Further work has not been seen to remedy this problem.

A series of papers have been published by Hunt *et al.* based on componentistic mini-fault tree models (Kelly & Lees; 1985a; 1985b; 1985c; Hunt *et al.*, 1993a; 1993b; 1993c; 1993d; 1993e). A large system is decomposed into items of equipment, where streams flowing in and out are assigned as dummy heat and tail units. Fault tree synthesis process is performed by tracing the fault propagation from one mini fault tree to another. Their mini-fault trees are generated from propagation equations, event statements, and decision tables. A propagation equation reflects qualitative relationship between inputs and outputs of equipment under normal and fail conditions. Event statements record the effects of all the failure modes and external conditions within a component. Decision tables are employed to indicate conditional changes of relationship among process variables. Mini-trees are also used for special functional and structural situations such as headers and dividers, process controls, and trip loops. The methodology was developed originally for continuous processes, and it is ready to be adapted to embrace batch or sequential processes such as startup and shutdown operations. Reverse flow and two-way propagation have been considered. Since mini-fault trees are more suitable for an automatic framework, this methodology includes top event model library, unit model library, and mini fault tree library. The requirements for storage and computational complexity are much higher compared to other methods. Although the method promises a reasonable level of automation, no help is provided for the component modeling and the recognition of system functionalities and structures, which remain the responsibility of the analysts. No further publication has appeared on this work subsequently since 1993.

2.2.3 Rule Based Methods

Elliott (1994) presented a knowledge-based approach to build a fault tree from a reliability block diagram. User interactions were required to input the knowledge-based if/then rules, and the rule interpreter was used to execute fault tree construction rules.

This methodology attempts to incorporate features of artificial intelligence. A fault tree reduction algorithm has been considered to minimize the size of fault trees, but the substantial work to create if/then rules is left to the user. In addition, how to handle control loops is not clear in this methodology.

2.2.4 Loop Based Methods

Shafaghi *et al.* (1984a, 1984b) proposed a systematic approach to construct fault trees by decomposing the plant into a set of control loops. Instead of decomposing a chemical plant into functional units like most other methods, they sought to divide it according to control loops. In their method digraphs were created manually for each control loop and for the plant as a whole. Control loops are treated as the skeleton of the plant. The interconnection and fault propagation among loops were used to construct intermediate trees and then combined with component digraphs to generate the final fault trees. As mentioned above, digraph models for the whole system and each control loop must be created manually, which remain unsolved. Moreover, interaction and fault propagation between control loops are modeled by connection tables between control loops. For some cases such as reactors and distillation columns, control loops are coupled together, and the connections between them are not necessarily clear and obvious. Equipment with coupled process controls is almost always critical in safety studies. The connection tables for those items of equipment could be difficult to generate and suspect of human omissions.

Another similar algorithm was later proposed by Chang and Yuan (Chang & Yuan, 2000). It seeks to modularize each control loop first to establish its unit model and then to construct fault trees from a reduced system that is constituent of control loops. This method is essentially similar to the Shafaghi *et al.* (1984a, 1984b)'s method. The drawbacks mentioned above for control loop based methods still remain unsolved.

2.2.5 Other Methods

Caceres and Henley (1976) proposed an algorithm to create fault trees from a process block diagram. However, this methodology is unable to manage feedforward loops, which reduces its value to the CPI.

Camarda *et al.* (1978) proposed an efficient simple algorithm for fault tree synthesis from reliability graphs. In their method, reliability graphs were employed to identify all the minimal path sets and then transformed into equivalent fault trees. Reliability graph is itself a tool to identify hazards. Development of reliability graph is not necessarily obvious and straightforward, especially for complex chemical processes.

De Vries (1990) developed a quantitative methodology to generate fault trees from a schematic diagram applicable only to electrical circuits.

Another fault tree construction methodology was presented by Kumamoto and Henley (1995). A terminology of "flow" was introduced to indicate any material, information, energy, activity, or phenomenon that can propagate through the system. Three values: flow rate, generation rate, and aperture are used to characterize each "flow". Equipment is categorized according to these three attributes, and fragmented semantic networks are developed for each type of equipment. A semantic network for the whole system is then produced from these fragments together with a system schematic. Fault tree generation is guided by recursive event development rules, which are generated from flow rate, generation rate, and aperture attributes. This method appears to be more advantageous than digraph based method in the sense that it can indicate relations other than digraph gains among process variables, equipment failures, human activities, and environmental conditions. However, equipment behavior, component interaction, and fault propagation are defined based on “flow” and its three attributes. Problems arise in the preparation and understanding of these flows and their attributes. Treatment of process control loops is not yet stated explicitly in this work.

2.3 Summary of Chapter II

A comprehensive review of the research work done on computer assisted fault tree synthesis has been presented in this chapter. Efforts have been carried out to overcome the problem of traditional manual construction of fault trees to facilitate the application of fault tree technique in the CPI. However, none of the above methods has yet been extensively used in practice. Strengths and limitations of each method have been discussed. Some methods are valid only for electrical systems, some are not applicable to complex systems, and many involve generation of a complicated system model, such as digraphs, decision tables, transition tables, and connection tables between control loops, which is equivalent to or sometimes even more substantial than generating fault tree models itself. In particular, fault tree logic to model the behavior of process loops and control loops are one of the most challenging parts and have been the subject of many literature debates for years. No entirely satisfactory algorithm has been published for fault tree synthesis, especially when control loops are encountered.

This research is aimed at developing a practical and efficient computer-aided fault tree synthesis methodology. Cause-and-effect models are generated automatically by the program from functional digraph tables, failure modes and effects tables and external events tables. Special features such as process controls, trip systems, bypasses, pressure relief systems, *etc*. are handled through special cause-and-effect models. This methodology works directly from process P&IDs, thus it avoid the tedious working of generating a complicated system modeling. The tree structure is expected to be concise and easy to understand. Details of this proposed methodology are presented in Chapter III.

CHAPTER III

METHODOLOGY*

3.1 Overview

The basic idea of the proposed methodology is to capture the cause-and-effect logic among equipment behaviors, human responses, and environment factors around each item of equipment directly into mini fault trees. Mini fault trees are employed to model the local cause and effect relationship of components. Special fault trees are developed to model the behaviors of special features such as control loops, trip systems, pressure relief systems, and bypasses. This methodology starts directly from a process block diagram, and fault trees created by this method are expected to be concise and easy to understand.

In order to computerize the entire procedure, the block diagram has to be read to represent the system topology. Configuration information, initiating states, and physical boundaries are embedded in the system block diagram.

---

*Part of this chapter is reprinted from *Journal of Loss Prevention in the Process Industries*, Vol. 15, No. 4, Wang Y., T.L. Teague, H.H. West, & M.S. Mannan, A New Algorithm for Computer-Aided Fault Tree Synthesis, 265-277, Copyright (2002), with permission from Elsevier.

*Part of this chapter is reprinted from *Journal of Loss Prevention in the Process Industries*, Vol. 16, No 5, Wang Y., W.J. Rogers, H.H. West, & M.S. Mannan, Algorithmic Fault Tree Synthesis for Control Loops, 427-441, Copyright (2003), with permission from Elsevier.

Once the top events have been selected, the algorithm proceeds to find the corresponding cause-and-effect models in the top event library to trace the top event down to deviations in the process variables, or prompt for user interaction if it does not exist in the library. Then intermediate events are developed, and relevant cause-and-effect models are generated directly from the digraphs, failure mode table, and external event table until a system boundary or primary events are reached. Control loops and special features are managed by their corresponding fault tree models (template trees), which will be addressed in detail later. Serial consistency is ensured during fault tree synthesis procedure, and parallel consistency is checked after the initial tree is constructed. After that, a simplification procedure is performed to make the fault tree concise and easy to read. Finally, user verification is required to ensure the completeness and correctness of the final fault tree.

Figure 3.1 is a simplified flow chart of the fault tree synthesis methodology, of which some steps are addressed in detail in the following sections and Chapter IV and Chapter V.

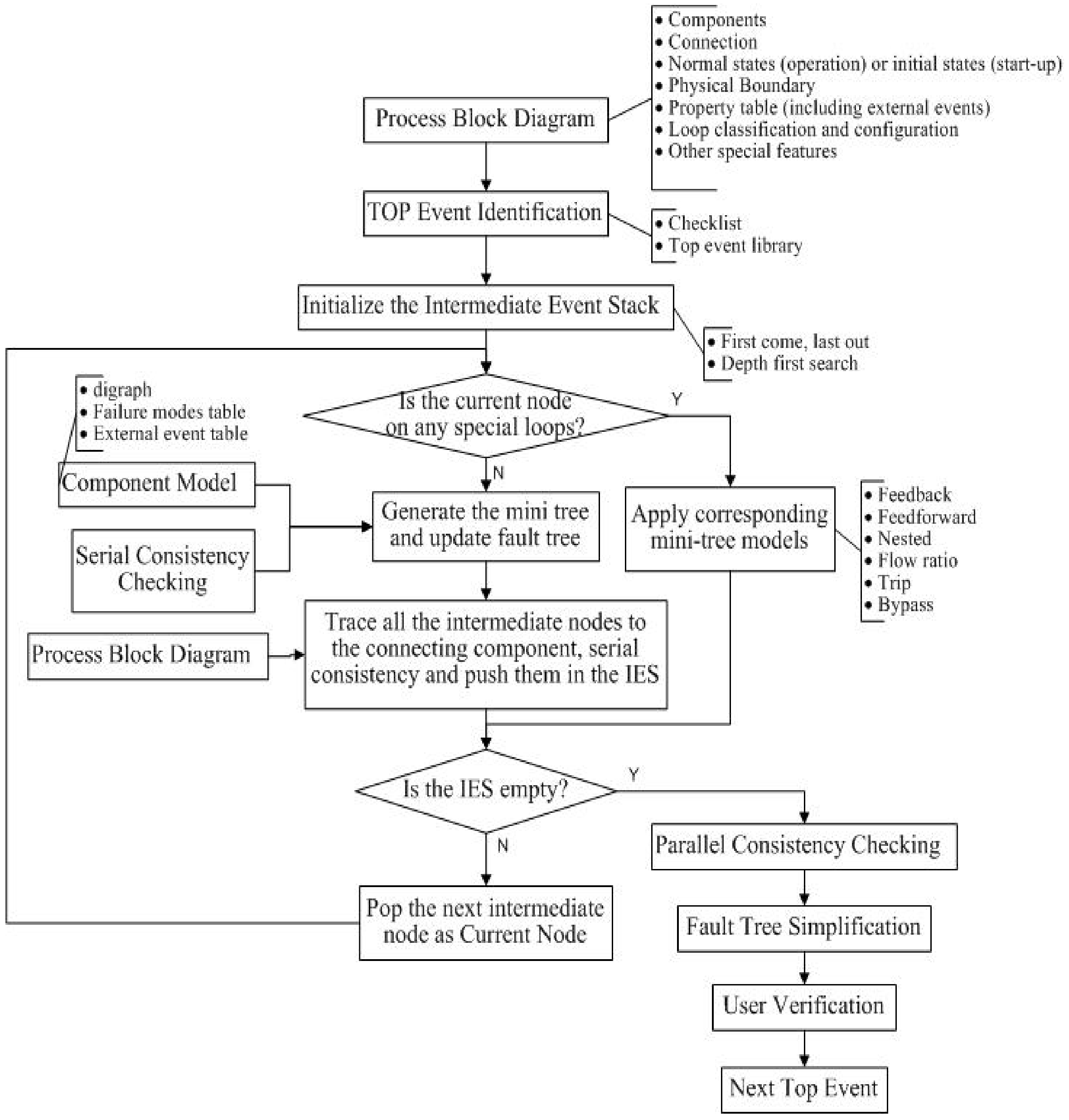

Process Block Diagram
• Components
• Connection
• Normal states (operation) or initial states (start-up)
• Physical Boundary
• Property table (including external events)
• Loop classification and configuration
• Other special features
TOP Event Identification
• Checklist
• Top event library
Initialize the Intermediate Event Stack
• First come, last out
• Depth first search
Is the current node on any special loops?
Y
N
• digraph
• Failure modes table
• External event table
Component Model
Serial Consistency Checking
Generate the mini tree and update fault tree
Apply corresponding mini-tree models
• Feedback
• Feedforward
• Nested
• Flow ratio
• Trip
• Bypass
Process Block Diagram
Trace all the intermediate nodes to the connecting component, serial consistency and push them in the IES
Is the IES empty?
Y
N
Parallel Consistency Checking
Fault Tree Simplification
User Verification
Next Top Event
Pop the next intermediate node as Current Node

Figure 3.1 A simplified flow chart of the fault tree synthesis methodology

3.2 Top Event

A top event is defined as an undesirable event or incident. The first step of FTA is to identify the potential top events. This is normally achieved by hazard identification analysis. The identification of top events is critical but often underestimated. Undesirable incidents can be categorized into three classes as human impacts, environmental impacts, and economic impacts (CCPS, 1989). Typical top events include fires, explosions, toxic materials, human injury, environment contamination, property damage, poor product yield/quality, legal liability, *etc.* It is difficult to automate top event identification. In this work, simple checklists are employed to assist analysts to identify potential incidents. However, this approach is not exhaustive and does not represent complete hazard identification. It is the responsibility of the user to ensure its completeness and correctness. Users can also specify their perceived incidents directly.

Checklists and corresponding fault tree models for fire, explosion, and toxic releases have been developed in this project. For instance, the basic idea in fire event identification is to examine every possible location for the presence of fuel, oxygen, and ignition sources. If the area is not classified as intrinsically safe, users can select possible ignition sources in a particular facility from a list of typical ignition sources in the chemical process industry, or add perceived ignition sources manually. If the area is classified as intrinsically safe, then ignition sources are normally related to maintenance activities and operation procedures that might render the area not intrinsically safe. For intrinsically safe areas, users are responsible to enter possible ignition sources. Figure 3.2 is a sample checklist for fire event identification in this algorithm.

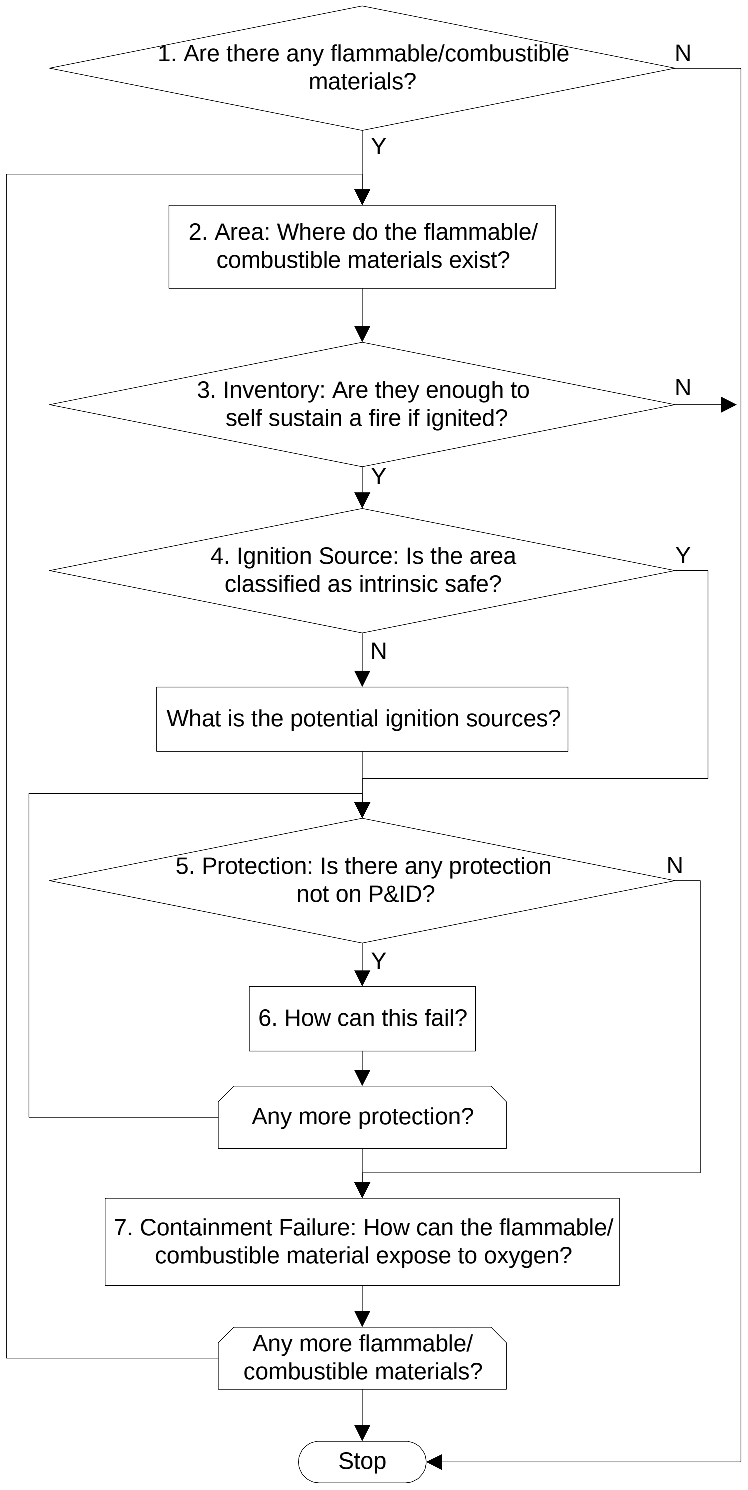

1. Are there any flammable/combustible materials?
N
Y
2. Area: Where do the flammable/ combustible materials exist?
3. Inventory: Are they enough to self sustain a fire if ignited?
N
Y
4. Ignition Source: Is the area classified as intrinsic safe?
Y
N
What is the potential ignition sources?
5. Protection: Is there any protection not on P&ID?
N
Y
6. How can this fail?
Any more protection?
7. Containment Failure: How can the flammable/ combustible material expose to oxygen?
Any more flammable/ combustible materials?
Stop

Figure 3.2 A sample checklist for “fire” event

This checklist can run repeatedly for any flammable/combustible materials that might be present. The answer to all the questions are simply yes/no or in AND/OR format. The corresponding resultant fault tree model for a fire event is shown in Figure 3.3. Ignition sources for intrinsically safe areas and not intrinsically safe areas are different. Users may revise the fault tree models for top event generated from checklists.

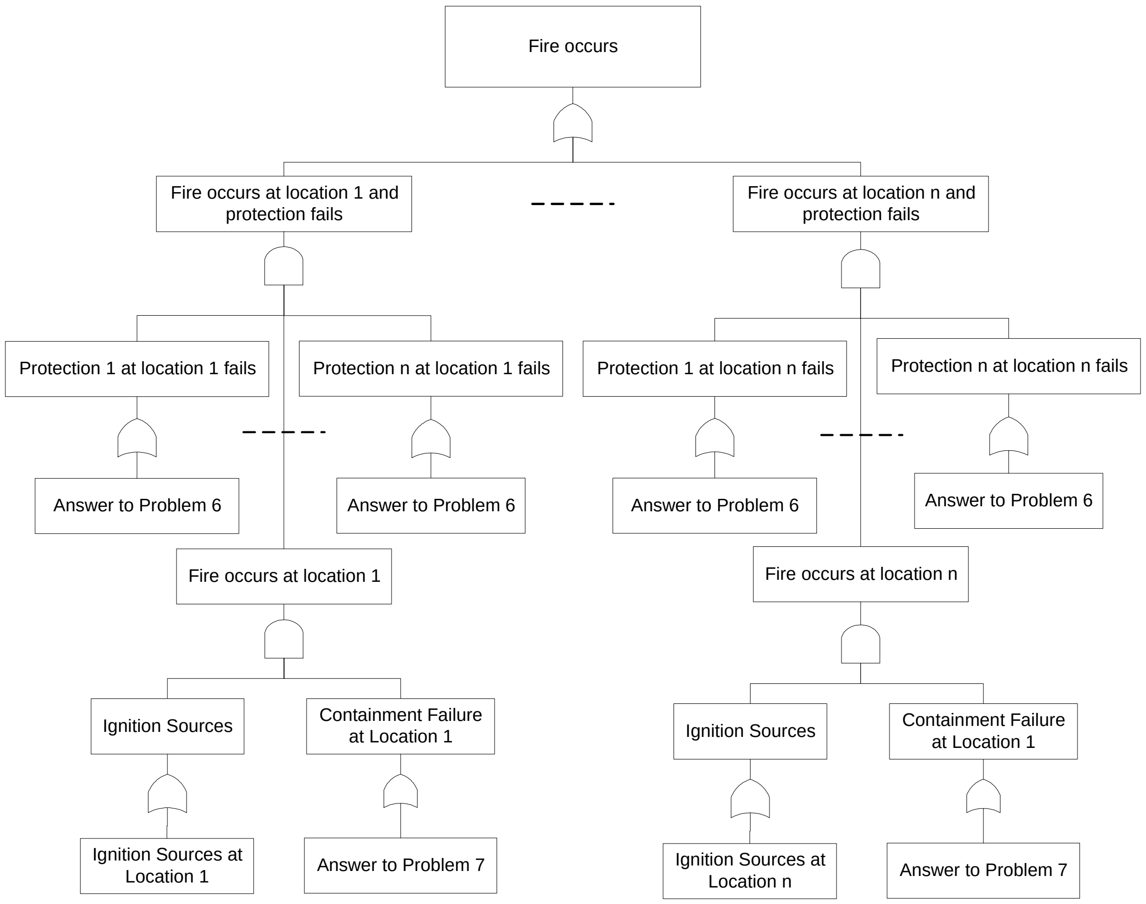

Fire occurs
Fire occurs at location 1 and protection fails
Fire occurs at location n and protection fails
Protection 1 at location 1 fails
Protection n at location 1 fails
Protection 1 at location n fails
Protection n at location n fails
Answer to Problem 6
Answer to Problem 6
Answer to Problem 6
Answer to Problem 6
Fire occurs at location 1
Fire occurs at location n
Ignition Sources
Containment Failure at Location 1
Ignition Sources
Containment Failure at Location 1
Ignition Sources at Location 1
Answer to Problem 7
Ignition Sources at Location n
Answer to Problem 7

Figure 3.3 The resultant fault tree model for ‘fire” event

3.3 Modeling Component

Chemical plants are usually very large and complex, and it is impractical and unnecessary to analyze the entire plant at one time. To gain a reasonable scope of study, the first task is to divide the plant into a set of functional units and then examine them one by one. Therefore, before applying the methodology, the physical boundary of the system must be well defined.

The information that must be gathered for good reliability analysis includes component interrelations, component failure characteristics, and accurate system specifications. In order to automate fault tree synthesis, the computer must identify the system components and their connections. In this work, a system block diagram is used to represent the system topology. Each block in a system block diagram represents one item of equipment. A solid line between components indicates the physical connection between them, and a dotted line represents information or signal transmission between components (typically in control loops, trip systems, and alarm systems). The configuration information for controls or special features is also required for a block diagram.

However, there is still one problem remaining to address. The behavior of each component in a chemical plant depends on time, internal states, and external environment. Each component in a system is related to other components (materials, equipment, plant personnel, *etc*.) and environment in a specific manner and identical components may behave differently in different environment. Before we can determine the correct cause-and-effect models to apply, the component state and environment factors must be clarified, which will be incorporated into a system block diagram. System identification requires a detailed description of component initial states and normal operation conditions. Initial conditions are specified for each component that has more than one operating states. The external event table records the external events and its effects associated with each plant, section, unit operation, and equipment. The lower level components inherit external events automatically from the upper level blocks.

As we have noticed, some failure behaviors of components are time-related. Furthermore, fault trees are instant "snapshots" of a system at a certain time. Start-up or shutdown conditions can generate different hazards compared to a steady state operation, and the time domain must also be specified.

In summary, system identification requires a careful delineation of component initial conditions. The time domain must also be specified; start-up or shutdown conditions can generate different hazards than a steady state operation. Once the system is defined clearly, we can then proceed with fault tree construction.

3.3.1 Fault Propagation within Component

Each chemical plant is unique, however, they have much identical equipment, materials, and even processes. For these common components, we have sufficient knowledge of their failure modes and effects. If we can utilize this common knowledge and automate this part, the labor and cost required in fault tree construction will be reduced significantly.

Qualitative cause-and-effect models are used to represent the functionality and relationship within components. The mini fault tree models for an individual component include three parts: 1. partial (simple) digraphs describing the first-principle relationship between the input and output variables; 2. failure modes and their effects; 3. external events and their effects. Digraphs are generated from qualitative confluences and heuristic knowledge of the equipment. The effect of a failure, human error, and environmental condition can cause one/several output deviations or change the relationships among variables. Failure modes and external events are integrated into cause-and-effect models directly during the fault tree synthesis to avoid unnecessary expanding of the digraph. The lower level components inherit external events automatically from the upper level blocks.

When constructing the fault tree, the algorithm searches for the appropriate digraphs, relevant failure modes, and external events of the current output variable deviations, generates mini-fault tree models automatically and then utilizes them to construct fault trees. Because of this approach, we not only reduce the storage requirement, the models are also straightforward and easy to understand. Users can edit the relationships among the variables, and add possible failure modes for the specific equipment directly in the database.

Digraphs in this algorithm are only partial digraphs showing the general functionality of a component between input and output under certain states. There are no published systematic guidelines for the generation of digraphs. An algorithm creating diagraphs directed from differential equations and algebraic equations was mentioned in

Maurya, Rengaswamy, and Venkatsubramanian (2001), which is yet to be detailed and proved. For most of the equipment, chemical processes, and unit operations in the CPI, many theoretical or empirical mathematical models have been developed to describe their behavior. These mathematical models typically involve fluid mechanics, mass transfer, heat transfer, and kinetics. However, in most cases, it is difficult, expensive, and sometimes impractical to solve the large set of all the unwieldy mathematical equations and present an accurate quantitative analysis even with the help of modern computers. Fortunately, it is not always necessary to solve the equations numerically to construct a digraph for a device or a process. In contrast to classical physics, qualitative physics describes physical systems in qualitative terms – qualitative states and qualitative differential equations (De Kleer & Brown, 1984). The concepts and theories of qualitative physics are not yet mature, but it is still useful to create digraphs systematically for relatively simple devices. Once input variables and output variables are specified, qualitative differential equations are drawn from the conventional mathematical models, and digraphs are derived from these equations. For heterogeneous and distributed system such as heat exchangers and distillation columns, mathematical equations, experimental data or heuristic knowledge can be used together to create the digraphs.

External event table includes the environmental conditions and external activities that will cause deviation or change the relationship between any input and output variables in the process under study.

Examples of digraph, failure modes and effects table, and external event table can be found in the case studies presented in Chapter V.

3.3.2 Fault Propagation between Component

An essential aspect of fault tree construction is tracing concerned process deviations, taking care of process safe guards in the meanwhile. In order to generate correct fault tree models, component interrelations and system topology, component failure characteristics, and accurate system specifications must be provided.

A system consists of components such as hardware, materials, and plant personnel, and is surrounded by its physical and social environment, and suffers from aging. The functionality of a component can be affected from within the component such as equipment failures or from outside factors such as the states of connected components. Deviations are assumed to initiate through components, and the environment, plant personnel, and aging can affect the system only through the system components.

Each component in a system is related to the other components in a specific manner, and identical components may have different characteristics in different systems. Therefore, we must clarify component interrelations and system topology. The interrelations and topology are found by examining plant piping, electrical wiring, mechanical couplings, information flows, alarm loops, and physical locations of components, which are represented by system block diagram in this methodology. Fault propagation among components are taken into account by tracing physical connections in system block diagram and examining the specific inputs to determine if the changes of these input can change the component state.

The system environment enters the cause and effect models by external event tables. The same external event may cause multiple equipment failures or process deviations. The system boundary must be provided to prevent the analysis from diverging. Fault propagation at the system boundary is treated as a basic event, and is not developed further. However, this basic event might be top events for the adjacent system units.

A condition or an event that causes multiple basic events is called a common cause. An example of a common cause is an external fire or flood that causes all supposedly redundant components to fail simultaneously. Common mode failures are always an important subject of FTA. Consider one system consisting of valves A and B, and assume that one of the valves is redundant. This valve system will be far more reliable than a system with a single valve, if one valve fails independently of the other. Coexistence of the two malfunctions is unlikely. However, if one valve is liable to fail under the same conditions as the other, the double-valve system is only slightly more reliable than the single-valve system. In this methodology, each basic event is tagged with a unique identifier. Common cause failures can be found during fault tree evaluation if some basic events cause more than one failure or deviation, and feed into multiple intermediate events.

3.4 Modeling Special Features

Special features such as process controls, trip systems, bypasses, pressure relief *etc*. are integral parts of the design and operation of a safe and productive chemical plant. How to handle special features is the central concern in the research of computer-aided fault tree synthesis for the CPI. Different types of control systems have different failure mechanisms and therefore different cause-and-effect models. Every special feature within the process under analysis needs to be identified and classified according to its structural characteristics and functionalities. Although it is highly desirable to automate the identification and classification of special features, it has been recognized that a mechanistic approach is not realistic and not always appropriate. In this work, control loops and their configurations are supplied as part of the system block diagram.

3.4.1 Modeling Process Controls

There are three circumstances under which a controlled process variable can deviate beyond its normal range (Lapp & Powers, 1977). Firstly, uncontrollable disturbances can drive the controlled variable to abnormal states even when the control loop is working properly. Secondly, a deviation can be caused by a controllable disturbance while the control loop is inactive. Finally, the control loop itself can cause process deviations upon certain malfunctions.

Based on the above thoughts, modeling of process control loops includes three essential attributes: constituent components, control capability (uncontrollable or controllable), and instrumental failure effects (inactive or unhealthy). A generalized fault tree model for process controls is proposed, which is shown in Figure 3.4.

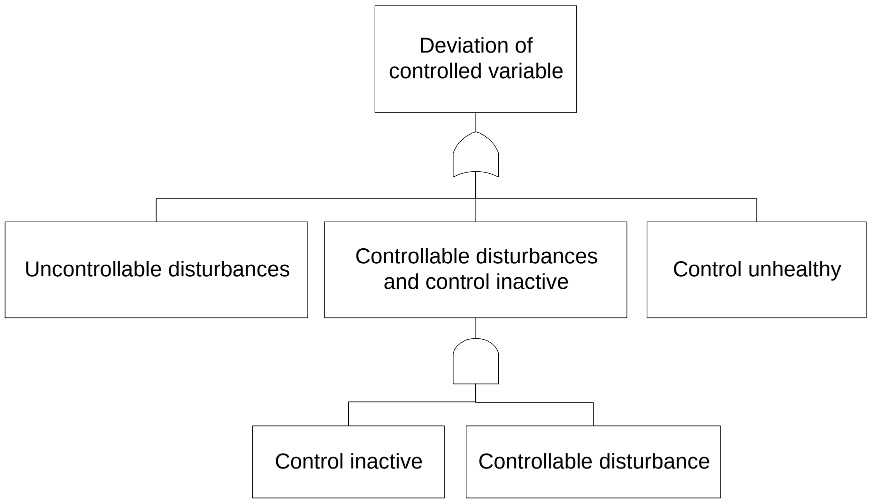

Deviation of controlled variable
Uncontrollable disturbances
Controllable disturbances and control inactive
Control unhealthy
Control inactive
Controllable disturbance

Figure 3.4 A generalized fault tree model for process controls

Uncontrollable disturbances refer to disturbances that are not captured by the control loop such as set point too high, large/fast disturbances in feedback controls, and uncompensated disturbances in feedforward controls. These disturbances will cause the controlled variable to deviate from the normal range regardless of the status of the control loop. When the control loop is inactive, such as a stuck component, the controlled variable will deviate upon a disturbance. When the control loop is unhealthy, the controlled variable will deviate even without disturbance. “Unhealthy” conditions are rendered by component malfunctions (fails high/low) or component reverse installation. Four most common types of instrument failures are considered in this work: “stuck”, “zero point drift high/low”, “fails high/low”, and “reversed”. The “reversed” faults of continuous control loops normally should be detected during the design, installation, or test phases. If other instrumental failures are possible, the user can add them manually by specifying the effects of these failures (control inactive or control unhealthy). Here we assume that the possibility of “two elements on the control loop malfunctioning simultaneously” is negligible, which is normally true in reality.

Feedback control, feedforward control, nested control, and flow ratio control are typically used in the CPI, of which negative feedback control and negative feedforward control are the most popular. Different types of control systems behave differently with regard to the same disturbance and have different tolerances toward certain instrument failures, and hence their cause-and-effect models. Therefore, each type of control mechanism has its own cause-and-effect fault tree model, which has three subtrees for “uncontrollable disturbances”, “controllable disturbances and control loop inactive”, and “control loop causes the deviation” events. Computer control, alarm system, and manual control systems can be viewed as special forms of feedback or feedforward systems. The instruments discussed here are not limited to traditional sensors, controllers, and control valves. Computers, human operators, digital devices, and logic voting systems can represent any of them and differ only in their failure mechanisms and failure rates. For instance, when evaluating conventional controllers, we normally consider hardware failures. However, there is a significant amount of human involvement in the design, operation, and maintenance of computer and manual controls, which experiences both software and hardware failure. Therefore we must take human activity into account when gathering data for computer controls and manual controls. A logical voting system is typically used for the purpose of redundancy, and mathematical treatment can be employed to derive its failure rate. Trip system, bypass, and overpressure relief can be viewed as special forms of feedforward loop. The treatments of these special features are slightly different from normal feedforward controls. Complex control systems such as cascade and ratio controls are dealt as a whole with their corresponding fault tree models.

Even for the same control scheme, the cause-and-effect model may differ with the controller type. Traditionally, controllers can be proportional (P), proportional-integral (PI), proportional-derivative (PD), and proportional-integral-derivative (PID). The integral function reduces the steady-state errors to zero and is also called "reset" action while the derivative function was proposed to speed up the action of the control loop and is also called "rate" action. There is no systematic discussion about the effects of various types of controllers on the fault tree synthesis. Galluzzo and Andow (1984) once reported some experimental work about the effect of proportional and proportional-and-integral controllers on fault tree synthesis. The special fault tree models for control loops must be modified especially when the controller has the integral function, which is quite common in the CPI. The effects of different conventional controllers on the fault tree synthesis will be discussed as well.

When control loops are encountered in fault tree synthesis, we are normally concerned with the deviations of the controlled variables instead of the intermediate outputs of sensors, controllers, and control valves. Therefore, in this research, only models for deviations of the controlled process variables are developed. Furthermore, most control loops prevent the controlled variable from deviating in both directions (positive and negative). The cause-and-effect models for negative deviations of the controlled variable are similar to those for positive deviations except for the case of uni-direction control loops. Though not discussed specifically in this dissertation, this methodology is ready to embrace uni-direction control loops through capability tables. The figures in the following section represent the proposed cause-and-effect unit models of a medium positive deviation of the controlled variable. An extension to a negative or large deviation is straightforward. In this dissertation, feedback controls, feedforward controls, nested controls, and ratio controls are discussed in detail. Trip systems, bypasses, and pressure relief are viewed as special forms of feedforward or feedback control. Complex controls can be handled by providing their corresponding fault tree models.

3.4.2 Modeling Feedback Loops

Feedback is the most popular control mechanism in the CPI. It is simple and easy to design, implement, and test. In particular, it can tolerate slow changes of controllers and control valves, such as zero point drifting failures. Figure 3.5 is a general block diagram of feedback controls. As denoted, sensor, controller, and control element here are not limited to traditional instruments, and can represent any kind of transmitter, programmable logic solver, and final action element.

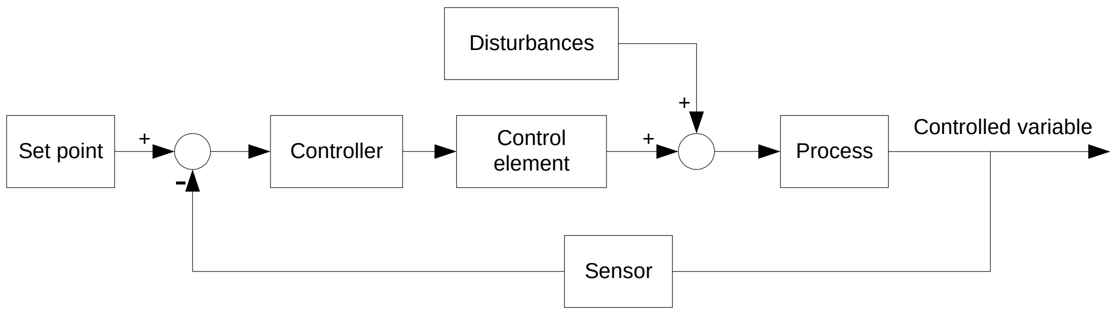


Figure 3.5 Feedback control block diagram

In contrast to feedforward control, feedback control is capable of capturing most of the process disturbances to the controlled variable except for set point deviations, but it may be saturated by large disturbances or ineffective for fast disturbances. Therefore, “uncontrollable disturbances” include large and fast disturbances and set point deviations. Feedback control is robust toward slow changes of controller and control valve, and only sensor zero point drifting and complete instrument failures (“fails high/low”) can render the control loop “unhealthy”. Normally complete instrument failures are treated as basic events since failure rate data available do not normally differentiate between “fail high or low”. If higher resolution is desired in some cases, the algorithm selects “high” or “low” according to the functionality of the elements. “High” is selected if the controlled variable increases with the certain output, and vice versa. Here we neglect the scenarios of “two elements on the control loop malfunctioning simultaneously”, whose failure rates are normally much lower than one element failure.

It is assumed that the process is normal at the moment the sensor becomes stuck. However, the measured value from the sensor is not necessarily equal to the set point under normal operation. In practice there are normally differences when the control loop is functioning properly. With a proportional or proportional-and-derivative controller, this discrepancy is not sufficient to cause a significant deviation of the controlled variable upon “sensor stuck” without other disturbances. Figure 3.6 represents the cause-and-effect fault tree model for a feedback control without integral action.

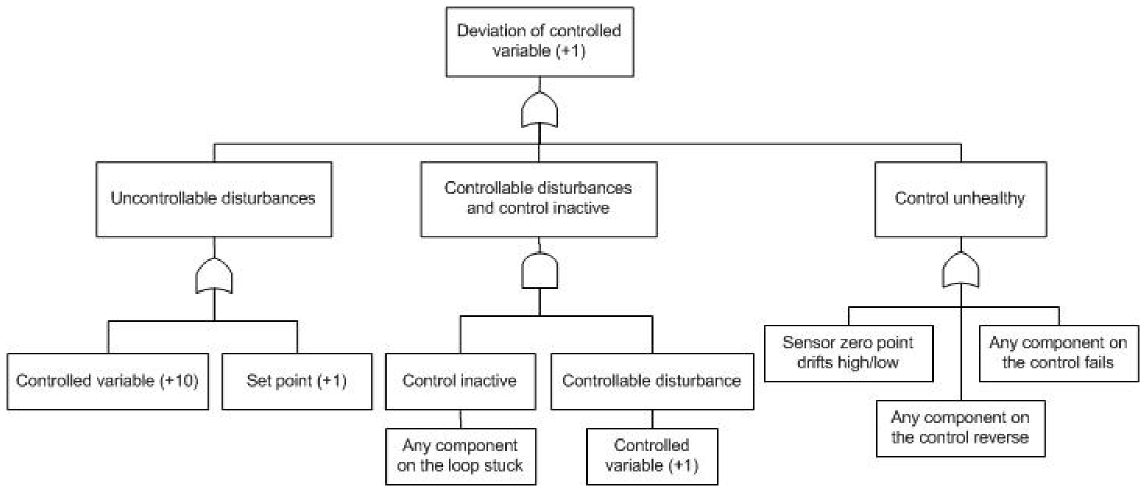

Deviation of controlled variable (+1)
Uncontrollable disturbances
Controllable disturbances and control inactive
Control unhealthy
Controlled variable (+10)
Set point (+1)
Control inactive
Controllable disturbance
Sensor zero point drifts high/low
Any component on the control fails
Any component on the loop stuck
Controlled variable (+1)
Any component on the control reverse

Figure 3.6 Fault tree model for feedback control without integral action

The subtrees of “controlled variable (+1/10)” represent the deviations of the controlled variable without the manipulating variable disturbances, which are actually disturbances. They are used in the fault tree models for special features in the follow context as well.

With integral action, the controller will keep on increasing or decreasing until saturated given a tiny difference between sensor signal and set point. The controlled variable will take off from the original value until the control valve is completely opened or closed. From the above discussion, we can see that “sensor stuck” makes the control inactive without an integral action, while it renders the control loop unhealthy when an integral action is present. The deviation direction (increase or decrease) of the controlled variable depends on the sign of the difference and the relationship between the difference and controlled variable. As a shortcut, we can decide this by the modes of the sensor. If the sensor is positive activated (as most sensors are), the controlled variable increases with the set point greater than the sensor output, and vice versa. Since “reset” action is widely used in chemical process controls, our cause-and-effect models must be modified to accommodate this scenario. Figure 3.7 shows the cause-and-effect fault tree model for a feedback control with integral action.

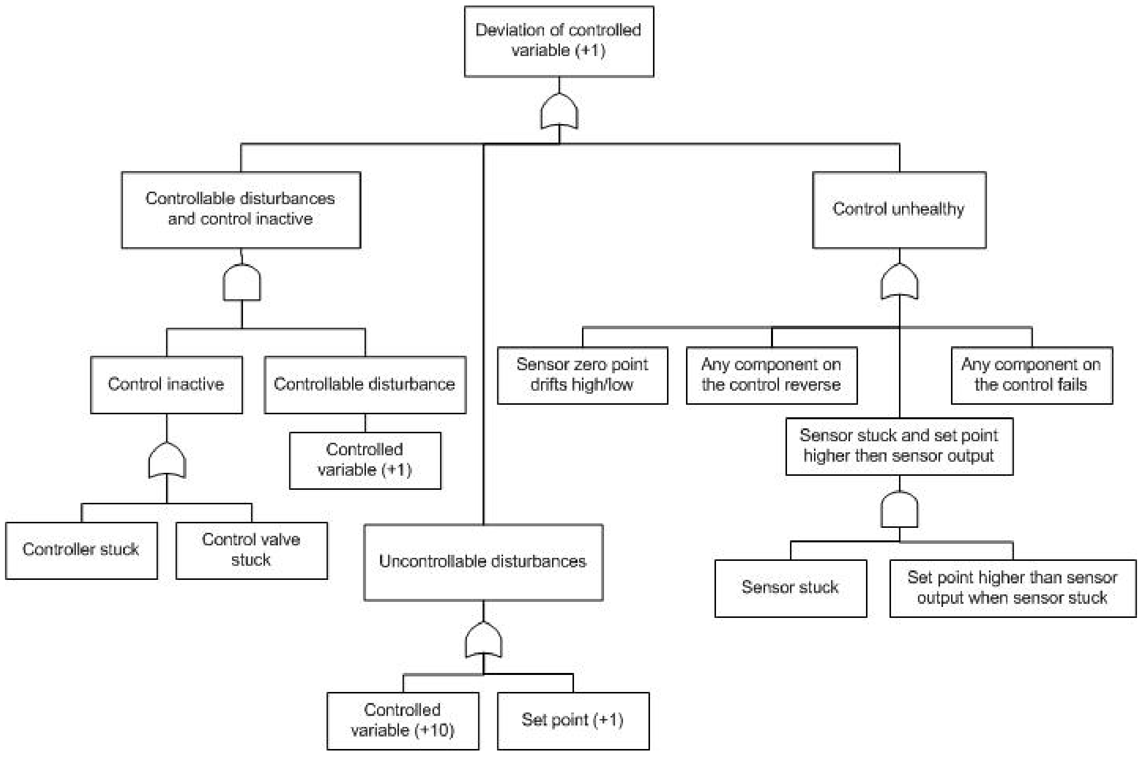

Deviation of controlled variable (+1)
Controllable disturbances and control inactive
Control unhealthy
Control inactive
Controllable disturbance
Controlled variable (+1)
Controller stuck
Control valve stuck
Uncontrollable disturbances
Controlled variable (+10)
Set point (+1)
Sensor zero point drifts high/low
Any component on the control reverse
Any component on the control fails
Sensor stuck and set point higher then sensor output
Sensor stuck
Set point higher than sensor output when sensor stuck

Figure 3.7 Fault tree model for feedback control with integral action

3.4.3 Modeling Feedforward Loops

Feedforward control is also called open-loop or input-compensated control. In principle, feedforward control is designed to compensate for certain kinds of disturbances, but it is not able to capture other disturbances. The general block diagram and the corresponding fault tree model of feedforward control is shown in Figures 3.8 and 3.9, respectively.

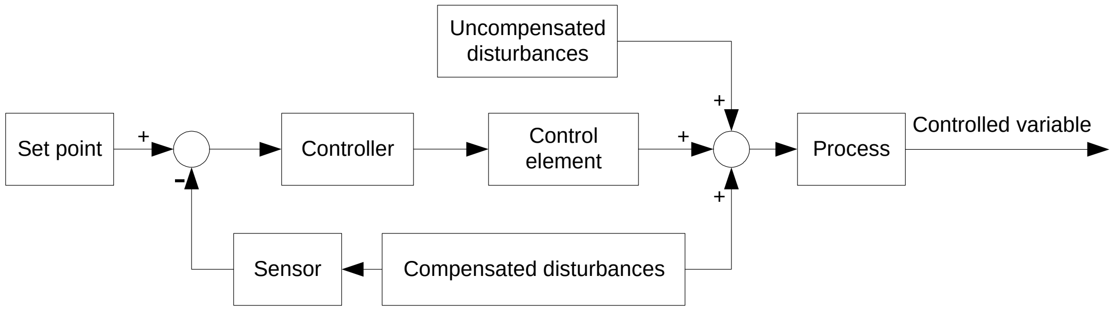


Figure 3.8 Feedforward control block diagram

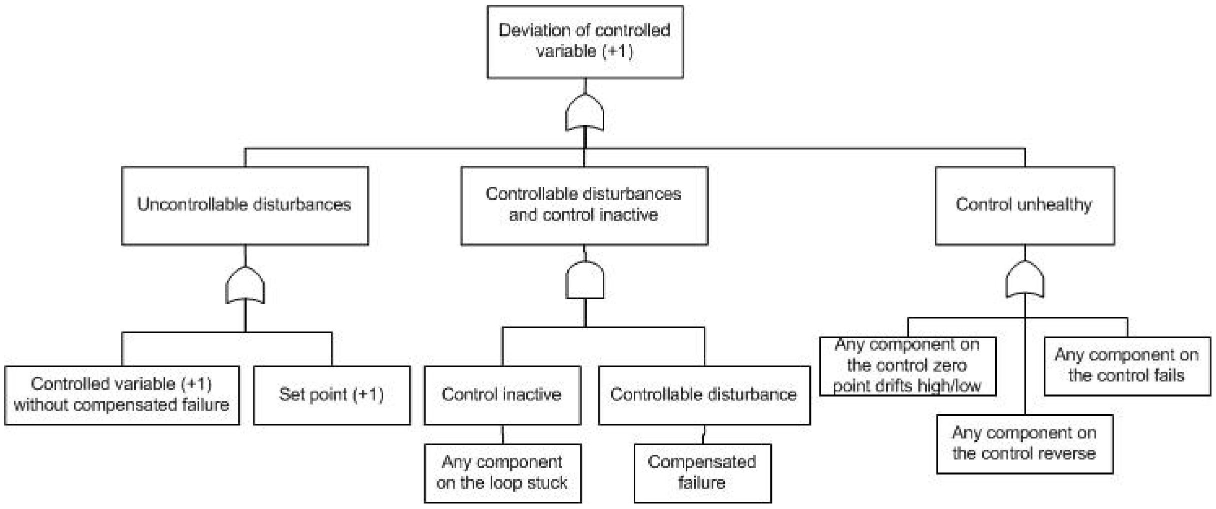

Deviation of controlled variable (+1)
Uncontrollable disturbances
Controllable disturbances and control inactive
Control unhealthy
Controlled variable (+1) without compensated failure
Set point (+1)
Control inactive
Controllable disturbance
Any component on the control zero point drifts high/low
Any component on the control fails
Any component on the loop stuck
Compensated failure
Any component on the control reverse

Figure 3.9 Fault tree model for feedforward control

In this work, trip systems are also categorized as special feedforward or feedback controls. However, instead of just trying to counteract a process disturbance, trip system will shutdown (or close) some process or sub-process upon certain exceeding-limit failures (such as pump stops, level too high, level too low, utility failures, *etc*.) to prevent further fault propagation or avoid more serious incidents. Since there is no controlled variable, in such cases, whenever we meet these specific conditions (pump stop, utility failures) in the fault tree synthesis procedure, we will add an AND gate and replace this condition, as in Figure 3.10.

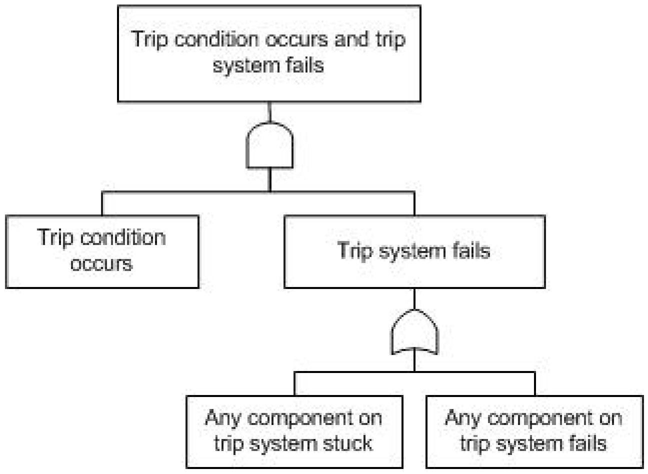

Trip condition occurs and trip system fails
Trip condition occurs
Trip system fails
Any component on trip system stuck
Any component on trip system fails

Figure 3.10 Fault tree model for trip system

The treatment for pressure relief and bypass protection is similar to that of trip systems. Pressure relief and bypass are typical uni-direction protections. Whenever the protection condition (high pressure or bypass conditions) is met during the construction of fault trees, an AND gate is used to replace this condition, which is shown in Figure 3.11 and Figure 3.12. Design errors such as “under design” of relief valve and “over design” of bypass line are not included here by default, but they can be entered by the users if they are desired to be considered.

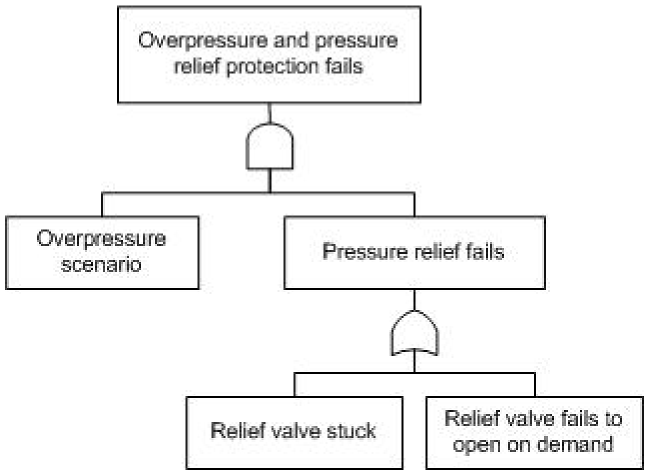

Overpressure and pressure relief protection fails
Overpressure scenario
Pressure relief fails
Relief valve stuck
Relief valve fails to open on demand

Figure 3.11 Fault tree model for pressure relief system

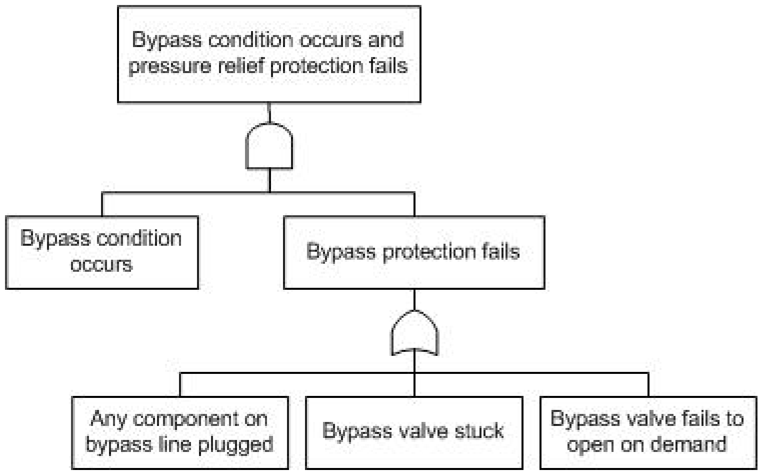

Bypass condition occurs and pressure relief protection fails
Bypass condition occurs
Bypass protection fails
Any component on bypass line plugged
Bypass valve stuck
Bypass valve fails to open on demand

Figure 3.12 Fault tree model for bypass system

3.4.4 Modeling Complex Controls

Although feedback and feedforward control are the most popular controls in the process industry, some complex process controls like nested control and flow ratio control are typically used in the process industry as well, especially for some critical equipment such as reactors and distillation columns. These places are almost always the focus of safety concerns, and are critical in the construction of system fault tree models.

Modeling nested control

Nested control is also called cascade control. A nested control with two feedback loops is shown in Figure 3.13.

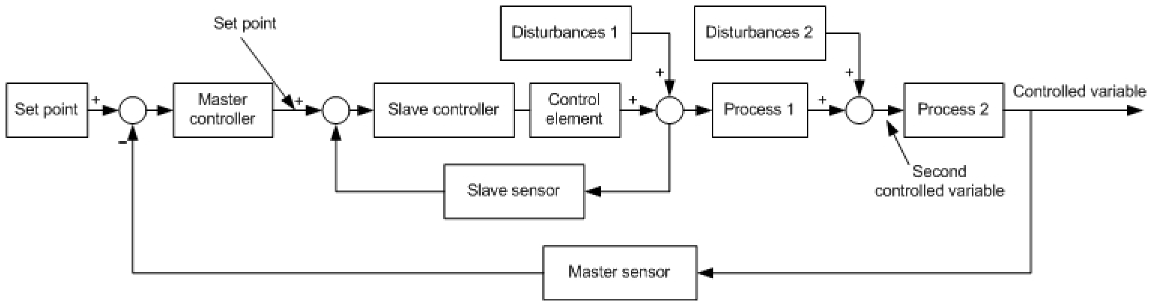


Figure 3.13 Nested control block diagram with two feedback loops

As we know, nested controls are normally adopted if the dynamic response of the slave loop is much faster than the master loop. Generally speaking, nested feedback control is efficient for controlling large/fast disturbances to the secondary controlled variable. Also nested control is robust to the zero point drifting of the master controller and slow changes occurring inside the slave control loop including the slave sensor. The disturbances entering the slave control loops can be corrected much faster by the design of a cascade control, and the effective dynamic lag of the master loop will be greatly reduced if designed properly. As we know, the performance of a cascade control relies on the relative speeds of the components included in the master and slave loops. The dynamic response of the slave loop must be much faster than the master loop. If the slave sensor is ideal and its transfer function is 1 (which is normally true), the forward gain of the slave loop is normally much larger than 1 so that the slave loop can be approximated with an equivalent gain of 1.

When the slave sensor is stuck, the forward gain of the master loop increases significantly. The stability of the control loop degrades with increasing gain. The closed loop will become oscillatory or unstable upon such a significant forward gain increase, which is confirmed by our simulations with Matlab ®. However, though slave sensor stuck failure will drive the control loop to abnormality, it is difficult to decide which direction it will go and more likely will oscillate. To obtain a conservative estimation, we will put "stuck slave sensor" under "unhealthy control" of controlled variable deviation. The user is responsible for analyzing its effects if more precision is desired.

The corresponding fault tree model for a nested control with two feedback loops is shown in Figure 3.14.

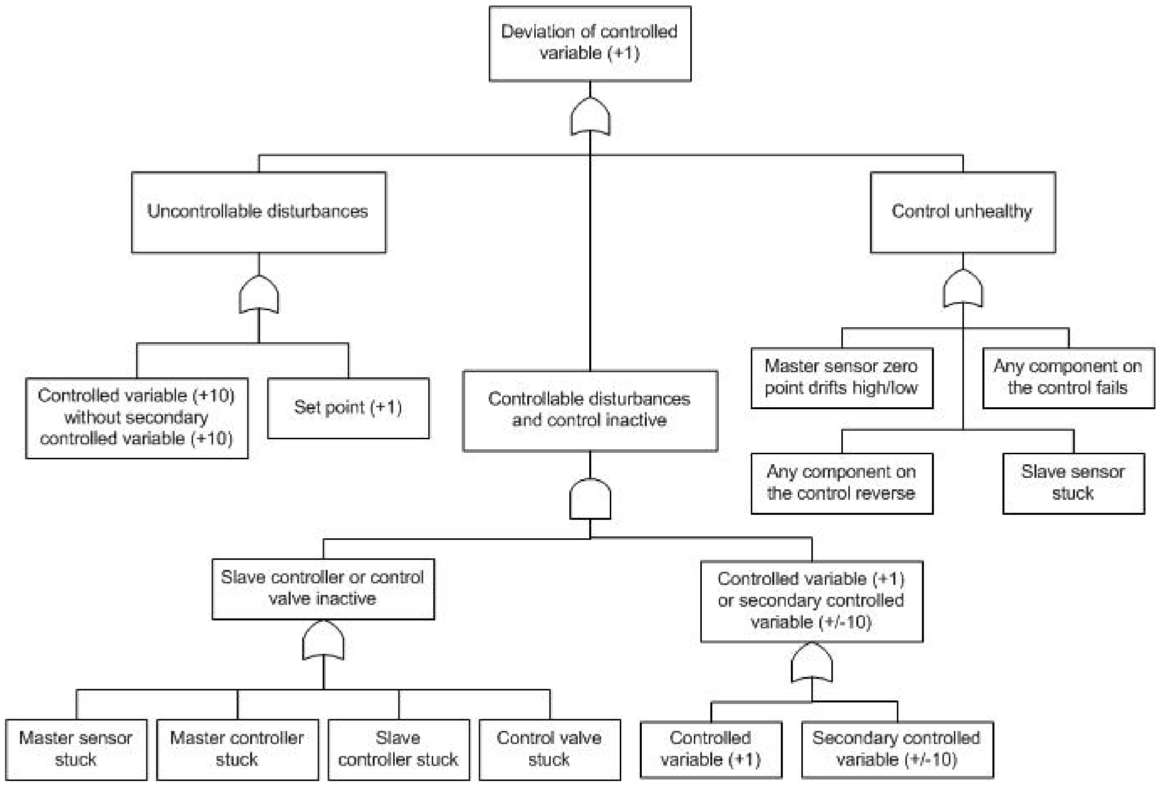

Deviation of controlled variable (+1)
Uncontrollable disturbances
Control unhealthy
Controlled variable (+10) without secondary controlled variable (+10)
Set point (+1)
Controllable disturbances and control inactive
Master sensor zero point drifts high/low
Any component on the control fails
Any component on the control reverse
Slave sensor stuck
Slave controller or control valve inactive
Controlled variable (+1) or secondary controlled variable (+/-10)
Master sensor stuck
Master controller stuck
Slave controller stuck
Control valve stuck
Controlled variable (+1)
Secondary controlled variable (+/-10)

Figure 3.14 Fault tree model for nested control with two feedback loops

The direction of the secondary controlled variable deviation is determined by the characteristics of the process (positive or negative). Here the slave controller and the control element are defined as common components for the master control loop and slave control loop. The algorithm decides the sign of the secondary controlled variable depending on the relationship between the primary and secondary controlled variables. The master controller and the slave controller can be combined into one item of equipment with master sensor signal, slave sensor signal, and set point as its inputs. For the inner control loop, the cause-and-effect model is the same as a simple feedback loop except that the set point of the inner control is determined by the master controller.

A feedback and a feedforward can be combined together to consist of a nested control as well. Figure 3.15 shows a general block diagram for this configuration.

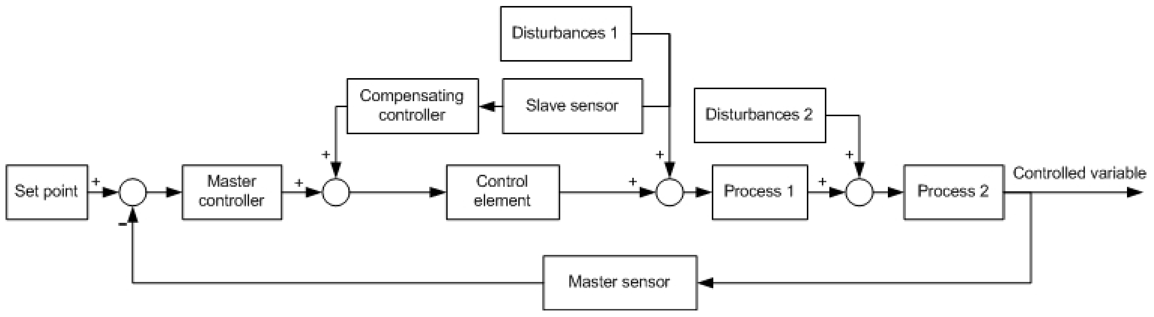


Figure 3.15 Nested control block diagram with one feedback loop and one feedforward loop

The major disturbance 1 will be compensated by the feedforward loop as well as the feedback control. This configuration can tolerate small zero point drifting of the slave sensor, compensating controller, master controller, and control valve. The fault tree model for this configuration is shown in Figure 3.16.

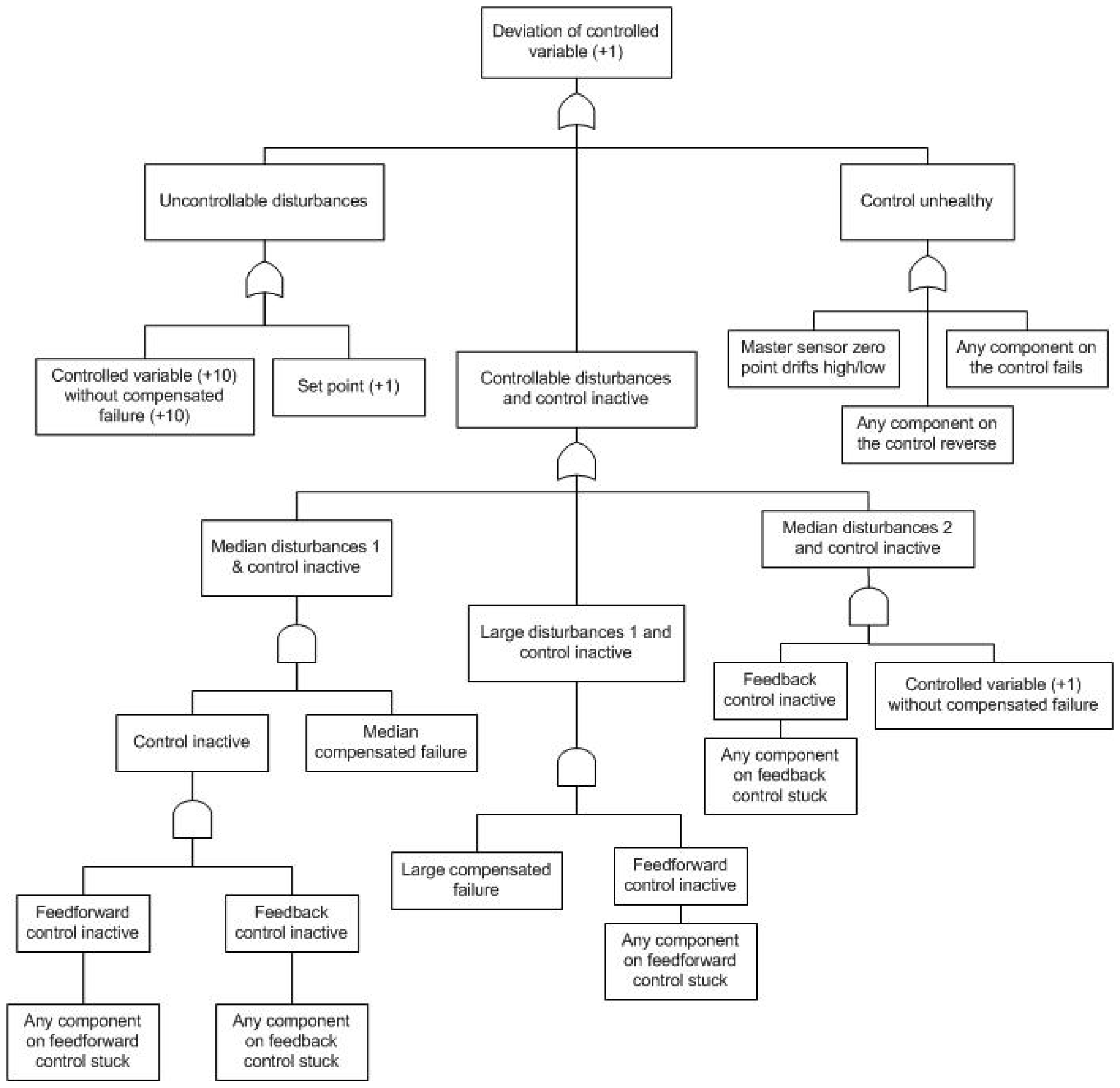

Deviation of controlled variable (+1)
Uncontrollable disturbances
Control unhealthy
Controlled variable (+10) without compensated failure (+10)
Set point (+1)
Controllable disturbances and control inactive
Master sensor zero point drifts high/low
Any component on the control fails
Any component on the control reverse
Median disturbances 1 & control inactive
Median disturbances 2 and control inactive
Large disturbances 1 and control inactive
Feedback control inactive
Controlled variable (+1) without compensated failure
Control inactive
Median compensated failure
Any component on feedback control stuck
Large compensated failure
Feedforward control inactive
Feedforward control inactive
Feedback control inactive
Any component on feedforward control stuck
Any component on feedforward control stuck
Any component on feedback control stuck

Figure 3.16 Fault tree model for nested control with one feedback loop and one feedforward loop

Other configurations of nested control are possible, and can be incorporated in this work by providing their corresponding fault tree models.

Modeling flow ratio control

Flow ratio control is employed in chemical processes to maintain, for example, the fuel/air ratio to a furnace or the flow ratio of two feed streams to a reactor. Here the flow ratio between two streams is the issue because it is critical to the product quality, process safety, or economic costs. Currently we consider only a constant flow ratio control, namely, the set point of the ratio between F_1 (master flow rate) and F_2 (slave flow rate) does not depend on time or other process variables. Flow ratio control can be open-loop or closed-loop. In an open-loop flow ratio control, the controller is essentially a ratio calculator, and its output is transmitted to a control valve directly. A general block diagram for an open-loop flow ratio control is shown in Figure 3.17.

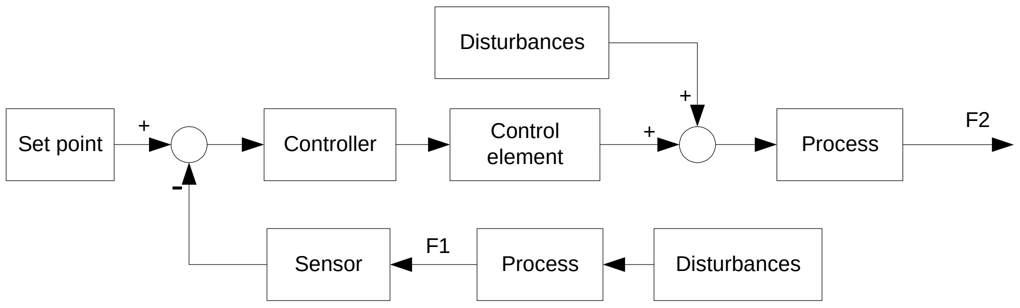


Figure 3.17 Open-loop flow ratio block diagram

Since there is no control over the master stream F_1, the cause-and-effect model is the same as that of the process without controls. Sometimes, fault tree models for the slave flow rate may be desired. The fault tree models for deviations of flow ratio are the same as that for the slave stream flow, which is presented in Figure 3.18. The master stream F_1 and slave stream F_2 are assumed to be independent without the flow ratio control, which is normally true.

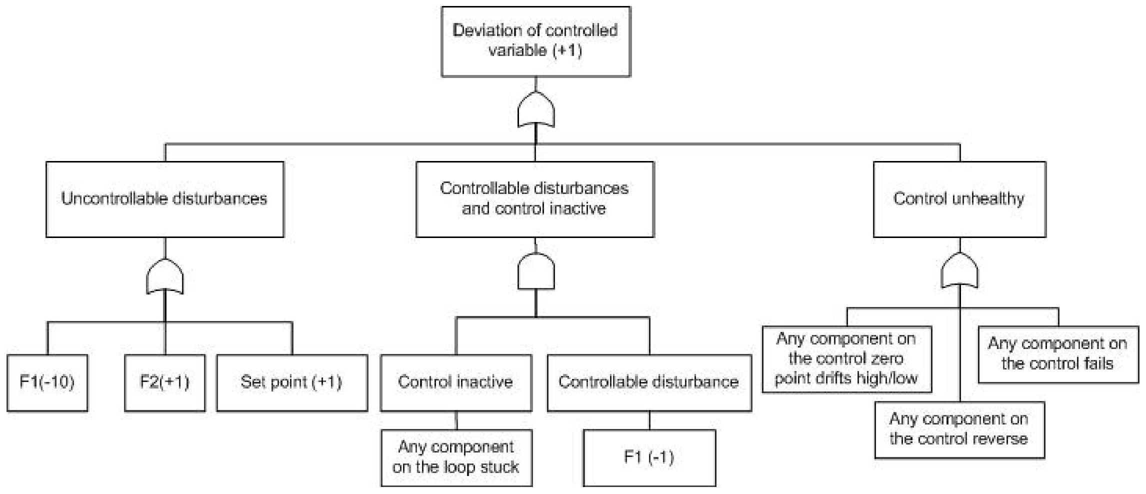

Deviation of controlled variable (+1)
Uncontrollable disturbances
Controllable disturbances and control inactive
Control unhealthy
F1(-10)
F2(+1)
Set point (+1)
Control inactive
Controllable disturbance
Any component on the loop stuck
F1 (-1)
Any component on the control zero point drifts high/low
Any component on the control fails
Any component on the control reverse

Figure 3.18 Fault tree model for open-loop flow ratio control

The open-loop flow ratio control is not robust to disturbances such as inlet pressure change of the slave stream. Instead engineers sometimes use closed-loop ratio control to overcome this disadvantage. The diagram of closed-loop flow ratio control is shown in Figure 3.19.

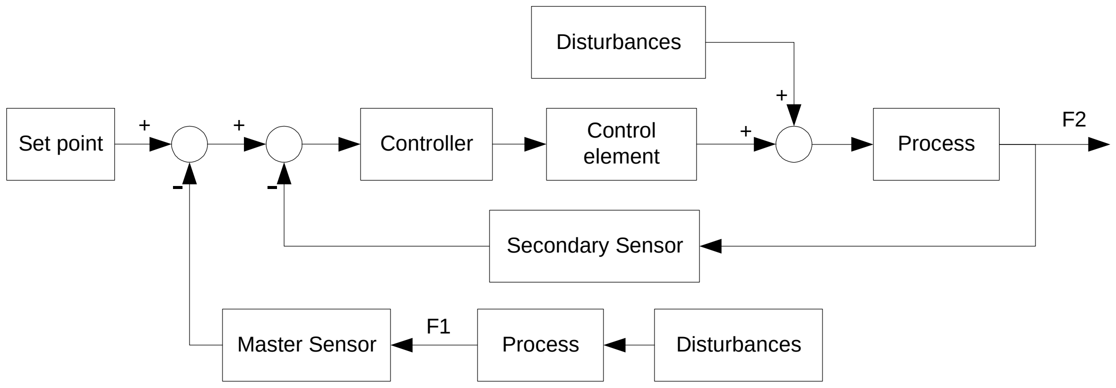


Figure 3.19 Slave stream close-loop flow ratio control block diagram

Here we still do not have any control over the master flow, and the cause-and-effect model for the master flow is the same as a normal process variable. The corresponding fault tree model for flow ratio and slave stream flow rate is shown in Figure 3.20.

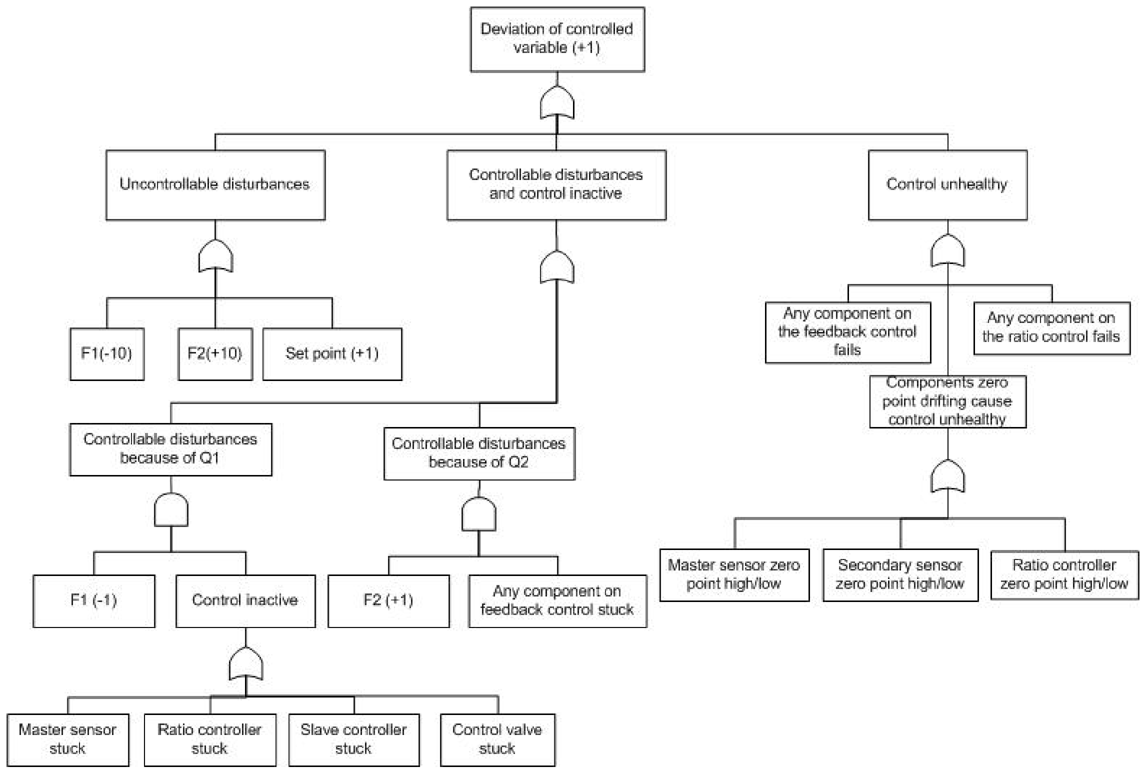

Deviation of controlled variable (+1)
Uncontrollable disturbances
Controllable disturbances and control inactive
Control unhealthy
F1(-10)
F2(+10)
Set point (+1)
Any component on the feedback control fails
Any component on the ratio control fails
Components zero point drifting cause control unhealthy
Controllable disturbances because of Q1
Controllable disturbances because of Q2
F1 (-1)
Control inactive
F2 (+1)
Any component on feedback control stuck
Master sensor zero point high/low
Secondary sensor zero point high/low
Ratio controller zero point high/low
Master sensor stuck
Ratio controller stuck
Slave controller stuck
Control valve stuck

Figure 3.20 Fault tree model for slave stream close loop flow ratio control

Flow ratio control can be implemented with two independent feedback controls. Set points are determined based on flow ratio desired. In this case, the fault tree model for feedback control loop will be applied if deviation of flow or flow ratio is traced.

3.5 Tree Rationalization

Tree rationalization includes consistency checking and tree simplification.

3.5.1 Event Consistency

It is essential to check the consistency among tree events to remove inconsistent and unreasonable events and prevent the fault tree construction procedure from going forever. There are two types of consistency – serial and parallel consistency.

Serial consistency is the consistency of events with its upper level events while parallel consistency is the consistency of events between two branches of the same AND gate. During fault tree construction, the methodology stores all the upper level parents, checks for serial consistency, and deletes inconsistent events whenever encountered and detected. However, parallel consistency checking must be performed after the initial fault tree construction. Events under different branches of the same AND gate are examined, and inconsistent events are removed.

It is also necessary to ensure that events in the fault tree do not violate or go beyond the system boundary. Process deviations at the system boundary are treated as basic events, which might be top events for adjacent systems. This step can be done during fault tree construction.

3.5.2 Tree Simplification

Fault trees drawn directly from a computer algorithm are normally opaque and contain many repeated events. It is not easy to read. A simplification procedure can make them concise and understandable to users. Two kinds of simplification are performed – algebraic simplification and tree simplification.

Algebraic simplification basically deals with certain or negligible events. When a certain event (with the probability of 1) is under an OR gate, algebraic simplification is performed to remove the parent gates until an AND gate is encountered. When a certain event is under an AND gate, this event is deleted directly. If an event with negligible probability is under an AND gate, removal of the whole parent gate will continue until an OR gate is met. If a negligible event is under an OR gate, this event is deleted directly.

Tree simplification mainly refers to structure suppression. All intermediate events with only one child are removed during tree suppression. Some Boolean manipulation is capable of converting the fault tree into an equivalent and simpler structure. During algebraic simplification and tree simplification, identical events under the same gate are removed automatically.

3.6 Summary of Chapter III

This chapter has been dedicated to describe the basic modeling technique and overall methodology. The basic idea of the proposed methodology is to capture the cause-and-effect logic among equipment behaviors, human responses, and environment factors around each item of equipment directly into mini fault trees.

Process block diagram is used to represent the system topology, configuration information, initiating states, and physical boundaries. Simple checklists are used to assist the analysts to identify potential hazardous incidents. Component modeling techniques are discussed in Section 3.3. Fault propagation within and between components is addressed.

Once the top event has been selected, the algorithm proceeds to find the corresponding cause-and-effect models in the database to trace the top event down to deviations in the process variables. Then intermediate events are developed, and relevant cause-and-effect models are generated directly from the digraphs, failure mode table, and external event table until a system boundary or primary events are reached. Control loops and special features will be managed by their corresponding fault tree models (template trees), which are discussed in detail in Section 3.4. Serial consistency checking is carried out during fault tree construction, and parallel consistency is performed after initial fault tree has been developed. Finally, a simplification procedure will make the generated fault tree concise and easy to understand.

The idea of this research is to make use of the available knowledge about common devices in the CPI in fault tree construction and reduce human force and expert time required. Though a computer will perform the dull part of the methodology, human interactions are required in many places to input system block diagrams and configurations, decide top events, and incorporate the cause-and-effect models for special process components and controls. Users are responsible for verifying the completeness and correctness of final fault tree models. User decision overrides a computer during or after tree construction.

CHAPTER IV

DATABASE STRUCTURE

A prototype computer package has been developed to demonstrate the capability of this methodology. To implement this methodology, user interface, algorithm/rules, and data are required. Software application architecture is shown in Figure 4.1. The computer language used to implement this program is Microsoft Visual Basic ®. Microsoft Access ® is the primary carrier for the database system. Process block diagrams and fault trees are in the form of Microsoft Visio ® files. Database system employed in this methodology is discussed in Chapter IV while user interfaces and algorithm flow charts are covered in Chapter V.

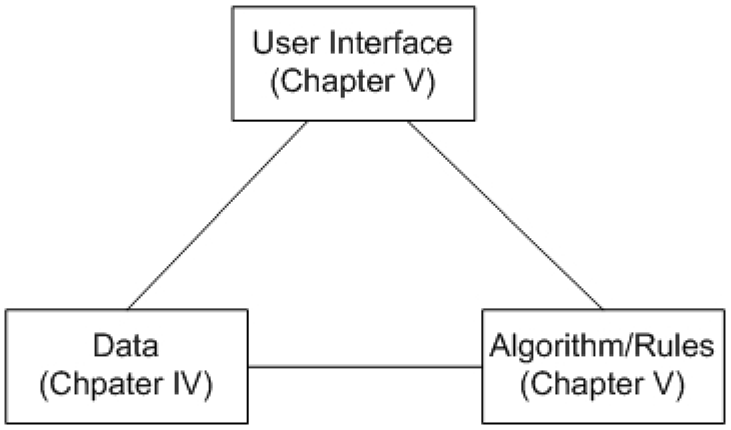


Figure 4.1 Software application architecture

Growing from a computerized analog of paper files and folders containing facts and data, database systems are now becoming more of a well-disciplined collection of information. An improper design of database might be poorly organized, inefficient, lack integrity, and compromise the use of database ultimately. On the other hand, a good database design not only meets the current needs but also incorporates future requirements.

A database begins with a concept of the overall application. The application should be able to take what you want and translate it into what the computer can handle. In this work, information about component modeling, equipment, and special features are required in the synthesis of fault trees. Many database software products are currently available on the open market. No matter how powerful these database management systems are to provide data manipulation and data management support, they require users to design the data structure and understand the details of the data, record contents, keys, *etc*. Microsoft Access ® is employed in this work to implement the database described hereafter.

Some normal forms have been developed toward database design to avoid redundancy, inconsistency, and anomalies. Though normalized design penalizes data retrieval in the sense that data may have to be retrieved from several tuples, it is the most valuable principle to reduce redundancy and remove anomalies, and thus facilitates database management and reduces storage requirements substantially. There are formal mathematical definitions for each normal form (Bagchi & Chaudhri, 1989). Their narrative and brief descriptions are listed as follows:

- The first normal form: Every data entry for each attribute is non-decomposable.
- The second normal form: Every “non-key” attribute in the relation is dependent on the key.
- The third normal form: No non-key attribute identifies another non-key attribute.
- The fourth normal form: No more than one multi-valued fact is allowed in the same relation.

There are two databases in this project – the public database and the specific database. The public database includes digraphs, failure modes and effects, and failure rate data for common devices and cause-and-effect models for common special features in the process industry. The specific database includes equipment, equipment state, equipment connections, cause-and-effect models for components and special features, external events, and initial events in a particular process. This methodology will first search the associated specific database for cause-and-effect models before it turns to the public database. In other words, the specific database takes priority over the public database.

4.1 Public Database

As pointed out in Chapter III, each chemical plant is unique, however, they have many universal equipment, materials, and processes. For those universal components, their functionality, failure modes, failure mechanisms, and failure effects have been explored extensively. This knowledge is stored in eight tables in the public database in this methodology. These eight tables and their structures are covered through Sections 4.1.1 to 4.1.3.

Note that numbers in the parentheses in the type column indicate the length of text. Primary keys are pointed out in the tables.

4.1.1 Tables Related to Component Models

During fault tree construction, component cause-and-effect models are directly drawn from digraph tables, failure modes and effects tables, and external events tables. External events are highly dependent on the system and its environment, which are better considered in the specific database. Digraph relationship and failure modes and effects of a certain component are influenced by its state in the system and the environment as well. However, they have a decent level of independence and can be extracted. System-independent information is included in the public database while system-dependent information enters the specific database for the particular process. Two tables in the public database, the public digraph table and the public failure modes and effects table, are related to component models, which are presented separately hereafter.

Public Digraph Table

The public digraph table is devoted to model the functionality of common components and the relationship between their inputs and outputs. The data fields and their data types are shown in Table 4.1.

Table 4.1 Design view of the public digraph table

Field	Type	Description
Index	AutoNumber (key)	Record index
Equipment type	Text (25)	Equipment type
Equipment Subtype	Text (25)	Equipment sub type
Character	Text (25)	Primary characteristics
Equipment State	Text (10)	Equipment state
Input Variable	Text (10)	Affecting variable
Output Variable	Text (10)	Affected variable
Relationship	Number	Influence of input on output
Comment	Text (100)	Further comments

Access lookup wizards are applied to the input variable, the output variable, and the relationship to assist users when entering data. Lookup relationships are defined in Table 4.2 and Table 4.3, respectively.

Table 4.2 Lookup relationship for “relationship”

Relationship	Description
-10	Large decrease
-1	Medium decrease
+1	Medium increase
+10	Large Increase

Table 4.3 Lookup relationship for “input variable” and “output variable”

Abbreviation	Process Variable
C	Concentration
Cin	Inlet concentration
Cout	Outlet concentration
Fcl,in	Cool inlet flow rate
Fcl,out	Cool outlet flow rate
Fh,in	Hot inlet flow rate
Fh,out	Hot outlet flow rate
Fin	Inlet flow rate
Fout	Outlet flow rate
L	Level
P	Pressure
Pin	Inlet pressure
Pout	Outlet pressure
Sigin	Input signal
Sigout	Output signal
T	Temperature
Tcl,in	Cool inlet temperature
Tcl,out	Cool outlet temperature
Th,in	Hot inlet temperature
Th,out	Hot outlet temperature
Tin	Inlet temperature
Tout	Outlet temperature
Uin	Power

Public Failure Modes and Effects Table

For many common materials, processes, equipment and control mechanisms, their failure modes, failure mechanisms and failure effects have already been explored.

This knowledge is especially useful to trace fault propagation in the construction of fault trees. Some failure modes introduce process deviations while others may change the relationships between input variables and output variables of a certain component. Failure modes and effects table stores this information, and its structure is shown in Table 4.4.

Table 4.4 Design view of the public failure modes and effects table

Field	Type	Description
Index	AutoNumber (key)	Record index
Equipment type	Text (25)	Equipment type
Equipment Subtype	Text (25)	Equipment sub type
Equipment State	Text (10)	Equipment state
Failure modes	Text (100)	Failure modes
Input Variable	Text (10)	Affecting variable
Output Variable	Text (10)	Affected variable
Relationship	Number	Influence of input on output
Comment	Text (100)	Further comments

Again, lookup wizards are applied to the input variable, the output variable, and the relationship, which have been shown in Table 4.2 and Table 4.3.

4.1.2 Tables Related to Failure Rate Calculation

Good design not only meets the current needs, but also considers flexibility for future development and extendibility. Although this dissertation does not include the evaluation of fault trees, tables that may be used in the calculation of fault tree models are developed for future research. These tables store the failure rate data and failure distribution for widely used equipment such as pump, seals, or valves in the chemical processes. Public failure rate tables are shared by all the fault tree files, covering the failure rate data for common components. Failure distributions are extracted from the failure distribution table and used together with data from the failure rate table to determine failure probabilities. The public failure rate table and the failure distribution table are presented in the following context.

Public Failure Rate Table

Failure probability is determined by failure rate and time together with failure distribution. The design view of the public failure rate table is shown in Table 4.5. Calculation scheme of basic events is presented in Section 4.2.4.

Table 4.5 Design view of the public failure rate table

Field	Type	Description
Index	AutoNumber (key)	Record index
Equipment Type	Text (25)	Equipment, process or human
Equipment Subtype	Text (25)	Equipment manufacture, ID etc.
Character1	Text (25)	Primary characteristic
Character2	Text (25)	Characteristic
Character3	Text (25)	Characteristic
Failure modes	Text (100)	Failure modes
Parameter1	Number	Failure rate parameter
Parameter2	Number	Failure rate parameter
Parameter3	Number	Failure rate parameter
Unit	Date (Y,M,D,H)	Unit of failure rate parameter
Description	Text (100)	Description of the failure
Source	Text (50)	Source of the failure rate data
Comment	Text (100)	Other comments

A lookup wizard is available for the unit with the listed values of Y, M, D, and H indicating year, month, day, and hour, respectively.

Failure Distribution Table

The failure distribution table as shown in Table 4.6 mainly takes care of different failure distribution functions. The purpose of making failure distributions into the format of a database table is to have the extendibility and flexibility to include future failure distributions. Any possible failure rate distribution can be easily included by simply adding a record in this table.

Table 4.6 Design view of the failure distribution table

Field	Type	Description
DistributionID	Number (key)	Failure distribution ID
Formula	Text (50)	Corresponding calculation formula
Range	Text (150)	The applicable area of this distribution
Comment	Text (100)	Further comments

4.1.3 Tables Related to Special Features

As mentioned in Chapter III, special features are critical in fault tree synthesis, hence the database representation of special features. According to the generalized fault tree model for special features described in Chapter III, there are four tables defined that pertain to special features, which are the special features table, the components table, the capability table, and the instrument failure effects table.

Some entries such as controlled variable, component ID, *etc* cannot be decided precisely in the public database in advance for any particular process. When users select to use a cause-and-effect model for special features in the public database for a specific process, the selected records are copied to the specific database for that process automatically. User inputs are required for equipment ID for all the components, controlled variables, or compensated failures.

Public Special Features Table

The special features table includes primary information regarding controlled variables, compensated failures, and special feature types (feedback, feedforward, bypass, trip, or relief). The table structure and data types are shown in Table 4.7.

Table 4.7 Design view of the public special feature table

Field	Type	Description
Special feature ID	Number (key)	Special feature ID
Type	Text (50)	Type of special feature (feedback, bypass etc)
EquipmentID	Text (10)	Equipment ID of the controlled variable
Controlled variable	Text (10)	Controllable variable
Compensated failure	Text (100)	Compensated variable (feedforward)
Comments	Text (100)	Other comments

Here equipment ID, controlled variable, and compensated failure are left with a default value. When users select a cause and effect model in the public database for a particular system, all the relevant records in the special feature components table, the capability table, and the instrument failure effects table are automatically updated accordingly. A lookup wizard is defined for “type” as in Table 4.8.

Table 4.8 Lookup relationship for “type” in the special feature table

Type	Description
Feedback	Feedback control
Feedforward	Feedforward control
Bypass	Bypass protection
Trip	Trip system
Relief	Pressure relief system
Nested	Nested control
Ratio	Flow ratio control

Public Special Feature Components Table

Information on constituent components of any special features is stored in the special feature components table. As mentioned above, components ID are given some default numbers in the public database. When a user selects the associated special feature ID, the methodology requires the user to input equipment IDs for constituent components. Related records in the capability table and instrument failure effects table are updated automatically while the user enters the data. The design view of the public special feature components table is shown in Table 4.9.

Table 4.9 Design view of the public special feature components table

Field	Type	Description
Index	AutoNumber (key)	Record index
Components ID	Text (10)	Equipment ID
Special feature ID	Integer	Special feature ID
Structure components	Text (50)	Components of the special feature
Comments	Text (100)	Other comments

Public Special Feature Instrument Failure Effects Table

Instrument failures of the special features play a fundamental role in the modeling of its cause-and-effect relationship. Different process control and safe guards have different tolerability toward the same instrument failures. According to the generalized cause-and-effect model for special features described in Chapter III, the effects of any instrument failure are categorized as either “inactive” or “unhealthy”. The program searches the instrument failures of the concerned special feature and adds them under the desired sub-tree according to their effects. The data structure design of the public special feature instrument failure effects table is shown in Table 4.10.

Table 4.10 Design view of the public special feature instrument failure effects table

Field	Type	Description
ID	AutoNumber (key)	Instrument failure ID
Special feature ID	Number	Special feature ID
Component ID	Text (10)	Equipment ID
Failure Modes	Text (100)	Instrument failure modes
Effects	Boolean	Unhealthy or inactive
Comments	Text (100)	Other Comments

For the sake of programming, the effects of instrument failures are represented by Boolean variables in the database. The lookup relationship defined for “effects” is shown in Table 4.11.

Table 4.11 Lookup relationship for special feature instrument failure “effects”

	Effects
Yes	Unhealthy
No	Inactive

Public Special Feature Capability Table

Different special features have different control or compensation capability with regard to process deviations. Even with the same special feature scheme, the capability of controlling or compensating faults depends on the way it is used in the process and the environment it is surrounded by. The public special feature capability table stores the information of capability toward process deviations and faults. The design view of the public special feature capability table is provided in Table 4.12.

Table 4.12 Design view of the public special feature capability table

Field	Type	Description
ID	AutoNumber (key)	Record index
Special feature ID	Number	ID
Component ID	Text (10)	Equipment ID
Affected variable	Text (10)	Process variable / failures
Deviation	Number	Deviation of affecting variable
Effects	Boolean	Uncontrollable or controllable
Comments	Text (100)	Other comments

The lookup wizard has been defined for “deviation”, which is similar to that of “relationship” shown in Table 4.2.

For the sake of programming, the capability of special features is represented by Boolean variables in the database. The lookup relationship defined for “capability” is shown in Table 4.13.

Table 4.13 Lookup relationship for special feature instrument failure “capability”

	Effects
Yes	Controllable
No	Uncontrollable

4.2 Specific Database

Contrary to the public database, specific databases are associated with each particular process, which is represented by a Microsoft Visio ® file in this work. Information regarding equipment, component modeling, top events, failure rates, and special features are included in this database. It should be emphasized that all tables in the specific databases are owned by the associated process only and not available for use by other processes.

4.2.1 Tables Related to Equipment

As described in Chapter III, each item of equipment in the process is represented by one block in the system block diagram. Some characteristics of the equipment are required in the methodology. Equipment ID, system boundary, equipment type, equipment characteristics, equipment states, and connections between them are stored in three tables, respectively. Data fields and data types of these three tables are presented in this section.

Equipment Table

Equipment table is dedicated to store information about equipment ID, boundary, equipment type, subtype, manufacturers, and other characteristics. Equipment type, subtype, and other characteristics are used to locate its failure rate data in the public database if they do not exist in the specific failure rate table. Data structure and types are given in Table 4.14. Equipment IDs are taken as the primary key for this table. It is unique and not allowed to be duplicated.

Table 4.14 Design view of the equipment table

Field	Type	Description
Equipment ID	Text (20) (key)	Unique equipment ID
Boundary	Boolean	System boundary
Equipment Type	Text (25)	Equipment type
Equipment Subtype	Text (25)	Equipment subtype
Character1	Text (25)	Character 1
Character2	Text (25)	Character 2
Character3	Text (25)	Character 3
Manufacturer	Text (25)	Manufacturer
Comment	Text (100)	Comments

The entry of system boundary is a Boolean variable to indicate whether the equipment is on a physical boundary or not. Deviations of input process variables of the equipment on a physical boundary will not be developed further and remain as basic events.

Equipment States Table

The cause-and-effect models for components are determined by not only its own characteristics but also its initial states and normal states in the system. Digraphs, failure modes and effects, external events, and initial events are directly dependent on the states of the equipment. Such kind of information is in the equipment states table. Only equipment that has more than one possible state has corresponding records in this table. Table 4.15 shows the data structure of the equipment states table.

Table 4.15 Design view of the equipment states table

Field	Type	Description
Equipment ID	Text (20) (Key)	Unique equipment ID
Initial State	Text (10)	Initial equipment state
Normal State	Text (10)	Normal state
Comment	Text (100)	Further comments

Topology Table

The connections between components are used to trace down concerned process deviations during fault tree construction. This information is stored in the topology table, which can be automatically obtained from process block diagram file by computer codes. The database design view of the topology table is shown in Table 4.16.

Table 4.16 Design view of the topology table

Field	Type	Description
ID	AutoNumber	Record index
FromEquipID	Text (20) (Key)	Starting equipment ID
FromPortName	Text (10)	Port name
ToEquipID	Text (20)	Ending equipment ID
ToPortName	Text (10)	Port name

Port Names are defined for each item of equipment, which is normally represented by “in” and “out”.

4.2.2 Tables Related to Component Models

Digraph relationship and failure modes and effects of certain component are influenced by its state in the system and the environment though they have certain degree of independence. System-independent information is included in the public database while system-dependent information for the particular process enters the specific database. Four tables in the specific database, the specific digraph table, the specific failure modes and effects table, the external events table, and the initial events table, are related to component models, which are presented separately hereafter.

Specific Digraph Table

Specific digraph tables are devoted to model the functionality of components in a particular process and the relationship between their inputs and outputs. The fields and their data types are shown in Table 4.17. Different from the public digraph table, equipment ID is available in the specific digraph table. All information about that particular equipment can be found in the equipment related tables.

Table 4.17 Design view of the specific digraph table

Field	Type	Description
Equipment ID	Text (20)	Equipment ID
Equipment State	Text (10)	Equipment state
Input Variable	Text (10)	Affecting variable
Output Variable	Text (10)	Affected variable
Relationship	Number	Influence of input on output
Comment	Text (100)	Further comments

As mentioned in the public database, Microsoft Access ® lookup wizards are applied to the relationship, the input variable, and the output variable to guide the user to enter data, which are shown in Table 4.2 and Table 4.3, respectively.

Specific Failure Modes and Effects Table

Chemical plants may contain materials, processes, equipment or control mechanisms that are not commonly found in the process industry. Failure modes, failure mechanisms, and failure effects for those components must be entered in the failure modes and effects table for the generated fault tree models to be correct and complete. A failure can lead to process deviations, or change the functional relationship between inputs and outputs. The input variable column is left blank if the failure mode only introduces process deviations. The data structure and data types of specific failure modes and effects table are presented in Table 4.18.

Table 4.18 Design view of the specific failure modes and effects table

Field	Type	Description
Equipment ID	Text (20)	Equipment ID
Equipment State	Text (10)	Equipment state
Failure modes	Text (100)	Failure modes
Input Variable	Text (10)	Affecting variable
Output Variable	Text (10)	Affected variable
Relationship	Number	Influence of input on output
Comment	Text (100)	Further comments

Again, lookup wizards are defined for the input variable, the output variable, and the relationship, which have been shown in Table 4.2 and Table 4.3.

External Events Table

External environment influences any process in a sophisticated way. The effects of external events are highly dependent on the process under study. No one can predict the effects of an external event without the information of the concerned process. Some external events are typical common cause failures such as external fires and flooding. All the external events and their effects are listed in the external events table. Similar to failure modes and effects, external events can either introduce process deviations or change the functional relationship between inputs and outputs. To save computer storage, external events can be defined not only to equipment, but also to the sub-system or system, in which all the components in the system inherit the external events. The data structure of external events table is shown in Table 4.19.

Table 4.19 Design view of the external events table

Field	Type	Description
EquipmentID	Text (20)	Affected equipment ID
Description	Text (100)	Description
Input Variable	Text (10)	Affecting variable
Output Variable	Text (10)	Affected variable
Relationship	Number	Relationship
Comment	Text (100)	Further comments

Again, a lookup wizard is defined for “relationship” to guide users when entering data, which is shown in Table 4.2.

Initial Events table

The cause-and-effect relationship for a particular component may be influenced by some initial events as well. The data structure is essentially the same as that of external events table, which is provided in Table 4.20.

Table 4.20 Design view of the initial events table

Field	Type	Description
EquipmentID	Text (20)	Affected equipment ID
Description	Text (100)	Description
Input Variable	Text (10)	Affecting variable
Output Variable	Text (10)	Affected variable
Relationship	Integer	Relationship
Comment	Text (100)	Further comments

A lookup wizard is defined for “relationship” as in Table 4.2.

4.2.3 Tables Related to Failure Rate Calculations

As pointed out in the previous section, some facilities may contain materials, processes, equipment, and control mechanisms that are not commonly found in the process industry. Failure rate data for those components might not be available in the public database. In some cases, failure rate data for some commonly used devices can be different from the data in the public database because of the unique environment and process conditions in a particular system. Under these circumstances, failure rates must be stored in the specific database to override the data in the public database. The data design view of the specific failure rate table is shown in Table 4.21. The fundamentals for calculating the failure rates remain the same, and failure distribution are extracted from the public database.

Table 4.21 Design view of the specific failure rate table

Field	Type	Description
Equipment ID	Text (20) (key)	Equipment, process or human
Failure Modes	Text (5)	Failure modes
Parameter1	Number	Failure rate parameter
Parameter2	Number	Failure rate parameter
Parameter3	Number	Failure rate parameter
Unit	Text (1)	Time unit of failure rate
Description	Text (100)	Description of the failure
Source	Text (50)	Source of the failure rate data
Comment	Text (100)	Other comments

A lookup wizard is available for the unit with the listed values of Y, M, D, and H indicating year, month, day, and hour, respectively.

4.2.4 Tables Related to Special Features

It is rather difficult and impractical to predict all the special features and their corresponding cause-and-effect models in the public database. In particular, the cause-and-effect models selected from the public database by the users are copied to the associated specific database to match the equipment IDs and facilitate the automatic fault tree construction procedure. Similar to that in public database, there are four tables defined in the specific database that pertain to special features, which are the special features table, components table, capability table and instrument failure effects table.

As mentioned in Section 4.1.3, whenever users choose a cause-and-effect model for special features in the public database for a specific process, the selected records are copied to the specific database for that process automatically. User inputs are required for equipment IDs for the components, controlled variables, or compensated failures.

Specific Special Features Table

The special features table includes information regarding the controlled variable, the compensated failure, and the special feature type (feedback, feedforward, bypass, trip, or relief). The table structure and data type are shown in Table 4.22.

Table 4.22 Design view of the specific special features table

Field	Type	Description
Special feature ID	Number (key)	Special feature ID
Type	Text (50)	Type of special feature (feedback, bypass etc)
Equipment ID	Text (10)	Equipment ID
Controlled variable	Text (10)	Controllable variable
Deviation	Number	Controlled deviation
Comments	Text (100)	Other comments

When users change equipment IDs, controlled variables, or compensated failures, all the relevant records in the special feature components table, capability table, and instrument failure effects table are automatically updated accordingly. A lookup wizard is defined for “type” as in Table 4.8. Here the field “deviation” is reserved for some uni-direction process control or protection.

Specific Special Feature Components Table

Information on constituent components of any special features is stored in the special feature components table. The design view of the specific special features components table is shown in Table 4.23.

Table 4.23 Design view of the specific special feature components table

Field	Type	Description
ID	AutoNumber (key)	Record index
Components ID	Text (10)	Equipment ID
Special feature ID	Number	Special feature ID
Components	Text (50)	Components of the special feature
Comments	Text (100)	Other comments

Specific Special Feature Instrument Failures Effects Table

According to the generalized cause-and-effect model for special features, the effects of any instrument failure are categorized as either “inactive” or “unhealthy”. The program searches the instrument failures of the concerned special feature and adds them under the desired sub-tree according to their effects. The data structure design of public special feature instrument failures effects table is shown in Table 4.24, which is essentially the same as that of public database.

Table 4.24 Design view of the public special feature instrument failures effects table

Field	Type	Description
ID	AutoNumber (key)	Instrument failure ID
Special feature ID	Number	Special feature ID
Component ID	Text (10)	Equipment ID
Failure Modes	Text (100)	Instrument failure modes
Effects	Boolean	Unhealthy or inactive
Comments	Text (100)	Other comments

Specific Special Feature Capability Table

Similar to the public special feature capability table, the specific special feature capability table stores the information of control or compensation capability of special features toward process deviations and faults. Its design view is shown in Table 4.25.

Table 4.25 Design view of the public special feature capability table

Field	Type	Description
ID	AutoNumber (key)	Record index
Special feature ID	Number	ID
Component ID	Text (10)	Equipment ID
Affected variable	Text (10)	Process variable / failures
Deviation	Number	Deviation of affecting variable
Capability	Boolean	Uncontrollable or controllable
Comments	Text (100)	Other comments

4.2.5 Top Events Table

Top event identification is crucial in the fault tree construction although its importance is always undermined. In this work, simple checklists are used to help the analyst identify potential hazards in the process. However, a detailed and systematic hazard identification technique is preferred. All top events associated with the process are saved in the top events table, whose data structure is defined in Table 4.26.

Table 4.26 Design view of the top events table

Field	Type	Description
Top Event ID	AutoNumber (key)	Record index
GateID	Number	Gate ID
Description	Text (250)	Top event description
Comment	Text (100)	Other comments

4.2.6 Table Related to Fault Trees

Fault tree files can take many forms. Textual, tabular, graphical, or database forms can all be used to represent fault trees. In this work, fault trees are stored in database tables for the sake of data manipulation. Prototype computer codes are programmed to convert database tables to Visio file for analysts to review and update fault trees. Four tables are associated with fault tree models: basic events table, gates table, fault tree table, and stack table, which are discussed separately in this section.

Among them stack tables are used in the programming to record all the intermediate gates that need further development.

Basic Events Table

Basic events are defined to be primary failures, external events, or human activities that are not to be developed further in a fault tree. It can be failures of sub-system, equipment, or equipment parts, decided by the analysis resolution desired. The fields and their types are presented in Table 4.27.

Table 4.27 Design view of the basic events table

Field	Type	Description
Name	Text (20) (key)	Name of the basic event
Equipment ID	Text (20)	Unique equipment ID (P&ID)
Distribution	Number	Failure distribution
Mission Time	Number	Mission time or failure rate
Unit	Text (1)	Time unit
Description	Text (250)	Description of the failure
Source	Text (50)	Source of the information
Probability	Number	Calculated probability
Comment	Text (100)	Further comments

Lookup relationship has been defined for “unit” and “distribution”, in which distribution is extracted from the failure distribution table in the public database. A lookup wizard is available for the unit with the listed values of Y, M, D, and H indicating year, month, day, and hour, respectively.

To calculate the probability of basic events, the corresponding failure distribution is decided by the distribution field, and then the failure rate is extracted from the specific database or the public database. Finally mission time is used with the failure rate to calculate the probability. This procedure for calculation of basic event probability is shown in Figure 4.2.

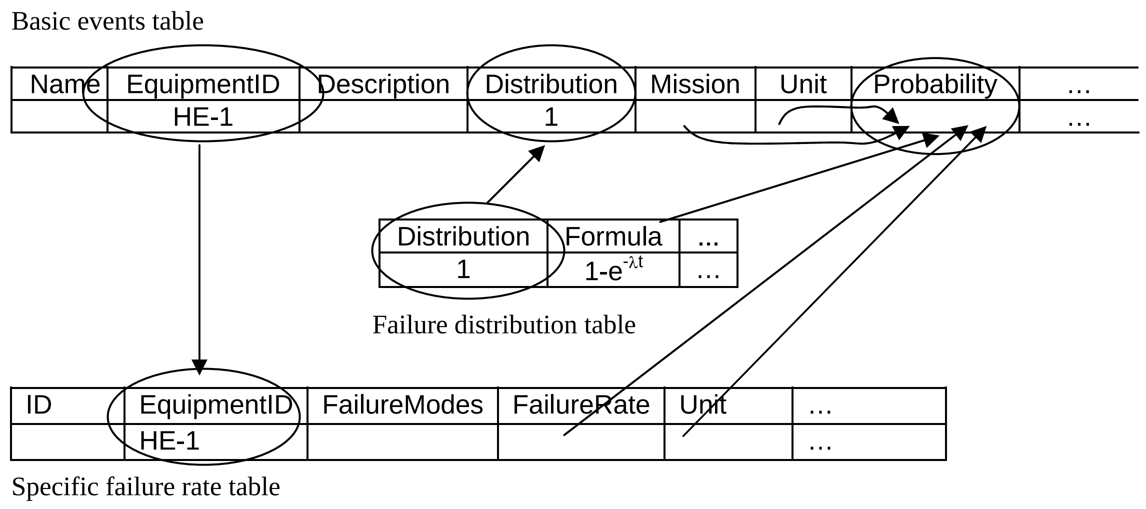


Figure 4.2 Calculation of basic event probability

This figure extracts failure rates or probability from the specific database. Calculation of basic event probability using failure rates from the public database is similar.

Gates Table

The records in the gates table are a one-to-one map of the intermediate gates in the fault tree. Data structure of the gates table is shown in Table 4.28.

Table 4.28 Design view of the gates table

Field	Type	Description
GateID	Number (Key)	Gate ID
Description	Text (250)	Description of the gate event
Completeness	Boolean	Completeness of the gate
EquipmentID	Text (20)	Equipment ID
Process variable	Text (10)	Process variable
Deviation	Number	Deviation
Probability	Number	Calculated probability
Time	Date	Date when calculated
Comment	Text	Further comments

Here a completeness field is used to indicate whether the gate is completed or not. If not, the gate is an intermediate event and needs to be developed further.

Fault Tree Table

The fault tree table stores the structure of fault trees, namely, the parent, its child and the logic between them. Many gates have more than one child, thus more than one record in the fault tree table. Data fields and their types are presented in Table 4.29.

Table 4.29 Design view of the fault tree table

Field	Type	Description
ID	AutoNumber (key)	Record index
GateID	Number	Gate ID
Type	Text (3)	Gate logic (AND,OR,EQU,XOR)
Child	Text (10)	Child of the gate

Stack Table

There is no definition of a data structure “stack” in Microsoft Visual Basic ®. The stack table is employed to serve the function of a stack, which is a typical “first come last out” data structure. All the intermediate events are pushed into the stack table in the program. AutoNumber is a data type in Microsoft Access ® such that it will be automatically assigned in an increasing order when a record is inserted. Each time the gate event with the maximum ID is popped out from the stack table to ensure “first come last out” principle. The design view of the stack table is presented in Table 4.30.

Table 4.30 Design view of the stack table

Field	Type	Description
ID	AutoNumber (key)	Record index
GateID	Number	Gate ID

4.3 Summary of Chapter IV

This chapter has been devoted to the discussion of database structures. A relational database system has been developed to implement the methodology. Microsoft Access ® is the database management system selected for this purpose.

There are two databases in this project – the public database and the specific database, discussed in Section 4.1 and 4.2, respectively. The public database includes digraphs, failure modes and effects and failure rate data for common devices and cause-and-effect models for common special features in the process industry. Eight tables are included in the public database and are divided into three types: tables related to component models (4.1.1), tables related to failure rate calculation (4.1.2), and tables related to special features (4.1.3). The specific database includes information on equipment, equipment state, equipment connections, cause-and-effect models for components and special features, external events, and initial events in a particular process. There are seventeen tables involved in the specific database. They are categorized into six types: tables related to equipment (4.2.1), tables related to component models (4.2.2), tables related to failure rate calculation (4.2.3), tables related to special features (4.2.4), top event table (4.2.5), and tables related to fault trees (4.2.6).

This methodology first searches the associated specific database for cause-and-effect models before it turns to the public database. In other words, the specific database takes priority over the public database.

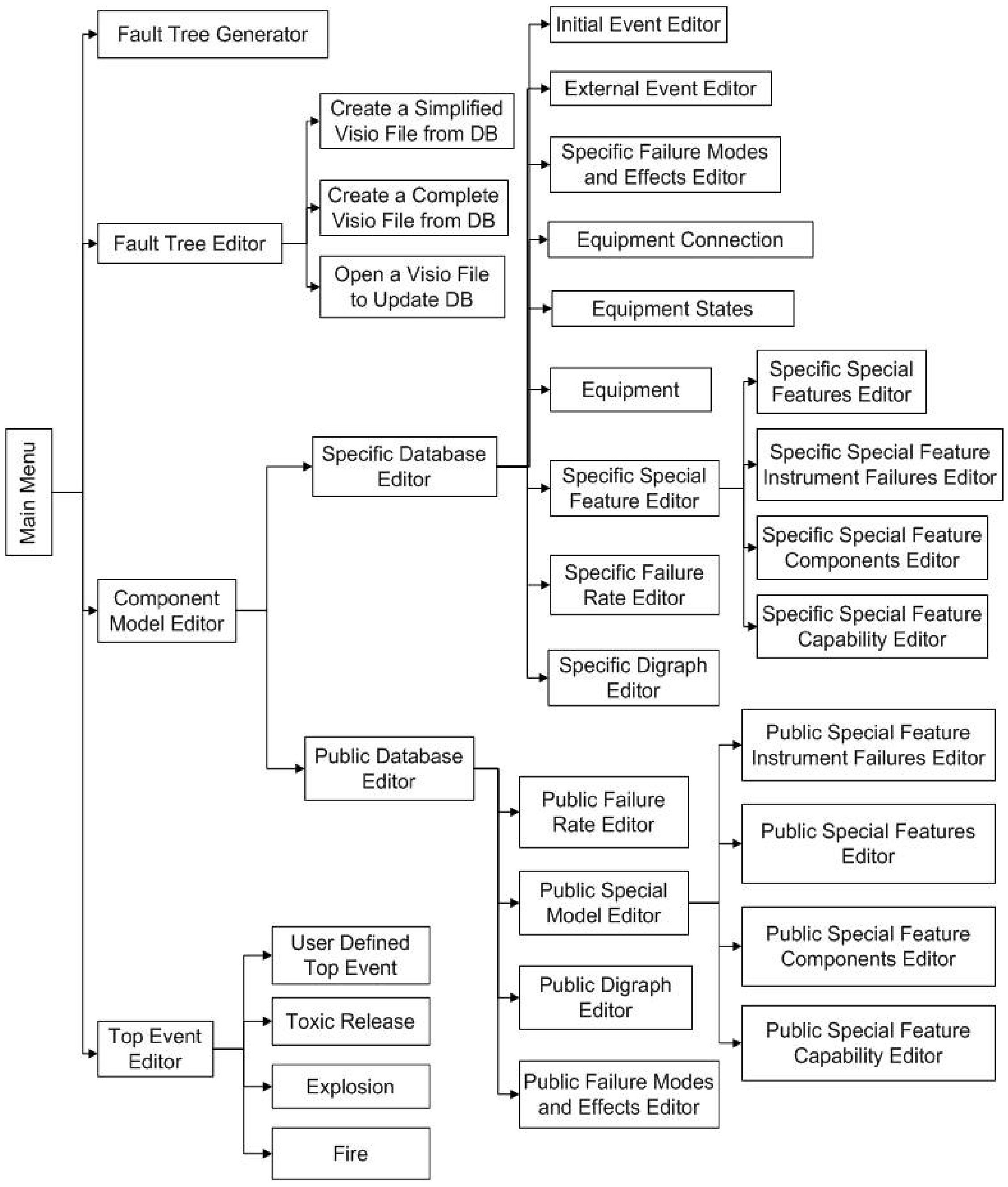
CHAPTER V

PROGRAM DESCRIPTION

5.1 Overview

A prototype computer package has been developed to demonstrate the capability of this proposed methodology. This prototype computer package consists of four major modules: top event editor, component model editor, fault tree generator, and fault tree editor, which are addressed in detail in the following sections. The computer language used to implement this program is Microsoft Visual Basic ®. Microsoft Access is the primary carrier for the database system. Users can modify records in the database through Microsoft Access ® directly. Database connectors are programmed so that users can add, update, and delete records in the public or specific database in the component model editor. Process block diagrams and fault trees are in the form of Microsoft Visio files.

As mentioned, this program consists of four major modules. System structure is shown in Figure 5.1. Only major error handlings such as “database not found” are considered in the programming. “Help” and “exit” buttons are provided for each module.



Main Menu
Fault Tree Generator
Fault Tree Editor
Create a Simplified Visio File from DB
Create a Complete Visio File from DB
Open a Visio File to Update DB
Component Model Editor
Specific Database Editor
Initial Event Editor
External Event Editor
Specific Failure Modes and Effects Editor
Equipment Connection
Equipment States
Equipment
Specific Special Feature Editor
Specific Special Features Editor
Specific Special Feature Instrument Failures Editor
Specific Special Feature Components Editor
Specific Special Feature Capability Editor
Specific Failure Rate Editor
Specific Digraph Editor
Public Database Editor
Public Failure Rate Editor
Public Special Model Editor
Public Digraph Editor
Public Failure Modes and Effects Editor
Public Special Feature Instrument Failures Editor
Public Special Features Editor
Public Special Feature Components Editor
Public Special Feature Capability Editor
Top Event Editor
User Defined Top Event
Toxic Release
Explosion
Fire

Figure 5.1 System structure

5.2 Top Event Editor

As shown in Figure 5.1, there are six command buttons in the top event editor, two of which are “help” and “exit”. Simple checklists are used to help users identify typical top events such as “fire”, “explosion”, and “toxic release”. A “User-defined top events” button is provided for the user to enter top events directly into the specific database.

The flow chart for “fire” event identification is presented here as an illustration. For fire to occur, fuel, oxygen, and ignition energy must be present. The basic idea in this algorithm is to examine every possible location for these three elements. If the inventory of a flammable/combustible material is enough to maintain a fire if ignited, the program will proceed to check whether the area is classified as intrinsically safe or not. If the answer is “no”, fire events are possible. Gates table, top events table, and fault tree table are then updated. The program then continues to prompt the user to enter information about protections not in the P&ID such as sprinkler system, dry chemical system, or other fire protection systems and their potential failure modes. The top events table, the gates table, and the fault tree table are updated according to the user inputs. Finally, a multi-check box is employed to incorporate the containment failure modes of flammable/combustible materials. This program iterates until all the places containing flammable/combustible materials have been treated. The flow chart of fire event identification is shown in Figure 5.2.

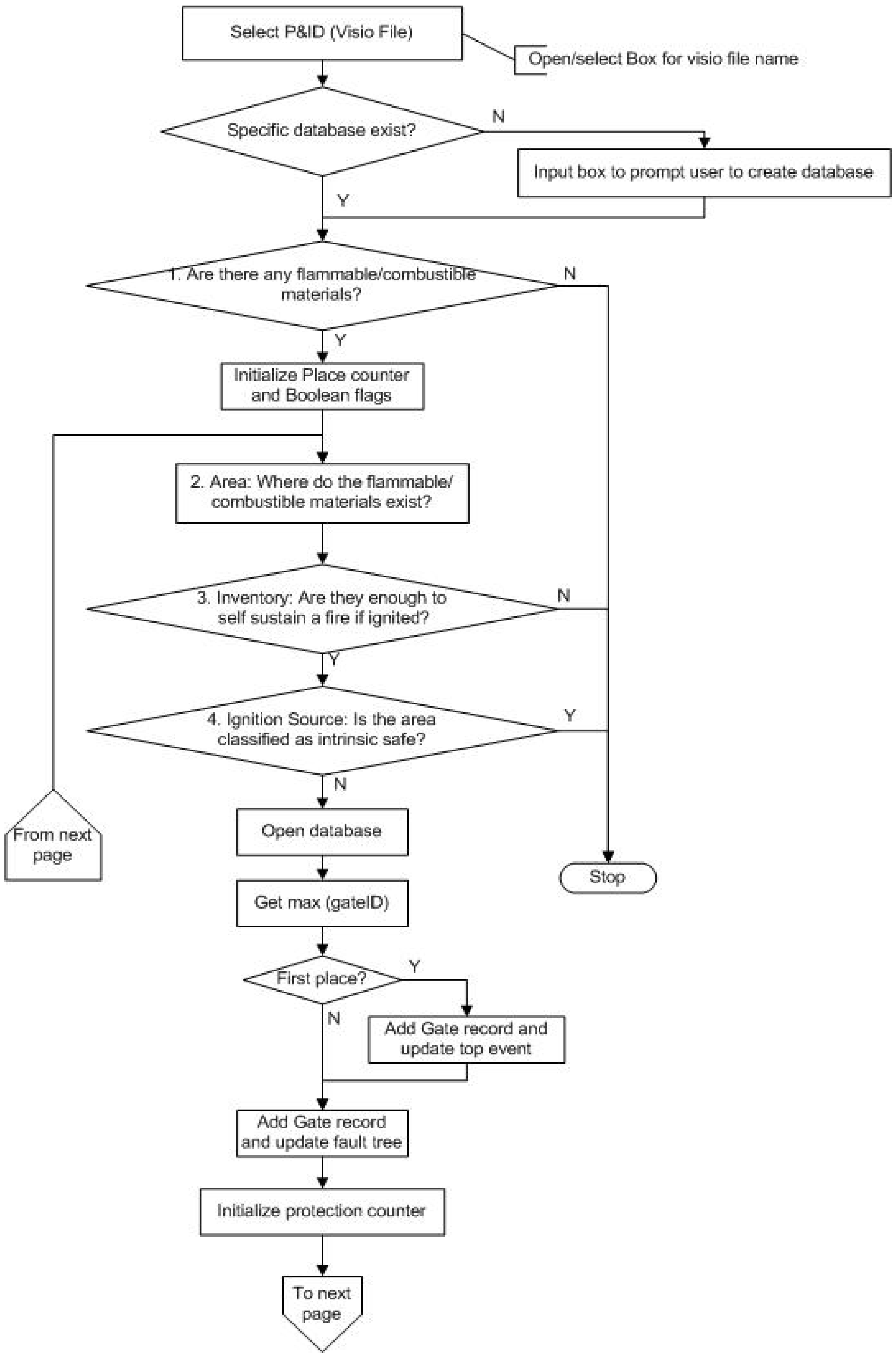

Select P&ID (Visio File)
Open/select Box for visio file name
Specific database exist?
N
Input box to prompt user to create database
Y
1. Are there any flammable/combustible materials?
N
Y
Initialize Place counter and Boolean flags
2. Area: Where do the flammable/ combustible materials exist?
3. Inventory: Are they enough to self sustain a fire if ignited?
N
Y
4. Ignition Source: Is the area classified as intrinsic safe?
Y
N
From next page
Open database
Stop
Get max (gateID)
First place?
Y
N
Add Gate record and update top event
Add Gate record and update fault tree
Initialize protection counter
To next page

Figure 5.2 Fire event identification flow chart

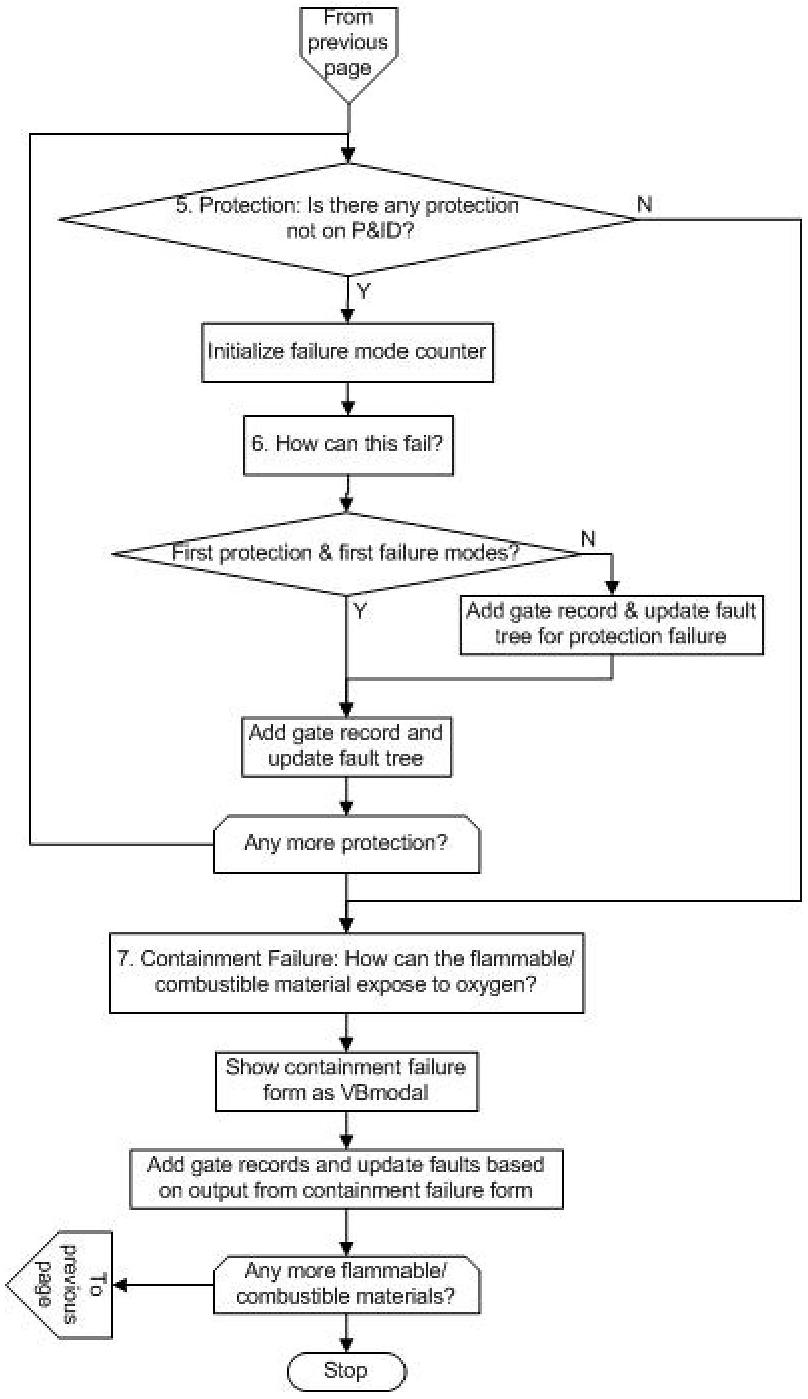
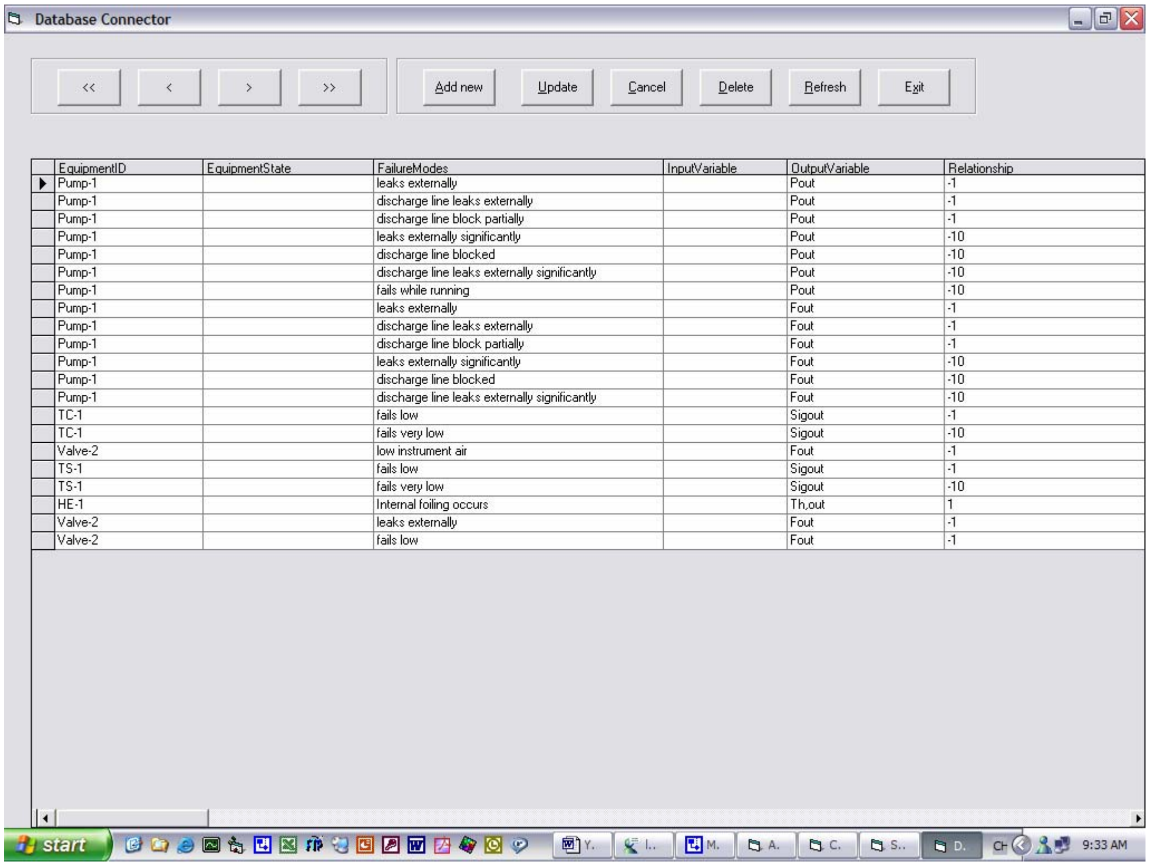

From previous page
5. Protection: Is there any protection not on P&ID?
N
Y
Initialize failure mode counter
6. How can this fail?
First protection & first failure modes?
N
Y
Add gate record & update fault tree for protection failure
Add gate record and update fault tree
Any more protection?
7. Containment Failure: How can the flammable/ combustible material expose to oxygen?
Show containment failure form as VBmodal
Add gate records and update faults based on output from containment failure form
Any more flammable/ combustible materials?
To previous page
Stop

Figure 5.2 continued

5.3 Component Model Editor

As mentioned in the overview section, Microsoft Access ® database management system has been employed to implement the public database and the specific database. Users can manipulate and manage data records directly in the Access interface. Database connectors for the public database and the specific database are provided in this work, especially for those who do not have the Microsoft Access ® software.

Entering the component modeling editor module, users will find two command buttons: public database editor and specific database editor. Each table described in Chapter IV is provided with a database connector. To improve the efficiency of programming, efforts have been carried out to provide a universal database connector via the Access ADO object. Only a database file name and table name need to be specified for each of the 25 table editors. The database connector for the specific failure modes and effects table is shown in Figure 5.3 as an example. All the other table editors take the same form as Figure 5.3.



EquipmentID	EquipmentState	FailureModes	InputVariable	OutputVariable	Relationship
Pump-1		leaks externally		Pout	-1
Pump-1		discharge line leaks externally		Pout	-1
Pump-1		discharge line block partially		Pout	-1
Pump-1		leaks externally significantly		Pout	-10
Pump-1		discharge line blocked		Pout	-10
Pump-1		discharge line leaks externally significantly		Pout	-10
Pump-1		fails while running		Pout	-10
Pump-1		leaks externally		Fout	-1
Pump-1		discharge line leaks externally		Fout	-1
Pump-1		discharge line block partially		Fout	-1
Pump-1		leaks externally significantly		Fout	-10
Pump-1		discharge line blocked		Fout	-10
Pump-1		discharge line leaks externally significantly		Fout	-10
TC-1		fails low		Sigout	-1
TC-1		fails very low		Sigout	-10
Valve-2		low instrument air		Fout	-1
TS-1		fails low		Sigout	-1
TS-1		fails very low		Sigout	-10
HE-1		Internal foiling occurs		Th,out	1
Valve-2		leaks externally		Fout	-1
Valve-2		fails low		Fout	-1

Figure 5.3 The specific failure modes and effects table editor

5.4 Fault Tree Generator

A fault tree generator builds fault tree models automatically by extracting information from the public database or the specific database. The principle and steps are discussed in Chapter III, and a detailed flow chart of the fault tree generator is presented in Figure 5.4. The default description for gates in the program is a combination of the equipment ID, the process variable, and the deviation. Users can modify them if desired.

5.5 Fault Tree Editor

A fault tree editor is provided to review and edit fault trees generated from the fault tree generator. As described in Chapter IV, fault tree structures are stored in the form of a database table in the specific database – fault tree table. Though it is efficient and easy for programming implementation, it is not easy to review and modify.

In order to solve the above mentioned difficulties, three functionalities are provided in this editor, namely, “open a Visio file to update DB”, “create a complete Visio file from DB”, and “create a simplified Visio file from DB”.

Fault trees drawn directly from a computer are normally diverging. A simplification procedure makes them concise and easy to read. This simplification step is discussed in detail in Section 3.5.2. Because fault tree evaluation is not covered in this dissertation, Boolean simplification is not implemented. The simplification function reads the fault tree table, performs a tree simplification procedure and stores the simplified tree as a new Visio file.

User verification is required to ensure the correctness of final fault trees. After reviewing the fault trees generated from computer codes, the user may change fault trees based on their own understanding. This new Visio file is read by “open a Visio file to update DB” button and manipulates database records to match the tree structure.

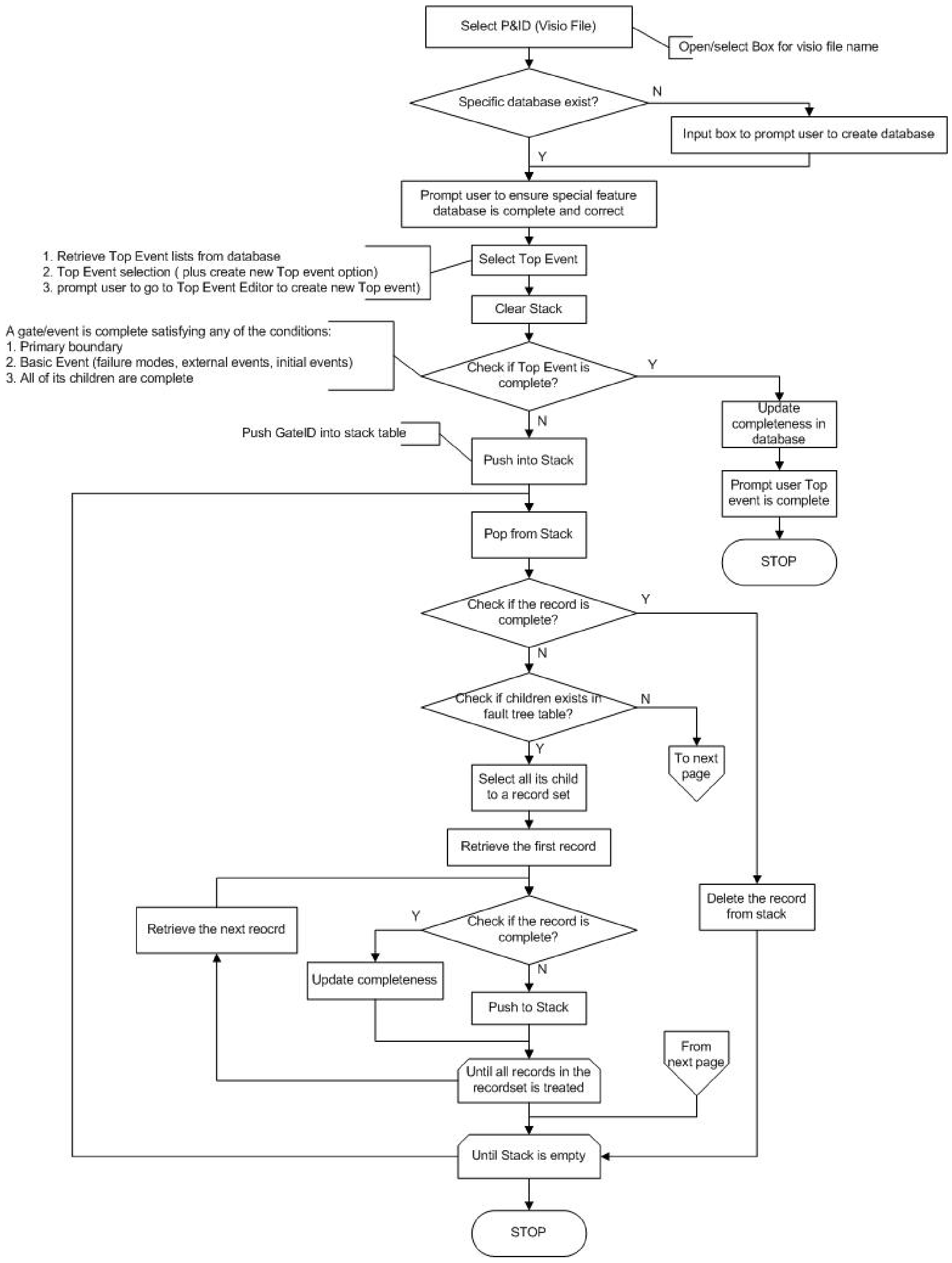

Select P&ID (Visio File)
Open/select Box for visio file name
Specific database exist?
N
Input box to prompt user to create database
Y
Prompt user to ensure special feature database is complete and correct
1. Retrieve Top Event lists from database
2. Top Event selection (plus create new Top event option)
3. prompt user to go to Top Event Editor to create new Top event)
Select Top Event
Clear Stack
A gate/event is complete satisfying any of the conditions:
1. Primary boundary
2. Basic Event (failure modes, external events, initial events)
3. All of its children are complete
Check if Top Event is complete?
Y
Update completeness in database
Prompt user Top event is complete
STOP
N
Push GateID into stack table
Push into Stack
Pop from Stack
Check if the record is complete?
Y
N
Check if children exists in fault tree table?
N
To next page
Y
Select all its child to a record set
Retrieve the first record
Delete the record from stack
Retrieve the next reocrd
Check if the record is complete?
Y
Update completeness
N
Push to Stack
From next page
Until all records in the recordset is treated
Until Stack is empty
STOP

Figure 5.4 Fault tree generator flow chart

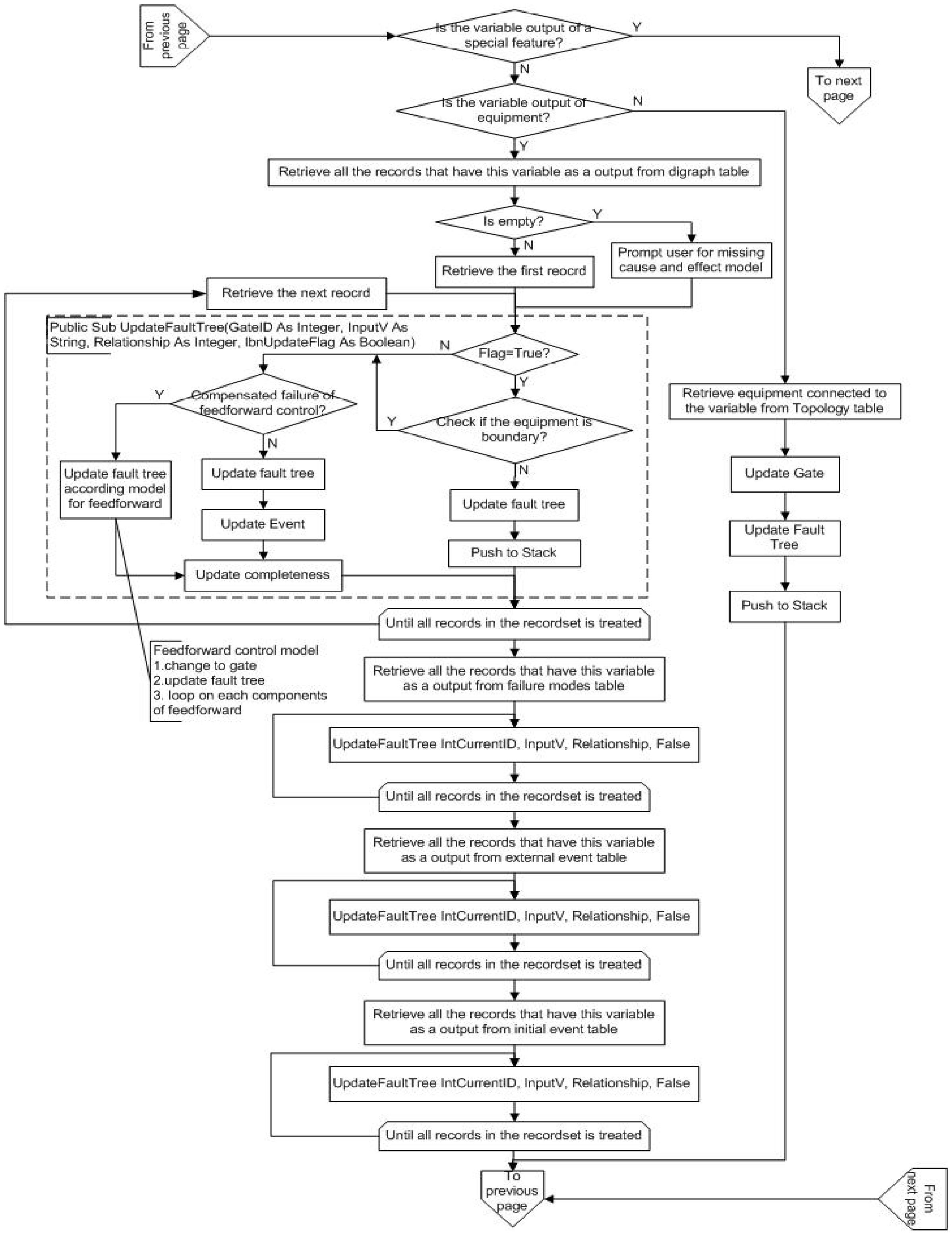

From previous page
Is the variable output of a special feature?
Y
N
To next page
Is the variable output of equipment?
N
Y
Retrieve all the records that have this variable as a output from digraph table
Is empty?
Y
N
Retrieve the first reocrd
Prompt user for missing cause and effect model
Retrieve the next reocrd
Public Sub UpdateFaultTree(GateID As Integer, InputV As String, Relationship As Integer, lbnUpdateFlag As Boolean)
N
Flag=True?
Y
Compensated failure of feedforward control?
Y
N
Check if the equipment is boundary?
Y
N
Update fault tree according model for feedforward
Update fault tree
Update Event
Update completeness
Update fault tree
Push to Stack
Retrieve equipment connected to the variable from Topology table
Update Gate
Update Fault Tree
Push to Stack
Until all records in the recordset is treated
Feedforward control model
1.change to gate
2.update fault tree
3. loop on each components of feedforward
Retrieve all the records that have this variable as a output from failure modes table
UpdateFaultTree IntCurrentID, InputV, Relationship, False
Until all records in the recordset is treated
Retrieve all the records that have this variable as a output from external event table
UpdateFaultTree IntCurrentID, InputV, Relationship, False
Until all records in the recordset is treated
Retrieve all the records that have this variable as a output from initial event table
UpdateFaultTree IntCurrentID, InputV, Relationship, False
Until all records in the recordset is treated
To previous page
From next page

Figure 5.4 continued

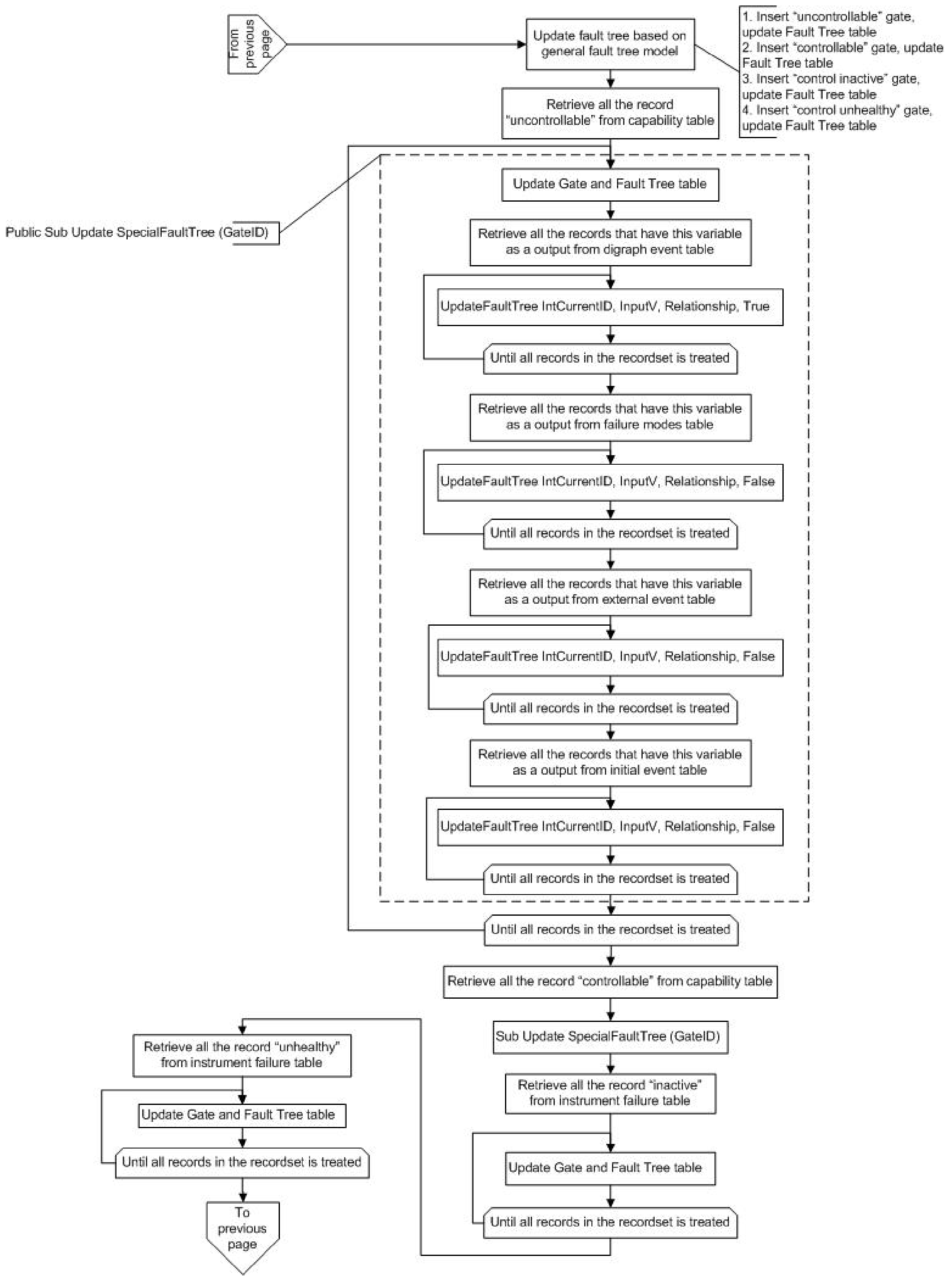
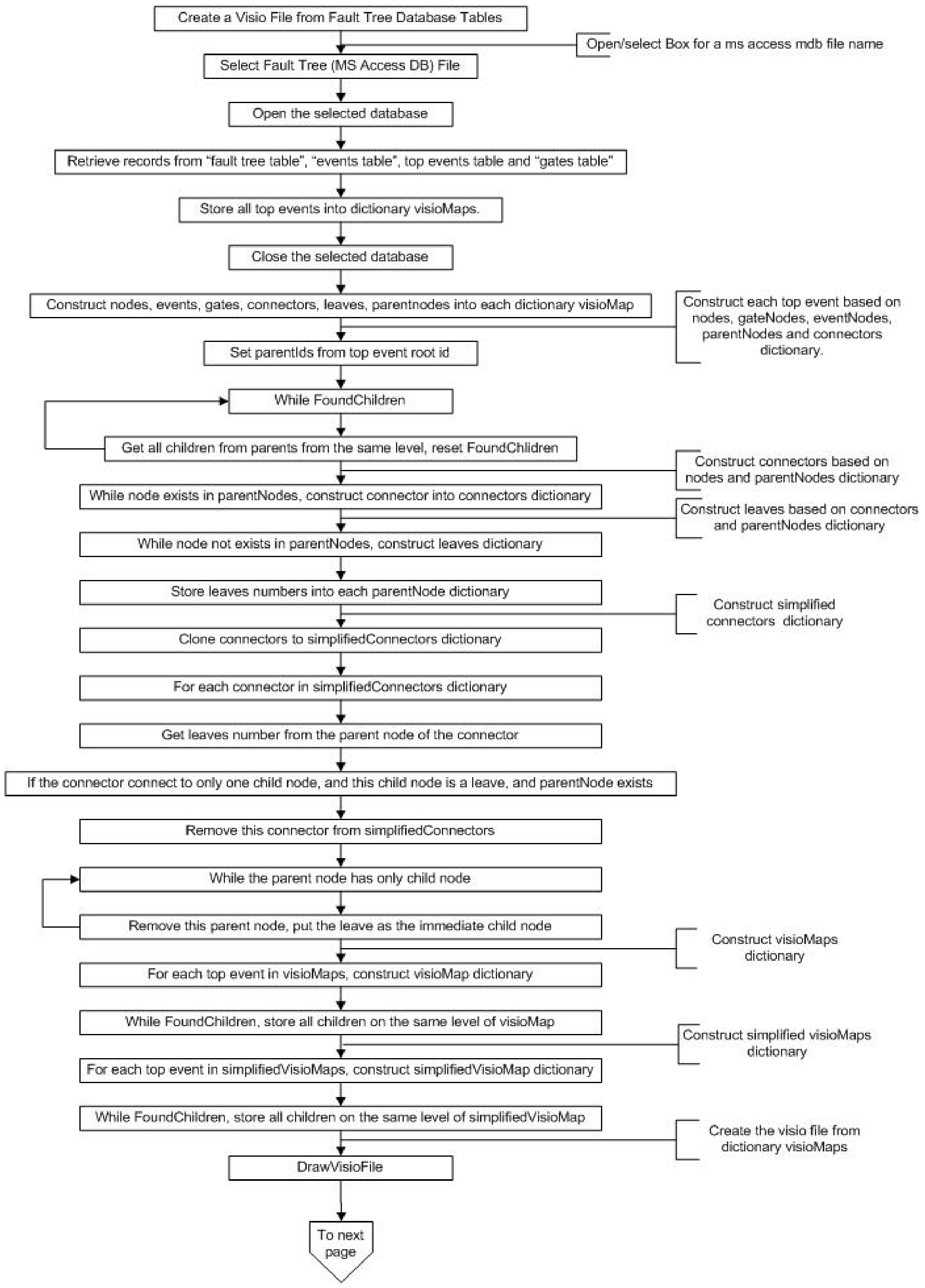

From previous page
Update fault tree based on general fault tree model
1. Insert "uncontrollable" gate, update Fault Tree table
2. Insert "controllable" gate, update Fault Tree table
3. Insert "control inactive" gate, update Fault Tree table
4. Insert "control unhealthy" gate, update Fault Tree table
Retrieve all the record "uncontrollable" from capability table
Public Sub Update SpecialFaultTree (GateID)
Update Gate and Fault Tree table
Retrieve all the records that have this variable as a output from digraph event table
UpdateFaultTree IntCurrentID, InputV, Relationship, True
Until all records in the recordset is treated
Retrieve all the records that have this variable as a output from failure modes table
UpdateFaultTree IntCurrentID, InputV, Relationship, False
Until all records in the recordset is treated
Retrieve all the records that have this variable as a output from external event table
UpdateFaultTree IntCurrentID, InputV, Relationship, False
Until all records in the recordset is treated
Retrieve all the records that have this variable as a output from initial event table
UpdateFaultTree IntCurrentID, InputV, Relationship, False
Until all records in the recordset is treated
Until all records in the recordset is treated
Retrieve all the record "controllable" from capability table
Sub Update SpecialFaultTree (GateID)
Retrieve all the record "inactive" from instrument failure table
Update Gate and Fault Tree table
Until all records in the recordset is treated
Retrieve all the record "unhealthy" from instrument failure table
Update Gate and Fault Tree table
Until all records in the recordset is treated
To previous page

Figure 5.4 continued

The procedures of these three functionalities are discussed in the following context, and their corresponding flow charts are presented to illustrate the detail implementation in the computer, respectively.

Information about top events, gate nodes, event nodes, and connections between them are extracted from the database and stored in “dictionaries”. “Dictionary” is a special data type defined in Microsoft Visual Basic, which has the typical function of a dictionary. Each top event is assigned with one Visio map, in which levels are defined. Each record in the Visio maps dictionary is a new page in the generated Visio document. Numbers of the leaves under each gate are calculated and stored in a dictionary as well. These numbers are used later to determine the distance between the nodes in the same level. To create a simplified Visio fault tree map, each record in the “connections” dictionary is scanned recursively to remove those gates having only one child. A well-structured Visio file for fault trees is drawn based on these dictionaries. Figure 5.5 is the flow chart for creating a Visio file for both the complete and the simplified version of fault trees since they share many functions and subroutines.

To update tables in the specific database from Visio files, top events, gate nodes, event nodes, and connections between them are extracted from the Visio file and recorded in two dictionaries – “nodes” and “parent nodes”. Then nodes, gates and events dictionaries are derived from the parent node and connector dictionaries. All the records containing the same node gate ID as records in the node dictionary are deleted, and then new records are inserted into the database based on the new dictionaries created. Figure 5.6 shows the flow chart for updating the database from Visio files.



Create a Visio File from Fault Tree Database Tables
Open/select Box for a ms access mdb file name
Select Fault Tree (MS Access DB) File
Open the selected database
Retrieve records from "fault tree table", "events table", top events table and "gates table"
Store all top events into dictionary visioMaps.
Close the selected database
Construct nodes, events, gates, connectors, leaves, parentnodes into each dictionary visioMap
Construct each top event based on nodes, gateNodes, eventNodes, parentNodes and connectors dictionary.
Set parentIds from top event root id
While FoundChildren
Get all children from parents from the same level, reset FoundChlidren
Construct connectors based on nodes and parentNodes dictionary
While node exists in parentNodes, construct connector into connectors dictionary
Construct leaves based on connectors and parentNodes dictionary
While node not exists in parentNodes, construct leaves dictionary
Store leaves numbers into each parentNode dictionary
Construct simplified connectors dictionary
Clone connectors to simplifiedConnectors dictionary
For each connector in simplifiedConnectors dictionary
Get leaves number from the parent node of the connector
If the connector connect to only one child node, and this child node is a leave, and parentNode exists
Remove this connector from simplifiedConnectors
While the parent node has only child node
Remove this parent node, put the leave as the immediate child node
Construct visioMaps dictionary
For each top event in visioMaps, construct visioMap dictionary
While FoundChildren, store all children on the same level of visioMap
Construct simplified visioMaps dictionary
For each top event in simplifiedVisioMaps, construct simplifiedVisioMap dictionary
While FoundChildren, store all children on the same level of simplifiedVisioMap
Create the visio file from dictionary visioMaps
DrawVisioFile
To next page

Figure 5.5 Flow chart for creating Visio files of fault trees

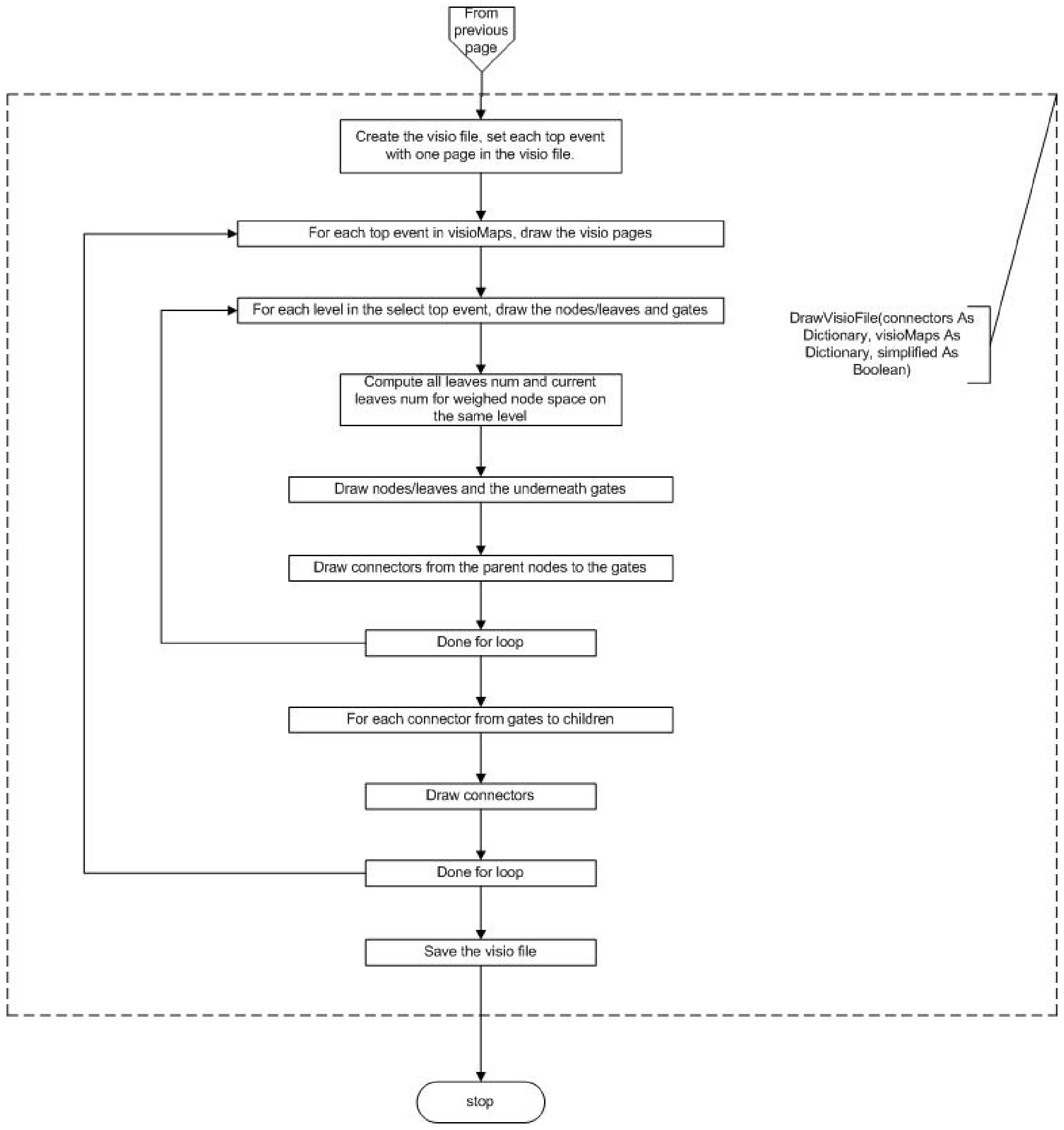

From previous page
Create the visio file, set each top event with one page in the visio file.
For each top event in visioMaps, draw the visio pages
For each level in the select top event, draw the nodes/leaves and gates
Compute all leaves num and current leaves num for weighed node space on the same level
Draw nodes/leaves and the underneath gates
Draw connectors from the parent nodes to the gates
Done for loop
For each connector from gates to children
Draw connectors
Done for loop
Save the visio file
stop
DrawVisioFile(connectors As Dictionary, visioMaps As Dictionary, simplified As Boolean)

Figure 5.5 continued

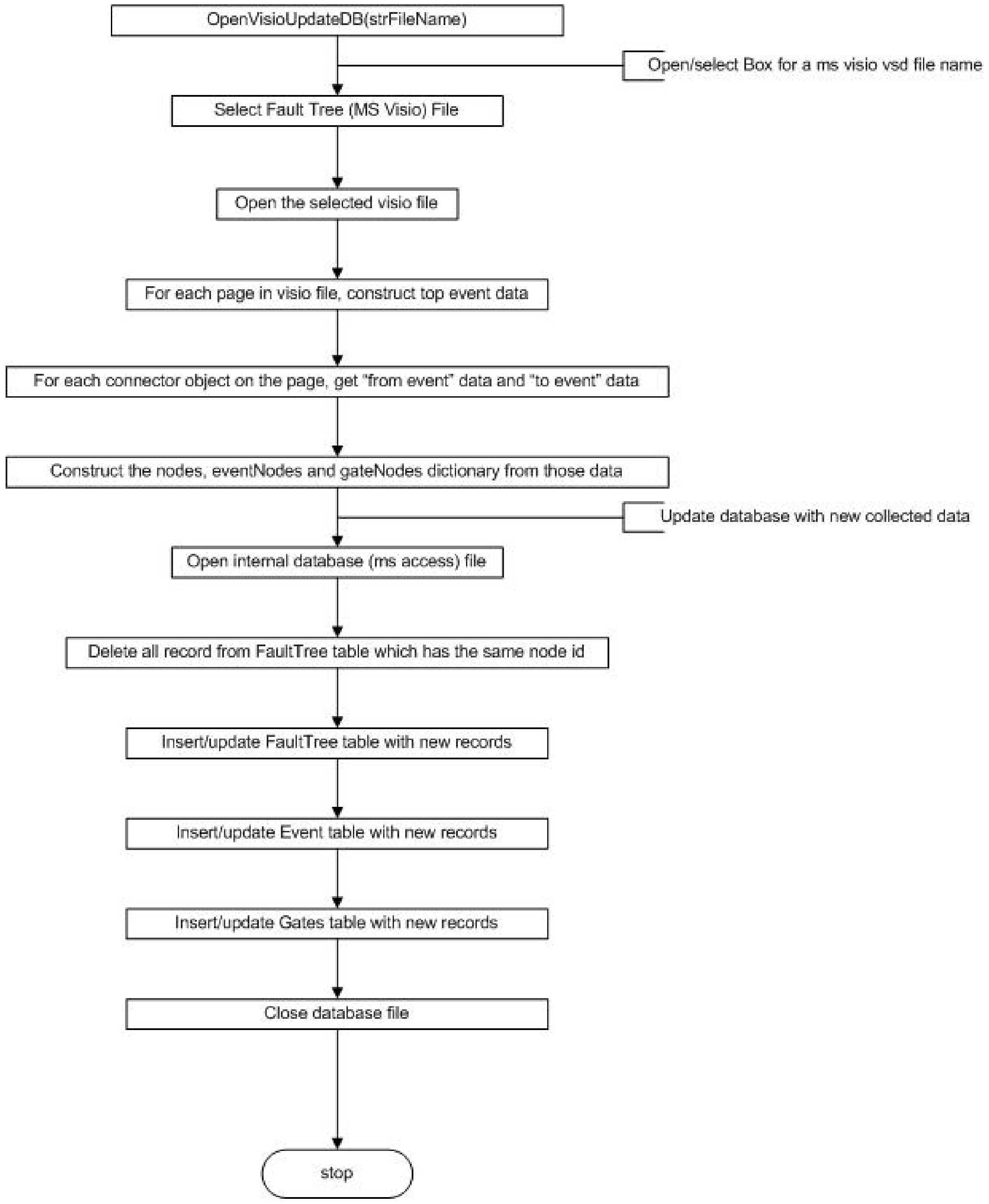

OpenVisioUpdateDB(strFileName)
Open/select Box for a ms visio vsd file name
Select Fault Tree (MS Visio) File
Open the selected visio file
For each page in visio file, construct top event data
For each connector object on the page, get "from event" data and "to event" data
Construct the nodes, eventNodes and gateNodes dictionary from those data
Update database with new collected data
Open internal database (ms access) file
Delete all record from FaultTree table which has the same node id
Insert/update FaultTree table with new records
Insert/update Event table with new records
Insert/update Gates table with new records
Close database file
stop

Figure 5.6 Flow chart for updating the database from Visio files

5.6 Summary of Chapter V

A prototype computer package has been developed to demonstrate the capability of this proposed methodology. This prototype computer package is presented in detail in this chapter. The Microsoft Visual Basic ® language is the primary language for coding, and the system structure is provided in the overview section. Four major modules, top event editor, component model editor, fault tree generator, and fault tree editor, are discussed separately in Sections 5.2, 5.3, 5.4 and 5.5.

The top event editor serves the purpose of helping the analyst identify potential top events and defining user-defined top events. Fire event identification is described as an example in Section 5.2. The component model editor is essentially database connectors, or table editors for both the public database and the specific database. There are a total of 25 database connectors, which are implemented through the Microsoft Access ADO object library. Users can modify records in the database in Microsoft Access directly, or database connectors are provided in the package especially for users without the Microsoft Access. The fault tree generator is the core of the proposed methodology. A detailed programming flow chart is presented in Section 5.4. Fault tree structures are stored as a table in the specific database. To facilitate the review and modification of fault trees, three functionalities are provided in fault tree editor, namely, “open a Visio file to update DB”, “create a complete Visio file from DB”, and “create a simplified Visio file from DB”. These functions are discussed in Section 5.5 in detail with their corresponding programming flow charts.

Only major error handlings such as “database not found” are considered in the programming. “Help” and “exit” are provided for each module.

CHAPTER VI

CASE STUDIES*

To be programmed in computer codes and applied in a real case, the proposed methodology must be tested against substantial examples. Any computer codes must be carefully examined and extensively verified before being used in real applications. It is not recommended to rely on the computer-aided approach wholly since many of the problems in system design can be discovered during the analysis process.

Two case studies are presented in this chapter to illustrate the methodology. The first one is a nitric acid cooling process adapted from Lapp and Powers (1977), which is a seminal publication in the history of computer-aided fault tree synthesis. The second one is an example of fire event identification for a storage tank.

*Reprinted from *Journal of Loss Prevention in the Process Industries*, Vol. 15, No. 4, Wang Y., T.L. Teague, H.H. West, & M.S. Mannan, A New Algorithm for Computer-Aided Fault Tree Synthesis, 265-277, Copyright (2002), with permission from Elsevier.

6.1 Case Study 1

6.1.1 Process Description

This process is adapted from Lapp and Powers (1977), which is to cool one hot nitric acid stream with cooling water before entering the reactor where it reacts with benzene to produce nitrobenzene. The process diagram is shown in Figure 6.1.

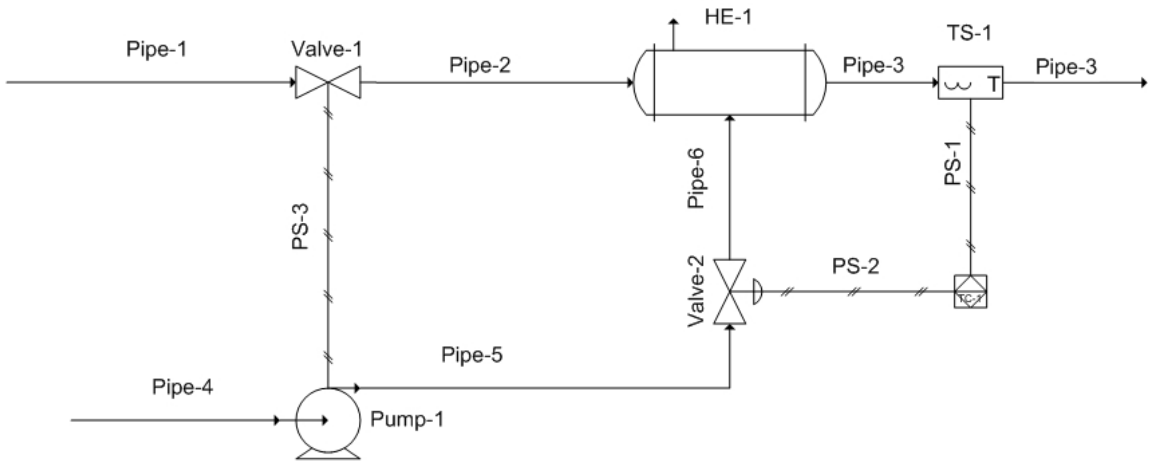


Figure 6.1 Process block diagram of case study 1

The trip valve and the control valve here are both air-to-open, which closes upon loss of instrument air. The signals of the temperature sensor and the controller increase with increasing inputs.

6.1.2 Database Inputs

For the above nitric acid cooling process, one of the identified incidents can be a runaway reaction in the benzene reactor. Many factors, such as a large external fire outside the reactor and loss of reactor temperature control system, can cause a runaway reaction. In our system boundary, assume that the only cause of this incident is too hot HNO_3 to the reactor. The top events table in the specific database is shown in Table 6.1 (user-defined top event). The corresponding inputs to the gates table in the specific database are shown in Table 6.2. Columns with no data are not shown.

Table 6.1 The top events table for case study 1

Index	Gate ID	Description	Comment
4	1	Too hot HNO3 enters the reactor	

Table 6.2 The gates table for case study 1

Gate ID	Description	Completeness	Equipment ID	Process Variable	Deviation
1	TS-1 Tout +1	FALSE	TS-1	Tout	1

Some information must be read from the Visio file and stored into the equipment table, equipment state table, and topology table. It can be extracted automatically by the computer from Visio files though not implemented in this work. In this particular case, the equipment state table is not used. Equipment table and topology table are shown in Tables 6.3 and 6.4. Some columns with no data are not shown here for brevity.

Table 6.3 The equipment table for case study 1

Equipment ID	Boundary	Equipment Type	Equipment Subtype
HE-1	FALSE	Heat Exchanger	Shell Tube
Pump-1	TRUE	Pump	Centrifugal
TC-1	FALSE	Controller	PLC
TS-1	FALSE	Temperature Sensor	Thermocouple
Valve-1	TRUE	Trip Valve	Pneumatic
Valve-2	FALSE	Control Valve	Pneumatic

Table 6.4 The topology table for case study 1

ID	FromEquip ID	FromPort Name	ToEquip ID	ToPort Name
1	HE-1	h,out	TS-1	in
2	Valve-1	out	HE-1	h,in
3	Valve-2	out	HE-1	cl,in
4	Pump-1	out	Valve-2	in

There are three tables relevant to component modeling, the specific digraph table, failure modes and effects table, and external events table, which are shown in Tables 6.5, 6.6 and, 6.7, respectively. For brevity, only the data to be used later in this nitric acid cooler example are listed. These tables are for illustration purpose only since the actual database may have more records. To reduce the size of the fault tree, pipeline and signal line failures are not taken into account in this example. In particular, the heat exchanger in the above system is special in that there will be an exothermic reaction if there is an internal leakage. This failure mode is neglected here to simplify the problem.

Table 6.5 The specific digraph table input for case study 1

Equipment ID	State	Input variable	Output variable	Relationship
TS-1		Tin	Tout	1
TS-1		Tin	Sigout	1
HE-1	Normal	Th,in	Th,out	1
HE-1	Normal	Fh,in	Th,out	1
HE-1	Normal	Tcl,in	Th,out	1
HE-1	Normal	Fcl,in	Th,out	-1
Valve-1		Tin	Tout	1
Valve-1		Fin	Fout	1
Valve-2		Tin	Tout	1
Pump-1		Tin	Tout	1
Pump-1		Pin	Pout	1
Pump-1		Pin	Fout	1
Pump-1		Fin	Fout	1
TC-1		Sigin	Sigout	1
Valve-2		Fin	Fout	1

Table 6.6 The specific failure modes and effects table input for case study 1

Equipment ID	Failure Modes	Output Variable	Relationship
Pump-1	leaks externally	Pout	-1
Pump-1	discharge line leaks externally	Pout	-1
Pump-1	discharge line block partially	Pout	-1
Pump-1	leaks externally significantly	Pout	-10
Pump-1	discharge line blocked	Pout	-10
Pump-1	discharge line leaks externally significantly	Pout	-10
Pump-1	fails while running	Pout	-10
Pump-1	leaks externally	Fout	-1
Pump-1	discharge line leaks externally	Fout	-1
Pump-1	discharge line block partially	Fout	-1
Pump-1	leaks externally significantly	Fout	-10
Pump-1	discharge line blocked	Fout	-10
Pump-1	discharge line leaks externally significantly	Fout	-10
TC-1	fails low	Sigout	-1
TC-1	fails very low	Sigout	-10
TS-1	fails low	Sigout	-1
TS-1	fails very low	Sigout	-10
HE-1	internal fouling occurs	Th,out	1
Valve-2	leaks externally	Fout	-1
Valve-2	fails low	Fout	-1

Table 6.7 The external events table input for case study 1

Equipment ID	Description	Output Variable	Relationship
Valve-2	loss of instrument air	Fout	-10
HE-1	large external fire	Th,out	10
HE-1	external fire	Th,out	1
Valve-1	low instrument air	Fout	-1
Valve-1	loss of instrument air	Fout	-10
Pump-1	loss of power supply	Fout	-10

The fault tree table and the gates table will be updated automatically during the procedure of fault tree generation. The stack table serves the functionality of a stack storing all the intermediate gates. The fault tree construction procedure will keep running unless the stack table is empty.

6.1.3 Results

After the table inputs are ready, clicking “fault tree generator” brings a top event list box, in which the listed items are extracted from top events table. The program will continue to construct fault trees for the top event once users select one item in the list box. The generated gates table is shown in Table 6.8. The default gate description generated from this program is a combination of the equipment ID, the process variable, and the deviation. Users can modify it if desired. The logic between the gates are stored in the fault tree table, as shown in Table 6.9. Again, some columns with no data such as probability, time, and comment have been omitted.

Table 6.8 The generated gates table for case study 1

Gate ID	Description	Completeness	Equipment ID	Process Variable	Deviation
1	TS-1 Tout +1	TRUE	TS-1	Tout	1
2	TS-1 Tin 1	TRUE	TS-1	Tin	1
3	HE-1 Th,out 1	TRUE	HE-1	Th,out	1
4	Uncontrollable disturbances	TRUE			0
5	Controllable disturbances and control inactive	TRUE			0
6	Controllable disturbances	TRUE			0
7	Control inactive	TRUE	Valve-2		0
8	Control unhealthy	TRUE	Valve-2		0
9	HE-1 Th,out 10	TRUE	HE-1	Th,out	10
10	HE-1 Th,in 10	TRUE	HE-1	Th,in	10
11	HE-1 Fh,in 10	TRUE	HE-1	Fh,in	10
12	HE-1 Tcl,in 10	TRUE	HE-1	Tcl,in	10
13	HE-1 Fcl,in –10	TRUE	HE-1	Fcl,in	-10
14	HE-1 large external fire	TRUE	HE-1		0
15	HE-1 Th,out 1	TRUE	HE-1	Th,out	1
16	HE-1 Th,in 1	TRUE	HE-1	Th,in	1
17	HE-1 Fh,in 1	TRUE	HE-1	Fh,in	1
18	HE-1 Tcl,in 1	TRUE	HE-1	Tcl,in	1
19	HE-1 Fcl,in -1	TRUE	HE-1	Fcl,in	-1
20	HE-1 Internal fouling occurs	TRUE	HE-1		0
21	HE-1 external fire	TRUE	HE-1		0
22	TS-1 stuck	TRUE	TS-1		0
23	TC-1 stuck	TRUE	TC-1		0
24	Valve-2 stuck	TRUE	Valve-2		0
25	TS-1 failure	TRUE	TS-1		0
26	TC-1 failure	TRUE	TC-1		0
27	Valve-2 failure	TRUE	Valve-2		0

Table 6.8 continued

Gate ID	Description	Completeness	Equipment ID	Process Variable	Deviation
28	Valve-2 Fout -1	TRUE	Valve-2	Fout	-1
29	Valve-2 Fin -1	TRUE	Valve-2	Fin	-1
30	Valve-2 low instrument air	TRUE	Valve-2		0
31	Valve-2 leaks externally	TRUE	Valve-2		0
32	Valve-2 fails low	TRUE	Valve-2		0
33	Pump-1 Fout -1	TRUE	Pump-1	Fout	-1
34	Pump-1 Pin -1	TRUE	Pump-1	Pin	-1
35	Pump-1 Fin -1	TRUE	Pump-1	Fin	-1
36	Pump-1 leaks externally	TRUE	Pump-1		0
37	Pump-1 discharge line leaks externally	TRUE	Pump-1		0
38	Pump-1 discharge line block partially	TRUE	Pump-1		0
39	Valve-2 Tout 1	TRUE	Valve-2	Tout	1
40	Valve-2 Tin 1	TRUE	Valve-2	Tin	1
41	Pump-1 Tout 1	TRUE	Pump-1	Tout	1
42	Pump-1 Tin 1	TRUE	Pump-1	Tin	1
43	Valve-1 Fout 1	TRUE	Valve-1	Fout	1
44	Valve-1 Fin 1	TRUE	Valve-1	Fin	1
45	Valve-1 Tout 1	TRUE	Valve-1	Tout	1
46	Valve-1 Tin 1	TRUE	Valve-1	Tin	1
47	Valve-2 Fout -10	TRUE	Valve-2	Fout	-10
48	Valve-2 Fin -10	TRUE	Valve-2	Fin	-10
49	Loss of instrument air and feedforward inactive	TRUE	Valve-2	Fout	-10
50	Loss of instrument air	TRUE	Valve-2	Fout	-10
51	Feedforward inactive	TRUE			0
52	Valve-1 stuck	TRUE	Valve-1		0
53	Pump-1 Fout -10	TRUE	Pump-1	Fout	-10
54	Pump-1 Pin -10	TRUE	Pump-1	Pin	-10
55	Pump-1 Fin -10	TRUE	Pump-1	Fin	-10

Table 6.8 continued

Gate ID	Description	Completeness	Equipment ID	Process Variable	Deviation
56	Pump-1 leaks externally significantly	TRUE	Pump-1		0
57	Pump-1 discharge line blocked	TRUE	Pump-1		0
58	Pump-1 discharge line leaks externally significantly	TRUE	Pump-1		0
59	Loss of power supply and feedforward inactive	TRUE	Pump-1	Fout	-10
60	Loss of power supply	TRUE	Pump-1	Fout	-10
61	Feedforward inactive	TRUE			0
62	Valve-1 stuck	TRUE	Valve-1		0
63	Valve-2 Tout 10	TRUE	Valve-2	Tout	10
64	Valve-2 Tin 10	TRUE	Valve-2	Tin	10
65	Pump-1 Tout 10	TRUE	Pump-1	Tout	10
66	Pump-1 Tin 10	TRUE	Pump-1	Tin	10
67	Valve-1 Fout 10	TRUE	Valve-1	Fout	10
68	Valve-1 Fin 10	TRUE	Valve-1	Fin	10
69	Valve-1 Tout 10	TRUE	Valve-1	Tout	10
70	Valve-1 Tin 10	TRUE	Valve-1	Tin	10

Table 6.9 The generated fault tree table for case study 1

ID	Gate ID	Type	Child
15	1	OR	2
16	2	OR	3
17	3	OR	4
18	3	OR	5
19	5	AND	6
20	5	AND	7
21	3	OR	8
22	4	OR	9
23	9	OR	10
24	9	OR	11
25	9	OR	12
26	9	OR	13
27	9	OR	14
28	6	OR	15
29	15	OR	16
30	15	OR	17
31	15	OR	18
32	15	OR	19
33	15	OR	20
34	15	OR	21
35	7	OR	22
36	7	OR	23
37	7	OR	24
38	8	OR	25
39	8	OR	26
40	8	OR	27
41	19	OR	28
42	28	OR	29
43	28	OR	30
44	28	OR	31
45	28	OR	32

Table 6.9 continued

ID	Gate ID	Type	Child
46	29	OR	33
47	33	OR	34
48	33	OR	35
49	33	OR	36
50	33	OR	37
51	33	OR	38
52	18	OR	39
53	39	OR	40
54	40	OR	41
55	41	OR	42
56	17	OR	43
57	43	OR	44
58	16	OR	45
59	45	OR	46
60	13	OR	47
61	47	OR	48
62	47	OR	49
63	49	AND	50
64	49	AND	51
65	51	OR	52
66	48	OR	53
67	53	OR	54
68	53	OR	55
69	53	OR	56
70	53	OR	57
71	53	OR	58
72	53	OR	59
73	59	AND	60
74	59	AND	61
75	61	OR	62
76	12	OR	63

Table 6.9 continued

ID	Gate ID	Type	Child
77	63	OR	64
78	64	OR	65
79	65	OR	66
80	11	OR	67
81	67	OR	68
82	10	OR	69
83	69	OR	70

As pointed out in Chapter IV, using database tables to represent the structure of fault trees are easy for the computer to manipulate data. However, it is not straightforward for users to review and modify fault trees. To solve this problem, this fault tree structure can be converted into Visio files by retrieving records from the gates table, events table, and fault tree table. The complete fault tree for case study 1 is shown in Figure 6.2 while the simplified fault tree is shown in Figure 6.3.

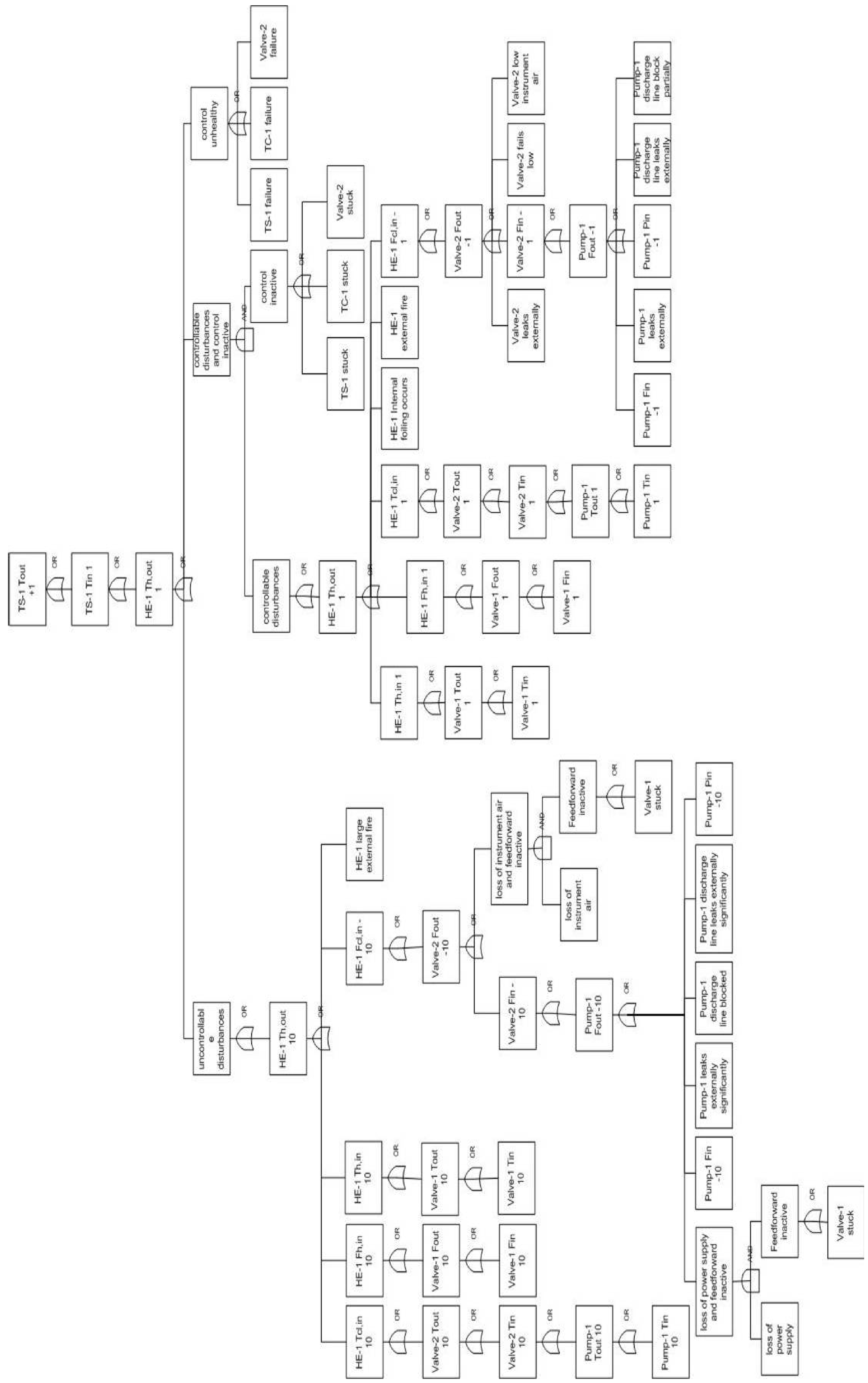


Figure 6.2 The complete fault tree for case study 1

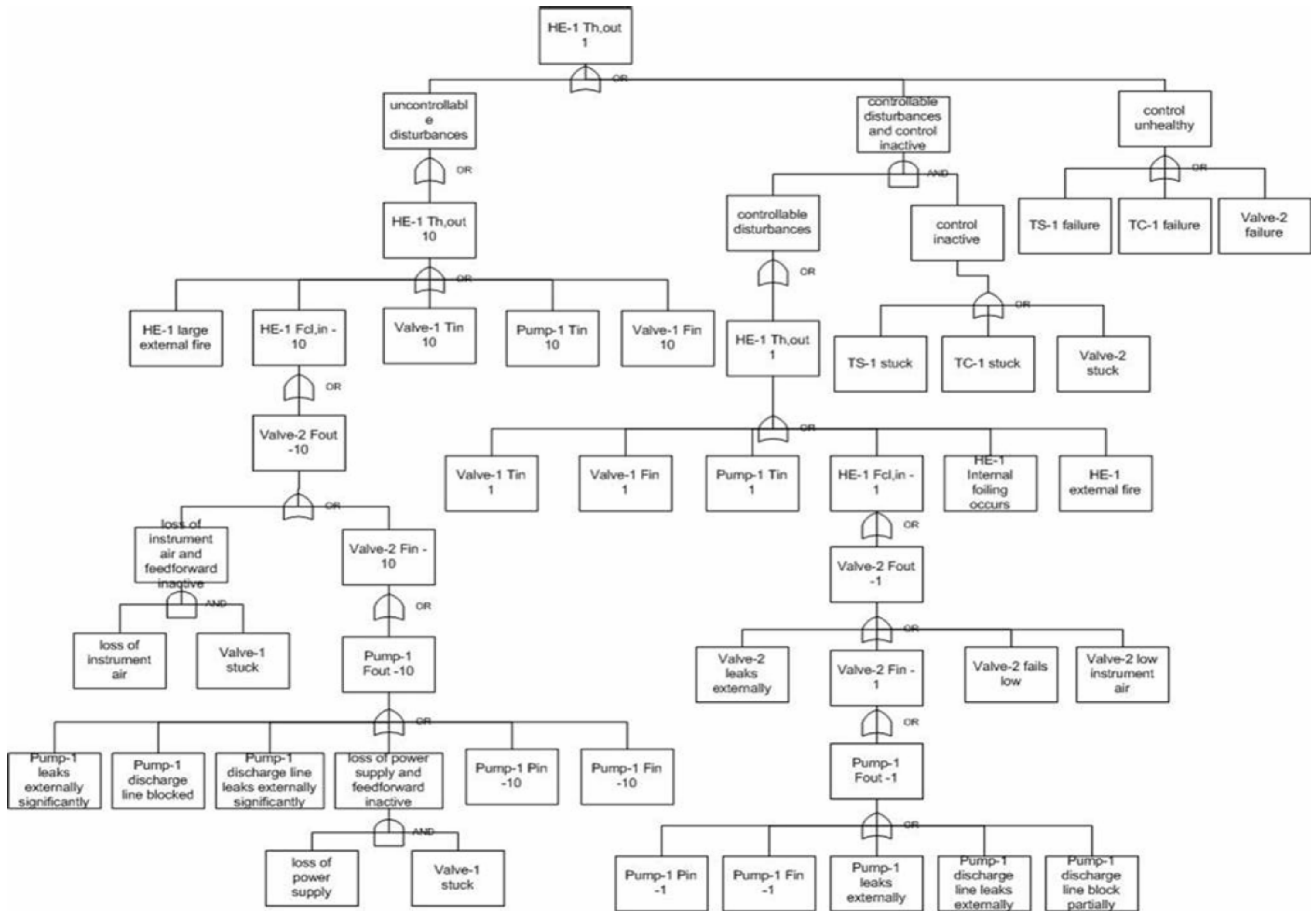


Figure 6.3 The simplified fault tree for case study 1

6.1.4 Comparison with Published Results

Analysis of fault tree models involves Minimal Cut Set (MCS) analysis. MCSs are the minimal combinations of basic events that can result in the top event. The procedure of MCS analysis transforms the fault tree into an equivalent two-level tree, which is an OR gate of all the possible MCSs. After algebraic analysis, the MCSs of the above simplified fault tree in Figure 6.3 are listed in Table 6.10.

Many failures in the fault tree in Figure 6.3 are not considered in the published fault tree in Lapp and Powers (1977). In order to compare these two fault trees, the analysis resolution must be consistent. By removing the MCSs containing those failures that are not considered in Lapp and Powers (1977, 1979) paper, we can obtain the following MCSs as shown in Table 6.11.

Table 6.10 The minimum cut sets for case study 1

Index	MCSs
1	HE-1 large external fire
2	Loss of instrument air, Valve-1 stuck
3	Pump-1 leaks externally significantly
4	Pump-1 discharge line blocked
5	Pump-1 discharge line leaks externally significantly
6	Loss of power supply, Valve-1 stuck
7	Pump-1 Pin(-10)
8	Pump-1 Fin(-10)
9	Valve-1 Tin(+10)
10	Pump-1 Tin(+10)
11	Valve-1 Fin(+10)
12	Valve-1 Tin(+1), TS-1 stuck
13	Valve-1 Tin(+1), TC-1 stuck
14	Valve-1 Tin(+1), Valve-2 stuck
15	Valve-1 Fin(+1), TS-1 stuck
16	Valve-1 Fin(+1), TC-1 stuck
17	Valve-1 Fin(+1), Valve-2 stuck
18	Pump-1 Tin(+1), TS-1 stuck
19	Pump-1 Tin(+1), TC-1 stuck
20	Pump-1 Tin(+1), Valve-2 stuck
21	Valve-2 leaks externally, TS-1 stuck
22	Valve-2 leaks externally, TC-1 stuck
23	Valve-2 leaks externally, Valve-2 stuck
24	Pump-1 Pin (-1), TS-1 stuck
25	Pump-1 Pin (-1), TC-1 stuck
26	Pump-1 Pin (-1), Valve-2 stuck
27	Pump-1 Fin (-1), TS-1 stuck
28	Pump-1 Fin (-1), TC-1 stuck
29	Pump-1 Fin (-1), Valve-2 stuck
30	Pump-1 leaks externally, TS-1 stuck
31	Pump-1 leaks externally, TC-1 stuck

Table 6.10 continued

Index	MCSs
32	Pump-1 leaks externally, Valve-2 stuck
33	Pump-1 discharge line leaks externally, TS-1 stuck
34	Pump-1 discharge line leaks externally, TC-1 stuck
35	Pump-1 discharge line leaks externally, Valve-2 stuck
36	Pump-1 discharge line blocked partially, TS-1 stuck
37	Pump-1 discharge line blocked partially, TC-1 stuck
38	Pump-1 discharge line blocked partially, Valve-2 stuck
39	Valve-2 fails low, TS-1 stuck
40	Valve-2 fails low, TC-1 stuck
41	Valve-2 fails low, Valve-2 stuck
42	Valve-2 low instrument air, TS-1 stuck
43	Valve-2 low instrument air, TC-1 stuck
44	Valve-2 low instrument air, Valve-2 stuck
45	HE-1 internal fouling occurs, TS-1 stuck
46	HE-1 internal fouling occurs, TC-1 stuck
47	HE-1 internal fouling occurs, Valve-2 stuck
48	HE-1 external fire, TS-1 stuck
49	HE-1 external fire, TC-1 stuck
50	HE-1 external fire, Valve-2 stuck
51	TS-1 failure
52	TC-1 failure
53	Valve-2 failure

Table 6.11 The new MCSs for case study 1

Index	MCSs
1	HE-1 large external fire
2	Loss of instrument air, Valve-1 stuck
6	Loss of power supply, Valve-1 stuck
7	Pump-1 Pin(-10)
9	Valve-1 Tin(+10)
11	Valve-1 Fin(+10)
12	Valve-1 Tin(+1), TS-1 stuck
13	Valve-1 Tin(+1), TC-1 stuck
15	Valve-1 Fin(+1), TS-1 stuck
16	Valve-1 Fin(+1), TC-1 stuck
24	Pump-1 Pin (-1), TS-1 stuck
25	Pump-1 Pin (-1), TC-1 stuck
42	Valve-2 low instrument air, TS-1 stuck
43	Valve-2 low instrument air, TC-1 stuck
48	HE-1 external fire, TS-1 stuck
49	HE-1 external fire, TC-1 stuck
51	TS-1 failure
52	TC-1 failure

As indicated by Lapp and Powers (1979), the probability of the event “Low air pressure of the cooling water control valve” and the event “control valve reversed” occurring simultaneously is negligible. We can simply replace the EOR (exclusive or) gate as OR gate. The MCSs for the published fault tree in Lapp and Powers (1977) are listed in Table 6.12, in which equipment names are adjusted to be consistent with this case study for easy comparison.

Table 6.12 The MCSs of the published fault tree

Index	**MCSs**
1	large external fire
2	Loss of instrument air
3	Pump-1 shutdown, Valve-1 reversed
4	Pump-1 shutdown, signal line plugged
5	Pump-1 Pin(-10)
6	Valve-1 Tin(+10)
7	Valve-1 Pin(+10)
8	Valve-1 Tin(+1), TS-1 stuck
9	Valve-1 Tin(+1), TC-1 stuck
10	Valve-1 Pin(+1), TS-1 stuck
11	Valve-1 Pin(+1), TC-1 stuck
12	Pump-1 Pin (-1), TS-1 stuck
13	Pump-1 Pin (-1), TC-1 stuck
14	Low air pressure, TS-1 stuck
15	Low air pressure, TC-1 stuck
16	External fire, TS-1 stuck
17	External fire, TC-1 stuck
18	TS-1 failure
19	TC-1 failure

The minimum cut set {Loss of instrument air, Valve-1 Stuck} is different from that in the published paper {Loss of instrument air}, because Lapp and Powers (1977, 1979) did not consider the fact that the trip valve will close upon loss of instrument air. The signal line failures are not considered in this example, so the minimum cut set {Pump-1 shutdown, signal line plugged} is not in the MCSs of the fault tree generated from the program. Except for the two sets mentioned above, the two minimum cut sets are consistent.

6.2 Case Study 2

This case study is to illustrate the “fire” top event identification procedure. Process description and the resultant fault tree are presented in Sections 6.2.1 and 6.2.2, respectively.

6.2.1 Process Description

Figure 6.4 shows a typical storage tank of flammable materials. A centrifugal pump is used to pump the materials to supply other processes. Here the process controls and safe guards are not shown since they are not related to fire event identification.

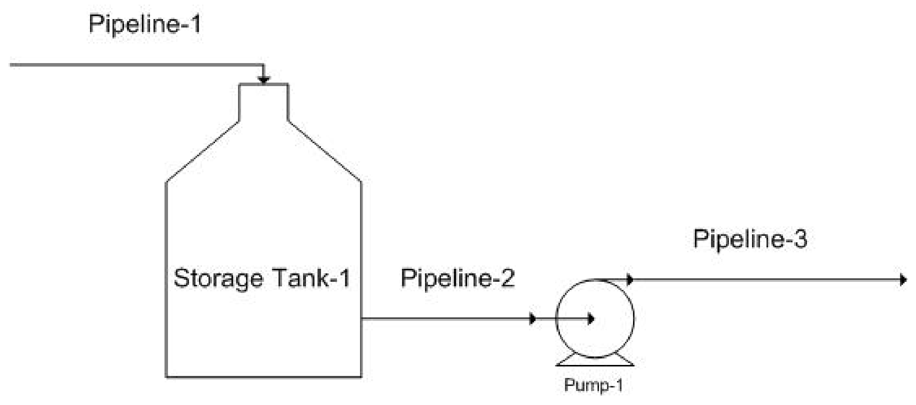


Figure 6.4 Process block diagram for case study 2

Assume that storage tank and centrifugal pump are the only two potential sources of fire in this process. The possible containment failure modes of the storage tank include leakage and corrosion. The pump might leak externally for the flammable materials to be exposed to air.

Suppose that sprinkler fire protection system and dry chemical protection system are available on the site to prevent a fire from occurring. The sprinkler system may fail upon pre-action valve failure, fusible link failure, or human error. The dry chemical system fails if the manual valve is stuck or the operator does not respond.

6.2.2 Results

The above information is entered into the computer while the program prompts the user to input certain information step by step in “fire event identification”. The generated gates table and fault tree table are shown in Tables 6.13 and 6.14, respectively.

Table 6.13 The generated gates table

Gate ID	Description	Completeness
1	Fire occurs	FALSE
2	Fire occurs in Storage Tank 1	FALSE
3	Ignition occurs in Storage Tank 1	FALSE
4	Protection 1 fails	FALSE
5	Pre-action valve failure	FALSE
6	Fusible link failure	FALSE
7	Human error	FALSE
8	Protection 2 fails	FALSE
9	Dry chemical valve stuck	FALSE
10	Operator fails to respond	FALSE
11	Containment failure of Storage Tank 1	FALSE
12	Leak externally at Storage Tank 1	FALSE
13	Erosion occurs at Storage Tank 1	FALSE
14	Fire occurs in Pump 1	FALSE
15	Ignition occurs in Pump 1	FALSE
16	Protection 1 fails	FALSE
17	Pre-action valve failure	FALSE
18	Fusible link failure	FALSE
19	Human error	FALSE
20	Protection 2 fails	FALSE
21	Dry chemical valve stuck	FALSE
22	Operator fails to respond	FALSE
23	Containment failure at Pump 1	FALSE
24	Leak externally at Pump 1	FALSE
25	Ignition sources at Pump 1	FALSE
26	Ignition sources at Storage Tank 1	FALSE

Table 6.14 The generated fault tree table

ID	Gate ID	Type	Child
15	1	OR	2
16	2	AND	3
17	2	AND	4
18	4	OR	5
19	4	OR	6
20	4	OR	7
21	2	AND	8
22	8	OR	9
23	8	OR	10
24	3	AND	11
25	3	AND	26
26	11	OR	12
27	11	OR	13
27	1	OR	14
28	14	AND	15
29	14	AND	16
30	16	OR	17
31	16	OR	18
32	16	OR	19
33	14	AND	20
34	20	OR	21
35	20	OR	22
36	15	AND	23
37	15	AND	25
38	23	OR	24

The resultant fault tree is shown in Figure 6.5. The subtrees of “Protection 1 fails” and “Protection 2 fails” appears twice in this fault tree. Their structures are shown in only one place for brevity. The structure is verified to be correct and complete.

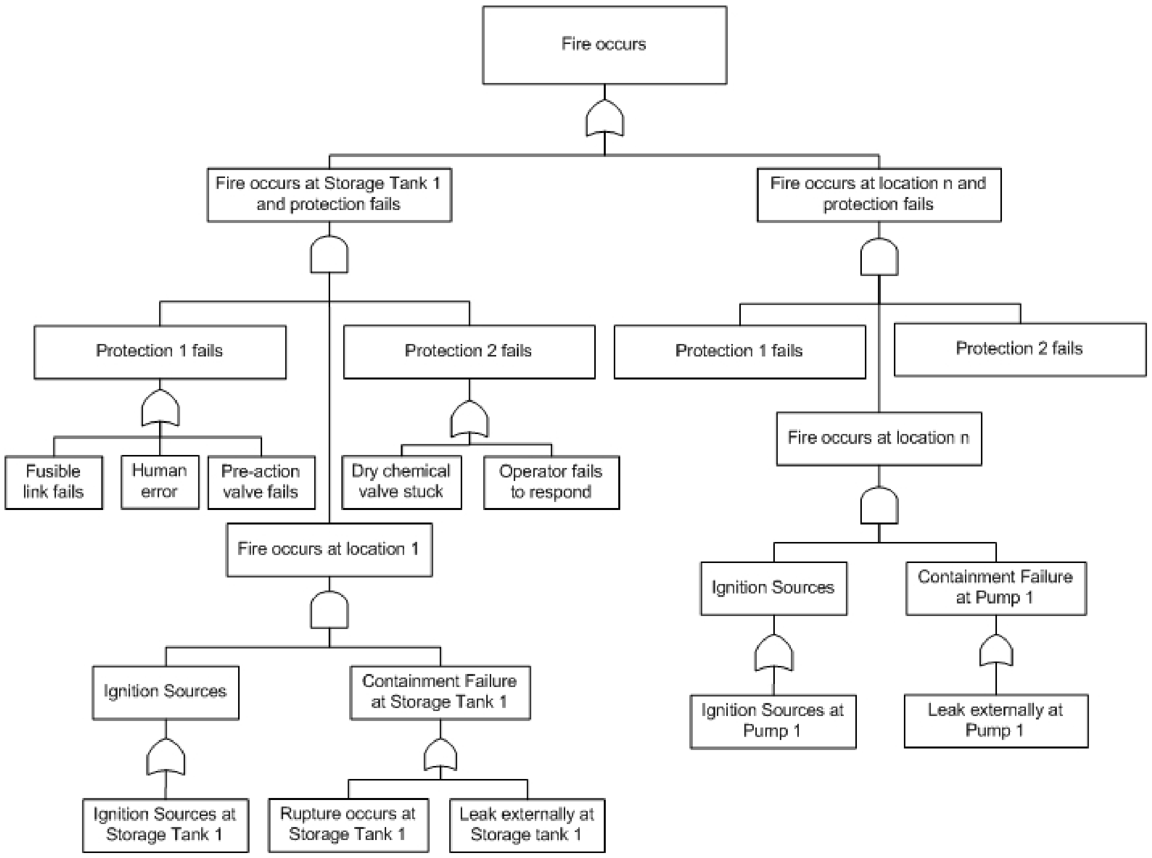

Fire occurs
Fire occurs at Storage Tank 1 and protection fails
Fire occurs at location n and protection fails
Protection 1 fails
Protection 2 fails
Protection 1 fails
Protection 2 fails
Fire occurs at location n
Fusible link fails
Human error
Pre-action valve fails
Dry chemical valve stuck
Operator fails to respond
Fire occurs at location 1
Ignition Sources
Containment Failure at Pump 1
Ignition Sources
Containment Failure at Storage Tank 1
Ignition Sources at Pump 1
Leak externally at Pump 1
Ignition Sources at Storage Tank 1
Rupture occurs at Storage Tank 1
Leak externally at Storage tank 1

Figure 6.5 The resultant fault tree for case study 2

6.3 Summary of Chapter VI

Two case studies are presented in this chapter to illustrate the methodology. The first one is a nitric acid cooling process adapted from Lapp and Powers (1977) paper, which is discussed extensively in the literature. Process description and database input are presented in Sections 6.1.1 and 6.1.2, respectively. The results from this methodology are discussed and compared with the published results, which are proved to be consistent.

The second one is an example of fire event identification for a storage tank. The resultant fault tree structure has been reviewed for correctness and completeness.

CHAPTER VII

DATA UNCERTAINTY

Though the example given in this chapter is SIL determination via FTA, this chapter is applicable not only to FTA but also to other QRA techniques and reliability studies.

The ideal situation in a reliability study is to have sufficient in-house data, however, due to various restrictions, laboratory data or data from generic data sources are often used in reliability studies, especially at the design stage. Transferring data from laboratory or generic data sources brings uncertainty. Traditionally, analysis uses only point values from generic data sources to conduct risk analysis. This can lead to misleading results in some cases. Data uncertainty and its impact are addressed in this chapter. The impact of data uncertainty on the calculation of SIL is discussed as an example in this chapter, and procedures are proposed to deal with data uncertainty in determining SIL for a safety instrumented system (SIS).

7.1 Sources of Failure Rate and Event Data

Failure rates depend on the equipment, the definition of failures, the process conditions, and the maintenance plan. The ideal situation for a reliability study is to have sufficient plant data from identical equipment from the same process. However, in many cases, in-house data are not always available. For new plants, there are essentially no historical failure rate data. In those cases, generic data from external sources must be used.

There are a number of generic data sources available from industry or government agencies. The choice of the appropriate generic failure rate data applicable for a particular chemical plant requires the knowledge of the background and data origins.

The Data Acquisition Working Party of the Mechanical Reliability Committee of the Institution of Mechanical Engineers in the United Kingdom examined principal available data sources and provided a summary table (Davidson, 1994). The applicable industry such as offshore, nuclear, industrial, or military industry has been pointed out, together with the equipment types that have been covered. CCPS (1991) provides a review of available databases and compiles equipment failure rates hierarchically. Upper and lower bounds are provided whenever available. It is probably the failure rate database most extensively used in the United States. OREDA (1984, 1988, 1997 and 2002) is dedicated to offshore gas and oil industry and provides failure rates, failure mode distribution, and repair times for offshore installations. There are a variety of other data sources available. A comprehensive compilation of failure rate and event data can be found in *Loss Prevention in the Process Industries* (Lees, 1996). It remains the user's responsibility to consult the original references and choose the best source for their plant.

7.2 Definition of Safety Integrity Level

A target SIL is defined during the safety requirement specification development for an industrial system (Summers, 1998). SILs are defined as 1, 2, 3, and 4 in the ISA/ANSI S84.01, as shown in Table 7.1 (ISA-TR84.00.02, 2002a), among which SIL 4 is not used in the process industry.

Table 7.1 Safety Integrity Level Performance Requirements (ISA-TR84.00.02)

Integrity Level	Safe Availability (%)	PFD[a]	Equivalent RRF[b]
4[c]	>99.99	<0.0001	>10,000
3	99.9-99.99	0.001-0.0001	1,000-10,000
2	99-99.9	0.01-0.001	100-1,000
1	90-99	0.1-0.01	10-100
0	(control - N/A)		

[a]PFD= Probability of failure on demand.
[b]RRF=Risk reduction factor (1/PFD).
[c]ISA and AIChE documents restricted to 3 levels.

According to the standards, the ability of an SIS to achieve a specific SIL must be validated at each stage of design and prior to any changes made to the design after commissioning. The entire operation, testing, and maintenance procedures and practices are also verified for agreement with the target SIL. Thus, determining SIL for an SIS and its validation is very important for conformance to ISA/ANSI-S84.01.

ISA-TR84.00.02-2002 (2002a, 2002b, 2002c, 2002d, 2002e) was provided to describe various methodologies for the determination of SIL of Safety Instrumented

Functions (SIF) and to reinforce the concept of the performance based evaluation of an SIS. Three techniques, simplified equations (ISA-TR84.00.02, 2002b), FTA (ISA-TR84.00.02-2002c), and Markov modeling (ISA-TR84.00.02, 2002d, 2002e) can be used to model the interaction and functionality of basic SIS components.

7.3 Data Uncertainty

Simplified equations, FTA, and Markov modeling all require failure rates of equipment and operators. Statistical data are used to calculate the overall system safety availability or SIL. There are many factors that determine the failure rates of equipment and the range of failure rates observed can be quite wide. It is necessary to consult the original sources to take full advantage of the available information. The failure rates of equipment can be influenced by a large number of factors, including design, specification, manufacture, application, operation conditions, maintenance, and environment. In the process industry, the operating conditions and environment can change dramatically for the same equipment. The most desirable information is to have sufficient plant specific data to determine the SIL for any SIF. However, in many cases, equipment has not been operating long enough to provide statistically valid data, internal collection may not be appropriate, or for new plant designs there is no possibility to collect any in-house failure data. Laboratory data and generic data are often used in determining the SIL of an SIS, especially at the design stage. Point values from these data origins are normally used to obtain a point estimation of SIL.

However, measuring and collecting data have uncertainty associated with them, and borrowing data from laboratory and generic data sources involves an element of uncertainty as well. Reliability data can often deviate by a factor of three or four, and a factor of ten is not unusual, as suggested by Kletz (1999). Nowadays, Monte Carlo simulation is extensively employed to treat data uncertainty. However, in the case of failure rate data, Monte Carlo simulation might be inappropriate since most of the available failure rates are only point values without information about their distributions. Only a few reliability databases provide upper and lower bounds and confidence levels of the failure rate data. Assumption of any failure distribution introduces an unpredictable uncertainty. Some modern reliability databases do provide upper and lower values on failure rates, like IEEE-Std-500 (IEEE-Std-500, 1984), OREDA (1984, 1992, 1997, 2002) and CCPS (1991). The EIREDA (1998) databank provides confidence levels with error factors. These upper and lower bounds and error factors can be used advantageously to derive ranges of failure rate data.

Using point values of failure rates may result in misleading evaluation of SIL for a SIS, which is illustrated in Figure 7.1. Using point values of some data, the result perfectly falls into SIL 3, while the data uncertainties associated with them might lead to a SIL 2 result.

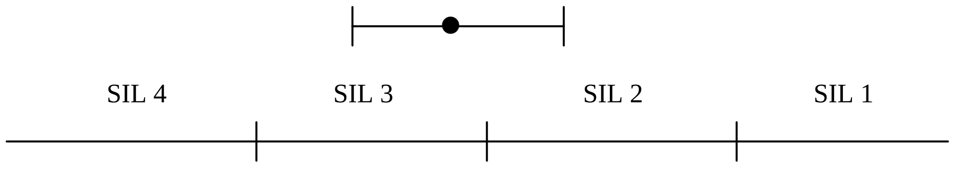


Figure 7.1 The impact of data uncertainty on SIL determination

The impact of data uncertainty on the calculation of SIL is discussed and illustrated in the example below. Here we propose a procedure to examine the impact of data uncertainty on the SIL estimation and deciding the inputs that may lead to a change of SIL. Due to the fact that SIL is a mono-increasing function of any single failure rate and failure rates are bounded by 0 and 1, ranges for input data that will not lead to a change of SIL are calculated by hand or with the help of a computer. These ranges are then compared to known ranges of these input data or expert opinions. Effort must be carried out to obtain more accurate values of those data that might actually lead to a change of SIL. This methodology helps the analyst to identify critical failure rate data that impact the SIL estimation and focus on these data to refine them. This procedure is elaborated further in the following example.

7.4 Example

We use the base case example from ISA-TR84.00.02-2002 (2002c), and assumptions are clearly stated in ISA-TR84.00.02-2002. As mentioned in ISA-TR84.00.02-2002, data in this example are for illustration purpose only. The example is an SIF to protect a storage tank by measuring pressure, flow rate, temperature, and liquid level. Logic voting is employed in the logic solver. The process diagram for the example is shown in Figure 7.2, and the schematic configuration of this SIF is shown in Figure 7.3. The fault tree model of probability of failure on demand (PFD) for this SIF example is shown in Figure 7.4.

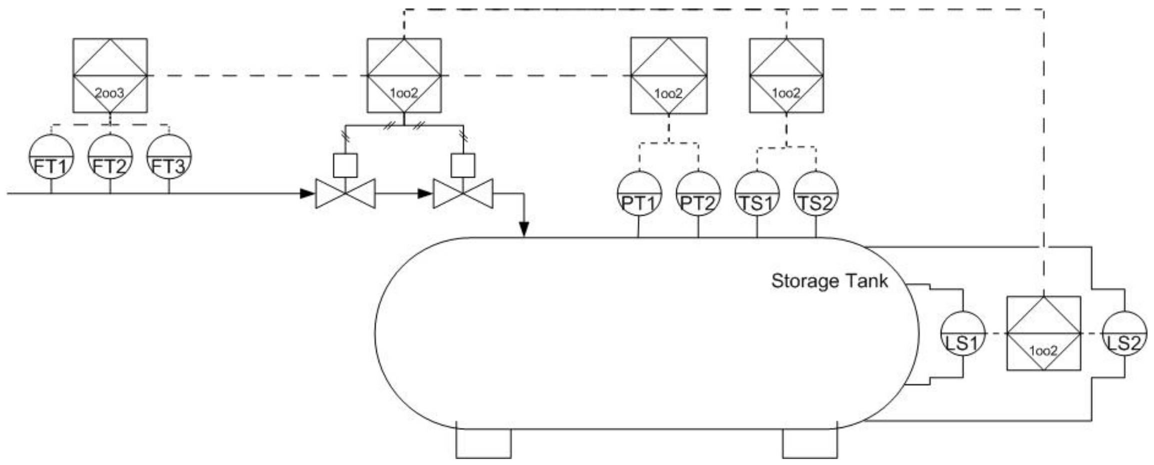


Figure 7.2 Process diagram of example (ISA-TR84.00.02, 2002c)

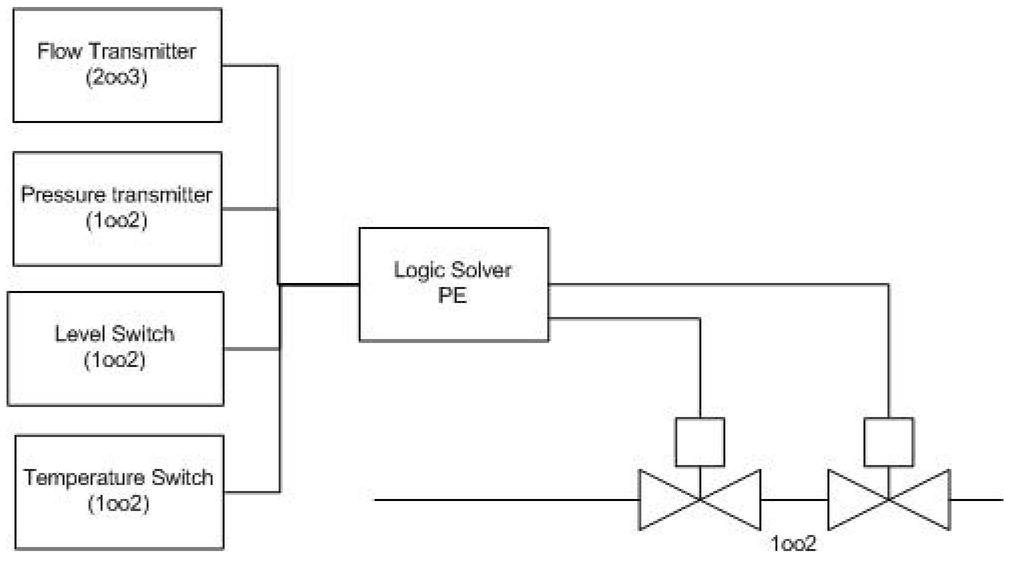


Figure 7.3 Schematic SIF configuration of example (ISA-TR84.00.02, 2002c)

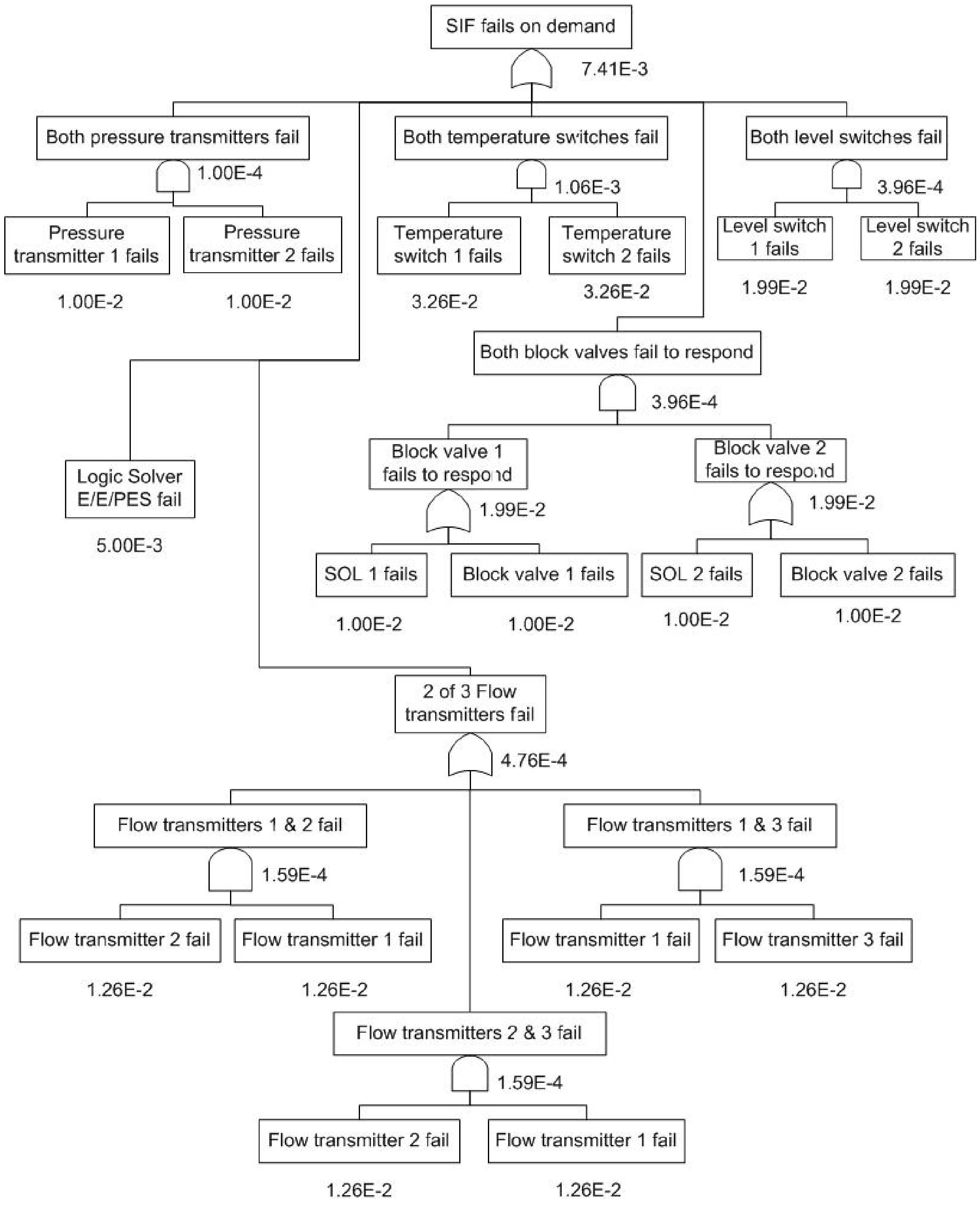


Figure 7.4 Probability of failure on demand fault tree for example SIF (Wang, West, & Mannan, 2004)

The PFD_{avg} data for the individual components used in this example are shown in Table 7.2. The calculated overall system PFD_{avg} is 7.41*E-3, which falls into SIL 2 according to Table 7.1. Then the tight upper bounds of the input data that will not lead to a change of SIL are calculated, and the results are shown in Table 7.2. Any deviation of the input data beyond these upper bounds results in a change in the SIL estimation.

Table 7.2 PFD_{avg} data and calculated ranges for the components (Wang, West, & Mannan, 2004)

Devices	PFD_{avg}	Upper bound that will not lead to a change of SIL
Flow Transmitters	1.26*E-2	3.16*E-2
Pressure Transmitters	1.00*E-2	5.20*E-2
Temperature Switch	3.26*E-2	6.06*E-2
Level Switch	1.99*E-2	5.50*E-2
Block Valves	1.00*E-2	4.55*E-2
Solenoid Valves	1.00*E-2	4.55*E-2
Logic Solver (E/E/PES)	5.00*E-3	7.60*E-3

The calculated upper bounds of these data are then compared to their actual ranges due to uncertainty with them. The actual ranges can be obtained from some data banks or from expert judgment. If the actual ranges of the failure rates have the possibility of exceeding beyond their calculated ranges, uncertainty with these particular data may lead to a change in the estimation of SIL. It will be necessary and beneficial to examine the data and extract more accurate values for those data. As suggested by Kletz

(1999), it is not uncommon for the failure rates to deviate by a factor of 3 or 4, and a factor of ten is not unusual. Though in this case, we do not have any actual ranges for these failure rates to compare with. From Table 7.2, we can notice that most of the calculated ranges fall within a factor of 1.5 ~ 5. Especially in the case of logic solver, a relatively small increase in failure rates (about 50%) will result in a lower estimation of SIL. More effort is required to examine and justify those data.

This example has a fault tree of moderate size for SIF in the process industry. In case of large fault trees, it is extremely difficult to obtain all these ranges by hand. However, this problem can be addressed with the help of computers. FTA is used in this example. However, extension to simplified equations, Markov modeling and other evaluation techniques is straightforward.

7.5 Summary of Chapter VII

This chapter discussed failure rate data sources, their uncertainty, and the impact of uncertainty on reliability studies. In the process industry, operating conditions and environment can change dramatically for the same equipment, and the range of failure rates observed can be quite wide. Traditional reliability studies using point values of failure rates may result in misleading conclusions. The impact of data uncertainty on the calculation of SIL is discussed as an example in this chapter. It may not be practical to research extensively to obtain more accurate estimation for all the failure rate data used in a reliability study, and procedures to deal with data uncertainty in determining SIL for an SIS are proposed to help engineers find the failure rate data that needs refinement and further examination. Effort must be carried out to obtain more accurate values of those data that might actually lead to a change of SIL. This procedure will guide SIS designers and process hazard analysts toward a more accurate SIL estimation and avoid misleading results due to data uncertainty.

Data sources are discussed in Section 7.1 briefly. Section 7.2 gives the definition of SIL. The current status of data uncertainty has been discussed in Section 7.3. An example from ANSI/ISA 84.01 is discussed in Section 7.4. Though the example given in this chapter is SIL determination via FTA, this chapter is applicable not only to FTA but also to other QRA techniques and reliability studies.

It has to be emphasized that the uncertainty with the model itself (simplified equations, FTA and Markov modeling) is not covered in this work.

CHAPTER VIII

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

8.1 Conclusions

Minor incidents and near misses are very common in the process industry, occurring daily. Major incidents have made the public aware of the potential hazards associated with industrial activities. Investigations of almost all the major incidents show that we could have avoided those tragedies with effective risk management programs. Despite the fact that FTA is one of the best tools available for a comprehensive reliability study, the current application of FTA in the CPI is unfortunately limited. One major barrier is the manual construction of fault trees, which can be highly subjective and dependent on the expertise of the analyst. While there are well-developed commercialized software packages for fault tree evaluation, no satisfactory methodology for fault tree synthesis has been published. The availability of a computer-based analysis methodology that simplifies the system understanding process and takes a structured approach to develop fault trees would greatly increase the attractiveness of FTA techniques in the CPI.

This dissertation proposes a systematic and computerizable computer-aided methodology for fault tree synthesis for chemical processes. The basic idea is to capture the cause-and-effect logic among equipment behaviors, human responses, and environment factors for each item of equipment directly into mini fault trees. Special fault tree models have been developed to manage special features such as control loops, trip systems, pressure relief systems, and bypasses. Consistency checking and fault tree simplification procedure are proposed, and fault trees created by this methodology are expected to be correct and concise. Ideally, FTA can be standardized through a computer package that reads information contained in process P&IDs and provides automatic aids to assist engineers in generating and analyzing fault trees.

A database structure has been designed and a prototype computer program is provided to illustrate and test the proposed methodology against some examples. The results generated from the program have been compared to published results and verified to be correct.

Dearth of failure rate data and the large uncertainty associated with the data are considerable problems in the application of FTA. This research proposed a deductive procedure to identify those data that may result in a wrong estimation of SIL for a SIS. This procedure guides SIS designers to focus on those data that require further refinement and avoid misleading results due to data uncertainty.

A computer is never and will never be a substitute for a human being. Many of the problems in the system design can be discovered during analysis of a process, and it is not recommended to rely wholly on computer-aided approaches.

8.2 Future Work

This work proposes a computer-aided methodology for fault tree synthesis. Though this work is originally developed for the application in the CPI, it has the extendibility to other fields such as electrical systems, the nuclear industry, and the aerospace industry.

Many special features are designed to prevent incidents, which are crucial in the synthesis of fault trees. In particular, complex chemical processes may not only have process control loop and safety-related functions, but also process loops. Every loop in the process under study needs to be identified but also classified according to its structural characteristics and functionalities. It is rather difficult to automate the loop identification and classification. In this work, configuration of special features is entered manually by users. Though it has been recognized that a mechanistic approach for the recognition of all the special features is not appropriate for fault tree synthesis in the CPI, guidelines and taxonomy to help analysts in identifying and classifying special features systematically is valuable and highly desired.

As mentioned, the ultimate goad of this research is to standardize the procedure of QRA and help decision makers to decide more formally and more cost-effectively. Risk is defined to be a function of both consequence and frequency. Fault trees are normally used to estimate the probability of concerned incidents, whereas a systematic and powerful consequence analysis methodology is valuable to obtain an overall measure of risk.

Finally, risk presentation and communication is a must for decision makers to understand the results from risk analysis correctly and efficiently and convert them to correct decisions for sustainable development of their enterprise.

REFERENCES

Allen, D.J., & Rao, M.S.M. (1980), New algorithm for the synthesis and analysis of fault trees, *Industrial & Engineering Chemistry Fundamentals, 19*, 79-85.

Andrew, P.K. (1980), Difficulties in fault tree synthesis for process plant, *IEEE Transactions on Reliability, 29*, 2-9.

Andrews, J., & Brennan, G. (1990), Application of the DIGRAPH method of fault tree construction to a complex control configuration, *Reliability Engineering, 28*, 357-384.

ANSI/ISA-S84.01 (1996), Application of safety instrumented systems for the process industries, *ISA*, Research Triangle Park, NC.

Arendt, J.S. (1990), Using quantitative risk assessment in the chemical process industry, *Reliability Engineering and System Safety, 29*, 133-149.

Bagchi, T.P., & Chaudhri, V.K. (1989), *Interactive Relational Database Design: a Logic Programming Implementation*, New York: Springer-Verlag.

Bossche, A. (1991a), Computer-aided fault tree synthesis I. System modeling and causal trees, *Reliability Engineering, 32*, 217-241.

Bossche, A. (1991b), Computer-aided fault tree synthesis II. Fault tree construction, *Reliability Engineering, 33*, 1-21.

Caceres, S., & Henley E.J. (1976), Converting block diagrams to fault trees, *Industrial & Engineering Chemistry Fundamentals, 15*, 128-131.

Camarda, P., Corsi, F., & Trentadue, A. (1978), An efficient simple algorithm for fault tree automatic synthesis from the reliability graph, *IEEE transactions on Reliability, R-27*, 215-221.

Carpignano, A., & Poucet, A. (1994), Computer assisted fault tree construction: A review of methods and concerns, *Reliability Engineering & System Safety*, *44*, 265-278.

CCPS (Center for Chemical Process Safety) (1989), *Guideline for Chemical Process Quantitative Risk Analysis*, 1st edition, New York: Center for Chemical Process Safety, American Institute of Chemical Engineers (AIChE).

Center for Chemical Process Safety (CCPS) (1991), *Guidelines for Process Equipment Reliability Data with Data Tables*, New York: Center for Chemical Process Safety, American Institute of Chemical Engineers (AIChE).

Center for Chemical Process Safety (CCPS) (1992), *Guidelines for Hazard Evaluation Procedures with Worked Examples*, New York: Center for Chemical Process Safety, American Institute of Chemical Engineers (AIChE).

Center for Chemical Process Safety (CCPS) (2000), *Guidelines for Chemical Process Quantitative Risk Analysis*, 2nd edition, New York: Center for Chemical Process Safety, American Institute of Chemical Engineers (AIChE).

Chang, C.T., Hwang, H.C., Hwang, K.S. & Hsu, D.S. (1997), The loop identification and classification algorithms for digraph-based safety analysis, *Computer & Chemical Engineering*, 21, 223-239.

Chang, C.T., & Hwang, H.C. (1992), New developments of the digraph-based techniques for fault-tree synthesis, *Industrial and Engineering Chemistry Research, 31*, 1490-1502.

Cheng, Y.L., & Yuan J. (2000), On structured fault tree construction by modularizing control loops, *Reliability Engineering and System Safety*, *67*, 161-173.

Crowl, D.A., & Louvar J. F. (2002), *Chemical Process Safety: Fundamentals with Applications*, 2nd edition, Upper Saddle River, N.J.: Prentice Hall PTR.

Davidson, J. (1994), *The Reliability of Mechanical Systems*, London, UK: Mechanical Engineering Publications Ltd for The Institution of Mechanical Engineers.

European Industry Reliability Data Bank (EIREDA) (1998), *European Industry Reliability Data Bank*, Varese, Italy: C.E.C.-J.R.C./ISEI 21020 ISPRA.

Elliott, M.S. (1994), Computer-assisted fault-tree construction using a knowledge-based approach, *IEEE Transactions on Reliability, 43*, 112-120.

Evans, R.A. (1981), Book review: Reliability engineering and risk assessment, *IEEE Transactions on Reliability, R-30*, 475.

Fussell J.B. (1973), A formal methodology for fault tree construction, *Nuclear Science & Engineering*, *52*, 421-432.

Galluzzo, M., & Andow, P.K. (1984), Failures in control systems, *Reliability Engineering, 7*, 193-211.

Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P., & Rushton, A.G. (1993a), The propagation of faults in process plants: 6. Overview of, and modeling of fault tree synthesis, *Reliability Engineering, 39*, 173-194.

Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P., & Rushton, A.G. (1993b), The propagation of faults in process plants: 7. Divider and header units in fault tree synthesis, *Reliability Engineering, 39*, 195-209.

Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P., & Rushton, A.G. (1993c), The propagation of faults in process plants: 8. Control systems in fault tree synthesis, *Reliability Engineering, 39*, 211-227.

Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P., & Rushton, A.G. (1993d), The propagation of faults in process plants: 9. Trip systems in fault tree synthesis, *Reliability Engineering, 39*, 229-241.

Hunt, A., Kelly, B.E., Mullhi, J.S., Lees, F.P., & Rushton, A.G. (1993e), The propagation of faults in process plants: 10. Fault tree synthesis –2, *Reliability Engineering, 39*, 243-250.

IEC 61508 (1998), Functional safety of electrical/electronic/programmable electronic safety related systems, Part 1, 3, 4, and 5, *International Electrotechnical Commission*, Final standard.

IEEE-Std-500 (1984), IEEE guide to the collection and presentation of electrical, electronic, sensing component, and mechanical equipment reliability data for nuclear-power generating stations, *IEEE*, New York.

ISA-TR84.00.02 (2002a), Safety Instrumented Functions (SIF) – Safety Integrity Level (SIL) evaluation techniques part 1: Introduction, Research Triangle Park, NC: The Instrumentation, Systems, and Automation Society (ISA).

ISA-TR84.00.02 (2002b), Safety Instrumented Functions (SIF) – Safety Integrity Level (SIL) evaluation techniques part 2: determining the SIL of a SIF via simplified equations, Research Triangle Park, NC: The Instrumentation, Systems, and Automation Society (ISA).

ISA-TR84.00.02 (2002c), Safety Instrumented Functions (SIF) – Safety Integrity Level (SIL) evaluation techniques part 3: Determining the SIL of a SIF via fault tree analysis, Research Triangle Park, NC: The Instrumentation, Systems, and Automation Society (ISA).

ISA-TR84.00.02 (2002d), Safety Instrumented Functions (SIF) – Safety Integrity Level (SIL) evaluation techniques part 4: Determining the SIL of a SIF via Markov analysis, Research Triangle Park, NC: The Instrumentation, Systems, and Automation Society (ISA).

ISA-TR84.00.02 (2002e), Safety Instrumented Functions (SIF) – Safety Integrity Level (SIL) evaluation techniques part 5: Determining the PFD of SIS logic solvers via Markov analysis, Research Triangle Park, NC: The Instrumentation, Systems, and Automation Society (ISA).

Kelly, B.E., & Lees, F.P. (1986a), The propagation of faults in process plants: 1. Modeling of fault propagation, *Reliability Engineering*, *16*, 3-38.

Kelly, B.E., & Lees, F.P. (1986b), The propagation of faults in process plants: 2. Fault tree synthesis, *Reliability Engineering*, *16*, 39-62.

Kelly, B.E., & Lees, F.P. (1986c), The propagation of faults in process plants: 3. An interactive, computer-based facility, *Reliability Engineering*, *16*, 3-38.

De Kleer, J., & Brown, J.S. (1984), A qualitative physics based on confluences, *Artificial Intelligence*, *24*, 7-83..

Kocza, G., & Bossche, A. (1997), Automatic fault tree synthesis and real-time tree trimming based on computer models, *Proceedings of Annual Reliability and Maintainability Symposium*, *Philadelphia*, *PA*, (January 13-16), pp. 71-75.

Krishna, K., Wang, Y., Saraf, S. R., Rogers, W.J., Baldwin, J.T., Gupta, J.P., & Mannan, M.S. (2003), Hydroxylamine production: Will a QRA help you decide?, *Reliability Engineering & System Safety, 81*, 215-224.

Kumamoto, H. &, Henley, E.J. (1995), Automated fault tree synthesis by semantic network modeling, rule based developed and recursive 3-value procedure, *Reliability Engineering & System Safety*, *49*, 171-188.

Kuo, D.H., Hsu, D.S., Chang, C.T., & Chen, D.H. (1997), Prototype for integrated hazard analysis, *AIChE Journal, 43*, 1494-1510.

Kuo, D.H., Hsu, D.S. & Chang, C.T. (1998), A prototype for integrating automatic fault tree/event tree/HAZOP analysis, *Computer Chemical Engineering, 21*, 923-929.

Lambert, H.E. (1979), Comments on the Lapp-Powers 'computer-aided synthesis of fault trees', *IEEE Transactions on Reliability, R-28*, 6-11.

Lapp, S.A., & Powers, G.J. (1977), Computer-aided synthesis of fault trees, *IEEE Transactions on Reliability, R-26*, 2-13.

Lapp, A.S, & Powers, G.J. (1979), Update of Lapp-Powers fault-tree synthesis algorithm, *IEEE Transactions on Reliability, 28*, 12-15.

Lees, F.P. (1996), *Loss Prevention in the Process Industries*, 2nd edition, Boston, MA: Butterworth Heinemann.

Maurya, M. R., Rengaswamy, R., & Venkatsubramanian, V. (2001), Systematic development and application of digraphs for process diagnosis and hazard analysis, *On-line fault detection and supervision in the chemical process industries: a proceedings volume from the 4th IFAC workshop*, Jejudo Island, Korea, (June 7-8), pp. 327-332.

NUREG-0492 (1998), Fault Tree Handbook, *U.S. Nuclear Regulatory Commission*.

OREDA (1984), *Offshore Reliability Data Handbook*, 1st edition, Hovik, Norway: Det Norske Veritas.

OREDA (1988), *Offshore Reliability Data Handbook,* 2nd edition, Hovik, Norway: Det Norske Veritas.

OREDA (1997), *Offshore Reliability Data Handbook,* 3rd edition, Hovik, Norway: Det Norske Veritas.

OREDA (2002), *Offshore Reliability Data Handbook*, 4th edition, Hovik, Norway: Det Norske Veritas.

Poucet, A. (1990), Stars - knowledge based tools for safety and reliability analysis, *Reliability Engineering & System Safety, 30*, 379-397.

Powers, G.J., & Tompkins, F.C. (1974), Fault-tree synthesis for chemical processes, *AIChE Journal, 20*, 376-387.

Shafaghi, A., Andow, P.K., & Lees, F.P. (1984a), Fault tree synthesis based on control loop structure, *Chemical Engineering Research and Design*, *62,* 101-110.

Shafaghi, A., Lees, F.P., & Andow, P.K. (1984b), An illustrative example of fault tree synthesis based on control loop structure, *Reliability Engineering & System Safety*, *8*, 193-233.

Summers, A.E. (1998), Techniques for assigning a target safety integrity level, *ISA Transactions, 37,* 95-104.

Taylor, J.R. (1982), An algorithm for fault-tree construction, *IEEE Transactions on Reliability, R-31*, 137-146.

Kletz, Trevor (1999), *HAZOP and HAZAN: Identifying and Assessing Process Industry Hazards*, 4th edition, Warwickshire, UK: Institution of Chemical Engineers.

De Vries, R.C. (1990), An automated methodology for generating a fault tree, *IEEE Transactions on Reliability*, *39*, 76-86.

Wang, Y., Rogers, W.J., West, H.H. & Mannan, M.S. (2003), Algorithmic fault tree synthesis for control loops, *Journal of Loss Prevention in the Process Industries*, 16, 427-441.

Wang, Y., Teague, T.L., West, H.H., & Mannan, M.S. (2002), A new algorithm for computer-aided fault tree synthesis, *Journal of Loss Prevention in the Process Industries*, *15,* 265-277.

Wang, Y., West, H.H., & Mannan, M.S. (2004), The impact of data uncertainty in determining safety integrity level, *Transactions of the Institute of Chemical Engineers, Part B: Process Safety and Environmental Protection* (accepted).

VITA

Yanjun Wang was born in Ningbo, China in 1975. She graduated with a B. En. In chemical engineering from Zhejiang University, China in 1997. After graduating, she worked as a UNIX system administrator in Computer Department of China Construction Bank, Ningbo Branch.

In August 1999, she enrolled at Chemical Engineering Department at Texas A&M University as a Ph.D. student. Since then, she has worked in the area of quantitative risk analysis under the guidance of Dr. Sam Mannan. She received her Ph.D. in December 2004.

Her permanent address is: No. 10 Tashui Road, Fenghua, Ningbo, Zhejiang Province, P.R. China.