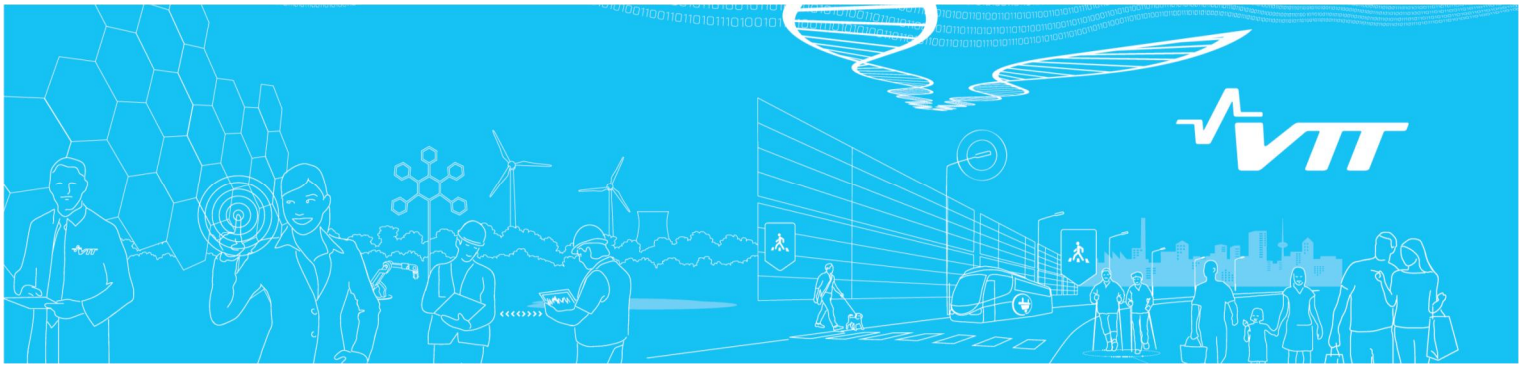


RESEARCH REPORT

VTT-R-03821-14

CORSICA Task 4.1 Hazard analysis methods of digital I&C systems

Authors: Ossi Teikari, VTT

Confidentiality: Public

Report's title

Hazard analysis methods of digital I&C systems

Customer, contact person, address	**Order reference**
VYR	4/2014SAF
Project name	**Project number/Short name**
Coverage and rationality of the software I&C safety assurance	85361 CORSICA
Author(s)	**Pages**
Ossi Teikari	64/
Keywords	**Report identification code**
Hazard analysis, FTA, FMEA, HAZOP, STAMP, STPA	VTT-R-03821-14

Summary

In this work we have reviewed three traditional hazard analysis methods, FTA, FMEA and HAZOP, as well as a recently developed STPA method that is based on the STAMP model. After giving an overview and case examples of the different methods, the methods are compared with each other, and their use in the context of nuclear domain I&C systems is discussed.

FTA, FMEA, and HAZOP have been around for decades. Compared to these traditional methods, STPA offers a very different approach for hazards analysis. STPA is based on control theory, and it was designed to better suit the analysis of complex, modern systems. Real-world experiences in the use of the method have been very positive so far. In some systems STPA has found faults that traditional hazard analysis methods have ignored. But despite the positive experiences, a larger use base and a larger amount of independent experience regarding the use of the method would be desirable. STPA has created major interest in many fields, so its use can be expected to increase.

Confidentiality	Public

Espoo 29.8.2014

Written by	**Reviewed by**	**Accepted by**
Ossi Teikari Research Trainee	Janne Valkonen Senior Scientist	Riikka Virkkunen Head of Research Area

VTT's contact address

VTT Technical Research Centre of Finland
P.O. Box 1000, FI-02044 VTT, Finland
Phone internat. +358 20 722 4520
Fax +358 20 722 4374

Distribution (customer and VTT)

VTT Kirjaamo, SAFIR2014 Reference group 2

Preface

This report has been prepared under the research project Coverage and rationality of the software I&C safety assurance (CORSICA), which is part of the Finnish Research Programme on Nuclear Power Plant Safety 2011–2014 (SAFIR2014). The research project aims to improve the safety evaluation of I&C systems in nuclear industry by improving consciousness of process assessment and rationality of integrated evaluation methods. In this work we have reviewed three traditional hazard analysis methods, FTA, FMEA and HAZOP, as well as a recently developed STPA method that is based on the STAMP model. After giving an overview of the different methods, in the end of the paper, we perform a review and a brief comparison between the methods, specifically in the context of nuclear domain I&C systems.

We wish to express our gratitude to the representatives of the organizations involved and all those who have given their valuable input in the meetings and discussions during the project.

Espoo, August 2014

Authors

Contents

Preface....2
Contents....3
Abbreviations....5
1. Introduction....7
2. Traditional hazard analysis methods....7
2.1 Background....7
2.2 FTA....7
2.2.1 Overview....7
2.2.2 Procedure....8
2.2.3 Software FTA....13
2.3 FMEA....14
2.3.1 Overview....14
2.3.2 Procedure....14
2.3.3 Software FMEA....15
2.4 HAZOP....15
2.4.1 Overview....15
2.4.2 Procedure....16
2.4.3 Software HAZOP....17
2.5 Combining methods – Case KNICS RPS....19
2.5.1 Overview....19
2.5.2 Software HAZOP....21
2.5.3 Software FTA....25
3. STAMP and STPA....29
3.1 Hazards in modern systems....29
3.2 STAMP....30
3.2.1 Background....30
3.2.2 Fundamental concepts....30
3.2.3 The nature of accidents according to STAMP....32

3.2.4 Developing a STAMP model 34
3.3 STPA hazard analysis method 35
3.4 Applying STPA – Case EPR MSIV 37
3.4.1 Overview 37
3.4.2 Accidents and hazards 38
3.4.3 Safety control structure 39
3.4.4 Process model variables 42
3.4.5 Unsafe control actions 43
3.4.6 Constraints and causal factors 46
3.4.7 Results 53
3.5 STPA in safety-guided design 53
4. STPA compared to traditional methods 57
4.1 General comparison 57
4.1.1 STPA vs. FTA 57
4.1.2 STPA vs. FMEA and HAZOP 57
4.1.3 Summary 58
4.2 Real-world experiences 60
5. Analysis methods in nuclear domain regulations and licensing 61
5.1 Current state 61
5.2 Potential benefits of STPA for licensing 62
6. Conclusions 63
References 64

Abbreviations

AOO	anticipated operational occurrence
CAST	causal analysis using system theory
CORSICA	coverage and rationality of the software I&C safety assurance
CVCS	chemical volume control system
DAS	diverse automation system
DOE	design of experiments
E/E/PE	electrical/electronic/programmable electronic
ECCS	emergency core cooling system
EPR	evolutionary power reactor
EPRI	electric power research institute
ESF	engineering safety features
ETA	event tree analysis
FBD	function block diagram
FMEA	failure mode and effects analysis
FTA	fault tree analysis
HAZOP	hazard and operability study
I&C	instrumentation and control
JAXA	Japan aerospace exploration agency
KNICS	Korean nuclear I&C systems project
MSIV	main steam isolation valve
NSSC	non-safety system controller
PA	postulated accidents
PLC	programmable logic controller
PM	priority module
PS	protection system
PWR	pressurised water reactor
PZR	pressurizer
QFD	quality function deployment

RPS	reactor protection system
SAFIR	Finnish research programme on nuclear power plant safety
SCRAM	emergency shutdown of a nuclear reactor
SCS	safety control system
SG	steam generator
SGTR	steam generator tube rupture
SI	safety injection
SPC	statistical process control
SSA	software safety analysis
STAMP	system-theoretic accident model and processes
STPA	system-theoretic process analysis

1. Introduction

During the last decades, the vast utilization of microprocessors and software has enabled engineers to design and build systems unlike ever before. Digitalization of old analogical systems has removed many of the physical boundaries limiting systems design. As a result, systems have grown more complex. I&C (instrumentation and control) systems where software and hardware together interact with the physical process and with other systems can include hidden faults and errors that might be critical for the correct operation of the system (and e.g. in the case of a nuclear reactor protection system, for the prevention of disasters potentially causing human casualties). Analysis of such systems is important, yet not simple. Various hazard analysis methods have been utilized in system hazard analysis. This report focuses on reviewing traditional and emerging techniques that are used, or could potentially be used, specifically in analysing I&C systems in the nuclear domain.

Chapter 2 presents an overview of three of the most commonly used traditional hazard analysis methods. Chapter 3 focuses on an emerging method called STPA (System-Theoretic Process Analysis), which is based on STAMP (System-Theoretic Accident Model and Processes). In the end of these sections, case examples are introduced to clarify the process of how the methods can be applied. Chapter 4 establishes a general comparison between the methods and discusses some results from real-world experiences in various fields. Related regulation and licensing issues in the nuclear domain are discussed in Chapter 5. Finally, conclusions of the study are presented in Chapter 6.

2. Traditional hazard analysis methods

2.1 Background

Three traditional hazards analysis methods called FTA (Fault Tree Analysis), FMEA (Failure Mode and Effects Analysis), and HAZOP (Hazard and Operability Study) are introduced in this chapter. These methods were specifically selected since they are used most extensively and they represent the standard practise in many domains. All of the methods have also been used in the nuclear energy sector.

All three introduced methods were invented several decades ago. When the methods were originally developed, systems were commonly built using mechanical and electrical components instead of a combination of hardware and software. Since digital technology finally spread to I&C systems old hazard analysis methods have also been used to analyse such modern systems – systems that are somewhat beyond the original context of the methods. Nowadays software is used more and more even in the most safety-critical contexts such as nuclear power plant I&C systems. Therefore, after giving an overview of the traditional application procedure for each method, some software-specific application issues for the methods are discussed.

2.2 FTA

2.2.1 Overview

FTA is a standard method for the assessment and improvement of reliability and safety. It has been and it is applied in various sectors, such as nuclear industry, air and space industry, electrical industry, chemical industry, railway industry, transport, software reliability, and insurance. FTA is an analytical technique, where an undesired state of the system is specified and then the system is analysed to find all realistic ways in which the undesired event can occur. The undesired state of the system, which is identified at the beginning of

the fault tree analysis, is usually a state that is critical from a safety or reliability standpoint and is identified as the top event. Top event is therefore an undesired event, which is further analysed with the fault tree analysis. [1]

The fault tree is a graphic model of the various parallel and sequential combinations of faults that can lead to the occurrence of the top event. The logical gates of the fault tree integrate the basic events to the top event. The basic events are events that are not further developed. They are the ultimate parts of the fault tree, which represent the undesired events and their failure modes, e.g., the component failures, the missed actuation signals, the human errors, etc. [1]. If fault tree analysis is used to evaluate the probability of the top event, failure data needs to be available for each basic event.

There also exists a related approach called ETA (Event Tree Analysis). Whereas fault trees are better suited for analysing the causes of a single event, event trees are used for defining the possible consequences of an initiating event. ETA is often combined with FTA to perform probabilistic safety assessments.

2.2.2 Procedure

This section follows the procedure used in [1], and the text is largely adapted from [1].

Following is one procedure of performing a FTA:

1) Identification of the objectives for the fault tree analysis

2) Definition of the top event of the fault tree

3) Definition of the scope, resolution, and ground rules of the fault tree

4) Fault tree construction

5) Qualitative fault tree evaluation

6) Preparation of the probabilistic failure database and connection of the basic events of the fault tree with probabilistic failure data

7) Quantitative fault tree evaluation

8) Interpretation of the fault tree analysis results

The procedure is based on the assumption that evaluation for the top event probability is wanted to be calculated. If failure probabilities are unknown or irrelevant (as is the case for software FTA), the fault tree is only used to model *how* the failure might happen, and quantitative steps 6 and 7 can be ignored. Each phase is discussed in more depth below.

1) Identification of the objectives for the fault tree analysis

Objectives of the fault tree analysis can include assessment of the failure probability of the system, fulfilment of the regulatory objectives, identification of the most important components of the system considering reliability, etc.

2) Definition of the top event of the fault tree

Top event, the undesired event to be analysed, is chosen based on the objectives of the analysis. For example, top event for analysing nuclear power plant safety could be “release of radiation to the environment”.

3) Definition of the scope, resolution, and ground rules of the fault tree

Scope includes defining which of the faults and contributors are included in the model and in the analysis, and which are not. Boundary conditions, including how treatment of the outside event is considered in the analysis, also have to be defined. E.g., it could be (rather unwisely) assumed that a nuclear power plant always has external power available, and situations where power both from the grid and emergency power sources are lost can be ignored.

Resolution includes defining to what extent larger individual components are broken into smaller pieces in the fault tree. E.g., defining if pumps are considered as single components or broken into parts including valves, lines etc.

Ground rules include general conventions such as the overall procedure how the tree is developed and rules for naming basic events and gates.

4) Fault tree construction

The fault tree can be identically developed either graphically or using Boolean equations. Figure 1 shows the most common fault tree symbols and logic gates used in the graphic representation of fault trees.

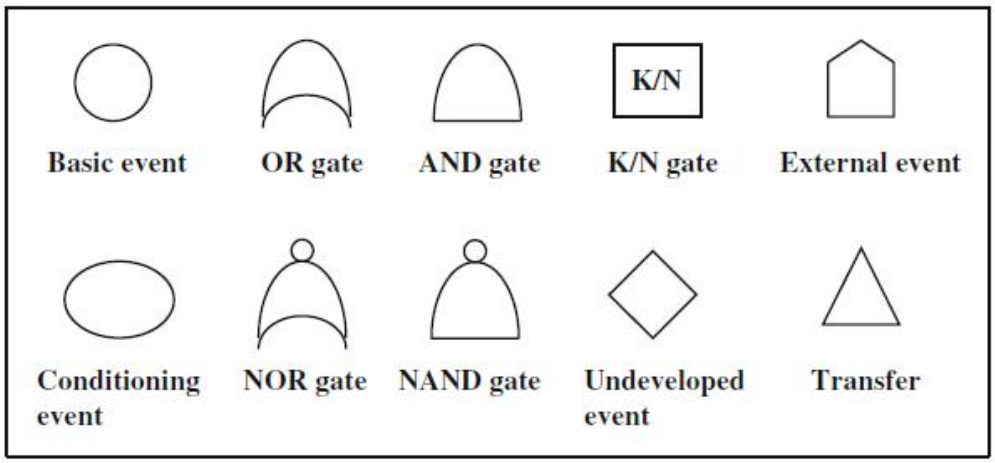
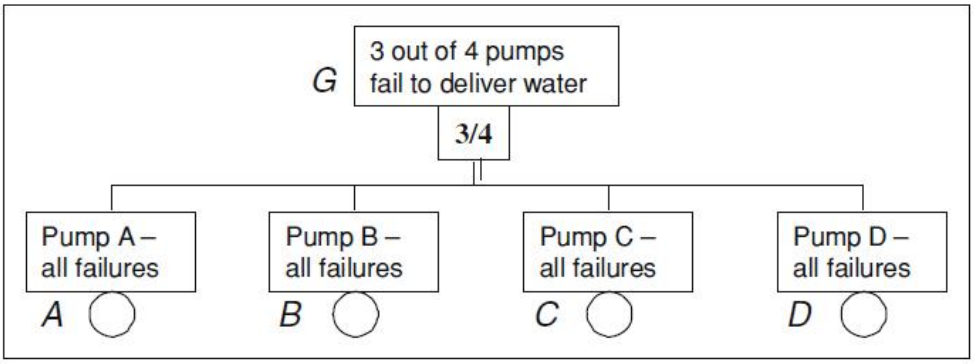


Figure 1. Fault tree symbols [1]

The basic event represents a failure in a system component. The K/N gate indicates that the output event occurs if K input events occur at the same time. This is often the case in a situation where there is redundancy built in the system, such as in the example of Figure 2, where the system consists of four pumps, and at least two of the pumps are required to supply sufficient amount of water.

The external event is an event, which is not a failure. For example, the change of operating mode of a system could be considered an external event. The conditioning event represents any conditions or restrictions that apply to any logic gate, such as the system’s current mode of operation. Undeveloped event is an event which is not developed further, because of its insignificance or due to unavailability of information. The transfer symbol is used to connect

fault trees between each other, such as connecting the fault tree of a system to the fault tree of its subsystem.



A	*B*	*C*	*D*	*G = 3/4 (A B C D)*
True	True	True	True	True
True	True	True	False	True
True	True	False	True	True
True	True	False	False	False
True	False	True	True	True
True	False	True	False	False
True	False	False	True	False
True	False	False	False	False
False	True	True	True	True
False	True	True	False	False
False	True	False	True	False
False	True	False	False	False
False	False	True	True	False
False	False	True	False	False
False	False	False	True	False
False	False	False	False	False

Figure 2. An example of the K/N gate [1]

The fault tree is constructed by connecting the top event to causing events by gates and refining these events further until only basic events are left to be connected. Figure 3 shows an example tree structure for a simple system, where the goal is to deliver water to a tank for 4 hours. This is achieved using two pumps, pumps A and B, each of which is independently capable of providing sufficient amount of water. The top event for the fault tree analysis, pumps failing to deliver enough water, can therefore happen only when both the pumps fail. Thus the use of AND gate for the top event is required. The tree is then completed by connecting both the pump failure events to their basic events through OR gates.

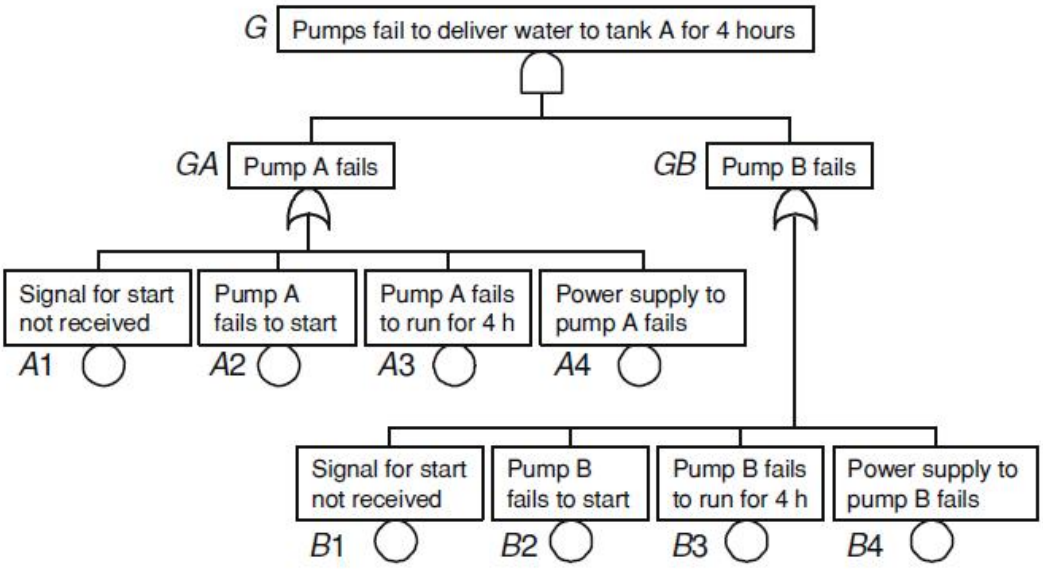


Figure 3. Fault tree for a simple example system [1]

The same graphic tree could be written identically as a set of Boolean equations. Equivalent equations for the tree in Figure 3 are:

$$G = GA \times GB, \tag{1.1}$$

for the AND gate, and

$$GA = A1 + A2 + A3 + A4, \tag{1.2}$$

as well as

$$GB = B1 + B2 + B3 + B4, \tag{1.3}$$

for the OR gates.

5) <u>Qualitative fault tree evaluation</u>

Qualitative fault tree evaluation involves finding the combination of basic events which can cause the top event to occur. If the representation is in graphic form, Boolean equations need to be written based on the logic gates and inputs. Then, using the Boolean algebra, an equation for the top event is obtained. This equation consists of sum of products of basic events. Using equations 1.1 – 1.3, the equation for the top event of the pump example would be:

$$G = (A1 + A2 + A3 + A4) \times (B1 + B2 + B3 + B4) \tag{1.4}$$

The top event equation can then be used to identify the minimal cut sets. A minimal cut set is a set of least amount of basic events that, occurring together, lead to the top event. Minimal cut sets in the pump example include 2 basic events (for example A1 and B1), since both the pumps need to fail in order for the top event to occur. Altogether the example tree has 16 minimal cut sets:

$$G = A1 \times B1 + A2 \times B1 + A3 \times B1 + A4 \times B1 + A1 \times B2 + A2 \times B2 + A3 \times B2 + A4 \times B2 + A1 \times B3 + A2 \times B3 + A3 \times B3 + A4 \times B3 + A1 \times B4 + A2 \times B4 + A3 \times B4 + A4 \times B4 \quad (1.5)$$

Top event G occurs if any of these minimal cut sets occur.

6) Preparation of the probabilistic failure database and connection of the basic events of the fault tree with probabilistic failure data

When the mechanism for the top event is discovered in the form of basic events, the next step is connecting these basic events with probabilistic failure data. This involves selecting the probabilistic model, preparing the failure database, and linking the selected model with data from the database.

Since the qualities and roles of components used in systems differ to a great extent, different probabilistic models are needed to model the probability of component failure. For example, a valve that has to remain open likely has a completely different failure probability than a valve of the same model that periodically has to switch between open and closed. The probability model has to be chosen for each component depending on its role in the system.

Next, sufficient data has to be gathered either from sources including, for example, existing databases, documents, reports, and industry experience. Once the probabilistic failure models are chosen and relevant data is available for each component, the data in the database is linked against each basic event to calculate the failure probabilities.

7) Quantitative fault tree evaluation

The failure probability of each basic event can be calculated using the selected probabilistic models and data gathered on each component. Then, basic event probabilities are used to calculate a probability for each of the minimal cut sets identified in Step 5. Under the assumption that the basic events are mutually independent, the probability of each minimal cut set can simply be defined as:

$$P_{MCS_i} = \prod_{j=1}^{m} P_{B_j} \quad (1.6)$$

where P_{MCS_i} is the minimal cut set probability, P_{B_j} is the basic event probability and m is the number of basic events in the minimal cut set.

After probabilities of each minimal cut set are known, the top event probability can be calculated using the following general equation:

$$P_{TOP} = \sum_{i=1}^{n} P_{MCS_i} - \sum_{i<j} P_{MCS_i \cap MCS_j} + \sum_{i<j<k} P_{MCS_i \cap MCS_j \cap MCS_k} - \cdots + (-1)^{m-1} P \bigcap_{i=1}^{m} MCS_i \quad (1.7)$$

where P_{TOP} is the top event probability, n is the number of minimal cut sets, and m is the number of basic events in the largest minimal cut set.

Equation 1.7 may be too complex for large fault trees. In these cases, a simplified version that considers only a limited number of summands may be used. For example, using three summands:

$$P_{TOP} = \sum_{i=1}^{n} P_{MCS_i} - \sum_{i<j} P_{MCS_i \cap MCS_j} + \sum_{i<j<k} P_{MCS_i \cap MCS_j \cap MCS_k} \quad (1.8)$$

Sometimes even using the first summand is enough. This approximation is called the first-order approximation.

For the pump example of Figure 3, if the failure probability for each component is 1E-2, the top event probability can be calculated as:

$$P_{TOP} = \sum_{i=1}^{16} 10^{-2} \times 10^{-2} - \sum_{1}^{15+14+\cdots+1} 10^{-2} \times 10^{-2} \times 10^{-2} \times 10^{-2} = 0.16 - 0.012 = 0.148$$

In addition to the top event probability, other factors can be calculated. These include Fussel-Vesely importance, risk achievement worth, risk reduction worth, Birnbaum importance, criticality importance and differential importance measure. Depending on the case, importance measures can be calculated for each of the basic events or for groups of basic events.

8) <u>Interpretation of the fault tree analysis results</u>

The interpretation of the fault tree analysis results is a phase where the qualitative and quantitative results of the fault tree analysis are considered together with the assumptions and limitations of the analysis, with the boundary conditions of the analysis and the resolution of modelling. It is a phase where the sensitivity and uncertainty of the results are evaluated before the final conclusions are made.

2.2.3 Software FTA

Software is fundamentally different from hardware. Whereas probability estimates can often be defined for the failure of hardware components, a similar estimation cannot be derived for software, since the operation of software is deterministic. Software is either correct or incorrect, and if a part of software was known to be incorrect, it would be fixed. Therefore the quantitative steps of FTA have to be ignored when performing a software FTA. However, the method can be used to discover potential logical errors in software, and it can be performed in various different abstraction levels.

One approach for the application of software FTA in digital I&C systems was developed in the KNICS (Korean Nuclear Instrumentation & Control Systems) project. Modern digital I&C systems are often built on programmable platforms such as PLC (programmable logic controller) that run on microcontrollers. These platforms usually run on pre-developed operational system software and include a library of standardised logic blocks, and application software is developed by connecting them together using a graphical language called FBD (function block diagram). Therefore it makes sense to use these logic blocks as the basis for the FTA. To help achieve this, Oh et al. [14] introduce the concept of fault tree templates. Fault tree templates are fundamentally fault trees that represent the inherent failure mode of a single logic block.

The procedure of using software FTA with fault tree templates involves first acquiring a FBD model of the software that is to be analysed and developing a fault tree template for each of the logic blocks appearing in the model. Next, the top event (e.g. an alarm variable is 0 when it should be 1) is chosen. The tree corresponding to the top event is then built based on the FBD by systematically applying a template to each logic block of the tree. A case example of the application of software FTA can be found in section 2.5.

2.3 FMEA

2.3.1 Overview

FMEA can be described as a systematic way of identifying failure modes of a system, item or function, and evaluating the effects of the failure modes on the higher level. The objective is to determine the causes for the failure modes and what could be done to eliminate or reduce the chance of failure. [5]

FMEA can be divided into various types, depending on the source or author. For example, one common classification is to divide FMEA into product FMEA and process FMEA. The product FMEA analyses the design of a product by examining the way that item's failure modes affect the operation of the product. The process FMEA analyses the processes involved in design, building, using, and maintaining a product by examining the way that failures in the manufacturing or service processes affect the operation of the product. For software-based systems both the product and process FMEA are appropriate. [5]

2.3.2 Procedure

The procedures of conducting FMEA may vary. In the following, a general form of the FMEA procedure based on [18] and [5] is introduced.

Figure 4 shows the main phases of the analysis process.

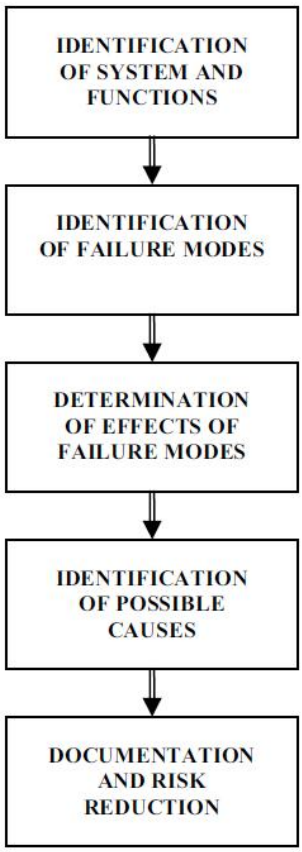


Figure 4. Main phases of FMEA [5]

Firstly, the system and its functions that are to be analysed have to be identified. To make sure that every member of the analysis team has a similar understanding of the system and

its potential problems, this step should be supported by creating a functional block diagram (for the system FMEA) or a process flowchart (for the process FMEA). The next step is to identify the failure modes of the system. This can be done e.g. by brainstorming. There exist several different worksheets to support the FMEA process. One of them should be chosen, and at this step of the FMEA, the team should start filling in the worksheet.

In the next phase, the effects and causes of the identified failure modes are identified. This phase may be supported by multiple techniques such as brainstorming, cause-and-effect analysis, QFD (quality function deployment), DOE (design of experiments), SPC (statistical process control), another FMEA, mathematical modelling, simulation, reliability analysis, and any other technique deemed suitable. The results are filled in the FMEA worksheet. In the last phase, the process is documented and corrective actions are taken to mitigate the identified failure modes.

2.3.3 Software FMEA

FMEA was originally developed to be used for mechanical and electrical systems. The procedure of using FMEA for software is rather similar. However, since software-based systems are fundamentally different from traditional electrical systems, some points need to be considered if the use of FMEA for software is desired.

Modern I&C systems are often built on programmable platforms using function block diagrams. Because the function block internals are usually not known, their failure modes are also unknown. Thus, usually the accuracy of software FMEA has to be limited on the function level. Failure modes of hardware components are generally quite well known, since operational experience and manufacturer's instructions can be used as a basis for their identification. For software components, however, failure modes are generally unknown. Identifying the ways in which software might "fail" (i.e. work in a flawed manner) is very difficult and applying the experience of software experts is required. [5]

2.4 HAZOP

2.4.1 Overview

HAZOP is a systematic method to analyse risks in a given system. It was originally developed for the chemical industry but has more recently been applied to various other types of systems, including nuclear power plants. It may be used for an existing system or during the system design phase before the system has been realised. HAZOP is usually conducted by a multidisciplinary team via HAZOP meetings and will likely take several weeks to complete.

The goal of the HAZOP process is to identify possible deviations from normal operations of the system and ensure that appropriate safeguards are in place to help prevent accidents. A design representation is used as the basis of the study. [21] Depending on the type of the system, the design may be represented by e.g. a P and ID (piping and instrumentation diagram), block diagram, data flow diagram or state transition diagram.

The unique feature in HAZOP is the use of guidewords to aid the analysis team in identifying possible hazards in a given system function. Guidewords such as "more" or "less" are paired with parameters such as "flow" to identify deviations such as "increased flow" or "decreased flow". This should ensure that each credible deviation from normal condition is systematically considered. [21] An example list of guidewords and parameters used in HAZOP is listed in Table 1.

Table 1. Common guidewords and parameters in HAZOP

Guideword	**Parameter**
No	Flow
More	Pressure
Less	Temperature
As well as	Level
Part of	Time
Reverse	Composition
Other than	…

2.4.2 Procedure

The procedure for performing a HAZOP analysis is the following [21]:

1. Define the system or activity

   Because the focus on HAZOP is identifying how the system might deviate from its normal operations, it is important to clearly define the intended functions for a system being analysed. Also, since systems usually interface with the environment and other systems, the boundaries for the system being analysed should be defined. This helps the analysis team avoid overlooking elements at interfaces.

2. Define the problems of interest for the analysis

   The problems of interest that the analysis will address are specified. These may include health and safety issues, environmental concerns, etc.

3. Subdivide the system or activity and develop deviations

   This phase is conducted before HAZOP team meetings to make the meetings more efficient. During the phase, the system or activity is divided into relevant functional sections. Sections should be small enough to ensure that no deviations are missed, but large enough to avoid excessive workload. Sections may include, for example, a section of piping, a tank, a part of a reaction, etc. Deviations are then derived from the sections by applying guidewords and parameters listed in Table 1 in a systematic manner:

   Guideword + parameter = deviation

   The type of the unit being analysed determines the applicable parameters for that section.

4. Conduct HAZOP reviews

   During the HAZOP team meetings, each deviation for each identified section is systematically analysed. The team defines the consequences of each deviation. If consequences include potential accidents, the team defines all the possible causes of the deviation. Then, the engineering and administrative controls that protect against

the deviation are identified. The team discusses if these controls are concerned adequate, and if not, recommendations for corrective actions are made. All the analysis done for each deviation is clearly documented.

5. Use the results in decision making

   Results from phase 4 are used in judging whether the system or activity is acceptable as is. Elements that most likely contribute to future reliability-related problems are identified, and specific suggestions for improving the system or activity are made. These may include e.g. equipment modifications and administrative policy changes. Also, benefits from the improvements are compared against their costs, and justification of allocating resources for the improvements is made.

The whole process should be precisely documented. Amount of documentation may grow substantial, and specialized software is often used to ease the documentation effort.

2.4.3 Software HAZOP

There is a limited amount of published research regarding the use of HAZOP for software systems. Lawrence [10] suggests that the HAZOP procedure, with a few modifications, may be used in software requirements, architectural design, detailed design and code implementation phases. For software HAZOP, Lawrence suggests the use of software qualities instead of parameters. These qualities are listed in Table 2. Additionally, instead of pairing qualities with guidewords, Lawrence establishes a set *guide phrases* for each quality. Example guide phrases for one of the software qualities are listed in Table 3.

Table 2. Software qualities [10]

Quality	Description
Accuracy	The term accuracy denotes the degree of freedom from error of sensor and operator input, the degree of exactness possessed by an approximation or measurement, and the degree of freedom of actuator output from error.
Capacity	The term capacity denotes the ability of the software system to achieve its objectives within the hardware constraints imposed by the computing system being used. The main factors of capacity are Execution Capacity (timing) and Storage Capacity (sizing). These refer, respectively, to the availability of sufficient processing time and memory resources to satisfy the software requirements.
Functionality	The term functionality denotes the operations which must be carried out by the software. Functions generally transform input information into output information in order to affect the reactor operation. Inputs may be obtained from sensors, operators, other equipment or other software as appropriate.

	Outputs may be directed to actuators, operators, other equipment or other software as appropriate.
Reliability	The term reliability denotes the degree to which a software system or component operates without failure. This definition does not consider the consequences of failure, only the existence of failure. Reliability requirements may be derived from the general system reliability requirements by imposing reliability requirements on the software components of the application system which are sufficient to meet the overall system reliability requirements.
Robustness	The term robustness denotes the ability of a software system or component to function correctly in the presence of invalid inputs or stressful environmental conditions. This includes the ability to function correctly despite some violation of the assumptions in its specification.
Safety	The term safety is used here to denote those properties and characteristics of the software system that directly affect or interact with system safety considerations. The other qualities discussed in this table are important contributors to the overall safety of the software-controlled protection system, but are primarily concerned with the internal operation of the software. This quality is primarily concerned with the effect of the software on system hazards and the measures taken to control those hazards.
Security	The term security denotes the ability to prevent unauthorized, undesired and unsafe intrusions. Security is a safety concern in so far as such intrusions can affect the safety-related functions of the software.

Table 3. Guide phrases for the 'functionality' quality (adapted from [10])

Software quality	Guide phrase
Functionality	Function is not carried out as specified (for each mode of operation)
	Function is not initialized properly before being executed

	Function executes when trigger conditions are not satisfied
	Trigger conditions are satisfied but function fails to execute
	Function continues to execute after termination conditions are satisfied
	Termination conditions are not satisfied but function terminates
	Function terminates before necessary actions, calculations, events, etc. are completed
	Function is executed in incorrect operating mode
	Function uses incorrect inputs
	Function produces incorrect outputs

The elements that are analysed may be requirements, architectural or detailed design elements, or code blocks, depending on which lifecycle phase the method is used in. The basis of the approach is determining which of the software qualities each element is related to, and then systematically considering deviations of the guide phrases corresponding to that quality. Some of the guide phrases might not be applicable to a certain case, and the suitable guide phrases should be selected as part of the software HAZOP procedure. The method is illustrated in a case example in the following section.

2.5 Combining methods – Case KNICS RPS

2.5.1 Overview

Because all of the introduced methods (FTA, FMEA and HAZOP) are somewhat different in nature and they all have their distinct characteristics, it is rather common to combine various methods when conducting hazard analysis for larger systems. When e.g. an I&C system that consists of hardware as well as software components is considered, it might be wise to choose different analysis methods for the system-level, hardware-level and software-level analysis. This approach was used in the design phase of a digital RPS (reactor protection system) developed in the KNICS (Korea Nuclear Instrumentation & Control) project.

Park et al. [7, 9] describe the process of conducting SSA (software safety analysis) for the KNICS project. A comprehensive approach was adopted, where SSA is performed at each phase of the software safety lifecycle. Overview of the SSA process is presented in Figure 5.

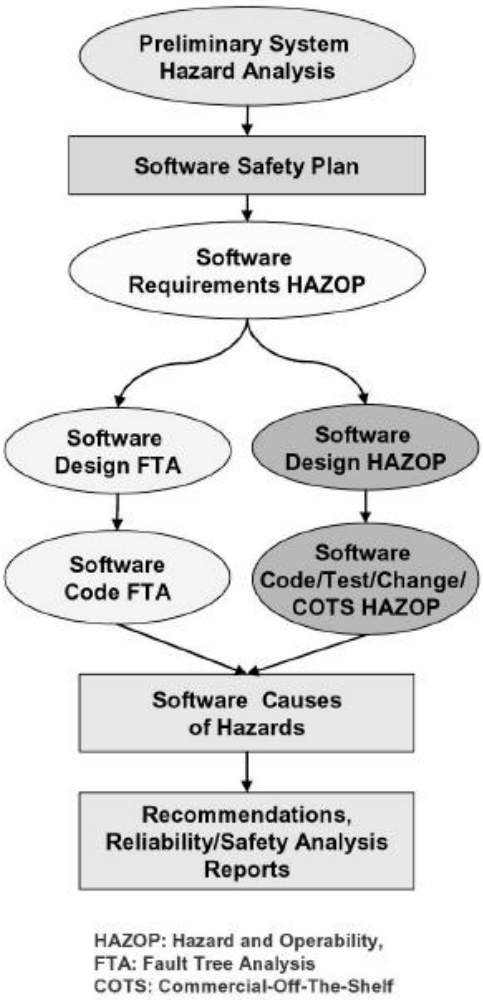


Figure 5. Software safety analysis for the KNICS project [16]

The process is following:

1. The process begins by creating a software safety plan based on a preliminary system hazard analysis.

2. The software-contributable system hazards are identified by reviewing the system FMEA results. These hazards are shown in Table 4.

3. Software HAZOP is utilized in the requirements phase.

4. In the design and implementation phase, the software HAZOP is applied to the software represented by a function block diagram. HAZOP is performed to evaluate the whole FBD design with respect to the software-contributable system hazards. This way the most potential areas for defects can be identified from the design.

5. A software FTA is then used to perform a detailed analysis of each of the defective areas to accurately identify the error in the logic or design.

Table 4. Software-contributable hazards of the KNICS RPS (adapted from [15])

No.	Software-contributable hazards	Criticality

		level
1	RPS cannot generate a trip signal when a trip condition for a process variable is satisfied.	4
2	RPS generates a trip signal when it should not generate a trip signal.	3
3	RPS cannot send qualified information of its operational status to the main control room for an operator.	2

It should be noted that FTA is a backward local analysis method, while HAZOP is a bidirectional broad-thinking method. Both methods are somewhat redundant and complementary. In the KNICS example, HAZOP was mainly used to identify consequences of deviations rather than identifying their causes. Thus, the forward analysis part of HAZOP was weighted much more.

The next two sub sections provide a more detailed explanation of phases 4 and 5.

2.5.2 Software HAZOP

In this example, only the most significant software-contributable hazard (item number 1 in Table 4) is considered. The hazard covers an event where the RPS doesn't generate a trip signal even though it should, i.e. the system fails to fulfil its most fundamental objective.

For the SWHAZOP and SWFTA to be applicable, the software-contributable system hazards and the interface points between them and the application software must be known. In the example system, output variables of all the trip logic software modules are the ones that act as interface points to the hazard. All these interface modules are listed in Table 5.

Table 5. Software modules interfacing with hazard no. 1 (adapted from [15])

Module	**Description**
PZR_PR_Hi Trip	Pressurizer Hi Pressure Trip
SG1_LVL_Lo_RPS Trip	SG-1 Low Level Trip
SG1_LVL_Lo_ESF Trip	SG-1 Low Level Trip for ESF
SG1_LVL_Hi Trip	SG-1 Hi Level Trip
SG1_PR_Lo Trip	SG-1 Low Pressure Trip
CMT_PR_Hi Trip	Containment Hi Pressure Trip
CMT_PR_HH Trip	Containment Hi-Hi Pressure Trip
SG1_FLW_Lo Trip	SG-1 Low Coolant Flow Trip
PZR_PR_Lo Trip	Pressurizer Low Pressure Trip
VA_OVR_PWR_Hi Trip	Variable Over Power Hi Trip
SG2_LVL_Lo_RPS Trip	SG-2 Low Level Trip

SG2_LVL_Lo_ESF Trip	SG-2 Low Level Trip for ESF
SG2_LVL_Hi Trip	SG-2 Hi Level Trip
SG2_PR_Lo Trip	SG-2 Low Pressure Trip
SG2_FLW_Lo Trip	SG-2 Low Coolant Flow Trip
LOG_PWR_Hi Trip	Log Reactor Power Hi Trip
DNBR_Lo Trip	Low DNBR Trip
LPD_Hi Trip	Hi LPD Trip
CPC_CWP Trip	CPC CWP

For the KNICS project, a software HAZOP approach similar to the one introduced in section 2.4.5 was adopted. The approach involves using guide phrases instead of guide words, and functional characteristics (software qualities) over traditional HAZOP parameters such as temperature. Guide phrases suitable for the case were carefully selected. These are presented in Table 6.

Table 6. Characteristics, guide phrases and deviation checklists for the FBD modules (adapted from [15])

Functional characteristic	**Guide phrase**	**Deviation checklist**
Accuracy	Below minimum range	What is the consequence if the sensor value is below its minimum range?
Accuracy	Above maximum range	What is the consequence if the sensor value is above its maximum range?
Accuracy	Within range, but wrong	What is the consequence if the sensor value is within its physical range but incorrect?
Accuracy	Incorrect physical units	What is the consequence if the input has an incorrect physical unit?
Accuracy	Wrong data type or data size	What is the consequence if the input has a wrong data type or data size?
Accuracy	Wrong physical address	What is the consequence if the input variable is allocated to a wrong physical address?
Accuracy	Correct physical address, but wrong variable	What is the consequence if a wrong input variable is allocated to a correct

		physical address?
Accuracy	Wrong variable type or name	What is the consequence if wrong type or name for an input /output/internal variable is used in the FBD module?
Accuracy	Incorrect variable initialization	What is the consequence if the input/output/internal variables are initialized incorrectly?
Accuracy	Wrong constant value	What is the consequence if the internal constant is given a wrong value?
Accuracy	Incorrect update of history variables	What is the consequence if the variable is updated incorrectly?
Accuracy	Wrong set point calculation	What is the consequence if the procedure for calculating a set point is incorrect?
Capacity	Erroneous communication data	What is the consequence if there is an error in the ICN data?
Capacity	Erroneous communication data	What is the consequence if there is an error in the SDL data?
Capacity	Unexpected input signal	What is the consequence when an unexpected input signal is arrived?
Capacity	Untimely operator action	What is the consequence if the operator commences a set point reset or an operating bypass function untimely?
Functionality	Function is not carried out as specified	What is the consequence if some portions in the FBD module have a defect or cannot perform the intended behavior?
Reliability	Data is passed to incorrect process	What is the consequence if the data is passed to an incorrect process?
Robustness	Incorrect selection of test mode	What is the consequence if the test mode is selected or changed unexpectedly?

Robustness	Incorrect input selection	What is the consequence if the input selection is incorrect?

In the software HAZOP process, all the items and checklists of Table 6 are iteratively applied to each of the modules in Table 5. The process takes a significant amount of time, and a well-established HAZOP team is essential for the effort. Table 7 presents results from applying HAZOP to a single module from Table 5. The chosen module is “SG1_FLW_Lo Trip”, which represents the steam generator no. 1 low coolant flow trip function.

Table 7. HAZOP analysis of SG1_FLW_Lo Trip module (adapted from [15])

Funct. characteristic	**Deviation checklist**	**Cause**	**Analysis**	**Effect**	**Criticality**	**Suggestion**
Accuracy	What is the consequence if the sensor value is below its minimum range?	Sensor failure	The TRIP_DECISION sub-module handles properly an out-of-range value, but it is carried out after all logical operations are done.	No effect on safety, but operability is poor.	2	It is desirable that a trip signal occurs at the front when an out-of-range sensor inputs value exists.
Accuracy	What is the consequence if the sensor value is above its minimum range?					
Accuracy	What is the consequence if the sensor value is within its physical range but incorrect?	Sensor failure or input conditioner malfunction	This is the problem at input conditioning processor.	Severe effect on safety.	4	Measures should be provided at input processor.
Accuracy	What is the consequence if the internal constant is given a wrong value?	Wrong constant value allocation	If MAXCNT is set to 0, the trip signal is always ON regardless of the trip condition status.	Poor operability	3	Need careful attention when assigning a value.

			If MAXCNT is too large, the trip signal is generated at much later time.	Violating the system response time	4	
Capacity	What is the consequence when an unexpected input signal is arrived?	ATIP error	No part performs an exceptional handling when ATIP sets up an erroneous test operation.	Wrong test execution	1	Augmented test mode selection.
Functionality	What is the consequence if some portions in the FBD module have a defect or cannot perform its intended behaviour?	Error in logic operation	Pre-trip is cancelled whenever it is triggered at the pre-trip sub-module.	Pre-trip is never functioning	3	Modify a pre-trip logic
			The hysteresis is not reflected in the trip logic sub-module because of using 19th previous value.	Inducing a trip malfunction	4	Modify trip logic.

2.5.3 Software FTA

Based on the results of HAZOP, the most critical software modules are chosen for closer inspection, and SWFTA is performed on them. A SWFTA approach similar to the one described in section 2.2.3 is used, including refined fault tree templates. The software is represented as a function block diagram, and the function blocks used were divided into five categories:

- Logic operations (AND, OR)
- Comparison (GE, GT, LE, LT, EQ)
- Selection (SEL)
- Algebraic operation (ADD, SUB, MUL, DIV, ABS)
- Timer (TON)

Figure 6 represents an example of a fault tree template for an ADD block.

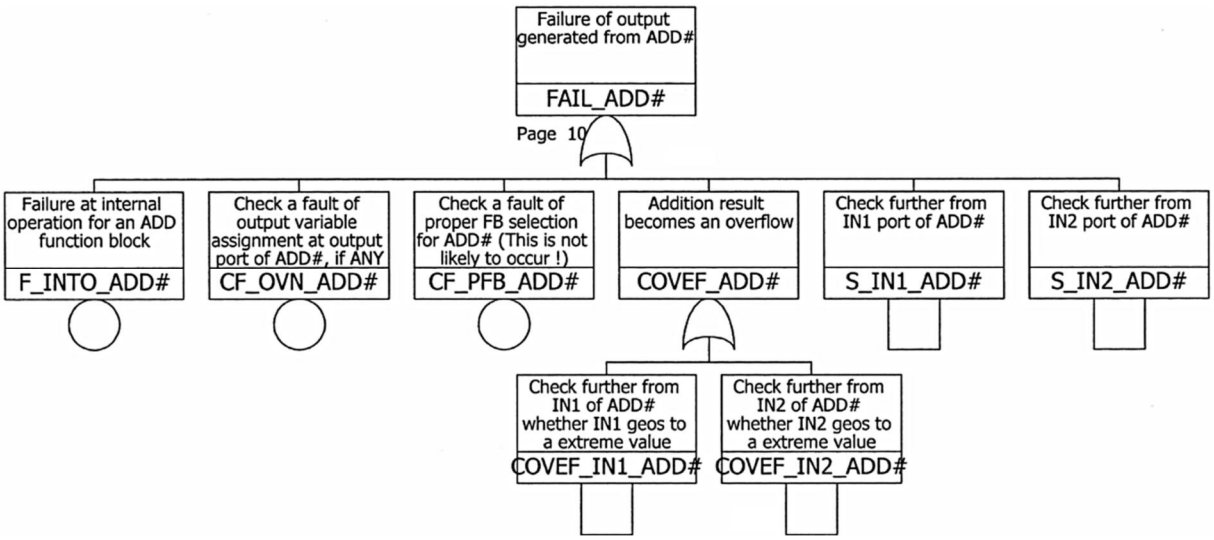


Figure 6. Fault tree template for an ADD function block

Fault trees were then constructed for each of the most critical software modules by placing the software-contributable hazard as the top event of the tree and applying fault tree templates for the FBD representations. As an example, the module “DNBR_LO trip” (see Table 5) is examined. Figure 7 shows the FBD for the module. The FBD should be read from left to right and from top to bottom (i.e. SEL1 -> AND1 -> OR1). Figure 8 shows the fault tree that was developed for the module. The top node of the tree represents the hazard where the software doesn’t initiate a trip signal even though trip conditions are met (i.e. hazard no. 1 in Table 4). In this case the hazard is realised when the value of _4_TRIP (at OR1) is 0 when the input trip variable _4_TRIP (at SEL1) becomes 1.

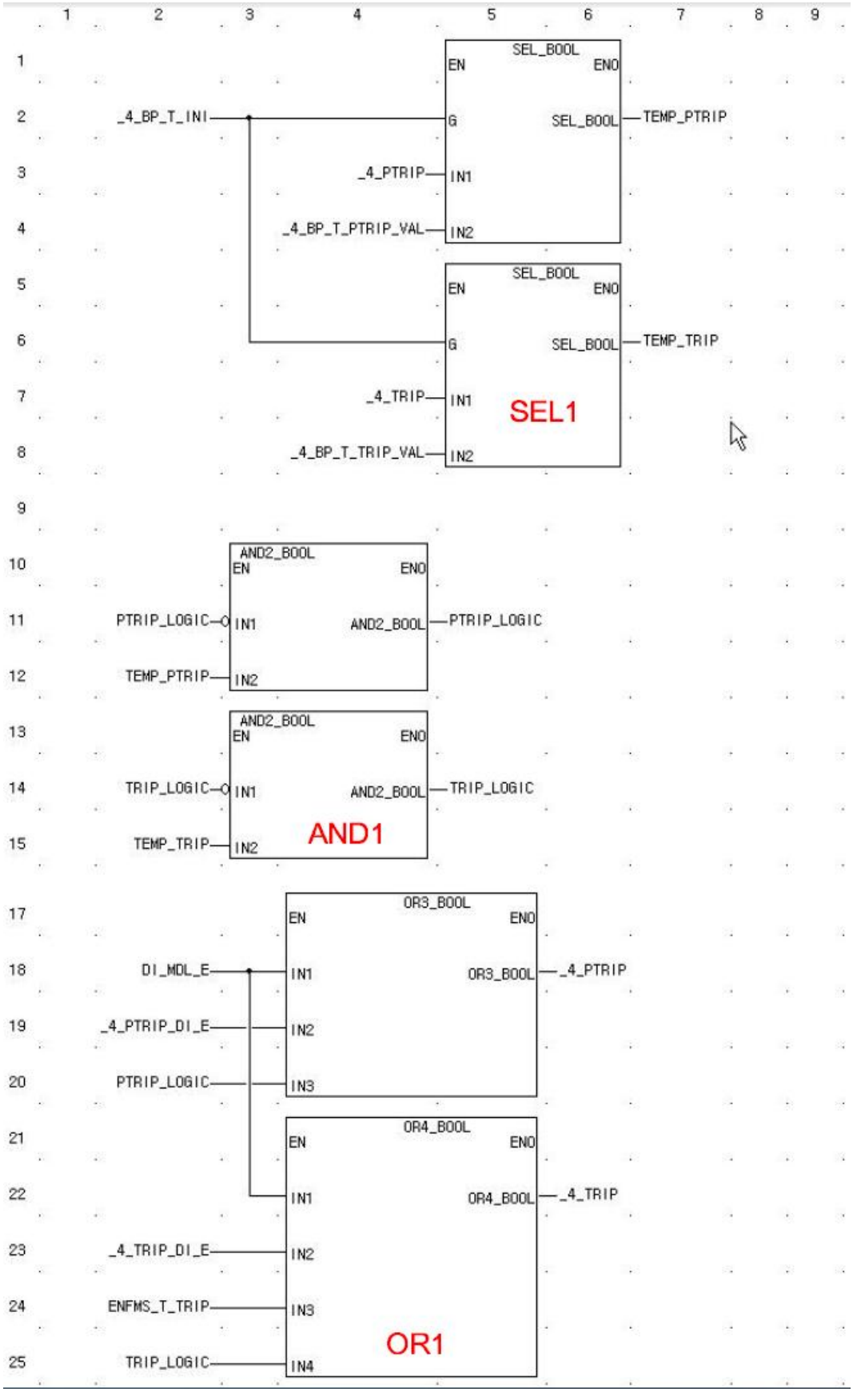
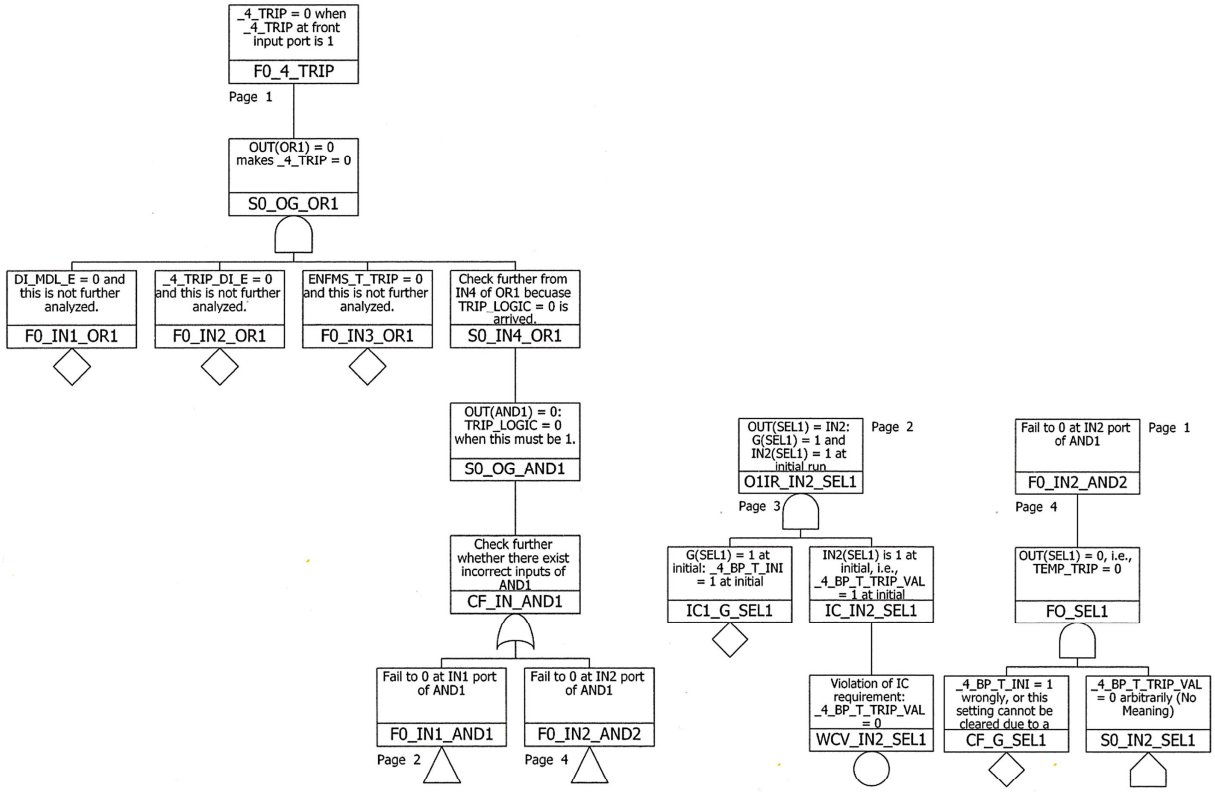


Figure 7. FBD for the module "DNBR_LO trip" [15]



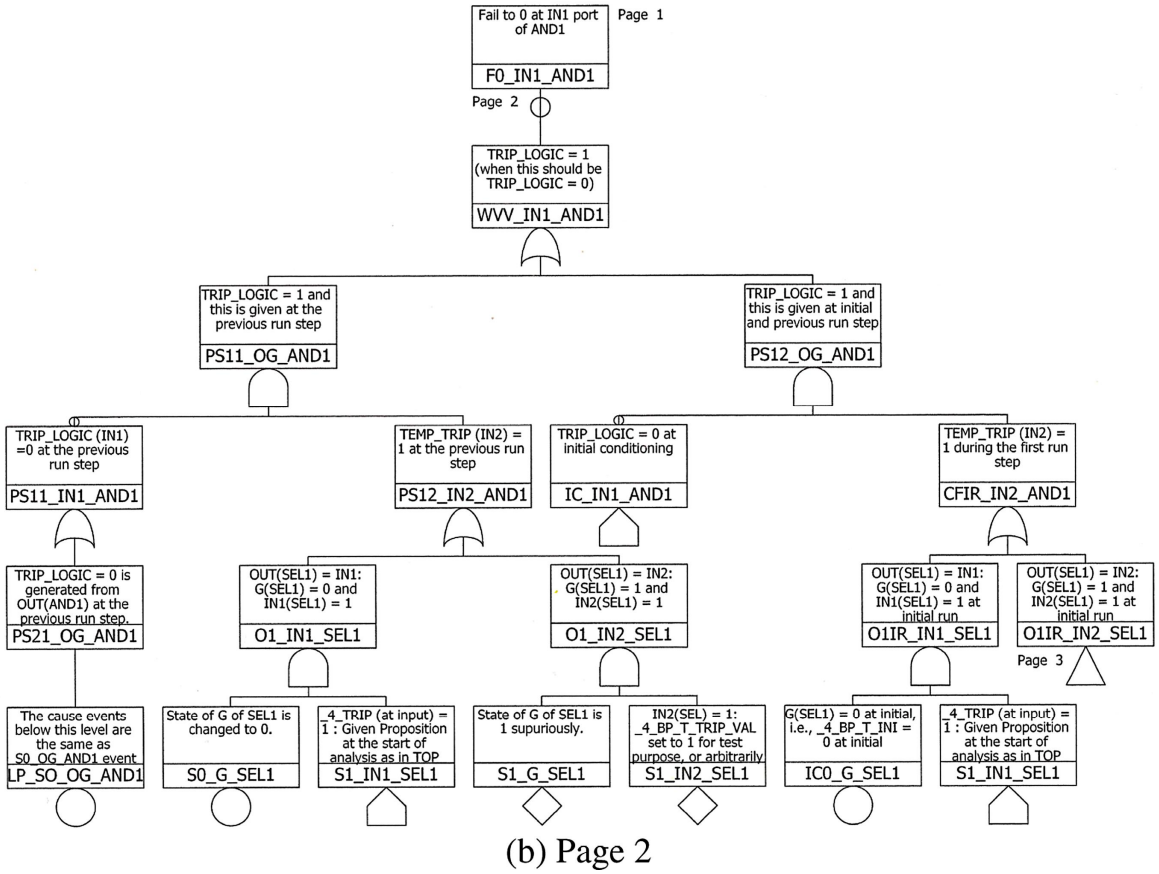


Figure 8. Fault tree for the module "DNBR_LO trip" [15]

Based on the SWFTA, two possible failures were recognised:

1. At the variable TEMP_TRIP: when the value of _4_BP_T_TRIP_VAL at SEL1 is 0, wrong input selection during the real trip operation (selecting _4_BP_T_INI = 0 instead of 1) leads to the top event. This failure was not considered further in this example, since wrong input selection is not in the scope of Figure 8.

2. At the input variable TRIP_LOGIC: the event LP_S0_OG_AND1 (LP means there is a loop within the tree) indicates that the events below this event are the same as with the upper event S0_OG_AND1. This means that the values of TRIP_LOGIC are toggling, which results in the coming and going trip signals 0 and 1.

The error described above was not found by formal verification process where the FBD was converted into a Verilog file and model checking on SMV model checker was performed on the Verilog file. The authors of the study found that HAZOP is capable of finding such a hazard in a small FBD as used here, but FTA is thought to be necessary when identifying a local defect.

3. STAMP and STPA

3.1 Hazards in modern systems

Nancy Leveson claims in her 2004 paper titled "*A new accident model for engineering safer systems*" [11] that new safety engineering approaches are required for modern systems. She states that one of the key factors supporting this claim is increased system complexity and networking of systems. Increasing complexity leads to new types of errors that can't be explained simply by individual component failures or operator errors. Leveson writes: "*Digital technology has created a quiet revolution in most fields of engineering, but system engineering and system safety engineering techniques have not kept pace. Digital systems introduce new ''failure modes" that are changing the nature of accidents. Many of the approaches that worked on electromechanical components—such as replication of components to protect against individual component failure (i.e., redundancy)—are ineffective in controlling accidents that arise from the use of digital systems and software*."

In particular, the use of software has removed some of the physical constraints present in analogical systems and enabled engineers to design systems where complexity is limited merely by the designer's imagination. Analysing the hazards of such systems is often extremely complex. Leveson writes: "*Software is an important factor here: it has allowed us to implement more integrated, multi-loop control in systems containing large numbers of dynamically interacting components where tight coupling allows disruptions or dysfunctional interactions in one part of the system to have far-ranging rippling effects. The problem is that we are attempting to build systems that are beyond our ability to intellectually manage: increased interactive complexity and coupling make it difficult for the designers to consider all the potential system states or for operators to handle all normal and abnormal situations and disturbances safely and effectively*."

In addition to increased complexity in systems, the societal structure surrounding them has also grown more complicated as risks and hazards in complex safety-critical systems can no longer be controlled merely by individuals. For example, controlling the risks of nuclear power plants often involves several regulatory bodies, laws and guidelines. Since the societal structures surrounding systems enforce their requirements on the system design and operation, these structures would also have to be included when analysing system hazards.

In her paper, Leveson criticizes traditional event-based hazard analysis methods. Most notably, she criticizes the inability of these methods to take into account organizational and leadership factors: "*Event-based models are poor at representing systemic accident factors such as structural deficiencies in the organization, management deficiencies, and flaws in the safety culture of the company or industry. An accident model should encourage a broad view of accident mechanisms that expands the investigation from beyond the proximate events*." Overall, event-based models tend to assign blame on failure of a single component or sub system, instead of focusing more on inadequate communication between different system components (such as nuclear regulator and nuclear power plant management).

3.2 STAMP

3.2.1 Background

The solution for the problems discussed above, as proposed by Leveson, is applying systems theory to modelling systems and their hazards. STAMP (System-Theoretic Accident Model and Processes) was first introduced in the 2004 paper [11] by Leveson. Her book titled "*Engineering a Safer World: Systems Thinking Applied to Safety*" [12] provides a comprehensive introduction on STAMP, STPA (a STAMP-based method for hazard analysis) and CAST (a STAMP-based method for accident analysis), as well as the underlying theory.

3.2.2 Fundamental concepts

STAMP includes three fundamental concepts:

1) The hierarchical control structure
2) Safety constraints
3) Process models

The aim of STAMP is to model systems based on systems theory, which means abandoning the models that use linear causality to explain hazards and accidents in systems. Instead, a system is modelled as a **hierarchical control structure** where the upper hierarchical control level controls the level below it, keeping the system in a safe state of equilibrium. For example, new nuclear control regulations from government regulators are communicated through several organizational levels all the way down to nuclear plant operators. Every level in this socio-technical structure has to be considered when overall system safety and hazards are analysed. In STAMP, systems consist of components (controllers, actuators, sensors and controlled processes), control actions, and feedback loops. Figure 9 shows an example of a typical hierarchical control structure.

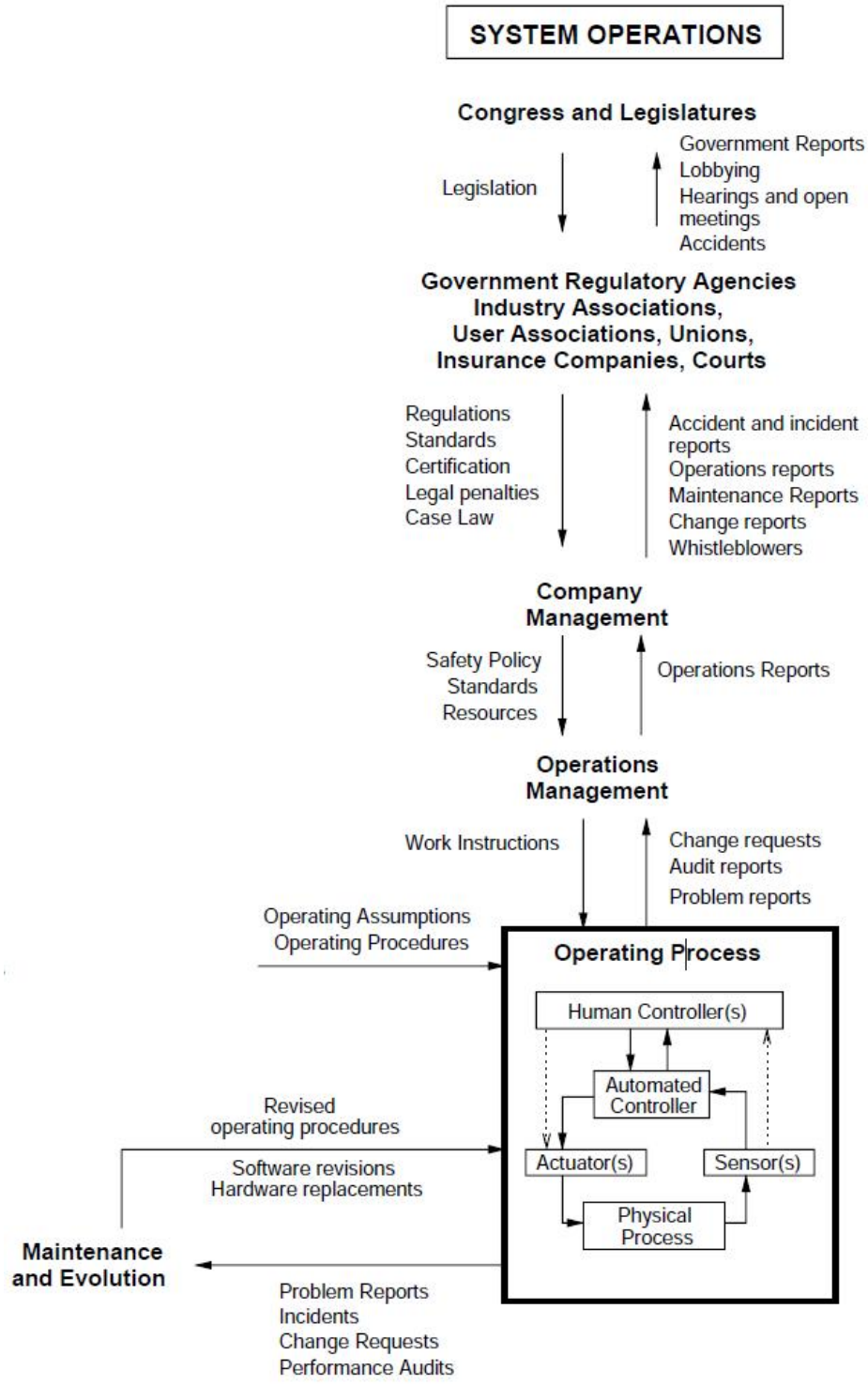


Figure 9. General hierarchical control structure (adapted from [12]*).*

Safety constraint is another fundamental concept in STAMP. A system has specific constraints which shall not be violated, or otherwise the system may be transformed into a hazardous state where an accident may happen. For example, a constraint for a cruise control system would be to ensure that the vehicle speed stays within certain values. If the constraint isn't properly controlled, vehicle speed may grow too high and a hazardous state, which might lead to an accident, is reached. Controllers try to ensure that constraints are not

violated by issuing control actions (downward arrows in Figure 9) and receiving feedback (upward arrows in Figure 9).

The third fundamental concept in STAMP is the **process model**. Having a model of the controlled process, at least in some detail, is a prerequisite for each controller in order for them to make correct control decisions. **Process model variables** represent the current state of the model (e.g. process model variables of a cruise controller are, at least, vehicle speed, acceleration, throttle level, and a variable indicating whether the cruise control system is on). Model of the process being controlled is required not just at the lower physical levels of the hierarchical control structure, but at all levels [12]. The concept should not be limited to the process models programmed into electrical controller devices; human mental models have to be considered process models as well. For example, the plant safety manager has to have some kind of a mental model of the plant's current safety situation in order for him to make correct decisions on revising safety principles.

3.2.3 The nature of accidents according to STAMP

The implication within STAMP is that accidents happen not due to a system component failure, but due to a component failing to enforce its constraint. Safety constraints can be violated in two ways:

1) The controller provides an unsafe control action. Four types of unsafe control actions are possible:

    a. A control action necessary for safety is not provided

    b. An unsafe control action is provided, which leads to hazard

    c. A control action necessary for safety is provided too late, too soon, or out of sequence

    d. A control action necessary for safety is stopped too soon or applied too long

2) Appropriate control actions were provided but not followed.

These same general factors apply at each level of the sociotechnical control structure, but the interpretation (application) of the factor at each level may differ.

Classification of accident causal factors starts by examining each of the basic components and determining how their improper operation may contribute to the general types of inadequate control [12]. This classification is shown in Figure 10.

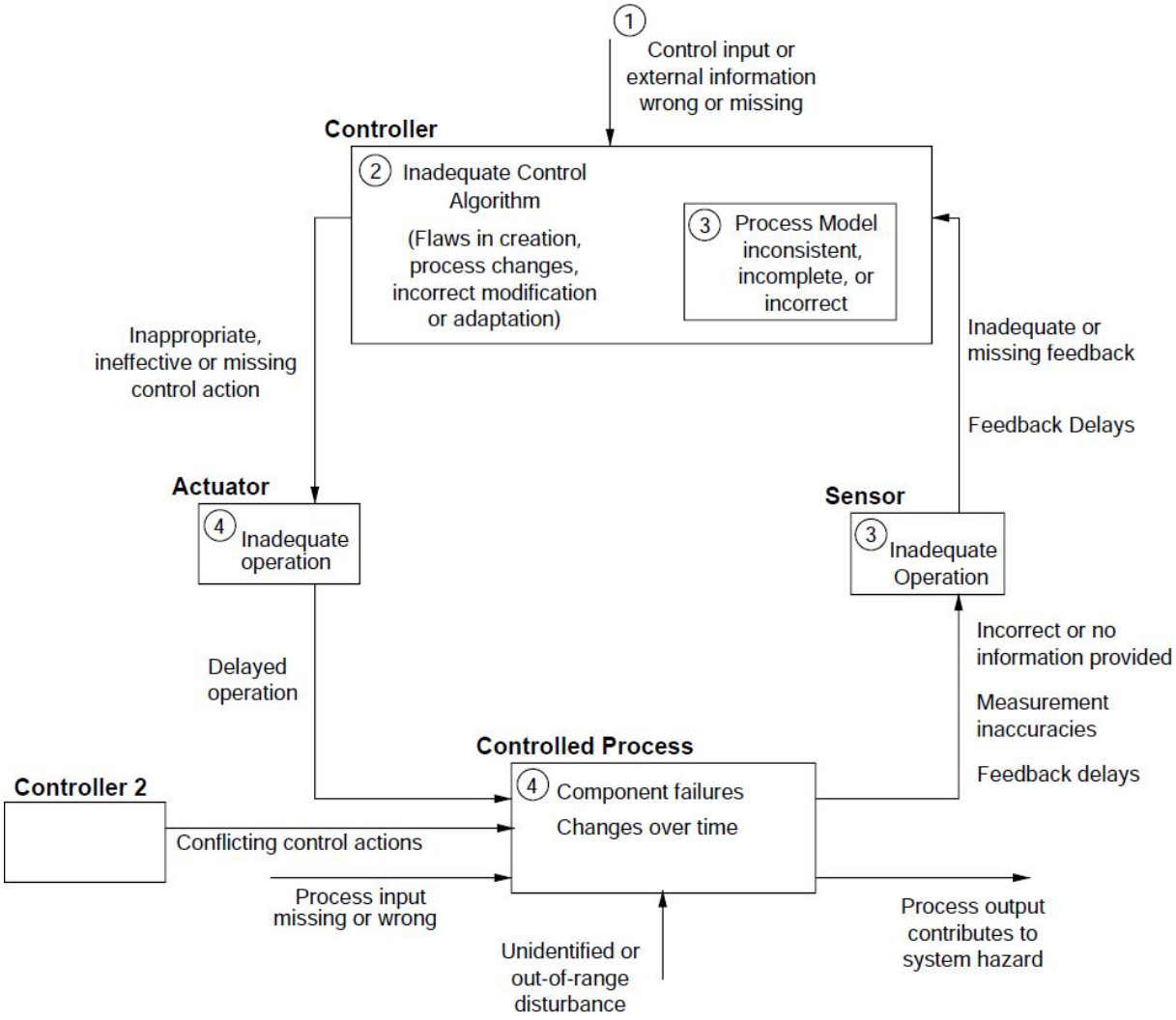


Figure 10. Classification of control flaws [12]

Control flaws related to the controller can be caused by three different factors. In Figure 10, these are numbered 1 – 3. Number 4 indicates flaws related to the actuator or controlled process.

1. Unsafe inputs

   Each controller in the hierarchical control structure is itself controlled by higher-level controllers. The control actions and other information provided by the higher level and required for safe behaviour may be missing or wrong. [12] This might lead to the controller not issuing the correct control action.

2. Unsafe control algorithm

   Algorithms in this sense are both the procedures designed by engineers for hardware controllers and the procedures that human controllers use. Control algorithms may not enforce safety constraints because the algorithms are inadequately designed originally, the process may change and the algorithms become unsafe, or the control algorithms may be inadequately modified by maintainers if the algorithms are automated or through various types of natural adaptation if they are implemented by humans. [12]

3. Inconsistent, incomplete, or incorrect process models

   Effective control is based on a model of the process state. Accidents, particularly component interaction accidents most often result from inconsistencies between the models of the process used by the controllers (both human and automated) and the actual process state. When the controller's model of the process (either the human mental model or the software or hardware model) diverges from the process state, erroneous control commands (based on the incorrect model) can lead to an accident. [12]

   The process model designed into the system may be wrong from the beginning, there may be missing or incorrect feedback for updating the process model as the controlled process changes state, the process model may be updated incorrectly, or time lags may not be accounted for. The result can be uncontrolled disturbances, unhandled process states, inadvertent commanding of the system into a hazardous state, unhandled or incorrectly handled process component failures, and so forth. [12]

4. Actuators and controlled processes

   The factors discussed so far have involved inadequate control. The other case occurs when the control commands maintain the safety constraints, but the controlled process may not implement these commands. One reason might be a failure or flaw in the reference channel, that is, in the transmission of control commands. Another reason might be an actuator or controlled component fault or failure. A third is that the safety of the controlled process may depend on inputs from other system components, such as power, for the execution of the control actions provided. If these process inputs are missing or inadequate in some way, the controller may be unable to execute the control commands and accidents may result. Finally, there may be external disturbances that are not handled by the controller. [12]

3.2.4 Developing a STAMP model

Any practical use of STAMP, such as its use for analysing accidents or STPA hazard analysis, begins with creating a STAMP model of the system being analysed. This includes four parts: identifying possible accidents, identifying system hazards, identifying system constraints, and creating the system safety control structure. STAMP is intended to be used in a top-down fashion. Modelling should start at a general level and proceed into more detailed subsystem and component models as required.

1. Identifying accidents

   STAMP uses the following definition of accident: "An undesired or unplanned event that results in a loss of human life or human injury, property damage, environmental pollution, mission loss, etc." In general, the definition of an accident in a given system comes from the customer and occasionally from the government for systems that are regulated by governmental agencies. Other sources might be user groups, insurance companies, professional societies, industry standards, and other stakeholders. [12]

   Accidents are not limited to events causing human damage. For example, failure of a manufacturing robot to fulfil its task (such as moving an item from place A to B) can be considered an accident even though no damage to humans occurs. Of course, another accident of this system would be the robot arm colliding with a person and resulting in human damage. In some cases, it might be useful to divide accidents into different levels based on their severity. In the robot example, accidents involving human damage would be in the top level followed by other accidents and levels.

2. Identifying hazards

   Within STAMP, hazard is defined as “a system state or set of conditions that, together with a particular set of worst-case environmental conditions, will lead to an accident (loss)” [12]. For example, a hazard for the robot system described above would be “A person enters the robot’s working area”, which might lead to an accident. In STAMP, hazards should be identified in a top-down fashion; they should be first identified in the system level. If hazards cannot be eliminated in the system level, they must be mapped into subsystem or component levels [12].

3. Identifying constraints

   After identifying hazards, system constraints that prevent the hazards from occurring should be identified. This is an important phase, since the results will be used later on when analysing potential failure mechanisms. An example constraint for the robot system would be “A person shall not be able to enter the robot’s working area when the robot is in operation”.

4. Creating the safety control structure

   It should be noted that STAMP may be used both for existing systems and in the design phase of new systems being developed. In the former situation, the control structure is already known and should be modelled. In the latter situation, the safety control structure is designed based on constraints that act as input to the design process. The case example in section 3.4 involves using STAMP for an existing system, while section 3.5 gives an introduction about the use of STAMP for developing a new system.

3.3 STPA hazard analysis method

System-Theoretic Process Analysis (STPA) is a hazard analysis method based on STAMP. Its goal is to discover causalities about how the system safety constraints can be violated. STPA can also be used either in the design phase of a new system or for an already-existing system. It has two main steps [12]:

1. Identifying unsafe control actions

   Identify the potential for inadequate control of the system that could lead to a hazardous state. Hazardous states result from inadequate control or enforcement of the safety constraints, which can occur because:

   - A control action required for safety is not provided or not followed.
   - An unsafe control action is provided
   - A potentially safe control action is provided too early or too late, that is, at the wrong time or in the wrong sequence.
   - A control action required for safety is stopped too soon or applied too long.

   In practice this step should be done by selecting a control action and creating a context table. Table 8 shows an example of a context table for a control action that opens a train door. The first column shows which control action is being analysed. The next three columns show the *process model variables* which together represent the *context* of the process (i.e. is the train moving, has the alarm been activated and

is what is the train position). The last three columns indicate whether the control action is hazardous or not in a given context.

Table 8. Example context table for opening a train door (adapted from [20])

Control action (CA)	Train moving?	Alarm?	Train position?	**Providing CA hazardous ?**	**Providing CA too early hazardous?**	**Providing CA too late hazardous?**
Open door	Yes	No	Doesn't matter	Yes	Yes	Yes
	Yes	Yes	Doesn't matter	Yes	Yes	Yes
	No	Yes	Doesn't matter	No	No	No
	No	No	Not aligned with platform	Yes	Yes	Yes
	No	No	Aligned with platform	No	No	No

A similar context table should be created for the case where CA is *not* provided. This way each of the control actions are systematically analysed, and as a result unsafe control actions are identified.

2. Identifying causal factors

   Determine how each potentially hazardous control action identified in step 1 could occur.

   - For each unsafe control action, examine the parts of the control loop to see if they could cause it. Design controls and mitigation measures if they do not already exist or evaluate existing measures if the analysis is being performed on an existing design. For multiple controllers of the same component or safety constraint, identify conflicts and potential coordination problems.

   - Consider how the designed controls could degrade over time and build in protection, including:

     - Management of change procedures to ensure safety constraints are enforced in planned changes.

     - Performance audits where the assumptions underlying the hazard analysis are the preconditions for the operational audits and controls so that unplanned changes that violate the safety constraints can be detected.

- o Accident and incident analysis to trace anomalies to the hazards and to the system design.

When evaluating each potentially hazardous control action, general types of control flaws presented in Figure 10 can be used as “guidewords”, much like guidewords are used in HAZOP.

The analysis process is introduced with a case example in the next section.

3.4 Applying STPA – Case EPR MSIV

3.4.1 Overview

Thomas et al. [20] conducted a case study of applying STPA to a part of an EPR (Evolutionary Power Reactor) I&C system. EPR is a type of PWR (pressurised water reactor) where the primary cooling system heats the water in the secondary cooling system through the SG (steam generator). Water in the SG evaporates into steam, operates a turbine and is condensed back to water before the cycle starts from the beginning. The SG prevents radioactive water from the primary system from mixing with the non-radioactive water in the secondary system. The PWR process is demonstrated in Figure 11.

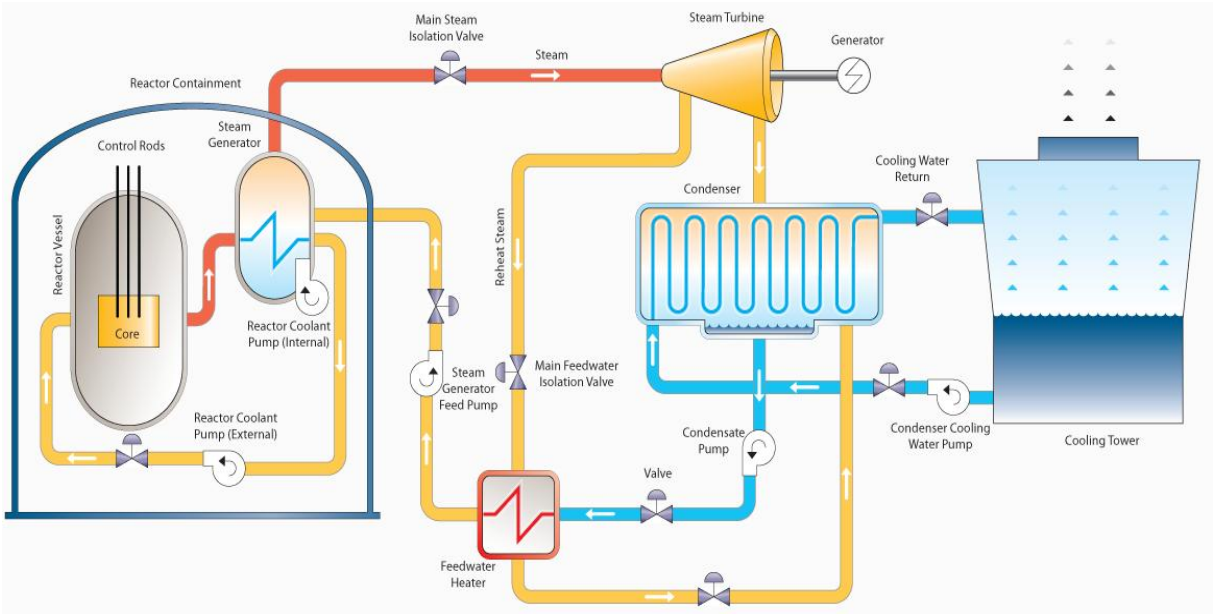


Figure 11. Pressurised Water Reactor process [4]

The system that STPA was applied to is responsible for closing the EPR MSIV (main steam isolation valve). The MSIV is located in the main steam line from the SG, and it is kept open in normal operational conditions. If a certain abnormality (e.g. a break in the main feedwater pipe to the SG) is detected, the MSIV may be closed to isolate the SG from the rest of the secondary cooling system. Closing the MSIV prevents the secondary cooling system from cooling the primary system. Thus, backup cooling systems have been built to provide adequate cooling in such situations. These systems include redundant SGs, turbine bypass valves, main steam relief isolation valves and main steam relief control valves, safety relief valves, the CVCS (chemical volume control system), and the ECCS (emergency core cooling system).

The analysis in the example begins by performing the STAMP process introduced in section 3.2 for the overall system. This means identifying accidents and hazards and creating a high-

level control structure for the system. Then the STPA process described in section 3.3 is performed specifically for the system that is responsible for closing the MSIV. The process includes systematic identification of unsafe control actions and causal factors. In the end, results of the analysis are summarised. The process is explained step by step in the following sub sections.

3.4.2 Accidents and hazards

Firstly, system-level accidents are considered. Accidents include not only events where human lives are lost but all other unacceptable events as well. These are summarised in Table 9.

Table 9. System-level accidents (adapted from [20])

Accident	Explanation
A-1	People injured or killed
A-2	Environment contaminated
A-3	Equipment damage
A-4	Loss of electrical power generation

Based on the accidents, system-level hazards can be identified. Hazards are summarised in Table 10 along with the corresponding accidents.

Table 10. System-level hazards (adapted from [20])

Hazard	Explanation	Related accident
H-1	Release of radioactive materials	A-1, A-2
H-2	Reactor temperature too high	A-1, A-2, A-3, A-4
H-3	Equipment operated beyond limits	A-3, A-4
H-4	Reactor shut down	A-4

H-1 refers to release of radioactive materials anywhere outside the primary system, regardless of quantity. It should be prevented to prevent exposure to people or the environment (A-1 and A-2). H-2 (reactor temperature too high) is a dangerous condition that can cause every system-level accident e.g. by melting of the fuel rods. H-3 (equipment operated beyond limits) includes operation beyond safe limits that causes reactor damage or damage to other equipment. H-4 (reactor shut down) includes any unplanned shutdown that results in a loss of electrical power generation.

3.4.3 Safety control structure

The system-level safety control structure is shown in Figure 12. The components inside the red box control the closing of the MSIV and they are analysed further with the STPA method. Figure 13 shows a more detailed safety control structure for the system highlighted in the red box.

The green arrow represents the communication between the MSIV controllers and other controllers. There are four controllers that may provide a control action to close the MSIV: the Operator, the NSSC (non-safety system controller), the PS (protection system), and the DAS (diverse automation system). These controllers send control actions to the MSIV PM (priority module), which uses a pre-programmed priority setting to determine which control actions to forward to the MSIV actuator.

If the operator detects a need to close the MSIV, he may issue a *Close MSIV* command to the PM. The PM determines which controller is in charge according to the priority setting, and forwards commands directly to the MSIV actuator. The operator may also send a *Close MSIV* command to the NSSC, which provides manual control for the MSIV. In this situation, the NSSC would normally forward the command from the operator to the PM, which would then forward the command to the MSIV actuator.

The PS is an automated system that can automatically detect some situations in which a *Close MSIV* command is necessary. In these situations the PS can provide the *Close MSIV* command to the PM which can forward the command to the MSIV actuator.

The DAS is a backup protection system that is used if there is a problem with the PS. The DAS can issue a *Close MSIV* command to the PM, which would normally forward the command to the MSIV actuator.

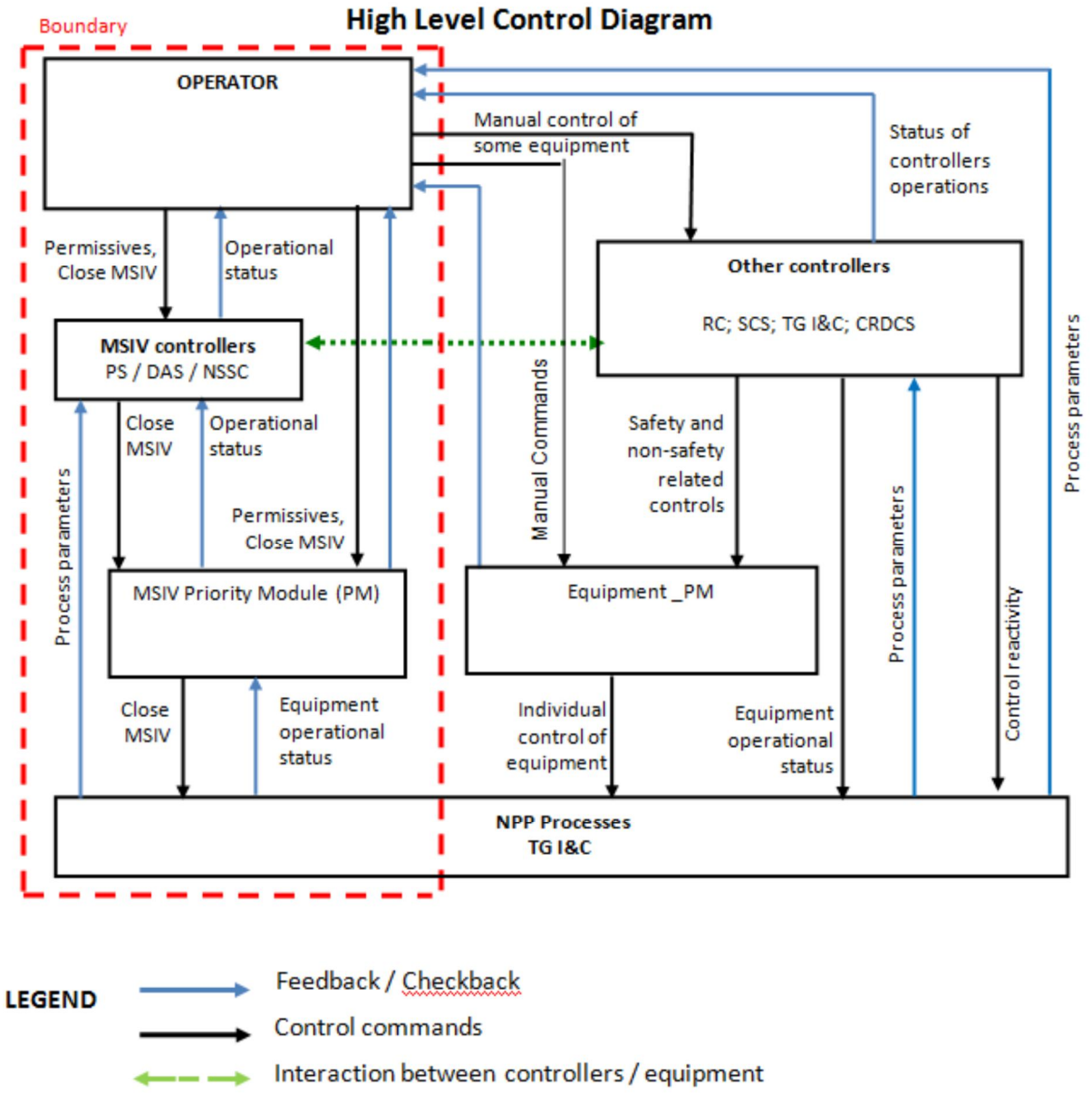


Figure 12. System-level PWR safety control structure [20]

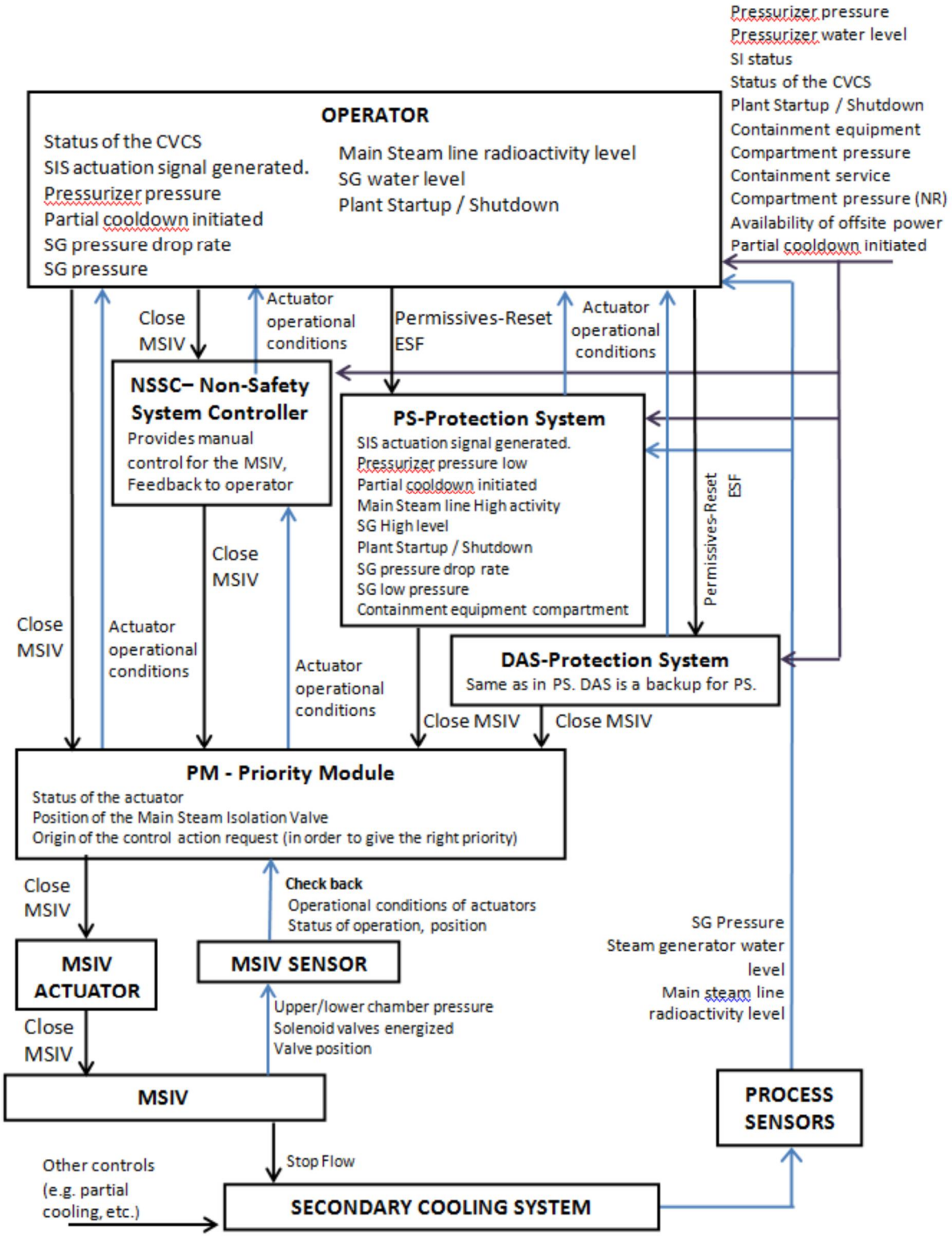


Figure 13. Safety control structure for the system responsible for MSIV operation [20]

A sensor provides feedback about the MSIV status directly to the PM. This sensor does not sense process variables such as pressure, temperature, or steam flux. Instead, it senses torque applied to the valve itself to detect if the valve has closed. The PM receives this feedback and can provide confirmation back to the controller that originally requested the MSIV closure.

Other process sensors report process variables to the controllers including various pressures, SG water level, and the operation of other backup systems. This information is used by the controllers to determine, among other things, whether the MSIV should be closed.

The controllers have responsibilities as follows:

Operator:

- Validate/inhibit permissives
- Bring the plant to a controlled shutdown in case of Anticipated Operational Occurrence (AOO) or Postulated Accidents (PA), such as leakage from primary into the secondary loop.
- Activate the safety engineering features (ESF)
- Start main steam line isolation when necessary
- Monitor parameters and look for abnormalities or trends (fault diagnostic)
- Operate the plant during startup
- Operate the plant during programmed shutdown
- Take actions in accordance to written guides upon any transient or emergency

PS - Protection System:

- Bring the plant to a controlled shutdown in case of Anticipated Operational Occurrence (AOO) or Postulated Accidents (PA), such as leakage from primary into the secondary loop.
- Activate the safety engineering features (ESF)
- Start main steam line isolation when necessary

DAS - Diverse Automation System

- Same as PS. DAS is a backup for PS.

NSSC - Non-Safety System Controller

- If an operator command to open/close MSIV is received, then send that command to PM
- If feedback is received from PM, then send that feedback to Operator.

PM - Priority Module

- Give access to control commands according to following priority: PS > DAS > SCS (safety control system) > Operator > NSSC
- Forward commands to MSIV actuator
- Forward feedback from MSIV actuator to the active controller
- Ensure that checkback is received when MSIV is closed (indicating that valve torque has reached its maximum)
- Check for any problems with MSIV actuator operability

3.4.4 Process model variables

The process model variables capture the information needed by each controller to decide what control action to provide. Different process model variables may be associated with each control action. The high-level process model variables associated with MSIV closure can be identified by considering the purpose of the MSIV. The MSIV remains open during normal plant operation and is only needed to control a few specific abnormal conditions. The

relevant high-level conditions can be derived from the system hazards and system description as follows:

- Steam generator tube rupture, which can cause an uncontrolled SG level increase and can release contaminated fluid into the secondary system
- Steam system piping leak, which can depressurize the SG and cause an overcooling transient and energy release into containment
- Feedwater system piping leak, which can depressurize the SG and cause an overcooling transient and energy release into containment

While these conditions could be caused by physical failures, the latter two could also be caused by design flaws or unsafe commands elsewhere in the system. For example, a leak in the main steam line could be caused by a physical failure (e.g. rupture in the line) or it could be caused by main steam relief valves that are opened inadvertently or at the wrong time. Both situations could require MSIV closure to prevent depressurization and an overcooling transient while the issue is investigated and resolved.

In addition to helping to mitigate the conditions above, the MSIV also controls the heat exchange that takes place within the SG. Before the SG is closed, other support systems may need to be engaged to provide adequate cooling. Therefore, information about additional cooling provided by other support systems (i.e. inadequate, adequate) may be needed for the decision to close the MSIV and should be included in the process model.

3.4.5 Unsafe control actions

When considering whether a potential control action is hazardous or not, it is important to avoid assuming that other defence barriers are intact or that they are appropriate, sufficient, and error-free. For example, even if there is an emergency feedwater system to provide the necessary cooling in the event of a relief valve inadvertently commanded open, it is still hazardous to inadvertently command the relief valve open. These hazardous actions must be included in the analysis and prevented regardless of other protective systems intended to mitigate unsafe behaviour. Table 11 summarizes the unsafe control actions identified for the command *Close MSIV*.

Table 11. Unsafe control actions for the Close MSIV command (adapted from [20])

Control action	Unsafe Control Actions			
	Not providing causes hazard	Providing causes hazard	Wrong timing or order causes hazard	Stopped too soon or applied too long
Close MSIV	Close MSIV not provided when there is a rupture in the SG tube, leak in main feedwater, or leak in main steam line [H-2, H-1, H-3]	Close MSIV provided when there is no rupture or leak [H-4]	Close MSIV provided too early (while SG pressure is high): SG pressure may rise, trigger relief valve, abrupt steam expansion [H-2, H-3]	N/A

		Close MSIV provided when there is a rupture or leak while other support systems are inadequate [H-1, H-2, H-3]	Close MSIV provided too late after SGTR (steam generator tube rupture): contaminated coolant released into secondary loop, loss of primary coolant through secondary system [H-1, H-2, H-3]	
			Close MSIV provided too late after main feedwater or main steam line leak [H-1, H-2, H-3, H-4]	

After identifying the unsafe control actions, a context table was constructed for the control action using the corresponding process model variables that were defined in the previous section. Table 12 shows the context table for *Close MSIV* provided by operator.

Table 12. Context table for the "Operator provides Close MSIV" control action (adapted from [20])

1	2	3	4	5	6	7	8	9
Control action	Steam generator tube	Condition of main feedwater pipe	Condition of main steamline	Operation of other support systems	Control action hazardous?	Control action hazardous if too late?	Control action hazardous if too early?	Not providing control action hazardous?
Close MSIV	Not ruptured	No leak	No leak	Adequate	H-4	H-4	H-4	No
	Ruptured	No leak	No leak	Adequate	No	H-1, H-2, H-3, H-4	H-3, H-4	H-1, H-2, H-3, H-4
	Not ruptured	Leak	No leak	Adequate	No	H-2, H-3, H-4	No	H-2, H-3
	Not ruptured	No leak	Leak	Adequate	No	H-2, H-3, H-4	No	H-2, H-3
	Ruptured	Leak	No leak	Adequate	No	H-1, H-2, H-3, H-4	H-3, H-4	H-1, H-2, H-3, H-4
	Not	Leak	Leak	Adequate	No	H-2, H-3,	No	H-2, H-3

	ruptured					H-4		
	Ruptured	No leak	Leak	Adequate	No	H-1, H-2, H-3, H-4	H-3, H-4	H-1, H-2, H-3, H-4
	Ruptured	Leak	Leak	Adequate	No	H-1, H-2, H-3, H-4	H-3, H-4	H-1, H-2, H-3, H-4
	Not ruptured	No leak	No leak	Inadequate	H-2, H-4	H-2, H-4	H-2, H-4	No
	Ruptured	No leak	No leak	Inadequate	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4
	Not ruptured	Leak	No leak	Inadequate	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-2, H-3
	Not ruptured	No leak	Leak	Inadequate	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-2, H-3
	Ruptured	Leak	No leak	Inadequate	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4
	Not ruptured	Leak	Leak	Inadequate	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-2, H-3
	Ruptured	No leak	Leak	Inadequate	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4
	Ruptured	Leak	Leak	Inadequate	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4	H-1, H-2, H-3, H-4

Column 1 in Table 12 is the control action being analysed while columns 2 to 5 correspond to the process model variables identified earlier. Column 6 specifies in which contexts it is hazardous to provide the Close MSIV control action. For example, row 1 describes a situation in which it is hazardous to close the MSIV: if there is no SG tube rupture, no main feedwater pipe leak, and no main steam line leak, then there is no need to close the MSIV. Closing the MSIV will cause H-4 (reactor shut down). If the operation of other support systems cannot make up for the additional heat exchange required, closing the MSIV will also lead to a loss of necessary cooling (H-2 in row 9 column 6).

In the case of SG tube rupture, keeping the MSIV open can cause not only equipment damage but also a more immediate shutdown (H-4) via SCRAM and can increase the amount of time the plant will need to remain shut down for repairs. The overfilling of the SG could allow water to enter the steam lines, damaging the delicate turbine pallets and requiring extensive time for repairs. In addition to actual damage, equipment can be overstressed and require more detailed inspections before the plant can be operational again. The additional contamination will also require more time to decontaminate and will result in the generation of more waste. Because keeping the MSIV open during a SG tube rupture will cause a more severe and prolonged shutdown than would otherwise occur with a contained SG tube rupture, H-4 is included in Table 12 for these cases. H-4 is not listed for other cases because it is assumed that keeping the MSIV open after a leak in the main steam line or main feedwater pipe will not cause a more severe or prolonged shutdown than if the MSIV is closed, although it does contribute to the other hazards listed.

When inspecting rows 10 to 16 in Table 12 more closely, it can be noticed that in these cases both providing the control action and not providing the control action are marked as hazardous. In other words, in these situations it is hazardous to close the MSIV yet hazardous to keep the MSIV open. In some cases, it is possible to revisit the design to eliminate the conflict and provide a safe option. If the conflict cannot be resolved, a decision must be made about what action should be taken in these contexts, that is, which is the least hazardous.

3.4.6 Constraints and causal factors

Each of the unsafe control actions from Table 12 can be translated into safety constraints as shown in Table 13.

Table 13. Safety constraints [20]

Unsafe Control Action	Safety Constraint
UCA 1: Close MSIV not provided when there is a leak (rupture in the SG tube, leak in main feedwater, or leak in main steam line) and the support systems are adequate	SC 1: MSIV must be closed when there is a leak (rupture in the SG tube, leak in main feedwater, or leak in main steam line) and the support systems are adequate
UCA 2: Close MSIV not provided when there is a main feedwater or main steam line leak and other support systems are inadequate	SC 2: MSIV must be closed when there is a main feedwater or main steam line leak and other support systems are inadequate
UCA 3: Close MSIV provided when there is a SGTR but support systems are inadequate	SC 3: MSIV must not be closed when there is a SGTR and support systems are inadequate
UCA 4: Close MSIV provided too early (while SG pressure is high)	SC 4: MSIV must not be closed too early while SG pressure is too high
UCA 5: Close MSIV provided too late after rupture/leak (in the SG tube, main feedwater, or main steam line)	SC 5: MSIV must not be closed too late after rupture/leak (in the SG tube, main feedwater, or main steam line)
UCA 6: Close MSIV provided when there is no rupture/leak	SC 6: MSIV must not be closed when there is no rupture/leak

As mentioned in section 3.2.3, there are two ways that a safety constraint can be violated:

1) An unsafe control action is provided

2) Appropriate control actions were provided but not followed

In the following, causal factors for case 1 are analysed for the Operator. A similar analysis can also be done for NSSC, DAS and PS, which, as described in 3.4.3, are the other controllers that may issue control actions for the MSIV system. The causal factors related to operator unsafe control actions are numbered 1 to 4 and shown in Figure 14, and they are used in the analysis.

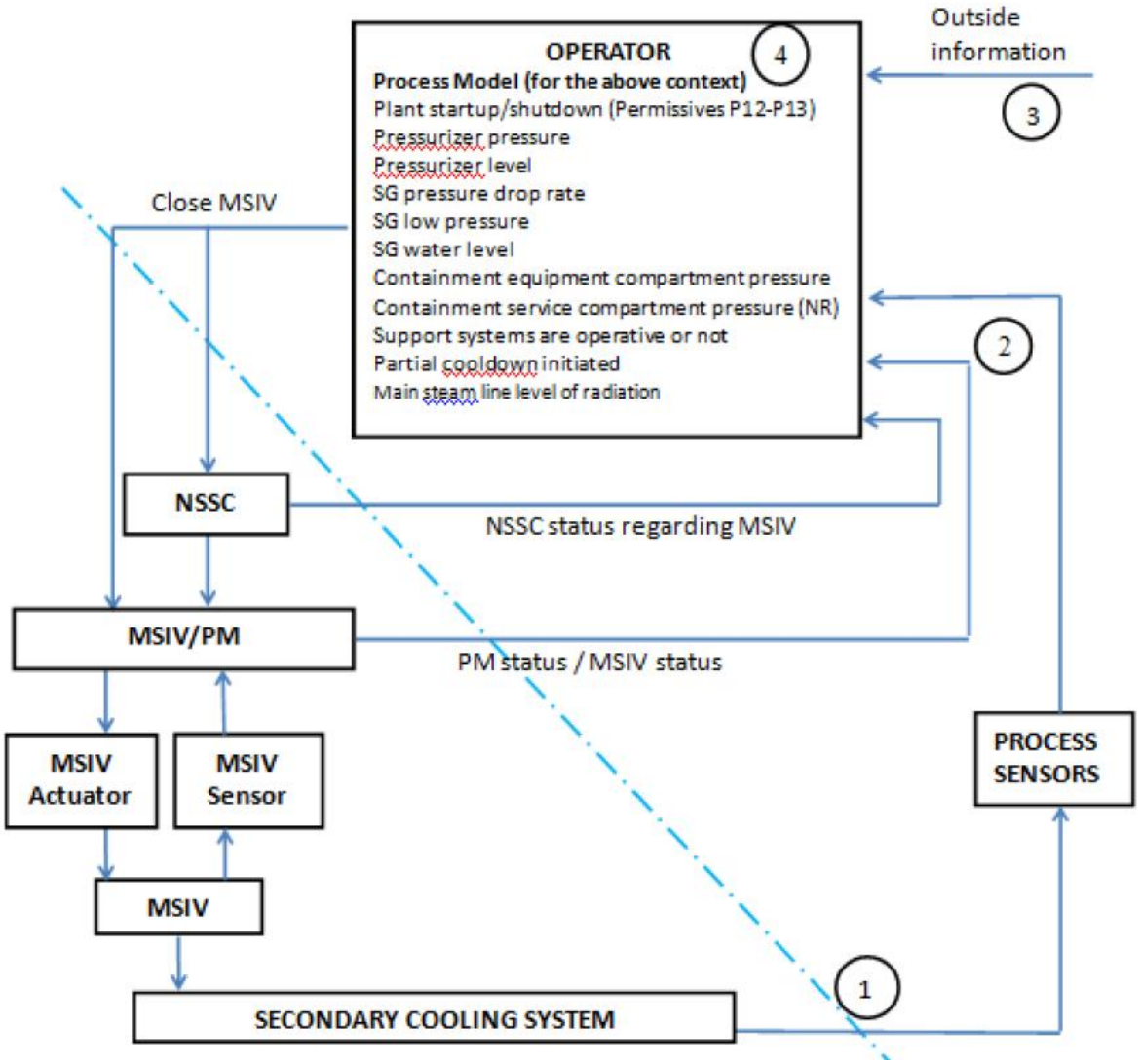


Figure 14. Causal factors leading to operator unsafe control actions [20]

UCA 1: Close MSIV command not provided when there is a leak (rupture in the SG tube, link in main feedwater, or leak in main steam line) and the support systems are adequate.

(1) Secondary cooling system (CVCS or emergency feedwater system)
  (a) Concurrent situation masks another. For example, a feedwater problem could happen concurrent with a SGTR, causing the SG water level to stay practically stable.
  (b) Situation that requires MSIV closure is masked. For example, NSSC engages PZR (pressurizer) heaters to make up for loss of RCS pressure during SGTR.
  (c) Event progresses too slowly to detect

(2) Process Feedback
  (a) SG level feedback missing, delayed, or incorrect
  (b) SG Pressure, or setpoints, is not correct or delayed
  (c) Steam generator water level delayed or incorrect
  (d) Main steam line activity not correctly indicated
  (e) Conflicting data indicating a false situation

(f) Voting system does not operate properly and gives wrong measures
(g) No indication of partial cool down initiated
(h) Failures in sensors, communication lines, or power
(i) PM reports both MSIV actuators as inoperable when they are
(j) PM reports MSIV already closed, when it is not
(k) NSSC reported as operational (or no feedback provided) when it is not

(3) Outside information
(a) PZR pressure delayed or missing
(b) PZR level incorrectly indicated as normal
(c) No indication of SI initiated
(d) Delayed indication of SI initiated
(e) Inappropriate permissives in effect5
(f) Wrong combination of indicators from the 4 divisions

(4) Operator
(a) Operator believes Steam Generator is not ruptured when it is ruptured
(b) Operator believes the main steam line has no leak when it has a leak
(c) Operator believes the main feedwater has no leak when it has a leak
(d) Operator confused about the procedure to be followed
(e) Operator confused because of conflicting indicators5
(f) Operator reluctant to shutdown the reactor, unsure if shutdown is necessary or warranted
(g) Operator under pressure not to trip reactor
(h) Operator waits for the PS to handle the situation (e.g. Operator recognizes possible SGTR but believes PS will handle it)
(i) Operator is not aware of the problem due to inadequate feedback (e.g. screen is frozen)
(j) Operator is not aware because NSSC is inoperative or providing inadequate information
(k) Operator closes the wrong valve
(l) Operator recognises the rupture/leak but believes other support systems are inadequate, and keeps MSIV open to maintain sufficient cooling capability.
(m) Operator uncertain whether a rupture/leak exists (there is a conflict between being conservative under uncertainty versus immediate manual spurious shutdown which costs money and may be discouraged. May also prefer to wait for the automated system to resolve the problem versus intervening under uncertainty)
(n) Operator believes NSSC is operational when it is not (could cause operator to provide command to an inoperative or disabled NSSC instead of directly to PM)

UCA 2: Close MSIV command not provided when there is a main feedwater or main steam line leak and other support systems are inadequate.

(1) Secondary cooling system (CVCS or emergency feedwater system)
(a) Concurrent situation masks another. For example, a feedwater problem could happen concurrent with a SGTR, causing the SG water level to stay practically stable.
(b) Situation that requires MSIV closure is masked.
(c) Event progresses too slowly to detect

(2) Process Feedback

(a) SG level feedback missing, delayed, or incorrect
(b) SG Pressure, or setpoints, is not correct or delayed
(c) Steam generator water level delayed or incorrect
(d) Conflicting data indicating a false situation
(e) Voting system does not operate properly and gives wrong measures
(f) No indication of partial cool down initiated
(g) Failures in sensors, communication lines, or power
(h) PM reports both MSIV actuators as inoperable when they are
(i) PM reports MSIV already closed, when it is not
(j) NSSC reported as operational (or no feedback provided) when it is not

(3) Outside information
(a) PZR pressure delayed or missing
(b) PZR level incorrectly indicated as normal
(c) No indication of SI initiated
(d) Delayed indication of SI initiated
(e) Inappropriate permissives in effect6
(f) Wrong combination of indicators from the 4 divisions

(4) Operator
(a) Operator believes the main steam line has no leak when it has a leak
(b) Operator believes the main feedwater has no leak when it has a leak
(c) Operator believes there is an SGTR that does not require MSIV closure when there is actually a main steam line or main feedwater leak that does require MSIV closure
(d) Operator confused about the procedure to be followed
(e) Operator confused because of conflicting indicators
(f) Operator reluctant to shutdown the reactor, unsure if shutdown is necessary or warranted
(g) Operator under pressure not to trip reactor
(h) Operator waits for the PS to handle the situation (e.g. Operator recognizes possible leak but believes PS will handle it)
(i) Operator is not aware of the problem due to inadequate feedback (e.g. screen is frozen)
(j) Operator is not aware because NSSC is inoperative or providing inadequate information
(k) Operator closes the wrong valve
(l) Operator recognizes the rupture/leak but because other support systems are inadequate, keeps MSIV open in an effort to maintain sufficient cooling capability.
(m) Operator uncertain whether a rupture/leak exists (there is a conflict between being conservative under uncertainty versus immediate manual spurious shutdown which costs money and may be discouraged. May also prefer to wait for the automated system to resolve the problem versus intervening under uncertainty)
(n) Operator believes NSSC is operational when it is not (could cause operator to provide command to an inoperative or disabled NSSC instead of directly to PM)

UCA 3: Close MSIV provided when there is SGTR but other support systems are inadequate

(1) Secondary cooling system

(a) A concurrent situation could mask another, other support systems could appear adequate but may not be, and automated systems could exacerbate the situation. For example, main steam line high radioactivity may be detected coincident with safety injection, making it difficult to detect whether partial cooldown was initiated by the automation.
(b) Loss of power

(2) Process Feedback
(a) SG level feedback not provided, delayed, or incorrect
(b) SG Pressure or setpoints are not correct, delayed, or missing
(c) Steam generator water level not correct, delayed, or missing
(d) Conflicting data indicating a false situation
(e) Voting system does not operate properly and gives wrong measures
(f) Failures in sensors, communication lines, or power

(3) Outside information
(a) Wrong combination of indicators from the 4 divisions
(b) PZR pressure delayed or missing
(c) False signal SI initiated

(4) Operator
(a) Operator thinks support systems are working when they are not. For example, NSSC may appear to be working but may not be because the screen is frozen. The operator may believe that a partial cool down was initiated by the automation because safety injection was engaged at the same time that main steam line radioactivity was detected
(b) Operator believes there is a main steam line or feedwater leak when there is actually an SGTR
(c) Operator knows support systems are working, but does not realize they are inadequate
(d) Operator confused about the procedure to be followed
(e) Operator confused because of conflicting indicators
(f) Operator does not realize other support systems are not operative (e.g. for maintenance or other reasons)

UCA 4: Close MSIV provided too early (while SG pressure is high)
(1) Secondary cooling system
(a) A concurrent situation could mask another. For example, a feedwater problem could happen concurrently with a SGTR, and the SG water level stay practically stable.
(b) Event progress too slowly to detect
(c) Actuation of NSSC could confuse Operator. For example, PZR heaters could make up for loss of RCS pressure

(2) Process Feedback
(a) SG level feedback not provided
(b) SG Pressure, or setpoints, is not correct
(c) Steam generator water level not correctly indicated
(d) Main steam line activity not correctly indicated
(e) Conflicting data indicating a false situation
(f) Voting system does not work properly and gives wrong measures
(g) Sensors failure

(3) Outside Information
  (a) PZR pressure delayed
  (b) PZR feedback missing
  (c) False feedback indicates PZR level is normal
  (d) No indication of SI initiated
  (e) No indication of partial cool down initiated
  (f) Permissives wrongly in effect7
  (g) Wrong combination of indicators from the 4 divisions

(4) Operator
  (a) Operator believes it is already safe to initiate action after indications confirm SGTR
  (b) Operator believes it is already safe to initiate action after indications confirm Main steam line break
  (c) Operator believes it is already safe to initiate action after indications confirm main feedwater break
  (d) Operator confused about the procedure to be followed
  (e) Operator confused because of conflicting indicators

UCA 5: Close MSIV command provided too late after rupture/leak (in the SG tube, main feedwater, or main steam line)

(1) Secondary cooling system
  (a) A concurrent situation could mask another one. For example, a feedwater problem could happen concurrently with a SGTR such that the SG water level stays practically stable.
  (b) Event progress too slowly to detect
  (c) Actuation of NSSC could confuse Operator. For example, PZR heaters could make up for loss of RCS pressure

(2) Process Feedback
  (a) SG level feedback not provided
  (b) SG Pressure, or setpoints, is not correct
  (c) Steam generator water level delayed
  (d) Main steam line activity not correctly indicated or delayed
  (e) Conflicting data indicating a false situation
  (f) Voting system does not work properly and gives wrong measures
  (g) Sensor failure
  (h) PM reports both MSIV actuators as inoperable when they are
  (i) PM reports MSIV as already closed, when it is not
  (j) NSSC reported as operational (or no feedback) when it is not

(3) Outside Information
  (a) PZR pressure delayed
  (b) PZR feedback missing
  (c) False feedback indicates PZR level is normal
  (d) No indication or delayed indication of SI initiated
  (e) No indication or delayed indication of partial cool down initiated
  (f) Permissives wrongly in effect
  (g) Wrong combination of indicators from the 4 divisions
  (h) Screen is blank or frozen/NSSC or PS provides no feedback

(4) Operator
- (a) Operator thinks it is not yet safe to initiate action after SGTR is confirmed
- (b) Operator thinks it is not yet safe to initiate action after main steam line leak is confirmed
- (c) Operator thinks it is not yet safe to initiate action after main feedwater leak is confirmed
- (d) Operator confused about the procedure to be followed
- (e) Operator confused because of conflicting indicators
- (f) Operator reluctant whether to shutdown the reactor
- (g) Operator under pressure not to trip reactor
- (h) Operator has a conflict between being conservative with uncertainty of whether there is a SGTR, or to do what it is expected, i.e. to wait for the automated system to resolve the problem. In other words, the operator tries to avoid spurious shutdown, which costs money and should be avoided.
- (i) Operator waits for the PS to handle the situation, does not act in time

UCA 6: Close MSIV provided when there is no rupture/leak

(1) Secondary cooling system
- (a) Feedwater pumps not working properly
- (b) Condenser leaking (loosing water)
- (c) Too much sludge in water (blocking water)
- (d) Object in water that could cut flux to SG
- (e) Spurious opening of relief valves

(2) Process Feedback
- (a) SG level feedback not provided
- (b) SG Pressure low (setpoints not correct)
- (c) Steam generator water level delayed or incorrect
- (d) False SG isolation signal8
- (e) Main steam line activity (false positive signal)
- (f) Conflicting data indicating a false situation where close valve would be needed
- (g) Voting system does not work properly and gives wrong measures
- (h) Sensor Failure

(3) Outside Information
- (a) PZR pressure indication delayed
- (b) PZR feedback missing
- (c) False PZR pressure feedback
- (d) False feedback shows PZR level as low
- (e) False signal of initiation of SI
- (f) False Partial cool down initiated signal
- (g) Startup/shutdown not recognized
- (h) Wrong combination of indicators from the 4 divisions

(4) Operator
- (a) Operator thinks Steam Generator Tubes are ruptured when they are not
- (b) Operator thinks the main steam line has a leak when it does not
- (c) Operator thinks main feedwater has a leak when it does not
- (d) Operator confused about the procedure to be followed
- (e) Operator confused because of conflicting indicators

(f) Blank screen induces operator to think situation is different
(g) False alarm of radiation
(h) Close wrong valve, other SG

3.4.7 Results

An example insight obtained from the analysis is the difficulty of detecting a SGTR (steam generator tube rupture) through the normal indicators, which can lead to a delayed response by the automated controllers and the operator. The current solution relies on the operator's ability to detect and intervene in certain cases. Relying on the operator, however, may not be effective because of other factors that will influence the operator decision-making process. It is important to identify the factors under which a component, like the operator, may not act adequately and use those factors to improve the design of the system. The alternative is to simply blame the operators after an accident or incident for any failure to detect and resolve the problem as it was assumed they would.

Using a hazard analysis method based on STAMP allows more extensive analysis that includes events in which nothing failed but the hazards arise due to unsafe interactions among components. The identification of weaknesses in the overall PWR design is possible using STPA because the STPA analysis examines the interactions between the various controllers and system components. These weaknesses are unlikely to be found by hazard analysis methods based on assumptions about accidents being caused by chains of component failure events.

3.5 STPA in safety-guided design

As was mentioned in section 3.2.4, STAMP and STPA may be utilized either for analysing an existing system design, or for designing a completely new system from the beginning. Leveson [12] uses the term safety-guided design for the latter case. The idea behind safety-guided design is to design safety into the system from the very beginning. The system design and STPA are performed concurrently. According to Leveson, this provides better results than designing the system independently, and then trying to identify its weaknesses and implementing safety functions on top of the existing design.

Leveson [12] describes a process of how STAMP and STPA may be embedded into the design process:

1. Identify hazards, system-level safety requirements and constraints (as described in section 3.2.4)

2. Try to eliminate the hazards from the conceptual design

3. If any of the hazards cannot be eliminated, identify the potential for their control at the system level

4. Create a system control structure and assign responsibilities for enforcing safety constraints.

5. Refine the constraints and design in parallel.

    a. Identify potentially hazardous control actions by each of system component that would violate system design constraints using STPA step 1. Restate the identified hazardous control actions as component design constraints.

b. Using STPA step 2, determine what factors could lead to a violation of the safety constraints.

c. Augment the basic design to eliminate or control potentially unsafe control actions and behaviours.

d. Iterate over the process, that is, perform STPA steps 1 and 2 on the new augmented design and continue to refine the design until all hazardous scenarios are eliminated, mitigated, or controlled.

The process is illustrated with an example regarding an experimental design of a Space Shuttle robotic Thermal Tile Processing System (TTPS). The TTPS has two main goals:

1. To prevent the space shuttle belly surfaces from absorbing water, the TTPS should inject a specialized waterproofing chemical into heat-resistant tiles on the lower surface of the space shuttle.

2. The TTPS should inspect the tiles for scratches, cracks, gouges, and other flaws.

The TTPS operates in a facility that establishes some fundamental environmental constraints on the system, for example:

1. The work area can be very crowded.

2. The mobile robot must enter the facility through personnel access doors 1.1 meters wide.

In **step 1** of the safety-guided design process, all the system-level hazards are identified and ranked based on severity. System-level safety-related requirements and design constraints are derived from the hazards.

Table 14. Examples of hazards and constraints in the TTPS (adapted from [12])

Hazard	Description	System-level safety design constraint
H2	The robot base is instable.	The mobile base must not capable of falling over even under worst-case operational conditions.
H7	Inadequate thermal protection.	The TTPS must not miss any tiles in the waterproofing or inspection process.
…	…	…

These constraints are refined further when the design process proceeds.

After constraints have been identified, an initial system architecture must be created to get started with the actual design process. In this case, the initial TTPS architecture consists of a base where the tools can be mounted, including a robot arm that performs the processing and inspection. A human operator is also included to supervise the robot movement. The TTPS control system is in charge of the non-movement activities and both the TTPS and the human operator share control of movement activities. There is also an automated robot work planner to provide the overall goals and tasks for the robot. Also needed are a location system to provide robot position information for the movement controller, and a camera to provide visual information for the human operator.

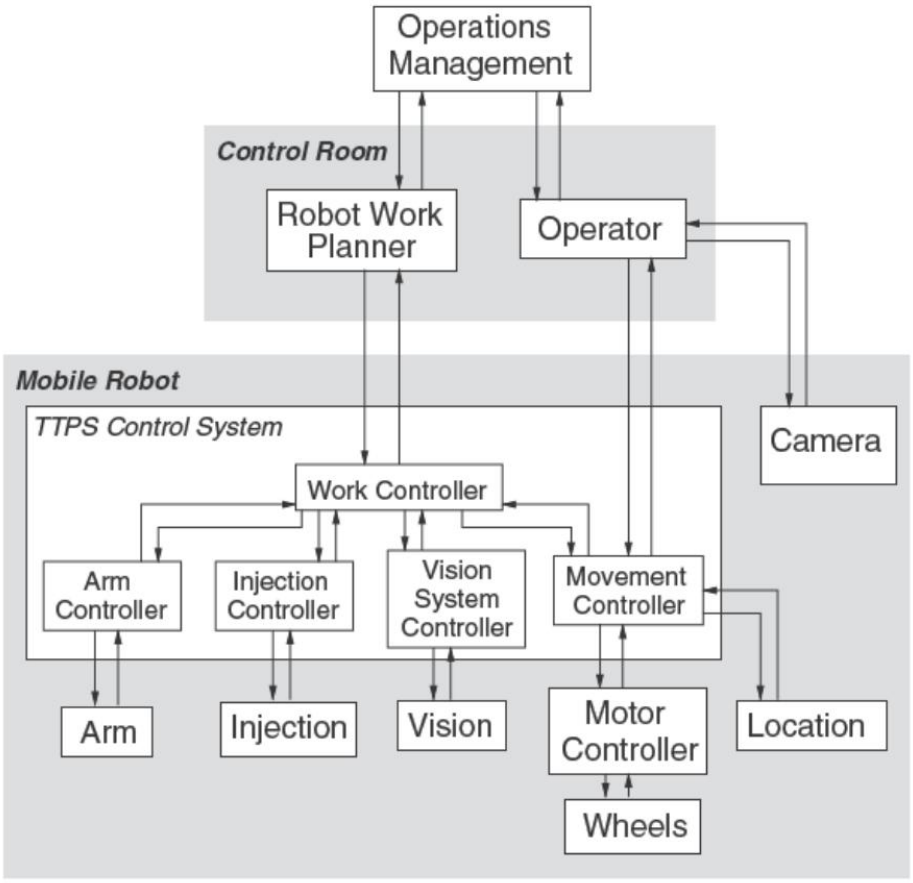


Figure 15. Initial TTPS control structure [12]

In **steps 2 and 3** of the process, hazards identified in step 1 should be either eliminated from the design, or adequately controlled. For example, the instability hazard H2 could be eliminated from the design by making the robot base so heavy that it cannot become unstable in any condition. This would, however, increase the damage caused if the base collides with a human, and make the base harder to move in an emergency situation. The base could also be made long and wide to prevent instability, but this could violate the environmental constraints and the robot might be unable to move through doors and in the crowded work area. A third option would be to control the instability hazard by including retractable stabilizer legs that extend only when the manipulator arm is extended.

The third option is selected at this point, but none of the design options should be ignored yet. After further analysis it might be discovered that making the robot base heavy is actually the best solution. New hazards and constraints can be identified for the case where stabilizer legs are included:

Table 15. More detailed interpretation of H2 (adapted from [12]).

Hazard	Description	System-level safety design constraint
H2.1	The manipulator arm is extended while the stability legs are not fully extended.	The manipulator arm must never be extended if the stabilizer legs are not extended.
H2.2	The mobile base moves while the	The mobile base must not move with

	stability legs are extended.	the stability legs extended.

In **step 4** of the process, when the new constraints are taken into account in the design, a refined control structure can be formed. Here it is assumed that the legs are controlled by the movement controller.

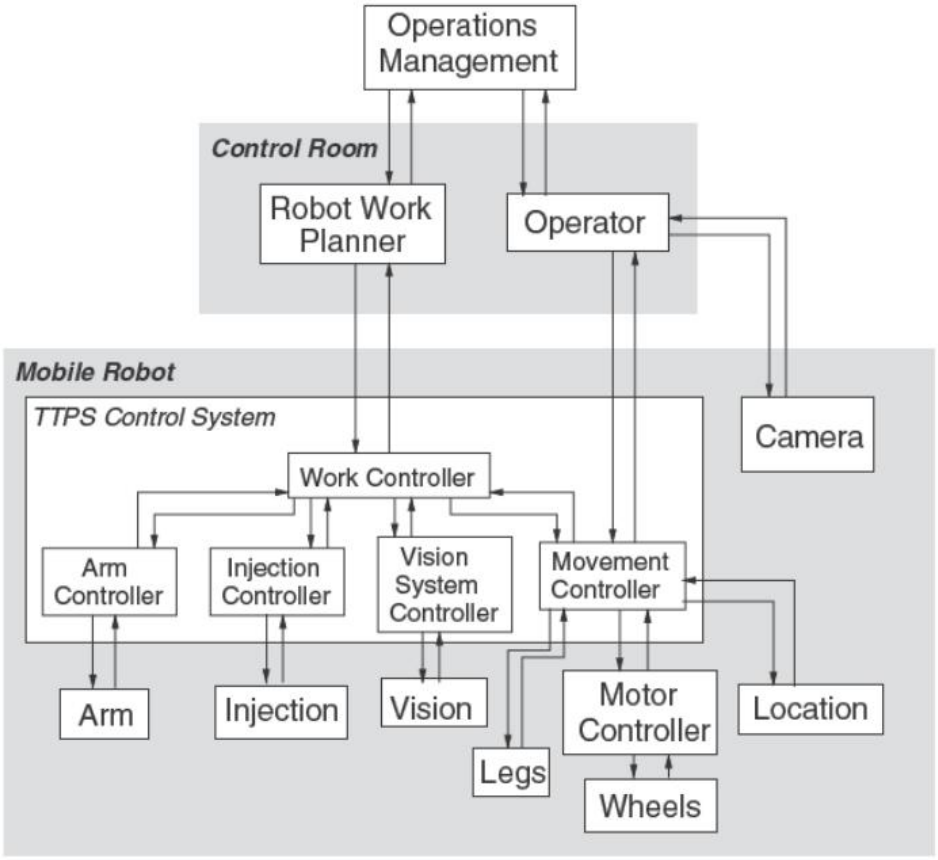


Figure 16. Refined TTPS control structure [12]

In **step 5**, using the refined control structure, STPA can then be used to identify hazardous control actions, which can be interpreted as safety constraints. Causal factors that could lead to these constraints being violated are also determined. The unsafe control actions should then be eliminated or controlled in the design. The whole step is iterated over until all hazardous scenarios are adequately eliminated or controlled.

The process therefore includes a top-down identification of hazards and factors that might lead to these hazards. The design is systematically refined from the initial assumption to the point where all the hazards should be handled. A more comprehensive explanation of the example can be found from [12].

4. STPA compared to traditional methods

4.1 General comparison

4.1.1 STPA vs. FTA

FTA is a top down analysis method that is based on an event chain model where accidents happen due to particular combinations of certain low-level events (basic events) occurring at the same time. The traditional FTA also includes a probabilistic element where probabilities of the basic events (often component failures) are estimated and used to calculate the probability of the accident. However, this quantitative step isn't always used (e.g. in the case of software FTA).

The event chain approach of FTA is fundamentally very different from the hierarchical control structure approach of STPA. STPA assumes that accidents happen due to incorrect control actions instead of being a chain reaction caused by failed components, and therefore it lacks the probabilistic considerations that FTA has. Accidents can indeed happen because of a component failure, but the reason behind that failure might be e.g. a bad decision made on the management level years before the accident. These kinds of complicated causal factors are difficult to include in fault trees. Also, in complex and large systems, the fault trees in FTA can grow very large and analysing them can be very time-consuming. Due to these reasons, compared to STPA, FTA is more suitable for local lower-level analysis of single failure modes.

An example about the use of FTA was introduced in section 2.5.3. In this example, hazards in the KNICS RPS software were first identified by other methods (FMEA and HAZOP). Then, since analysing the whole system with FTA would have been too time-consuming, FTA was applied only on the most significant hazards. This kind of limited FTA analysis seems like the way how FTA might potentially be used in the future. STPA, however, is generally utilized on a higher level to analyse whole systems, and even systems along with their surrounding societal structures.

4.1.2 STPA vs. FMEA and HAZOP

Both HAZOP and STPA offer some guidance for the analysis process. HAZOP does this in the form of guidewords. As discussed in chapter 2.4, pre-selected guidewords are combined with system parameters to form deviations. Each deviation is then systematically analysed to discover its cause and effects. STPA, being based on control structures and unsafe control actions, includes a general classification of control flaws for each type of component. This classification, introduced in Figure 11, can be used much like the guidewords in HAZOP when figuring out the control flaws regarding a certain component. FMEA, however, doesn't offer guidance in the identification of failure modes or their causes.

All three methods require both domain knowledge and knowledge of the method itself. However, STPA might initially require more training on the STPA procedure than HAZOP of FMEA. In HAZOP, for example, it is sufficient when one member of the team (the study leader) has significant experience of applying the method, and it is his responsibility that the method is followed correctly. The rest of the team members should have more comprehensive domain knowledge about the design and use of the system. Also, since all the traditional methods have been used for decades, and since the whole philosophy of STAMP & STPA differs so much from the traditional methods, new users of STPA are likely to require guidance to get started.

Compared to FMEA and HAZOP, STPA was designed to have a more comprehensive scope. In theory, it broadens the scope of HAZOP and FMEA by taking into account the

societal control structure and having a stronger emphasis on component interactions and human factors. Theoretically it should be able to discover the same hazards as HAZOP and FMEA, and hazards that were not discovered by the two methods.

4.1.3 Summary

Table 16 establishes a summary where the four methods were compared based on certain characteristics.

Table 16. Comparison between the methods based on certain characteristics

	FTA	**FMEA**	**HAZOP**	**STPA**
Year originated	1962	1950s	1960s	2002
Basis	Fault trees; accidents occur due to a chain of component failures.	Systematic analysis of components.	Analysis of components based on deviations.	Control theory; hierarchical control structure and unsafe control actions.
Qualitative / quantitative	Both	Mainly qualitative (probabilistic data may be used to calculate Risk Priority Numbers, but quantification isn't as fundamental as with FTA)	Qualitative	Qualitative
Required inputs	System documentation Component failure data (if probability steps are applied)	System documentation	System documentation	STAMP model (depending on the abstraction level, building the model may require comprehensive information on the socio-technical structure, design specification, etc.)
Minimum size of the analysis team	1	4	4	1
Level of expertise (on the method) required of the analysis team	Medium to high	Medium	Medium	High

Direct results	Fault trees that show how each failure can occur due to specific basic events. Probability estimates for each failure (if applied).	A list of items/components. Failure modes and the effects of these failure modes.	A list of components and their deviations. The effects and causes of each deviation.	STAMP model. Unsafe control actions and their effects. Causal factors (what are the causes for each unsafe control action).
Potential weaknesses	Narrow focus. Fault trees can grow very complex when analysing large systems. Gathering component failure data and calculating probabilities can be time-consuming.	No guidance for the identification of failure modes or their causes – relies a lot on the domain expertise of the analysts.	Time-consuming. Requires a rather large analysis team.	Requires significant amount of input data to build a comprehensive STAMP model. Members of the analysis team need substantial expertise on the method. Can be time-consuming, especially if analysts are inexperienced. In spite of high interest, currently limited amount of independent experiences* available.
Potential strengths	Probabilistic failure estimates can be used for certain applications (e.g. Safety Integrity Levels).	Existing FMEA worksheet templates make the process clearer.	Guidewords offer guidance on the systematic identification of deviations.	Modelling systems based on control theory should theoretically provide a wider scope compared to other methods. This should enable the identification of new kinds of flaws e.g. regarding inadequate communication between components. Also offers guidance on the identification of unsafe control actions.

				Modern, emerging method.

*Meaning applications without direct contribution by Ms. Leveson or her students.

All of the methods have their individual strengths and weaknesses. A clear weakness of FTA and FMEA when comparing them with HAZOP and STPA is the lack of guidance in the analysis. In FTA this means there is no guidance on which hazards should be analysed by the method, i.e. which events should be selected as top events of the trees. In FMEA this means there is no guidance on how to identify failure modes or their effects and causes. Engineers involved in the analysis are solely responsible for these actions, which makes the analysis highly subjective. On the other hand, when using HAZOP, one must be able to select the right guidewords and parameters for the case. Also, when utilizing STPA, even creating the control structure might prove difficult, especially if the analysts are not experts on the method and/or the system design is constantly evolving.

Traditional methods such as FTA or FMEA work better on finished system designs. One significant benefit that STPA has over traditional methods is that it can be better applied to systems in their design phase. According to Ishimatsu et al. [9]: "*System designs have become so complex that waiting until a design is mature enough to perform a safety analysis on it is impractical. The only practical and cost-effective safe design approach in these systems is to design safety in from the beginning*." In safety-driven design, the analysis is performed in parallel with the design process, not after it.

There is very limited reported experience in combining STPA with traditional hazard analysis methods. However, creating some sort of a unified hazard analysis framework, combining both traditional event-based methods and a systematic method such as STPA, seems like an interesting course of study.

4.2 Real-world experiences

EPRI (Electric Power Research Institute) compared different methods on a real nuclear power reactor design. These methods included FTA, ETA, HAZOP, FMEA, STPA, and a few others. According to STPA Primer [13], STPA was the only method that found a scenario for a real accident that had occurred on that plant design, and which the analysts did not know about.

Song [17] applied STPA to the Shutdown System 1 of the Darlington nuclear power plant located in Canada. The same system had previously been analysed with FMEA, which made it possible to compare results obtained from STPA with the ones obtained from FMEA. Song states that when compared to the FMEA analysis results, STPA found more hazards, failure modes and causal factors. However, none of these were particularly significant. Song concludes that: "*The results indicate that not only more hazards, failure modes and causal factors related to the trip computer were identified, but more components and their interactions were considered. This is because STPA analyzes hazards and causal factors in a systematic way. It considers not only components themselves, but also the interactions among components or between operator and components. By using a safety control structure and a general control laws classification to analyze causal factors of each identified hazard, STPA may help the analyst to find more failure modes and causal factors*."

In a joint research project between Massachusetts Institute of Technology and Japan Aerospace Exploration Agency (JAXA), Ishimatsu et al. [9] applied STPA to HTV (H-IIB Transfer Vehicle). HTV is an unmanned vehicle that is launched by a rocket to carry cargo to the International Space Station. JAXA has traditionally used component failure based hazard analysis methods such as FTA and FMEA. The results from the STPA analysis were

compared against the existing results obtained from FTA. The causal factors that were identified by STPA included all the hazard causes of the FTA. STPA also found additional hazards that were not identified by the FTA. The authors conclude: "*We found that causal factors other than component failures such as process model inconsistency, causal factors with regard to "delay of command," "delay of feedback," and "acknowledgment of control action" are not identified by FTA*."

Fleming et al. [3] utilize STPA in the safety analysis of NextGen. NextGen represents the transformation of the United States air traffic control system from ground-based to satellite-based targeting reduced flight delays, reduced environmental impact and improved safety of air traffic. The former safety analysis of NextGen is based on FTA and is documented in a document called DO-312. Fleming et al. discuss several fundamental differences between the underlying philosophies of the DO-312 method and STPA, and how this leads to different kinds of results. The FTA approach models causal factors with fault trees, and assigns probabilities to basic events, even to events that represent human error. According to the authors, probability for human error cannot be verified. Also, the FTA approach doesn't discuss any means how to prevent these errors; it only assumes they happen randomly. FTA represents communication errors by simply software of hardware failures within the communication system, although there might be many additional causes for flawed communication. The STPA approach included the basic communication errors that the FTA did, but also included additional reasons for communication errors as well as guidance for understanding human errors.

5. Analysis methods in nuclear domain regulations and licensing

5.1 Current state

The regulatory requirements on nuclear power plant safety by STUK (the Finnish nuclear safety authority) are presented in the YVL series of documents. YVL B.1 [19] titled "Safety design of a nuclear power plant" addresses failure tolerance analysis. In clause 352, it states that a failure tolerance analysis should be performed systematically for each system that performs a safety function: "*A failure tolerance analysis shall assess one functional complex at a time, with due regard both to the system that performs a safety function and its auxiliary systems. The analysis shall address each component that, in the event of a failure, may affect the successful execution of the safety function performed by the system following a specific initiating event. The analysis shall address all modes of failure for all the components affecting the system performing the safety function. Depending on the applicable failure criterion, the analysis shall focus on one failure at a time and examine its impact in terms of the operation of the system*." Clause 5241 of the document addresses the analysis of I&C systems: "*The effects of the failures and errors of the controls and functions performed by the I&C systems shall be analysed as functional entities. Functional entities may consist of system-internal structures, and they may cross the interfaces between systems. The functional entities selected for analysis shall be justified. The analysis shall account for all possible failure modes of the I&C systems."* It is quite obvious that some of the traditional methods listed in this document, e.g. FMEA and FTA, are suitable for this kind of analysis. However, the YVL B.1 doesn't name specific methods that should be used.

The document titled "Licensing of safety critical software for nuclear reactors - Common position of seven European nuclear regulators and authorised technical support organisations" [2] (later called CP2013) clearly states that failure and hazard analysis has to be done and that "*the possible failure modes of the architecture that may compromise the safety functions have been taken into account*", and that "*adequate exception handling mechanisms and hazard mitigating functions have been included in the design*" (clause

2.2.3.8). It also states that the software requirements have to address the results of the system failure and hazard analysis (clause 2.3.3.1.4). However, CP2013 doesn't clearly list specific methods that are suitable for such analyses.

IEC 61508 is a generic safety standard that is also used in the nuclear domain. IEC 61508-2 [6] is a part of the standard that considers requirements for the functional safety of E/E/PE (electrical/electronic/programmable electronic) safety-related systems. IEC 61508-3 [7] is a part of the standard that includes similar requirements for safety-related software. Both IEC 61508-2 and IEC 61508-3 provide recommendations on the use of failure analysis methods (the term failure analysis largely corresponds to the term hazard analysis used in this report). Table B.5 in IEC 61508-2 considers the use of failure analysis when validating the safety of E/E/PE systems. Table B.4 in IEC 61508-3 lists methods such as cause consequence diagrams, event tree analysis, fault tree analysis, and software functional failure analysis as possible techniques to be used when analysing safety-related software. Both standards base their recommendations on the Safety Integrity Level of the target system. Also, both standards reference Table B.6.6 in IEC 61508-7 [8], which provides a list and description of suitable failure analysis methods, including, e.g., FMEA, ETA, and FTA.

5.2 Potential benefits of STPA for licensing

A case example [20] of applying STPA to a part of an EPR MSIV closing system was introduced in section 3.4. After conducting the case study, the authors discussed ways by which STPA could improve the existing NRC safety assurance framework. Four most important use cases identified were [20]:

1. Classification of components as safety-related vs. non-safety-related

   The authors argue that classification of components to safety-related and non-safety-related isn't as straightforward to software components as it is to traditional electro-mechanical components. In the case example a non-safety-related controller named NSSC can slow down the closure of the MSIV by e.g. reporting erroneous feedback to the operator, and thus affect safety-related functions. STPA does not fundamentally classify controllers to safety-related or non-safety-related. In Step 1 STPA considers, for each controller, unsafe control actions and hazards that the actions can contribute to. Thus, the outputs of Step 1 could be used to classify components or to verify existing classifications.

2. Broadening the scope of the analysis

   Operator errors, flaws in safety culture and organizational and human factors can be as dangerous for the system as component failures. STPA can be used to capture these factors.

3. Assisting authorities in understanding applicant functional designs

   As part of STPA, a STAMP model of the system is developed. This model can help regulatory authorities understand the system better than the traditional system design documentation.

4. Enhancing the review of candidate designs

   The output of Step 1 of STPA shows the hazards of the system, the unsafe control actions, and which hazards each of the control actions connect to. The output of Step 2 provides identification of possible causes for the unsafe control actions. The outputs form a traceable chain and can be used for comparing against safety requirements of

an existing system and finding potential insufficiencies, or generating new requirements and mitigation measures that have to be met.

Even though the study was conducted for the NRC in a U.S. context, the benefits listed above are universal.

The nuclear domain has traditionally been rather slow in adopting new policies to regulations and standards. Despite the positive experiences, STPA is still a fairly young method and, compared to traditional methods, represents a very new approach to safety. It is still uncertain whether it will emerge to become widely used in the domain.

6. Conclusions

Microprocessor-based I&C systems and the use of software enable designers to design more complex systems than ever before. This digital technology is used in all kinds of systems, even in the most safety-critical contexts, such as the nuclear domain. However, software-based systems can contain hidden faults that might be extremely difficult to detect. Effective methods are needed to mitigate the possibility of errors and hazards in such systems.

FTA uses fault trees to analyse faults in systems. The advantages of fault trees include the possibility to take into account probabilities of the occurrence of basic events. However, the analysis is limited to faults that occur due to chain of events. This suits well when analysing hazards due to hardware component failures, but not as well when analysing hazards in software-based systems or large systems (fault trees have the potential to grow very complex).

FMEA is a systematic way of analysing system hazards, and it has a wider focus than FTA. FMEA, however, doesn't offer any guidance for the identification of failure modes or their causes – it relies a lot on the domain expertise of the analysts. HAZOP and STPA offer some of this guidance. HAZOP achieves this through the use of guidewords, and STPA offers guidance on the identification of unsafe control actions.

Overall, STPA offers a rather different approach on hazards analysis. The method is based on control theory, and it was designed to also take into account communication faults between system components and the societal structure surrounding the system. It widens the scope of traditional methods and should theoretically enable the identification of new kinds of flaws. Real-world experiences in the use of the method have been very positive so far. STPA also shows potential regarding its use for licensing purposes in the nuclear domain. But despite the positive experiences, STPA is still a fairly young method with a limited use base. Also, quite a significant amount of knowledge about the method is required for one to be able to apply it to real-world cases. The method has, however, created major interest in many fields and its use can be expected to increase in the future.

References

1. Čepin, M. 2011. Assessment of Power System Reliability. Springer, 61–87.
2. European Commission's Advisory Experts Group, 2013. Nuclear Regulators Working Group, Licensing of safety critical software for nuclear reactors - Common Position of seven European nuclear regulators and authorized technical support organizations, Revision 2013.
3. Fleming, C. et al. 2012. Safety Assurance in Nextgen. NASA Technical Report. 36 p.
4. Flowserve WWW page. Retrieved 19/08/2014. Available: http://www.flowserve.com/Industries/Power-Generation/Nuclear.
5. Haapanen, P. & Helminen, A. 2002. Failure Mode and Effects Analysis of Software-based Automation Systems. Helsinki, Finland: STUK-YTO-TR 190. 35 p.
6. International Electrotechnical Commission, IEC 61508, Functional Safety of Electrical/Electronic/Programmable Electronic Safety Related Systems – Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems. Second edition, IEC 2010.
7. International Electrotechnical Commission, IEC 61508, Functional Safety of Electrical/Electronic/Programmable Electronic Safety Related Systems – Part 3: Software requirements. Second edition, IEC 2010.
8. International Electrotechnical Commission, IEC 61508, Functional Safety of Electrical/Electronic/Programmable Electronic Safety Related Systems – Part 7: Definitions and abbreviations. Second edition, IEC 2010.
9. Ishimatsu, T., Leveson, N. et al. 2011. Modeling and Hazard Analysis using STPA. Conference of the International Association for the Advancement of Space Safety, Huntsville, AL, USA. 10 p.
10. Lawrence, J.D. 1995. Software Safety Hazard Analysis, UCRL-ID-122514, Lawrence Livermore National Laboratory. 24 p.
11. Leveson, N.G. 2004. A new accident model for engineering safer systems. Safety Science 42 (4), 237–270.
12. Leveson, N.G. 2012. Engineering a Safer World: Systems Thinking Applied to Safety. Cambridge, MA, USA: The MIT Press. 534 p.
13. Leveson, N.G. et al. An STPA Primer, Version 1. Retrieved: 06/08/2014. Available: http://sunnyday.mit.edu/STPA-Primer-v0.pdf. 80 p.
14. Oh, Y. et al. 2005. Software Safety Analysis of Function Block Diagrams using Fault Trees. Reliability Engineering and System Safety 88, 215–228.
15. Park, G.Y. et al. 2007. Safety Analysis of Safety-Critical Software for Nuclear Digital Protection System. Computer Safety, Reliability, and Security - Lecture Notes in Computer Science Volume 4680, 148-161.
16. Park, G.Y. et al. 2008. Fault Tree Analysis of KNICS RPS Software, Nuclear Engineering and Technology 01/2008. 12 p.
17. Song, Y. 2012. Applying System-Theoretic Accident Model and Processes (STAMP) to Hazard Analysis. MSc. Thesis, McMaster University, Ontario, Canada. 95 p.
18. Stamatis, D. H. 2003. Failure Mode and Effect Analysis - FMEA from Theory to Execution (2nd Edition). Milwaukee, WI, USA: ASQ Quality Press. 442 p.
19. STUK, 2013. Radiation and Nuclear Safety Authority, Guide YVL B.1 Safety design of a nuclear power plant.
20. Thomas, J. et al. 2012. Evaluating the Safety of Digital Instrumentation and Control Systems in Nuclear Power Plants. NRC-HQ-11-6-04-0060, NRC Research report. 66 p.
21. USCG Risk-Based Decision-Making Guidelines (Vol. 3, chapter 10). 34 p.