



**NTNU – Trondheim**  
Norwegian University of  
Science and Technology

# Using reliability growth testing to reveal systematic faults in safety-instrumented systems

**Rowland Mbah**

Reliability, Availability, Maintainability and Safety (RAMS)

Submission date: June 2014

Supervisor: Marvin Rausand, IPK

Co-supervisor: Professor Mary Ann Lundteigen, IPK

Norwegian University of Science and Technology  
Department of Production and Quality Engineering



# RAMS

Reliability, Availability,  
Maintainability, and Safety

## Using reliability growth testing to reveal systematic faults in safety-instrumented systems

Rowland Mbah

June 2014

MASTER THESIS

Department of Production and Quality Engineering

Norwegian University of Science and Technology

Supervisor : Professor Marvin Rausand



**MASTER THESIS**  
**Spring 2014**  
**for stud. techn. Rowland Mbah**

**Using reliability growth testing to reveal systematic faults in safety-instrumented systems**

**(Bruk av pålitelighetsvekst-testing for å avdekke systematiske feil i instrumenterte sikkerhetssystemer)**

Systematic faults strongly influence the availability of safety-instrumented systems (SISs). Systematic faults may be introduced in all the lifecycle phases of a SIS, but this thesis is mainly concentrated on systematic faults that are introduced in the design and development phases. SIS reliability requirements are specified in the international standard IEC 61508 and the associated sector-specific standards. According to these standards, the SIS system integrator must use a formalized system for identification and avoidance of systematic faults. A possible approach is to use reliability growth testing for this purpose.

The main objective of this master thesis is to study, evaluate, and discuss to what extent reliability growth testing of a SIS is a suitable approach for identifying and avoiding systematic faults. If the conclusion is affirmative, another objective is to establish guidelines for reliability growth testing for this purpose and to identify challenges and pitfalls related to such testing.

Questions to be addresses and discussed as part of this master's project are:

1. What do we mean by systematic faults of a SIS?
2. How can systematic faults be classified?
3. What are the relationships between systematic faults and common-cause failures (CCF)?
4. What types of systematic faults can be introduced in the design and development phases?
5. What do we mean by reliability growth testing?
6. Which models and methods are available for reliability growth testing?

7. What are the strengths and weaknesses of the various models and methods for reliability growth testing?
8. Can reliability growth testing be used to identify systematic faults?
9. How can such testing be implemented?
10. What are the challenges and pitfalls related to reliability growth testing in relation to systematic faults?

Following agreement with the supervisor(s), the questions may be given different weights.

The assignment solution must be based on any standards and practical guidelines that already exist and are recommended. This should be done in close cooperation with supervisors and any other responsibilities involved in the assignment. In addition it has to be an active interaction between all parties.

Within three weeks after the date of the task handout, a pre-study report shall be prepared. The report shall cover the following:

- An analysis of the work task's content with specific emphasis of the areas where new knowledge has to be gained.
- A description of the work packages that shall be performed. This description shall lead to a clear definition of the scope and extent of the total task to be performed.
- A time schedule for the project. The plan shall comprise a Gantt diagram with specification of the individual work packages, their scheduled start and end dates and a specification of project milestones.

The pre-study report is a part of the total task reporting. It shall be included in the final report. Progress reports made during the project period shall also be included in the final report.

The report should be edited as a research report with a summary, table of contents, conclusion, list of reference, list of literature etc. The text should be clear and concise, and include the necessary references to figures, tables, and diagrams. It is also important that exact references are given to any external source used in the text.

Equipment and software developed during the project is a part of the fulfilment of the task. Unless outside parties have exclusive property rights or the equipment is physically non-moveable, it should be handed in along with the final report. Suitable documentation for the correct use of such material is also required as part of the final report.

The student must cover travel expenses, telecommunication, and copying unless otherwise agreed.

If the candidate encounters unforeseen difficulties in the work, and if these difficulties warrant a reformation of the task, these problems should immediately be addressed to the Department.

**The assignment text shall be enclosed and be placed immediately after the title page.**

Deadline: 10 June 2014.

Two bound copies of the final report and one electronic (pdf-format) version are required according to the routines given in DAIM. Please see <http://www.ntnu.edu/ivt/master-s-thesis-regulations> regarding master thesis regulations and practical information, inclusive how to use DAIM.

Responsible supervisor:

Professor Marvin Rausand  
E-mail: [marvin.rausand@ntnu.no](mailto:marvin.rausand@ntnu.no)  
Telephone: 73592542

Co supervisor:

Professor Mary Ann Lundteigen  
E-mail: [mary.a.lundteigen@ntnu.no](mailto:mary.a.lundteigen@ntnu.no)  
Telephone: +47 930 59 365

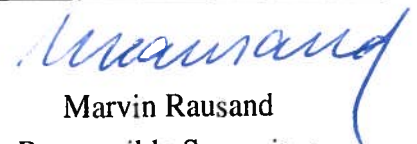
**DEPARTMENT OF PRODUCTION  
AND QUALITY ENGINEERING**



Per Schjølborg

Associate Professor/Head of Department

---



Marvin Rausand  
Responsible Supervisor

## **Preface**

This master's thesis was written during the spring semester 2014, at the Department of Production and Quality Engineering, Faculty of Engineering and Technology, NTNU. The thesis is part of the two-year International master's programme in RAMS at NTNU.

The title of the thesis is "Using reliability growth testing to reveal systematic faults in safety-instrumented systems". The main objective of this thesis is to develop a procedure for how to identify and correct systematic faults by reliability growth testing and the TAAF (test-analyze-and-fix) cycle. This is a challenging topic that may have great effects on the development of high-reliable safety-instrumented systems. Relevant challenges and pitfalls of reliability growth testing in relation to systematic faults are identified and guidelines for such tests proposed. It is assumed that the reader has some knowledge in safety and reliability analysis. It is preferable to have some knowledge of the standards IEC 61508, IEC 61511 and IEC 61014.

Trondheim, June 24, 2014

Rowland Mbah



## **Acknowledgment**

I would like to express my deepest and sincere gratitude to Professor Marvin Rausand for his excellent guidance, understanding and encouragement throughout the period of writing this report. I also thank my family members and friends for their support.

R.M.

## Summary

This master thesis studies the effects of systematic faults in the development phase of a safety-instrumented system, especially the relation between systematic faults and operational common-cause failures. Safety-instrumented systems are used widely in many industry sectors to detect the onset of hazardous events and mitigate the consequences to humans, the environment and material assets.

Systematic faults are non-physical faults introduced due to design errors or mistakes. Unidentified systematic faults represent a serious problem, as their safety effects are unpredictable and are not normally susceptible to a statistical analysis like random faults. In addition to safety effects, there can also be economic losses through product recalls, high warranty costs, customer dissatisfaction and loss of market share. Reliability growth testing is the same as TAAF (test-analyze-and-fix) testing of a product early in the design and development phases of the product life cycle when design changes can be made readily in response to observed failures. Reliability growth testing, if applied in the development phase of a safety-instrumented system helps to overcome the disadvantages of doing the test in other phases, because it can be costly, highly inconvenient and time consuming in these phases.

The main focus of the thesis is to study, evaluate, and discuss to what extent reliability growth testing of safety-instrumented systems is a suitable approach for identifying and avoiding systematic faults, and develop guidelines for reliability growth testing to achieve this purpose. The thesis builds on concepts, methods and definitions adopted from two major standards for safety-instrumented system applications: IEC 61508 and IEC 61511, and IEC 61014: Programmes for reliability growth. The development of procedures on how to identify and correct systematic faults by reliability growth testing are inspired by these three standards and other relevant literature found during the course of the master thesis project.

The main contributions of this thesis are:

1. Illustrative examples of fire and gas detection and mitigation systems, car airbag and mo-

bile phone have been used to develop procedures on how reliability growth testing is used to identify and correct systematic faults.

2. Detailed discussion of systematic faults, common-cause failures and the relationship between them have been presented. It has been established that systematic faults give rise to common-cause failures, which dominate the reliability of safety-instrumented systems.
3. Detailed discussion of reliability growth testing, its models and methods, and strengths and weaknesses of the models and methods have been provided. Both continuous and discrete models are studied. The Duane model, which is an example of a continuous model is commonly used because of its simplicity and graphical presentation.
4. The challenges and pitfalls of reliability growth testing in relation to systematic faults are discussed. The major challenge is the introduction of new failure modes, especially in case of software testing.
5. Measures to handle systematic faults revealed during the test have been provided. The measures include: use of diverse and redundant channels, design reviews, use of simple designs, use of competent designers, training and re-training of designers and use of reliability analysis to identify causes of faults [such as Failure Modes Effects and Criticality Analysis (FMECA)]

# Contents

Preface . . . . .	i
Acknowledgment . . . . .	ii
Summary . . . . .	iii
<b>1 Introduction</b>	<b>2</b>
1.1 Background . . . . .	2
1.2 Objectives . . . . .	4
1.3 Limitations . . . . .	4
1.4 Structure of the Report . . . . .	5
<b>2 Introduction to Safety-Instrumented Systems</b>	<b>6</b>
2.1 Definition of Safety-Instrumented Systems . . . . .	7
2.2 Safety- Instrumented Function . . . . .	8
2.3 Safety Integrity Level . . . . .	10
2.4 Determination of Safety Integrity Level . . . . .	11
<b>3 Introduction to Safety Life Cycle and Product Life Cycle</b>	<b>13</b>
3.1 Introduction . . . . .	13
3.2 Overall Safety Life Cycle . . . . .	14
3.3 Objectives . . . . .	14
3.4 Safety Life Cycle and Standards . . . . .	14
3.4.1 IEC 61508 and Safety Life Cycle Classification . . . . .	14
3.5 Product Life and Product Life Cycles . . . . .	19
3.5.1 Product Life . . . . .	19

3.6	RAMS-Oriented Product Life Cycle Model . . . . .	20
3.7	Activities Performed in each Phase of the Product Life Cycle to Obtain High Reliability . . . . .	22
<b>4</b>	<b>Systematic Faults/Failure Classification</b>	<b>26</b>
4.1	Failure or Fault? . . . . .	26
4.2	Systematic Failures versus Systemic Failures . . . . .	27
4.3	Systematic Failure versus Random Hardware Failure . . . . .	28
4.3.1	Definition of Systematic Faults by Standards and Guidelines . . . . .	30
4.3.2	Importance of Systematic Faults . . . . .	34
4.3.3	Types of Systematic Faults that can be Revealed in the Design and Development Phase . . . . .	35
<b>5</b>	<b>Common-Cause Failures</b>	<b>38</b>
5.1	Introduction . . . . .	38
5.2	Definition of Common-Cause Failure . . . . .	38
5.3	Causes of Common-Cause Failure . . . . .	40
5.4	Relationship between Systematic Failure and Common-Cause Failure . . . . .	42
5.4.1	Types of Systematic Faults that do not Lead to a Common-Cause Failure . . . . .	43
5.5	Design Measures Against CCF . . . . .	44
<b>6</b>	<b>Reliability Growth Testing</b>	<b>47</b>
6.1	Literature Review of Work done on Reliability Growth Testing . . . . .	48
6.2	Reliability Growth Models and Methods . . . . .	49
6.2.1	Continuous Growth Models . . . . .	49
6.2.2	Discrete Models . . . . .	52
6.3	Strengths and Weaknesses of Various Models and Methods . . . . .	53
6.3.1	Duane Model . . . . .	53
6.3.2	AMSAA Model . . . . .	53
<b>7</b>	<b>Reliability growth testing for systematic faults</b>	<b>55</b>
7.1	Introduction . . . . .	55

<i>CONTENTS</i>	1
7.2 Illustrative Examples . . . . .	55
7.2.1 F&G Detection and Mitigation System . . . . .	56
7.2.2 Mobile Phone (Smartphone) . . . . .	57
7.2.3 Airbag . . . . .	59
7.3 Challenges and Pitfalls of Reliability Growth Testing in Relation to Systematic Faults	62
7.3.1 Pitfalls During RGT Planning . . . . .	63
7.3.2 Pitfalls During the Execution of Test . . . . .	64
7.3.3 Pitfalls During Analysis of Test Data . . . . .	65
7.4 Handling of Systematic Faults Revealed During the Test . . . . .	66
<b>8 Summary</b>	<b>73</b>
8.1 Summary and Conclusions . . . . .	73
8.2 Recommendations for Further Work . . . . .	75
<b>A Acronyms</b>	<b>76</b>
<b>Bibliography</b>	<b>79</b>

# Chapter 1

## Introduction

### 1.1 Background

The importance of safety-instrumented systems (SISs), especially in the oil and gas industries cannot be overstated; SISs play a major role in keeping the risk arising from manufacturing, process or transportation activities within tolerable limits. A SIS comprises sensors, logic solvers, and final elements as shown in Figure 2.1

Systematic faults influence the availability of SIS negatively. It is beneficial to reveal and correct systematic faults at the design and development phases of the SIS life cycle because it can be costly, highly inconvenient and time-consuming after these phases. SIS reliability requirements are specified in the international standards [IEC 61508 \(2010\)](#), [IEC 61511 \(2003\)](#) and other sector-specific standards. These standards provide a general framework and requirements for the design, development and operation of a SIS. According to these standards, the SIS system integrator must use a formalized system for identification and avoidance of systematic faults. Reliability growth testing (RGT) and the TAAF (Test, Analyze, and Fix) cycle are the approaches used for this purpose in this thesis.

A lot of work has been done on RGT by [Quigley and Walls \(1999\)](#); [Walls and Quigley \(2001\)](#) and [Krasich et al. \(2004\)](#). The RGT models and methods used to assess reliability growth are classified into *continuous* and *discrete* models ([Duane, 1964](#); [Crow, 1975](#)). The Duane and AM-SAA (Army Material Systems Analysis Activity) models are examples of continuous models; the

Wolman model and the Lloyd/Lipow model (Lloyd and Lipow, 1963) are examples of discrete models. The Duane model is used even by strong advocates of the AMSAA model because it is very simple to use and can be presented graphically.

Systematic faults and common cause faults have been studied by Gentile and Summers (2006) and Hauge et al. (2004). A systematic fault may affect all forms of components be it hardware or software. The fault may be introduced in the design or the manufacturing process, or it may be related to maintenance or modification activities. A common-cause failure (CCF) is a failure that occurs when failures of separate channels are triggered concurrently. The failures are considered concurrent if the interval between the failures is too short for repair or recovery measures to be taken (NP-T-1.5, 2009). Systematic faults are of great importance because they may give rise to CCFs, which dominate the reliability of a SIS. Systematic faults do not give rise to CCF if the redundant channels are not challenged and also if the failures affect only one channel.

In order to demonstrate how RGT is used to reveal faults, illustrative examples of fire and gas detection and mitigation systems, car airbag, and mobile phone were used. A mobile phone, though, not a SIS, is used for a clearer understanding of the principle. RGT as used in the thesis, can be formal, semi-formal or informal. Formal testing means testing in the laboratory in controlled conditions and an environment that mimics the real world in a limited manner. Informal testing is releasing the product to a limited number of customers or to employees of the manufacturer to use and keep appropriate information relating usage mode and intensity, operating environment, failure modes and report back to the design team for necessary corrections in design. The systematic faults revealed are handled and the SIS is modified, as the consequences of not doing so can have great impact on the SIS. The IEC 61508 focuses on qualitative measures for handling systematic faults by using requirements and checklists. Documents and guidelines on how the standards should be adopted are provided in Lundteigen and Rausand (2006) and NOG 070 (2004). The two main methods used to minimize the number of faults in SIS are *fault avoidance/prevention* and *fault detection/removal*. These two methods apply measures such as; design reviews, use of redundant and diverse channels, use of simple design, use of competent designers, and training of designers.



## 1.2 Objectives

The main objective of this master thesis is to study, evaluate, and discuss to what extent RGT of a SIS is a suitable approach for identifying and avoiding systematic faults. If the conclusion is affirmative, another objective is to establish guidelines for RGT for this purpose and to identify challenges and pitfalls related to such testing. To achieve these objectives, the following sub-objectives are deduced:

1. Explain the meaning of systematic faults of a SIS
2. Present the classes of systematic faults
3. Describe the relationships between systematic faults and CCF
4. Present the types of systematic faults that can be introduced in the design and development phases of the SIS lifecycle
5. Explain the meaning of RGT
6. Present the models and methods available for RGT
7. Discuss the strengths and weaknesses of the various models and methods for RGT
8. Verify if reliability RGT can be used to identify systematic faults
9. Identify how such tests can be implemented
10. Present and discuss the challenges and pitfalls related to RGT in relation to systematic faults

## 1.3 Limitations

The focus of this thesis is on using RGT to reveal systematic faults in SISs that are introduced in the design and development phases. It does not include other phases in the SIS lifecycle. Moreover, random hardware failures are outside the scope of this thesis, as it is restricted to systematic faults.

Also, for the purpose of this thesis, RGT can be carried out by formal, semi-formal and informal methods. These methods are explained in details in this report.

## **1.4 Structure of the Report**

The rest of the report is structured as follows. Chapter 2 gives a brief introduction to SISs and Safety Integrity Level (SIL) based on IEC 61508 and IEC 61511. Chapter 3 presents overview of the overall safety lifecycle and product lifecycle, also activities performed in each phase of the product lifecycle to obtain high reliability are discussed. Chapter 4 discusses systematic faults, its importance and types of systematic faults that can be revealed in the design and development phase. Common-cause failures and its relationship with systematic faults are presented in chapter 5. Chapter 6 deals with RGT, its models and methods, the strengths and weaknesses of the models and methods. Chapter 7 discusses RGT for systematic faults using some illustrative examples. Finally, Chapter 8 provides concluding remarks and issues requiring further research.

# Chapter 2

## Introduction to safety-instrumented systems and safety integrity level

Most well-designed systems have protection equipment or other features to protect people, the environment, and other assets against harm should failures or dangerous deviations occur in the system (Rausand, 2011).

SIS have been used for many years to perform safety-instrumented function in the process industries (IEC 61508, 2010). The main purpose of a safety-instrumented system (SIS) is to provide the necessary protection by bringing the plant or equipment to a safe state if a hazardous event occurs. In other words, SISs are designed to respond to deviations in normal operation and maintain a safe state for the system.

They are used in many sectors of our society: for example, as emergency shutdown systems in hazardous chemical plants, fire and gas detection and alarm systems, pressure protection systems, and dynamic positioning systems for ships and offshore platforms (Rausand, 2011). SISs are subject to specific expectations about performance, like high reliability and short response time. There are some key properties that impact the performance of the SIS both in design and operation, therefore is need for laid down guidelines on how to operate and design such systems (Hoem, 2013).

Several standards have been issued setting requirements to the SIS (Rausand and Høyland, 2004). Almost every safety-related system has at least one electrical, electronic, or programmable electronic (E/E/PE) component. The International Electrotechnical Commission (IEC) pub-

lished the international standard IEC 61508 *Functional safety of electrical/ electronic/ programmable electronic (E/E/PE) safety-related systems*, with the last version from 2010. This is the main standard for the construction of safety-critical systems such as SIS worldwide.

The sizes of SIS applications range from the smallest High Integrity Pressure Protection System (HIPPS) containing just few signals up to large emergency shutdown (ESD) systems with several thousand input and output points. The challenge to anyone designing a complex system such as programmable electronic system is to determine how much rigour/ assurance/ confidence is necessary for the specified safety performance level (Bell, 2006). IEC 61508 has come up with two concepts, which are fundamental to its application, with the following basis:

- Safety integrity level: That it is possible to quantify the random hardware failures and therefore estimate whether the target failure measure has been achieved.
- Safety lifecycle: That it is not possible to quantify those elements giving rise to systematic failure behaviour.

## 2.1 Definition of Safety-Instrumented Systems

A SIS is an independent protection layer that is installed to mitigate the risk associated with the operation of a specified hazardous system, which is referred to as the Equipment Under Control (EUC) (Rausand and Høyland, 2004). The main purpose of SIS is to prevent hazardous event by putting the EUC in safe state. SIS is referred to as E/E/PE safety-related system in IEC 61508. In other industry sectors, safety related systems has their names such as Instrumentation and Control (I&C) in the nuclear industry, and safety-related electrical control systems for machinery. In IEC 61511: Part 1 - Framework, definitions, system, hardware and software requirements, this safety related system is defined as a safety-instrumented system:

☞ **Safety-instrumented system:** SIS is used to implement one or more safety-instrumented functions. A SIS is composed of any combinations of sensors, logic solvers and final elements (IEC 61511, 2003). This is shown in Figure 2.1.

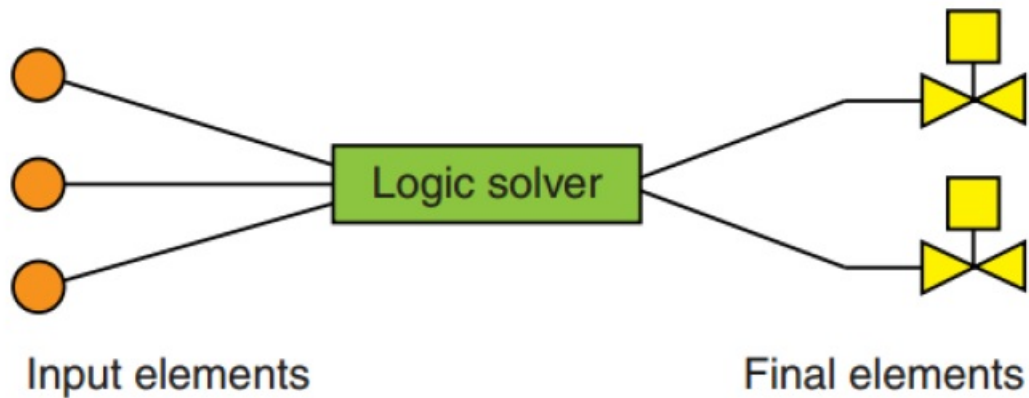


Figure 2.1: A simplified illustration of a SIS (Lundteigen, 2009).

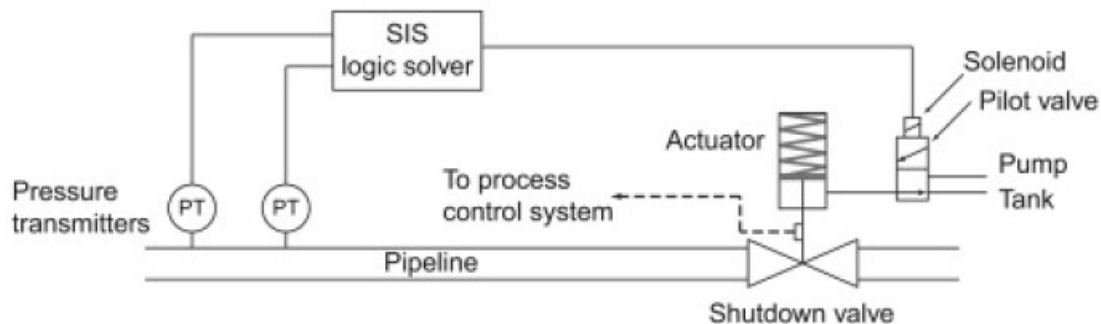


Figure 2.2: A simplified illustration of HIPPSI (Lundteigen and Rausand, 2009).

An example of a SIS can be a HIPPS which includes at least a single logic solver, a single shutdown valve as a final element, and two pressure transmitters giving input to the logic solver. This is illustrated in Figure 2.2.

## 2.2 Safety- Instrumented Function

A SIS is a physical system that performs one or more SIFs. The basis for the performance evaluation of the SIF is safety targets through hazard analysis and risk assessment (ISA-TR84.00.02, 2002). Safety-instrumented function (SIF) are in IEC 61511: Part 1 defined as;

☛ **Safety-instrumented function:** a safety function with a specified safety integrity level which

is necessary to achieve functional safety and which can be either a safety instrumented protection function or a safety-instrumented control function (IEC 61511, 2003).

A SIF may fail in two main failure modes. The modes are;

- *Fail-to-function mode*: When the specified deviation occurs in the EUC, and the SIS is not able perform the required SIF.
- *Spurious activation*: The SIF is performed without the presence of a predefined deviation (process demand) in the EUC. This failure mode is also known as *Spurious trip or False alarm* (Rausand, 2011).

Taking into account the HIPPS example given in the previous section, one of the SIFs can be, to fully close the shutdown valve (SDV) within 5 seconds after the pressure transmitters have detected overpressure in the pipeline. Highly reliable SISs like this one are often achieved by fail-safe design principles (e.g., the valves closes automatically upon loss of utility sytems) (Lundteigen and Rausand, 2009). A fail to fuction mode in this example can be if the SDV fully closes first after 7 seconds. While a spurious trip has occured if the SDV closes even though the pressure in the separator is within the safe limit because of a false alarm from the pressure transmitters. In other words, this particular SIF will not function as safety barrier if the valve closes too slowly. The failure mode is a description of fault(s) that caused the valve not to close within the time limit, and the criticality of the failure will obviously increase with the deviation from the target closing time.

It is important to realize that a failure mode is a manifestation of the failure as seen from outside, that is, the termination of one or more functions (Rausand and Høyland, 2004). Another example *Internal leakage* is a failure mode of a shutdown valve, since the valve loses its required function to "close flow". Wear of the valve seal, however, represents a cause of failure and hence not a failure mode of the valve. In this example, an investigation is needed to determine if the failure is a random hardware failure or a systematic failure. From the outside, one can easily discover that a valve has failed and also whether the failure is caused by wear and tear (random hardware failure) or by wrong installation (systematic failure). Failure classification will be discussed in details in the next chapter.

## 2.3 Safety Integrity Level

Safety integrity is a performance measure for safety functions. In IEC 61508, safety integrity is a fundamental concept, defined as:

☞ **Safety integrity:** probability of an E/E/PE safety related system (can be a SIS) satisfactorily performing the specific safety functions under all stated conditions within a stated period of time (IEC 61508, 2010).

It is the freedom of safety functions from flaws. IEC 61508 specifies the safety integrity requirements for each safety function in terms of either

- the average probability of a dangerous failure on demand of the function for a low demand mode of operation or
- the average frequency of a dangerous failure of the safety function for a high demand or a continuous demand mode of operation (IEC 61508, 2010)

These frequencies are defined as probability of failure on demand (PFD) or probability of failure per hour (PFH). "Failure on demand" here means failure likely to be observed when a demand occurs and should not be mixed up with the probability of a failure due to a demand. The frequencies are calculated from failure rates of dangerous undetected failures. It should be noted that probability of failure on demand are only accounting for random hardware failure in IEC 61508 and IEC 61511.

*Safety Integrity Level* (SIL) is an indication of the required level of protection against failure i.e., integrity level indicates the degree to which a component must be free from flaws. The problem is that random failures can be designed against while systematic failures are not known in advance and any further design may introduce further errors. In the standards, four SILs are given, and presented in Table 4.1, where SIL 1 is the lowest and least reliable, and SIL 4 is the highest and most reliable. The principle of SIL is that it should be allocated early in the lifecycle to systems, functions and components. The allocated integrity levels then define appropriate degree of rigour and scrutiny applied in the development. Intuition is that better development techniques will result in inclusion of fewer systematic flaws.

Table 2.1: SIL table (IEC 61511, 2003)

SIL	Low-demand (PFDavg)	High-demand (PFH)
4	$\geq 10^{-5} to < 10^{-4}$	$\geq 10^{-9} to < 10^{-8}$
3	$\geq 10^{-4} to < 10^{-3}$	$\geq 10^{-8} to < 10^{-7}$
2	$\geq 10^{-3} to < 10^{-2}$	$\geq 10^{-7} to < 10^{-6}$
1	$\geq 10^{-2} to < 10^{-1}$	$\geq 10^{-6} to < 10^{-5}$

This specified target ranges for the PFD or the PFH is the quantitative requirements for the hardware of a SIS (Lundteigen and Rausand, 2009). In addition, the SIL outlines the requirements for architectural constraints. In addition, the SIL outlines the requirements for architectural constraints and together with the systematic safety integrity constitutes the qualitative reliability the SIF should comply with. IEC 61508 distinguish between three categories of safety integrity:

- Hardware safety integrity
- Systematic safety integrity
- Software integrity

Software integrity is relating to systematic failures in a dangerous mode of failure that are attributable to software (IEC 61508, 2010). To meet a specified SIL requirement, it is necessary to demonstrate that all parts achieve the specified SIL. If, for example, it is confirmed that a SIF meets SIL 2 in terms of hardware safety integrity, we cannot claim compliance to this SIL unless the systematic and software integrity also meets SIL 2 (Lundteigen, 2009).

## 2.4 Determination of Safety Integrity Level

There are several methods used to determine or assign a SIL and there is no one method suitable for all applications. Any method adopted should put into consideration factors that influences method of determining SIL. It is shown by IEC 61508 (2010) that the following methods listed below are used. Reference should be made to the standards for more details.

- ALARP method (as low as reasonably practicable)



- Quantitative method of SIL determination
- Risk graph method
- Layers Of Protection Analysis (LOPA) method

To choose an appropriate method to assign SIL for a safety instrumented function, the factors below should be considered as stated in ([IEC 61508, 2010](#)):

1. The risk acceptance criteria.
2. Mode of operation of the safety function.
3. Confidence that the resulting residual risk meets specified criteria.
4. Severity of consequences.
5. Occurrence of common cause failures and demand causes.
6. Use of more than one method to determine SIL.
7. Knowledge and experience of persons undertaking the SIL determination.
8. The existing approach used for SIL determination.

# Chapter 3

## Introduction to Safety Life Cycle and Product Life Cycle

### 3.1 Introduction

A SIS is installed on the *equipment under control* (EUC) to perform the designated safety function. An EUC can be any equipment (e.g., machinery, apparatus or plant used for manufacturing, processing, transportation and so on) that the SIS is installed for. Several qualitative and quantitative aspects should be taken into account while implementing a SIS. These aspects normally encompass the whole life span of the SIS, i.e., each relevant activity from manufacturing, through use, to discarding. Accordingly, it has been shown by (IEC 61508, 2010) and (IEC 61511, 2003) the relevant requirements through the overall safety life cycle (SLC). This chapter discusses the main issues in the overall SLC as established in IEC 61508. The overall SLC is a fundamental concept of IEC 61508 because the sum total of all the technical and non-technical requirements of a SIS during design, development, operation and maintenance to decommissioning are addressed using the overall SLC as a framework. This general framework creates common understanding between parties involved (e.g., suppliers, system integrators, operators, consultants, regulatory bodies and so on) in the life span of the SIS. That is, the requirements in the SLC serve as common references for parties who are taking part in any phase (Jigar, 2013).

## 3.2 Overall Safety Life Cycle

The overall SLC is the foundation of the requirements in IEC 61508. All requirements and parts of the standard are directly linked to this fundamental concept. Broadly, the purpose is to demonstrate the development and documentation of a safety plan and its execution until de-commissioning. It enables us to approach all activities during the whole life span of the SIS in a systematic manner so that the required safety performance can be achieved.

## 3.3 Objectives

Key objectives of SLC are highlighted below.

1. Provide optimal design of SIS and other safety related systems that matches the risk reduction with process risk.
2. Ensure consistent, cost-effective design of SIS and other safety related systems while reducing error by verifying procedures.
3. Reduce likelihood of immature failures, minimize maintenance by selection of proper technology and correct specification of equipments.

## 3.4 Safety Life Cycle and Standards

SLC is a framework for implementing and managing safety systems throughout its lifecycle. This framework has enjoyed wide acceptance and thus have been adopted by many recognized standards. Some of the notable standards that use the concept of SLC are IEC 61508, IEC 61511, and ANSI/ISA-S84-01-1996 (replaced by ANSI/ISA-84.00.01-2004). ANSI/ISA-S84-01-1996 being the first to introduce the concept of SLC ([Ali, 2007](#)).

### 3.4.1 IEC 61508 and Safety Life Cycle Classification

The overall SLC model can be classified into three phases namely; analysis, realization and operation.

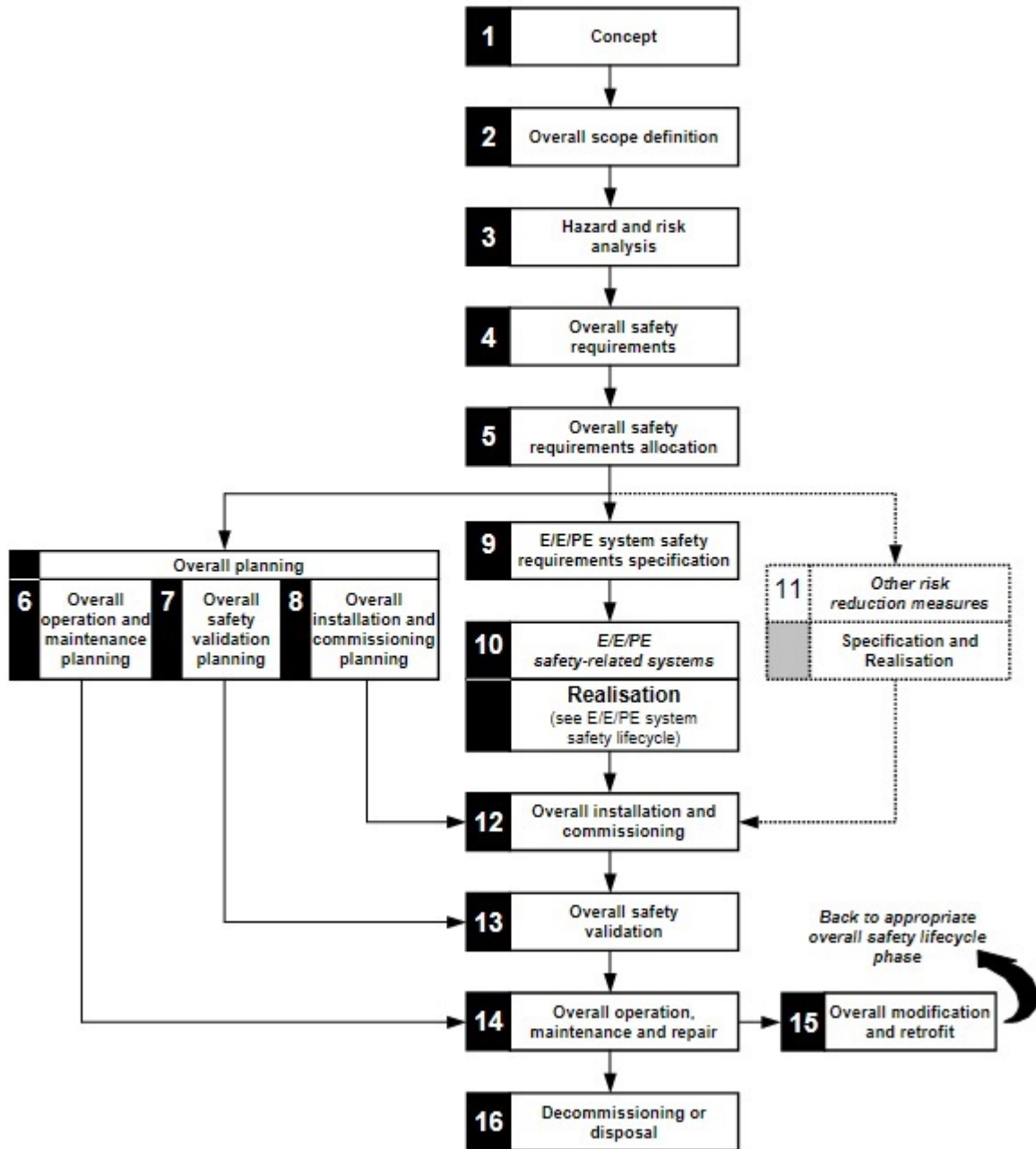


Figure 3.1: Overall Safety Life Cycle  
(IEC 61508, 2010)

1. *The Analysis phase* This part takes into account identification of hazards and hazardous events, the probability of these hazardous events occurring and their potential consequences when they occur. This phase also considers the layer of protection available to prevent and mitigate the hazard or hazardous event and the need to implement a SIS to prevent it. A document called Safety Requirements Specification (SRS) should be established after allocating the overall safety integrity requirements to the SIS.
2. *The Realization phase* This phase deals with the design and fabrication of E/E/PE safety related systems. It also includes other risk reduction systems or facilities. This thesis report deals majorly with this phase. This phase can be further classified into three: pre-design and development, design and development, and post-design and development.

### **Pre-Design and Development**

Realization begins with describing the design/architecture of the SIS under the so-called *SIS design requirements specification*. It is primarily derived from the SRS, but in this case design requirements are specified at subsystem, element and/or component level. The specification should contain detailed descriptions in terms of hardware and software of the SIS. Since it is the foundation for the realization of the SIS, great care should be taken to be able to demonstrate in detail how the requirements in the SRS are fulfilled. In addition, appropriate methods should be implemented to avoid some misspecification problems. The development of SIS design requirements specification is an iterative process where it is updated over time and becomes more mature as the design progresses. Like any other activity in the SLC, the document needs to be comprehensive and traceable for those who may use it in any activity in the SLC.

### **Design and Development**

This phase is normally performed parallel with *safety validation planning* (box 7 in Figure 3.1). This is a plan containing proper procedures (activities) to be implemented to validate whether or not the SIS performs the safety function as desired, i.e., a plan to demon-

strate, before commissioning, that the SIS satisfies all the requirements in the SRS and SIS design requirements specification. For example, the plan may include the environment under which the test is to take place, the pass/fail criteria (policy), words on how to deal with the results and other related issues. Design and development is a broad, time consuming and resource-intensive part of the overall SLC. The design, hardware or software, of a SIS shall meet all requirements related to

- (a) Hardware safety integrity
- (b) Systematic safety integrity
- (c) Architecture of integrity circuits (ICs) with one chip redundancy
- (d) System behaviour on detection of a fault
- (e) Data communication processes

*Systematic failures* During design and development of hardware and software, necessary attentions should be given to avoid systematic failures. Often due their nature, it is difficult to avoid or control such failures. There are always some residuals left in the system, especially the ones related to operational failures. However, it is possible to reduce their effects up to some extent by exploiting appropriate methods, procedures and/or documentation. It is also necessary to make sure that the design could not cause other safety system(s) to fail. Analysis should be conducted to determine this, and if it causes failure, the design has to be changed (preferred) or the likelihood has to be reduced. This masters thesis report shall be dealing mainly on how to reveal this systematic faults especially in the design and development phase. Systematic faults shall be extensively discussed in the next chapter.

## **Post-Design and Development**

Once the design and development of the SIS is completed, components/ elements/ sub-systems need to be integrated correctly to have a complete SIS. It shall then be tested to see whether they interact each other as intended or not. It should also be tested to see whether or not it performs only its intended function. If tests reveal any unpleasant

result, design change or modification is required and the procedures should again obey all the requirements. Whenever changes are made, version numbers should be written clearly on the document. It is part of the realization phase to provide procedures that will be implemented during operation and maintenance activities so as to maintain the achieved performance. It may consist issues related to the proper actions that need to be done by the operators (e.g., during start up, fault condition, shut down), maintenance procedures (e.g., fault diagnosis, repair, revalidation), documentation of system failure and component failure data, and so on. SIS validation can be considered as the final activity of the realization phase. This has to be done according to the plan developed in the beginning of the phase. It encompasses all aspects of the requirements. The integrated system needs to be tested for validity against the requirements in the SRS and SIS design requirements specification as well as the procedures developed to be implemented during operation and maintenance. If validation activity is not satisfactory, appropriate modifications, corrections or enhancements should be done. After these activities the system should be revalidated, reverified, and documented with another version. In parallel with the realization phase, it has suggested in the standard to make an *overall* plan in relation to

- Operation and maintenance
- Safety validation
- Installation and commissioning

3. *The Operation phase* This is the final part and covers startup, operation, maintenance and decommissioning of the E/E/PE safety related systems.

To implement the standard properly, from the onset the organization should appoint one or more persons who take care of one or more of the activities in the overall SLC. Persons need to be competent and knowledgeable for the activity they are assigned (see clause 6 of IEC 61508-1).

## 3.5 Product Life and Product Life Cycles

### 3.5.1 Product Life

The useful life of a product is the age beyond which the product is deemed to be unsuitable for further use due to its inability to perform satisfactorily. This is a random variable due to variation in manufacturing and/or usage. For a repairable product, a component can fail several times over its useful life and is restored to operational status through corrective actions (Murthy et al., 2008).

Considering new products, a related concept is the time for which a consumer uses the purchased product before it is replaced by a new one. This is called *period of ownership*. This is also a random variable as different consumers keep the purchased product for different lengths. If consumers keep the products for the useful life, then the products are scrapped at the end of their useful life. In this case, there are no second-hand products. If the period of ownership is shorter than the useful life, a market for second-hand products is created.

From the *consumer's perspective*, the product life cycle is the time from the purchase of an item to its discarding when it reaches the end of its useful life or being replaced earlier due to obsolete technology or the item being no longer in vogue. The life cycle here is divided into three phases:

- Acquisition
- Operation and maintenance
- Discard (and leading to replacement by new one)

From the *manufacturer's perspective*, the concept of *marketing* and *manufacturing* arises.

- *Marketing*: The product life cycle is the period from the instant the product is launched on the market to the time when it is withdrawn from the market and involves the following four phases:

1. Introduction. This phase is characterised with low sales
2. Growth. This phase has a rapid increase in sales





Figure 3.2: Phases of product life cycle

Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6	Phase 7	Phase 8
Pre-development			Development		Post-development		
Front-end	Design		Development		Production	Post-production	

Figure 3.3: Product life cycle model  
(Murthy et al., 2008).

3. Maturity. This phase has a near constant sales
  4. Decline. The phase is characterised with decreasing sales
- *Manufacturing*: The product life cycle is the period from the initial conception of the product to the final withdrawal of the product from the marketplace. It is divided into five phases as shown in Figure 3.2.

### 3.6 RAMS-Oriented Product Life Cycle Model

Even though many new product development models discuss product performance and specifications, no model have addressed this in an effective manner. Murthy et al. (2008) proposes a new model that is more appropriate for making decisions relating to performance and specification in new product development. It is closely linked to the product life cycle model and involves three stages and three levels as shown in Figure 3.3. The three stages are as follows:

*Stage I (Pre-development/Specification)*: This stage is concerned with a non-physical (or abstract) conceptualization of the product with increasing level of detail.

*Stage II (Development/Verification)*: This stage checks whether the prototype product can satisfy the requirement.

*Stage III (Post-development/Substantiation)*: This stage is concerned with the remainder of the product life cycle (e.g., production, sale, use) subsequent to the new product development.

The three levels are as follows:

*Level I (Business level)*: This level is concerned with the linking of business objectives for a new product to the desired product attributes.

*Level II (System, i.e., product level)*: This level is concerned with linking the product attributes to product characteristics.

*Level III (Component level)*: This level is concerned with linking product characteristics to lower level product characteristics, at an increasing detail.

The RAMS-oriented model has eight phases as shown in Figure 3.3:

**Phase 1 (Stage-I, Level-I)**: In this phase the need for a new product is identified and the decision related to the product attributes (customer's view of the product) are made from the top management level of the business.

**Phase 2 (Stage-I, Level-II)**: In this phase the product attributes are translated into product characteristics (engineer's view of the product).

**Phase 3 (Stage-I, Level-III)**: In this phase the detailed design (proceeding from product to component) of the product is carried out in order to arrive at a set of specifications that will ensure that the product has required characteristics.

**Phase 4 (Stage-II, Level-I)**: This phase deals with the product development/verification, from component to product - and ends up with the product prototype.

**Phase 5 (Stage-II, Level-II)**: In this phase the prototype is released to a limited number of consumers to evaluate the customers' assessment of the product features.

**Phase 6 (Stage-III, Level-III)**: This phase deals with the production of products starting from component and ending with the product for release to customers.

**Phase 7 (Stage-III, Level-II)**: This phase looks at field performance of the product taking into account the variability in usage intensity, operating environment, and so on, from the customers perspective.

The link between the phases of the RAMS-oriented model and the product life cycle phases is illustrated in Figure 4.2

This thesis report will concentrate mainly on phases 3, 4 and 5.

Phase 1	Phase 2	Phase 3	Phase 4	Phase 5	Phase 6	Phase 7	Phase 8
Pre-development			Development		Post-development		
Front-end	Design		Development		Production	Post-production	

Figure 3.4: Relationship between RAMS-oriented model and Product life cycle model (Murthy et al., 2008).

### 3.7 Activities Performed in each Phase of the Product Life Cycle to Obtain High Reliability

The motivation of the manufacturer to integrate reliability into the product life cycle can be attributed to the conflicting interests of the manufacturer and the consumer. The manufacturer wants the product realisation process to be minimized with respect to time and cost, conversely, the consumer expects the products to operate reliably and to incur little or no maintenance costs.

☛ Reliability is defined as the probability that a product performs its intended functions without failure under specified conditions for a specified period of time.

The aim of reliability engineering is to ensure that a product is operated without failure, which in reality is not feasible. Reliability engineering is implemented by taking structured and feasible actions that maximize reliability and minimize the effects of failures. To accomplish this objective, the steps taken according to Yang (2007) are:

- Build maximum reliability into a product during the design and development stage.
- Minimize production process variation to assure that the process does not appreciably degrade the inherent reliability.
- Initiate appropriate maintenance operations to alleviate performance degradation and prolong product life for a product that have been deployed to the market.

The following reliability techniques are employed in the three steps: *reliability planning and specification, allocation, prediction, robust reliability design, failure mode effects and criticality analysis (FMECA), fault tree analysis (FTA), accelerated life testing, degradation testing, reliability verification testing, stress screening, and warranty analysis*. The reliability tasks are adapted to fit the needs of specific products and enforced throughout the product life cycle. They add value to the products in the product realization process.

In the product planning phase (i.e., **Phase 1**), a team of reliability experts is formed whose duty is to set a reliability target that will translate customer expectations into engineering requirements. *Functional analysis and quality function deployment (QFD)* is done at this stage where functions of product/system is identified. The functions to be involved in RAMS analysis should also be decided at this stage. *Reliability history analysis* where customer feedback, test data, and warranty failure data of the prior-generation product are also collected and analyzed at this phase. The analysis should indicate whether customer wants were not reasonably satisfied and reveal areas to be improved upon. The reliability decisions made in this phase will have a great impact in the other stages of the product life cycle. For instance, setting a reliability target has strong effect on cost, time to market, and competitiveness (Yang, 2007). An immensely ambitious target would lead to a high design and development cost and prolong the product realization process and losses may be incurred. Conversely, a low reliability target will undermine competitiveness by losing customers.

In **Phase 2**, *reliability planning and specification* with the objective of establishing a reliability target that is economically achievable and develop an effective reliability program to reach or exceed the target. Results from QFD and reliability history analysis are used at this stage. Also *reliability modeling* where product reliability are modelled according to the structure of the product which may be in series, parallel or more complex configurations. Product reliability is expressed as a function of component reliabilities. This relationship is useful in reliability allocation, prediction, and analysis (Yiliu).

The objective of reliability tasks carried out in **Phase 3** is to design-in the reliability of the product while designing-out potential failure modes. The tasks include *reliability allocation*; where the established reliability targets are equally distributed to lower-level structures (subsystems, modules, or components) of the product. The reliabilities allocated to a structure be-

comes the reliability target of that structure. *Reliability prediction*; In the early design stage, predicted reliabilities are used for comparing design alternatives and components, identifying potential design issues, determining if a design meets the allocated reliability target, and projecting reliability in the field. *Robust design*; This is used to build robustness and reliability into the products in the design stage through the implementation of a three-stage process of concept design, parameter design and tolerance design. *Failure Modes, Effects and Criticality Analysis (FMECA)*; This is done to reveal potential failure modes, analyze effects and determine causes of failure. This process is intended to detect design errors that have been built into a design so that corrective actions can be carried out.

In **Phase 4, reliability testing**; this is done in the early design stage for the purpose of comparing design options, uncovering failure modes, estimating reliability, and verifying a design. Testing a product to failure at normal operating condition is not economically feasible, therefore, accelerated life tests at higher stress levels, which shorten test time and reduce test costs are conducted. *Reliability growth testing*; this is performed to assess current reliability, identify and eliminate faults, and forecast future reliability.

In **Phase 5, reliability improvement** is carried out to improve the product when the results of the tests are not good so as to meet reliability requirements and safety standards. Reliability growth analysis is performed to achieve product reliability targets. Reliability growth analysis consist of improving products whenever failure shows up during testing, this is known as test-fix test approach, or after the test, called the test-find test approach.

In **Phase 6, production control and assurance** is done to assure that the process results in the manufacture of uniform and reliable products. To maintain a stable process over time, process control plans and charts are implemented to monitor the process and help identify special causes as soon as they emerge. *Stress screening* is done to reveal latent defects due to material flaws, process variation or inadequate design. Defective products will fail in early service time and should be eliminated before the product is shipped to customers.

In **Phase 7, warranty and maintenance plan development**, is carried out when the product is ready for marketing. The warranty repair cost may be estimated from warranty repair modeling or reliability prediction.

In **Phase 8, field failure tracking** intended to collect failure information from warranty re-

pairs and customer complaints are carried out. Also *warranty data analysis* done to estimate field reliability, project warranty repair numbers and costs, monitor field failure modes and patterns. Early detection of unusual failure modes and high failure probability can promote corrective actions to change ongoing manufacturing process and repair strategies. *Total cost and profit analysis*; After a certain period of delivery of the products, information about selling, warranty, customer satisfaction can be collected. It is proper to evaluate the product performance for the target of the company, and then generate new ideas for new products.

# Chapter 4

## Systematic Faults/Failure Classification

The importance of failure classification in the lifecycle assessment of a SIS cannot be over-emphasized. The types of failures and their potential effects on the SIS help us to fully understand the lifecycle requirements. The ambiguous separation between a random failure and systematic failure is one of the reasons for the concept of systematic failures to be contested.

### 4.1 Failure or Fault?

This masters thesis report deals with systematic faults but the expression used in the reliability framework of SIS and mostly in the standards is systematic failures. This section tries to explain the relationship between a failure and a fault, and why there is no consistency in the use of only one expression.

☞ **Failure** is the termination of the ability of an item to perform a required function.

It is important to note that after a *failure*, the item has a *fault*. Failure is an *event*, as distinguished from fault, which is a *state*. This concept does not apply to items consisting of software only.

☞ **Fault** is the state of an item characterized by its inability to perform a required function, excluding the inability during preventive maintenance or other planned actions, or due to lack

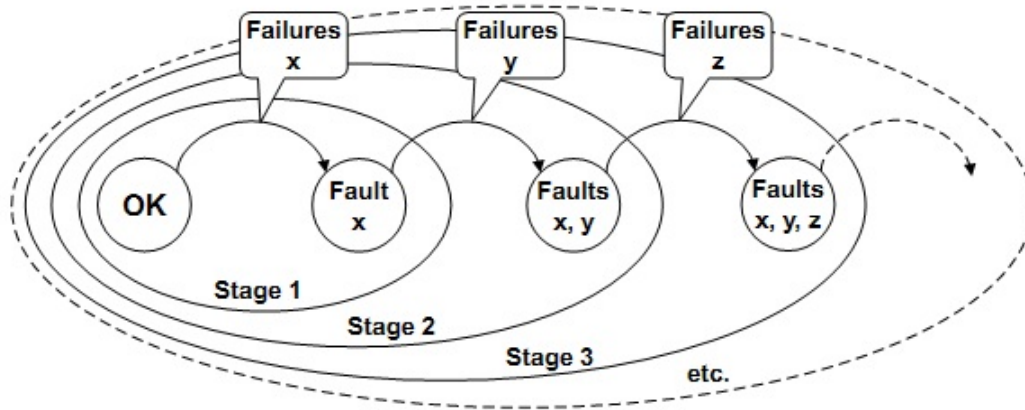


Figure 4.1: Relationship between failure and fault concepts (ISO/TR 12489, 2013).

of external resources.

A fault is often the result of a failure of the item itself, but may exist without prior failure. From the above definitions, it can be said that a fault may not give rise to a failure, but that all failures result in faults. This implies that a systematic failure is a systematic fault. A fault of an item may result from a failure of the item or from a deficiency in an earlier stage of the lifecycle, such as specification, design, manufacture, or maintenance. The relationship between the concept of failure and fault is illustrated in Figure 4.1.

The *Failure x* occurs at stage 1 and leads to the state *Fault x* which is not detected. From stage 2 point of view *Fault x* is a pre-existing fault. The *Failure y* occurs at stage 2 and lead to the state *Faults x,y* which is not detected. From stage 3 point of view *Fault x,y* is a pre-existing fault. The process continues in that order.

## 4.2 Systematic Failures versus Systemic Failures

Systematic failures are failures occurring in a deterministic way when given conditions are encountered. It should not be confused with systemic failures. Systemic failures (holistic failure) are failures at system level which cannot be simply described from the individual component failures of the system (ISO/TR 12489, 2013).

Systemic/holistic principles have been concisely summarized by Aristotle by "The whole is



more than the sum of its parts". Components only have failure modes. Those failure modes become dangerous, safe or spurious only when the components are implemented into a safety "system". This is why dangerous, safe or spurious failures are typical systemic failures. For example, the failure "fail to close" of a valve is dangerous only if it belongs to a safety system closing this valve on demand. Otherwise this failure mode does not matter.

### 4.3 Systematic Failure versus Random Hardware Failure

It has been shown by [IEC 61508 \(2010\)](#), [IEC 61511 \(2003\)](#), [Hauge et al. \(2010\)](#) and [Hauge et al. \(2013\)](#) that failures can be categorised based on the failure cause. The two classes are: *random hardware failure* and *systematic failure*. This is illustrated in [Figure 4.2](#).

**Random hardware failures** are failures due to physical causes and only apply to the simple hardware components within a system. This type of failure is caused by effects such as corrosion, thermal stressing, and wear-out. Due to their random nature, statistical information can be produced from testing and historical data about this type of fault. Thus, the average probability, and hence the risk, associated with the occurrence of a random fault can be calculated.

ISA-TR84.00.01 in line with the concept of [Table 4.1](#), defines fault as a state resulting in two types of events:

☛ **Faults** can result in two types of failures:

- Random failures, a spontaneous failure of a component.
- Systematic failures (or errors), a hidden fault in design or implementation ([ISA-TR 84.00.04](#))

The difference between random and systematic failure is summarized in [Table 4.1](#) as reproduced from Annex G in ISA-TR84.00.04.

**Systematic failures** as defined in PDS are failures related to a particular cause other than natural degradation. They are due to human errors during specification, design, operation, maintenance and decommissioning phases of the lifecycle. The scope of the report is only on the design and development stage.

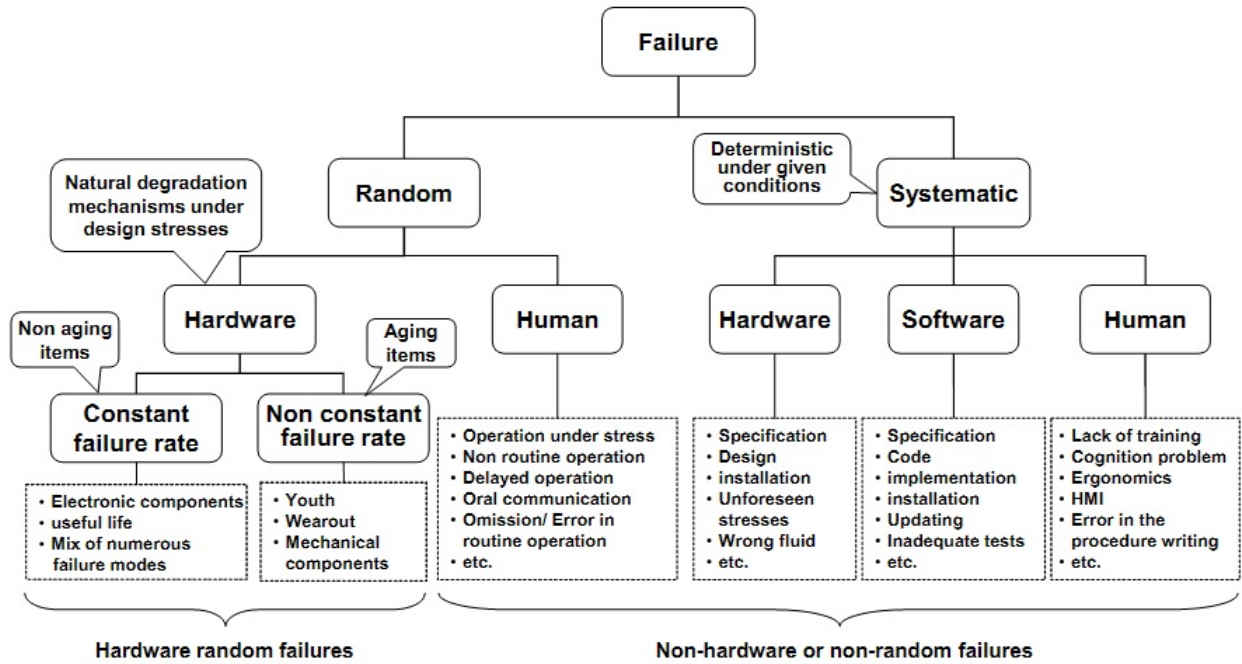


Figure 4.2: Failure classification by cause (ISO/TR 12489, 2013).

Table 4.1: Summary of the important differences between random and systematic failures (ISA-TR 84.00.04)

	Random failures	Systematic failures
Will always occur under the same conditions	No	Yes
Effectively prevented by redundancy	Yes	No
Effectively prevented by diversity in redundancy	Yes	Partially

### 4.3.1 Definition of Systematic Faults by Standards and Guidelines

IEC 61508 is the most important standard for design, implementation and maintenance of reliable SISs. The main feature that distinguishes the functional safety standards from other safety related standards is that, functional safety standards are performance based whereas many other standards are prescriptive standards (Jin). This includes work process, procedures, and tools for the user to decide, based on the desired safety integrity level, how the SIS should be designed, implemented, operated and maintained.

Most international and national standards concerning functional safety have similar definitions for SIS, SIF, EUC, common cause failures (CCF) and other terms within the framework. ANSI/ISA-84-01 and IEC 61508/61511 also refer to each other.

When it comes to failure classification and handling of these failures, there are different methods and guidelines. A suitable standard should provide guidelines to help the user to understand the correct meaning of the definition, get a sufficient knowledge of the principles and framework, and in this case understand what a systematic fault is, how to classify them, the relationship between systematic faults and CCFs and the types of systematic faults that can be introduced in the design and development phases of the life cycle of the SIS.

#### IEC 61508 and IEC 61511

The definitions of a systematic failure and a fault given in IEC 61508-4 and in IEC 61511-1 are identical:

☛ **Systematic failure:** failure related in a deterministic way to a certain cause, which can only be eliminated by a modification of the design or of other relevant factors (IEC 61508, 2010; IEC 61511, 2003).

#### The ISA-TR84.00.04 (2005 edition)

This technical report with the title *Guidelines for the implementation of ANSI/ISA-84.00.01-2004* deals with SIS for the process industries and implementation of IEC 61511. The technical report includes 16 informative annexes providing guidance from the ISA-SP84 committee on a wide

range of topics, where some are related to failure classification and handling of systematic faults. It defines systematic failures in this way:

☞ **Systematic failures** are due to mistakes or errors made in the SIF design and management and cause the SIF to fail every time a particular set of conditions occurs.

This definition is similar to other previous definitions but was presented in more clear terms. Three types of errors that can lead to systematic failures are presented in the standard as discussed below:

☞ **Specification errors:** mistakes and omissions made during the design process, such as incorrect actuator sizing or inappropriate materials of construction selection. These errors exist from the point of installation and remain throughout the SIS's life.

☞ **Equipment errors** such as a bad installation, improper bypassing, and poor maintenance, may occur at any stage in the safety life cycle.

☞ **Software errors:** may arise from errors in the initial programming or be introduced when the software is modified, intentionally or unintentionally.

The type of errors and the examples given, demonstrate the high degree of difficulty and impossibility of quantifying or predicting how often systematic failure leads to SIF failure.

#### **ISO/TR 12489 (2013 edition)**

This technical report *Petroleum, petrochemical and natural gas industries - Reliability modelling and calculation of safety systems*, sees systematic failures as failures that consistently occur under particular conditions of handling, storage, or use. The cause of a systematic failure originates in the specification, design, manufacture, installation, operation or maintenance. Its occurrence is precipitated by particular conditions of handling, storage, use or maintenance. Corrective maintenance without modification will usually not eliminate the failure cause. A sys-

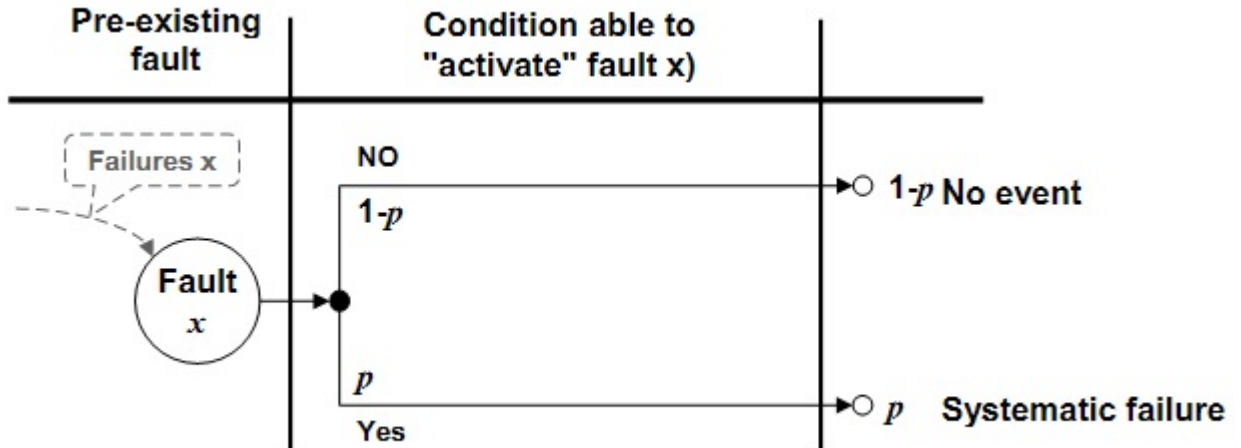


Figure 4.3: Illustration of systematic failure (ISO/TR 12489, 2013).

tematic failure can be reproduced by deliberately applying the same conditions, e.g., in verifying the failure cause. Systematic failures are non-random failures. In operation, a systematic failure is a manifestation of a systematic fault (i.e., a pre-existing state of the system). The software systematic failures, called "bugs" are examples of systematic failures: they are due to pre-existing bugs (i.e., faults) and they occur when the input data activate them. Systematic failures are in this standard considered a kind of dependent failures. Depending on the safety of the system under study, the related cause may never, always or randomly happen (e.g., probability  $p$  in Figure 4.3) (ISO/TR 12489, 2013). The three cases presented in the standard address systematic faults in connection with CCFs, which will be discussed in the next chapter.

### PDS Method Handbook (2003 edition)

The PDS<sup>1</sup> method is used to quantify safety unavailability and loss of production for SIS. The handbook has adopted the classification from IEC 61508/61511, and divides failures into *random hardware* and *systematic* failures. However, the method accounts for all relevant failure categories but also gives a detailed breakdown of systematic failures as shown in Figure 4.4.,

■ Systematic failure are failures that can be related to a particular cause other than natural

<sup>1</sup>PDS is the Norwegian acronym for "reliability of computer-based safety systems"

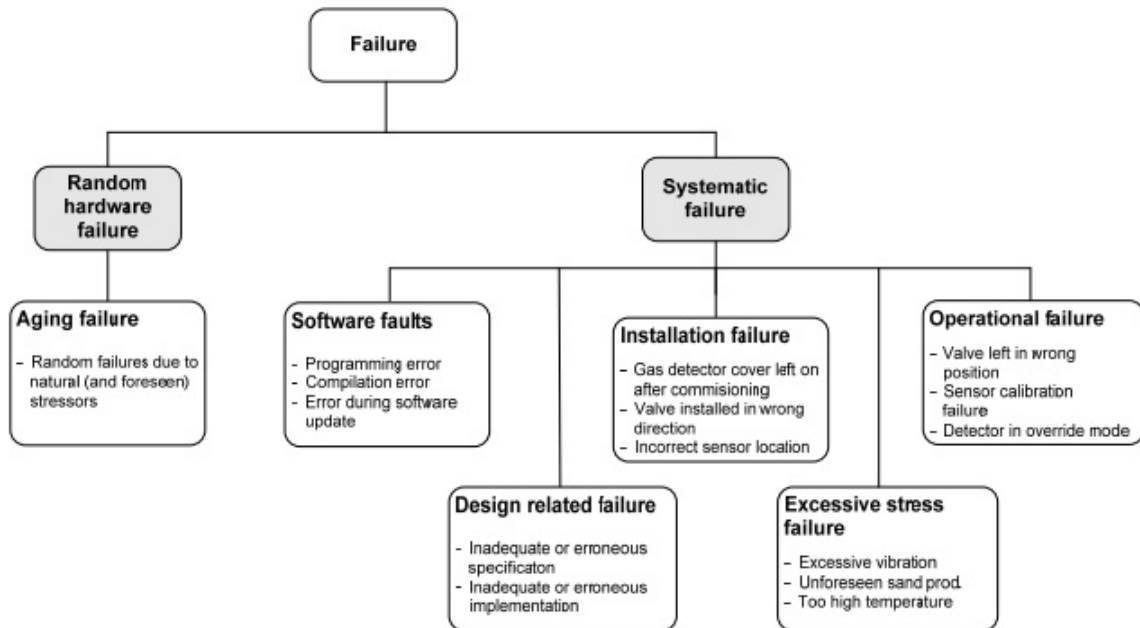


Figure 4.4: Possible failure classification by cause of failure (Hauge et al., 2013).

degradation. Systematic failures are due to errors made during specification, design, operation and maintenance phases of the lifecycle. Such failures can therefore normally be eliminated by a modification, either of the design or manufacturing process, the testing and operating procedures, the training of personnel or changes to procedures and/or work practices.

Systematic failures are further split into:

- **Software** faults can be caused by programming errors, compilation errors, inadequate testing, unforeseen application conditions, change of system parameters, etc. Such faults are present from the point where the error in code is developed until the fault is detected either through testing or through improper operation of safety function. How this fault can be revealed through reliability growth testing is discussed in chapter 7.
- **Hardware related** systematic failures are failures (other than software faults) introduced not only during the design phase of the equipment but also during modifications/repairs.
- **Installation** failures are failures introduced during the last phases prior to operation, i.e., during

installation or commissioning. If detected, such failures are typically removed during the first months of operation and such failures are therefore often excluded from data bases.

- **Excessive stress** failures are caused by overstressing the component beyond its design specification. This may be caused by either internal or external influences to the medium.
- **Operational** failures are caused by human errors during operation, testing, maintenance or repairs.

#### **NOG 070 guideline (2004 edition)**

The Norwegian Oil and Gas Association (NOG, formerly known as OLF) has developed a guideline, NOG 070, to support the implementation of the two IEC standard series, IEC 61508 and IEC 61511. In the regulations from the Norwegian Petroleum Safety Authority, specific references are given to the IEC standards and the NOG 070 guideline. Nevertheless, NOG 070 recommends the use of the PDS method from the previous section for the quantification of loss of safety.

In line with other definitions given in this report, this guideline defines systematic faults as:

☛ **Systematic faults** are faults in hardware and software introduced during specification, design, operation or maintenance/testing, which may result in a failure of the safety function under certain conditions (e.g., for particular input signals states) ([NOG 070, 2004](#)).

#### **4.3.2 Importance of Systematic Faults**

Systematic failures/errors are a major source of CCFs, because of the potential to disable redundant devices which dominate the reliability of SIS ([Gentile and Summers, 2006](#)).

All software faults are systematic, thus demonstrating the safety of software relies upon assessing the likelihood of this type of fault. Software within safety critical systems is increasing in size and extent of use, and the result of this is that risks associated with software systematic faults becoming more prevalent. The level of authority given to and complexity associated with software within safety critical applications means that it is of great importance to assess and argue about the effects of software with respect to system safety. It is not possible to use statis-

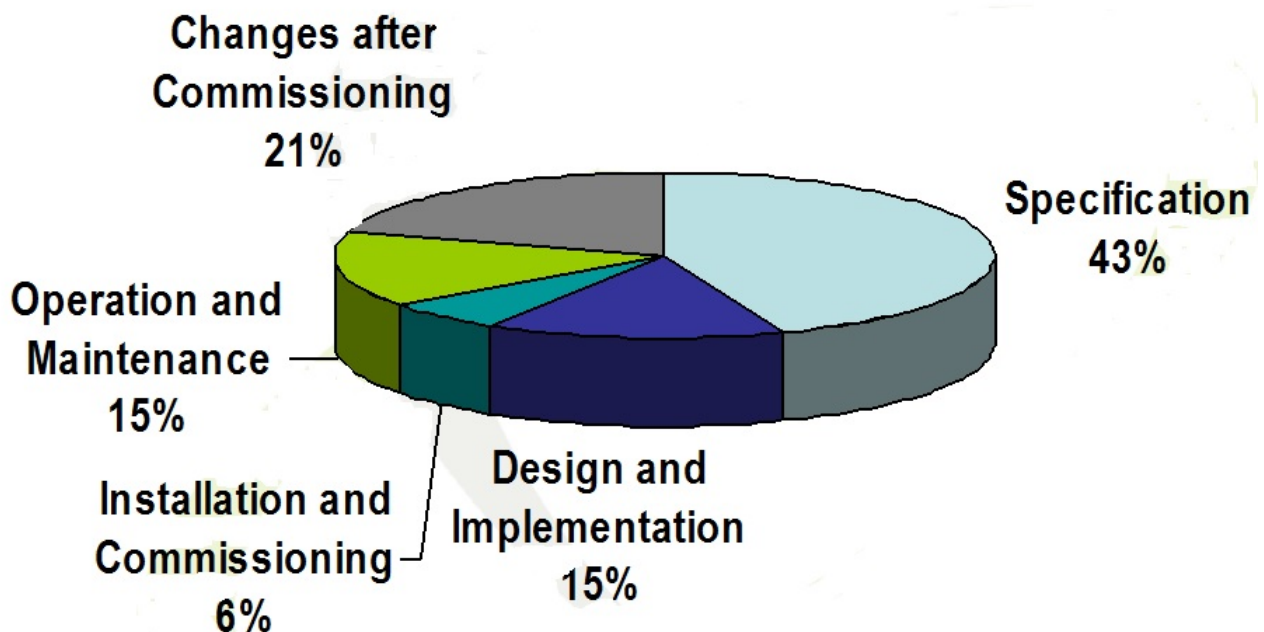


Figure 4.5: Failure cause of control system by lifecycle phase (HSE, 2003).

tical analysis to predict the probability of systematic faults, thus for software it is not possible to quantify the risks.

### 4.3.3 Types of Systematic Faults that can be Revealed in the Design and Development Phase

The Health and Safety Executive (HSE, 2003) classifies the causes of failure of control systems by lifecycle phase and represent it on a pie chart as shown in Figure 4.5. It is acknowledged that because of the small sample size the results of the analysis have low statistical significance, and therefore care needs to be taken in using these results to generalise for all control system failures. Even so, there are many useful lessons to be learned from summaries of incidents such as these.

The analysis suggests that most control system failures may have their root causes due to inadequate specifications. In some cases this was because insufficient hazard analysis of the EUC had been carried out; in others it was because the impact on the specification of a critical failure



mode of the control system had not been assessed. The control system needs to be continually reviewed throughout all lifecycle phases, both from the perspective of the EUC and the detailed design and implementation of the control system itself. Otherwise the end result is a machine, or plant, with inadequate protection against the hazard. Other studies provide support for these conclusions. In the area of software development a number of studies have shown that errors made during specification account for most software faults and failures (HSE, 2003).

Figure 4.5 shows that over 50% of the faults occur within the specification, design and development phases which is the main focus of this thesis. Types of systematic faults that can be revealed in these phases of the lifecycle includes:

### **Systematic software faults**

Software faults are systematic faults. The causes of software failures can either be error of omission or an error of commission. Systematic faults are mainly due to human error. An ***Error of omission*** is an error that results from something that was not done. This includes (Son and Kim, 2009) and (Herrmann, 1999):

- Incomplete or non-existent requirements
- Undocumented assumptions
- Not adequately taking constraints into account
- Overlooking or not understanding design flaws or system states
- Not accounting for all possible logic states
- Not implementing sufficient error detection and recovery algorithms

Error of omission results from the inability of software designers to understand system requirements from functional domain specialists. Domain experts tend to take for granted things which are familiar to them, but which are usually not familiar to the person eliciting the requirements.

***Error of commission*** are caused by making a mistake or doing something wrong in a software development process. Examples of errors of commission include (Herrmann, 1999):

- Logic errors

- Faulty designs
- Incorrect translation of requirements in software
- Incorrect handling of data
- Faulty system integration

A typical example of logic errors is the inadequate use of logic constructs - CASE constructs or IF/THEN/ELSE constructs - resulting in an unintended output ([Herrmann, 1999](#)). Adequate care should be taken by software engineers against errors when using logic constructs because of different evaluation and execution mechanisms of the constructs.

Examples of *hardware related systematic fault* can be a failure arising from incorrect, incomplete or ambiguous system specification

# Chapter 5

## Common-Cause Failures

### 5.1 Introduction

As part of the SIS lifecycle design process, the SIS should also be evaluated (not only for its SIL but) for its potential for CCF. IEC 61508 states that to maintain the safety integrity of SIFs, CCFs needs to be controlled. The U.S. Nuclear Regulatory Commission (NRC) has been spearheading the research on CCF through the development of CCF models, collection of data on CCF, and analysis of the data. The biggest effort in this regard is made by the Nuclear Energy Agency (NEA) and the International Common Cause Data Exchange (ICDE). Also the aviation industry and the Norwegian oil and gas industry have done some work in this field.

### 5.2 Definition of Common-Cause Failure

☛ **Common-cause failures** is a subset of dependent failures in which two or more component fault states exist simultaneously, or within a short time interval, and are a direct result of a shared cause (NUREG/CR 6268, 1998).

This definition from the nuclear industry suggests that the components do not have to fail at the same time, but the components must be in a fault state at the same time, or nearly the same time. For a SIF where the sensor subsystem has  $n$  items, two or more dangerous undetected (DU) failures can occur at different times in the same proof test interval. Because the DU failures

are undetected, the items remain in DU fault states until the proof test is carried out. According to this definition, a CCF of DU failures has therefore occurred if two or more DU faults (with a shared cause) are revealed in a proof test (Rausand, 2014).

The international standard IEC 61508, Part 4 Definitions and abbreviations, defines:

☞ **Common-cause failure.** Failure that is the result of one or more events, causing concurrent failures of two or more separate channels in a multiple channel system, leading to system failure (IEC 61508, 2010)

There are two issues raised in this definition.

- Common-cause failures is specified to be "one or more events". A condition, such as "higher humidity than specified", is not an event, and can therefore not be a common-cause according to the IEC 61508 definition.
- The term "concurrent failures" as used in the definition implies that the failure events must occur concurrently or close in time but failure is an event that takes place at a specific time. This negates the definition given for CCF by the nuclear industry.

Rausand (2014) combined the two definitions from the IEC 61508 and the nuclear industry to come up with the definition below.

☞ **Common-cause failure.** Failure, that is the direct result of a shared cause, in which two or more separate channels in a multiple channel system are in a fault state simultaneously, leading to a system fault.

**Categories of CCF:** CCFs can be classified into two:

- *Shock failures.* Simultaneous failures that occur at the same time due to shock or
- *Multiple failures* that occur over a certain period due to an increased stress (e.g., temperature, humidity, vibrations).

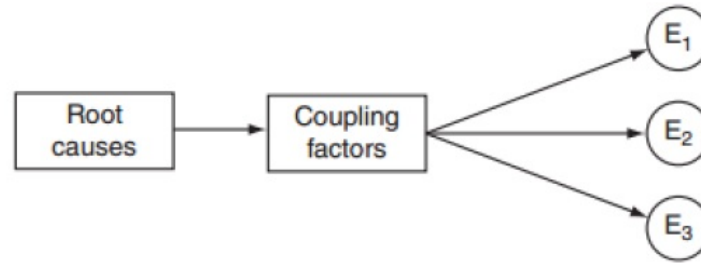


Figure 5.1: Relationship between root cause, coupling factors and failed items  $E_1$ ,  $E_2$ ,  $E_3$  (Rausand, 2011).

It is important to note that increased stress may not lead to a CCF of the second category but may increase the independent failure rates of the affected channels. In some cases, the increased stress may imply that the constant failure rate assumption is no longer valid and that the channels will have increasing failure rate functions (Rausand, 2014).

There can be multiple failures with a shared cause that are not a CCF, this is described with the term *Multiple failure with a shared cause* (MFSC). This is a failure that is a direct result of a shared cause, in which two or more items are in fault state simultaneously.

### 5.3 Causes of Common-Cause Failure

It has been shown by Paula et al. (1991), Rausand (2011) and also by Rausand (2014) that it is useful to split the shared CCF cause into two elements, a *root cause* and a *coupling factor*.

☞ **Root cause:** The root cause of a specified failure is the most basic cause that, if corrected, would prevent recurrence of this and similar failures.

In root cause analyses, the series of causes is pursued until the fundamental, correctable cause has been identified (DOE-NE-STD-1004-92, 1992). The study of root causes helps system designers to include in the design of the system control measures for reducing the predisposition to both single failures and CCFs. Causes of CCFs in complex systems have been linked to human actions and not following the laid down procedures. A study of centrifugal pumps in nuclear power plants shows that the root causes of about 70% of all CCFs are related to this cause

Table 5.1: Examples of root causes of CCF events in the design and development phase of SIS life cycle ([Hokstad and Rausand, 2008](#))

Cause type	Examples of specific cause
Design requirements Inadequate specifications	Failure of designer to predict an accident Failure of designer to recognize what proactive action is needed
Design error or inadequacy in design realization	Inadequate facilities for operation, maintenance, testing or calibration
Design limitations	Financial, spatial

([Miller et al., 2000](#)).

Practically, root causes of component failures are rarely and not often revealed from failure cause descriptions. They are determined through root cause analyses in addition to the use of checklists of generic root causes ([DOE-NE-STD-1004-92, 1992](#)).

☛ **Coupling factor:** A property that makes multiple components susceptible to failure from a single or shared cause.

Such properties include: same design, same hardware, same installation staff, same maintenance or operation staff, same procedures, same environment and same location. Studies of CCFs in nuclear power plants indicate that the majority of coupling factors contributing to CCFs are related to operational aspects ([Miller et al., 2000](#)).

Coupling factors explains why several components are affected by the same root cause, and are called coupling mechanisms. For example, same inadequate design is chosen for several valves. Poor design can be a root cause, and when the same poor design is carried out on several components, it is then a coupling factor. The root cause and the coupling factor constitute a CCF.

Investigations show that little or no CCF data from the oil and gas industry is available for public use. The nuclear industry industry has been leading the development of CCF models with respect to collection and analysis of data, and has since the early 1970's considered CCFs in their risk analysis. The Nuclear Energy Agency (NEA) has initiated the International Com-

mon Cause Data Exchange (ICDE) project to encourage collection and analysis of data related to CCF events. However, the actual data available to the public are restricted. Data on centrifugal pumps as presented in [Miller et al. \(2000\)](#) shows the distribution (of root causes and coupling factors) among the events with complete CCF:

- Root causes: 70% human actions and procedural deficiencies, 20% hardware related
- Coupling factor attributes: 66% operational, 33% hardware related

Though centrifugal pumps are not part of SIS in the oil and gas industry, the product development has similar levels of safety integrity requirements, and components are within the same category as SIS. Hence, data related to CCF event can be assumed to be valid for analysis of SIS.

## 5.4 Relationship between Systematic Failure and Common-Cause Failure

From previous sections of the report, it has been established that CCF is the concurrent failure of two or more structures, systems or components (SSCs) due to the triggering of a single systematic fault or causally related faults by a single specific event. The triggering event may be related to time, data or hardware. The term "common-mode failure" is not used because CCF is more inclusive of the effects related to the failure ([NP-T-1.5, 2009](#)).

A systematic fault affects all forms of component, be it hardware or software. The fault may be introduced during the design or the manufacturing process, or it may be related to maintenance or modification activities. Faults introduced during design and development are of particular importance here. A triggering mechanism is a specific event or operating condition, which causes SSCs to fail due to the latent fault. Thus a systematic failure is related in a deterministic way to a certain cause ([IEC 61508, 2010](#)). The failure will always occur when the design fault is challenged by the triggering mechanism. In order for such a failure to occur, the following conditions must be present:

- A systematic fault must exist in multiple components in the integrated system;

- A triggering event must occur to challenge the systematic fault.

A CCF is a systematic failure that occurs when failures of separate SSCs are triggered concurrently. The failures are considered concurrent if the interval between the failures is too short for repair or recovery measures to be taken (NP-T-1.5, 2009).

A given system may contain systematic faults. However, the faults do not become failures until they are challenged, and the failures are not harmful unless they impair the safety function when it is required. This implies that, not all faults become failures, and not all failures are harmful. Further, a given failure is not a CCF until multiple channels of a redundant system are affected. Therefore, it is not important to ensure that software faults are removed from the system design but it is necessary to demonstrate that software does not contain faults which may be triggered to become harmful, that can in turn lead to CCF. Since it is impossible to assure that software faults have been removed in the system design, except in situations where 100% functional testing can be performed, it is then necessary to incorporate defensive measures into the design to ensure safety of the system.

#### **5.4.1 Types of Systematic Faults that do not Lead to a Common-Cause Failure**

From the previous section, it has been established that the existence of a triggering mechanism and a latent fault is necessary to cause the coincidental failure of two or more channels. Redundant channels are rendered ineffective when the triggering mechanism causes the concurrent failure of the channels thereby leading to a CCF.

Systematic faults listed below do not lead to CCF if they are not challenged and also if the failures affect only one channel:

- Errors in standard functions, such as trigonometric functions, which can result in incorrect calculations
- Operating outside the valid parameter bounds of an algorithm or standard function can result in incorrect calculations or out of range results



- Loss of accuracy due to poorly constructed algorithms
- Inaccurate calculations or incorrect trip determinations
- Improper safety design can result in failure of the safety function when needed
- Errors in amplitude quantisation and sampling frequencies can affect transient response resulting in incorrect operation of the control system
- Inconsistencies in data communication protocols can cause incorrect operation
- Errors in software libraries can result in a variety of improper operations

## 5.5 Design Measures Against CCF

Prevention of CCFs in SISs starts at the top level of the whole SIS architecture. IEC 61508-6 presents three overall measures that can be used to reduce the probability of CCFs. These are:

1. Reduce the overall number of random hardware and systematic failures. (This reduces the areas of the ellipses in Figure 5.2 leading to a reduction in the area of overlap)
2. Maximize the independence of the channels (separation and diversity). (This reduces the amount of overlap between the ellipses in Figure 5.2 whilst maintaining the area.)
3. Reveal non-simultaneous CCFs while only one, and before a second, channel has been affected, i.e. use diagnostic tests or proof test staggering.

Two main methods are used to minimize the number of faults in SISs: *fault avoidance/prevention* and *fault detection/removal*.

**Fault avoidance/prevention** approaches are used in the design and development phase to reduce the number of faults introduced during this phase of SIS life cycle. The strategies applied here should be able to address sources of faults listed in section 4.3.3.

The main fault avoidance principles as listed in (NP-T-1.5, 2009) are:

- Making functional specification less complex in system/software design;

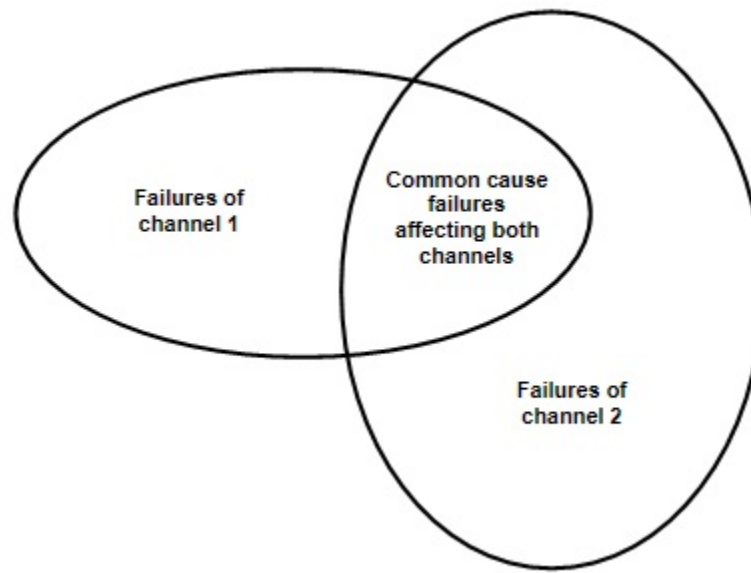


Figure 5.2: Relationship of CCFs to the failures of individual channels (IEC 61508, 2010).

- Application of well defined development processes with defined activities, well-specified deliverables and documentation, and clear attribution of responsibilities;
- Use of appropriate methods and tools;
- Use of competent, knowledgeable personnel, having sufficient resources, equipment and time;
- Application of suitable rules and guidelines for each activity of the development process, providing reasonable defence against systematic faults and CCFs.
- Use of dependable and well understood components and platforms. Components may be software components, hardware components or equipment integrating hardware and software.
- Taking into consideration lessons learned from past mistakes and faults in similar systems in order to improve development processes, methods and tools, rules and guidelines, as well as training.

**Fault detection/removal.** The techniques employed here are testing, formal inspection and formal design proofs. They are applied during system/software development, mainly in verification and validation activities. Other measures will be discussed in the chapter 7.

# Chapter 6

## Reliability Growth Testing

To help equipment and systems meet the required operational reliability, the concept of reliability growth testing and management has been developed for equipment/system development programmes.

☛ **Reliability growth** is defined as the positive improvement of the reliability of an equipment through the systematic and permanent removal of failure mechanisms ([Handbook, 1998](#)).

The success of reliability growth depends on the extent to which testing and other improvement techniques have been used during development and production to reveal design and fabrication flaws and the rigor with which these flaws are analyzed and corrected. There are three essential elements involved in achieving reliability growth:

1. Detection of failure causes (by analysis and test)
2. Feedback from failure causes identified
3. Effective redesign effort based on problems identified

Failure causes are detected through testing, and the testing process controls the rate of reliability growth. The reliability growth process is therefore called test, analyze, and fix (TAAF). Reliability growth testing (RGT) is only one aspect of the reliability growth management programme.

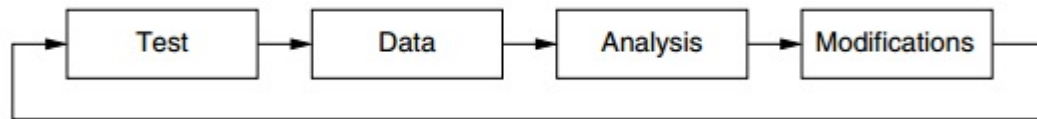


Figure 6.1: Test, analyse, and fix (TAAF) cycle  
(Murthy et al., 2008).

➤ **Reliability growth testing** is the same as TAAF testing of a product early in the development cycle when design changes can be made readily in response to observed failures (Murthy et al., 2008).

The objectives of RGT includes:

- Identify problems as early as possible in the design phase
- Verifying improvements in design reliability
- Identifying any necessary design changes
- Determining if the product design has to be improved to meet the specified reliability

RGT ensures that development testing is systematically carried out to track the progress of reliability improvement efforts and foresee system reliability given the anticipated rate of improvement. The aim here is to improve reliability and not to assess reliability. The item is exposed to environmental stresses simulating actual usage until failure occurs. The failure is analysed and the causes identified. Design modifications are made to avert or drastically reduce the likelihood of recurrence of the same failure mechanism. The TAAF process is an iterative process and is illustrated in Figure 6.1.

## 6.1 Literature Review of Work done on Reliability Growth Testing

The TAAF process begins in the design phase and consists of tests that are structured in such a way that it will expose the item to the kinds of stresses it is expected to go through in the field. TAAF test design principles, and relationship of TAAF to other testing programmes have been

discussed by [Priest \(1988\)](#), [Handbook \(1998\)](#), [IEC 61014 \(2003\)](#) and [Murthy et al. \(2008\)](#).

A number of reliability growth models have been developed to monitor the progress of the development programme and the improvements in reliability of the item under consideration. The works of [Lloyd and Lipow \(1963\)](#), [Amstadter \(1971\)](#), [Dhillon \(1983\)](#), [Handbook \(1998\)](#) and [Quigley and Walls \(1999\)](#) are in this regard, highly appreciated in this thesis report. The theoretical and practical criticisms of these models are well documented and compared in [Krasich et al. \(2004\)](#) and [Walls et al. \(2005\)](#).

A failure reporting and corrective action system (FRACAS) is sometimes initiated to record failure data gathered through test and improvement programme. The FRACAS is a closed-loop reporting system that is a parallel to the TAAF cycle. More information on FRACAS can be found on ([O'Connor, 2002](#)).

## 6.2 Reliability Growth Models and Methods

The purpose of estimating the potential reliability growth is to aid management in scheduling. In some cases, a reliability growth estimate is used to determine the expected or estimated amount of test time needed to reach a reliability level. Both *continuous* and *discrete* models have been developed to assess reliability growth ([Duane, 1964](#)) and ([Crow, 1975](#)).

### 6.2.1 Continuous Growth Models

Continuous growth models were developed for repairable products in which reliability is measured in terms of mean time between failures (MTBF). The cumulative failure rate is given by:

$$\omega_c(t) = \frac{E(N(t))}{t} \Rightarrow MTBF_c(t) = \frac{t}{E(N(t))} \quad (6.1)$$

The rate of occurrence of failures (ROCOF) is:

$$\omega_i(t) = \frac{d(E(N(t)))}{dt} \Rightarrow MTBF_i(t) = \frac{1}{\omega_i(t)} \quad (6.2)$$

Where;

- $MTBF_c(t)$  is the cumulative MTBF at time,  $t$
- $MTBF_i(t)$  is the instantaneous MTBF at time,  $t$

Examples of continuous growth models are the Duane model and the Army Material Systems Analysis Activity (AMSAA) model.

### Duane model

The Duane model is empirical and shows linear relationship between cumulative MTBF and time ( $T$ ) when the natural logarithm ( $\ln$ ) function is applied in reliability growth analysis to show the effects of corrective actions on reliability. After accelerated testing it is possible to estimate the MTBF, which is considered the initial MTBF in the Duane model. The equation for reliability growth in the Duane model is:

$$MTBF_c(t) = k \cdot t^\alpha \quad (6.3)$$

- $MTBF_c$  - Cumulative mean time between failure
- $t$  - Cumulative time
- $k$  - A constant which depends on equipment complexity, and design for reliability
- $\alpha$  - Reliability growth rate

By taking the logarithm of the Duane model:

$$\ln MTBF_c(t) = \ln k + \alpha \ln t \quad (6.4)$$

This is a straight line with a log-log scaled axis. The expected number of failures  $E(N(t))$  at time  $t$  can be expressed as

$$E(N(t)) = \frac{1}{k} \cdot t^{1-\alpha} \quad (6.5)$$

The ROCOF is then

$$\omega_i(t) = \frac{d}{dt} E(N(t)) = \frac{1-\alpha}{k} \cdot t^{-\alpha} \quad (6.6)$$

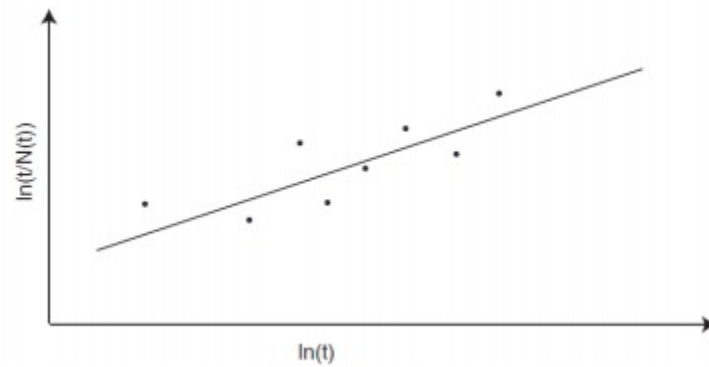


Figure 6.2: Duane reliability growth curve (Yiliu).

Thus the "instantaneous" or current MTBF curves are straight lines displaced from the cumulative plot by a factor  $\frac{1}{1-\alpha}$ , which shows up as a fixed distance on a logarithmic plot.

$$MTBF_i(t) = \frac{1}{\omega_i(t)} = \frac{k}{1-\alpha} \cdot t^\alpha \quad (6.7)$$

Thus

$$MTBF_i(t) = \frac{MTBF_c(t)}{1-\alpha} \quad (6.8)$$

### AMSAA model

This model attempts to track reliability within the series of growth testing cycles, referred to as phases. After the conclusion of each design change (cycle), the failure rate decreases if you do not introduce any new errors when you improve the system (Introduction of new errors is often a problem of software development). However, before subsequent testing, the failure remains constant.

Assume that modifications are made at  $t_i$ ,  $0 < t_1 < t_2 < \dots < t_k$ . Let  $N_i$  be the number of failures that occur in interval  $(t_{i-1}, t_i)$ . We assume that  $N_i$  has a Poisson distribution with intensity  $\omega_i$ , such that

$$\Pr(N_i = n) = \frac{[\omega_i(t_i - t_{i-1})]^n}{n!} e^{-\omega_i(t_i - t_{i-1})} \quad (6.9)$$



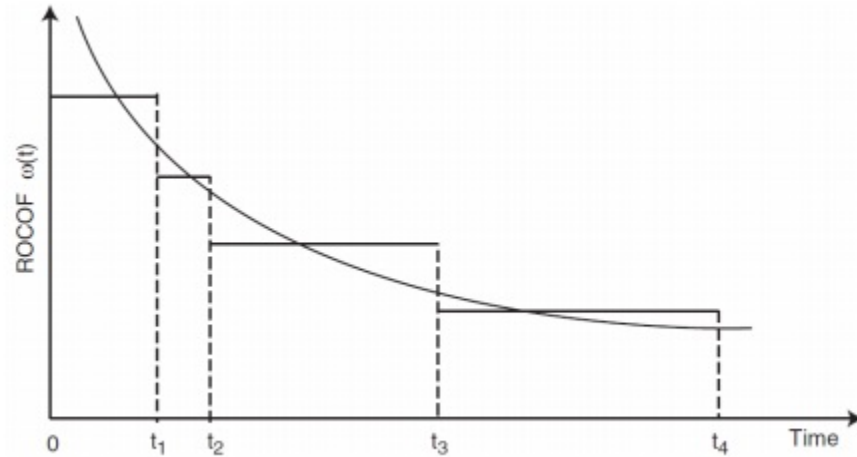


Figure 6.3: AMSAA reliability growth curve (Yilieu).

The mean number of failures in this interval is

$$E(N_i) = \omega_i(t_i - t_{i-1}) \quad (6.10)$$

The time to failure  $T_i$  of a prototype in the interval  $(t_{i-1}, t_i)$  is hence exponentially distributed with  $\omega_i$ . The MTBF in this interval is

$$MTBF_i = \frac{1}{\omega_i} \quad (6.11)$$

### 6.2.2 Discrete Models

Discrete models differ from continuous models because they measure reliability in terms of a go/no-go situation, such as for a missile or rocket. Products that either fail or operate when called into service are modeled by discrete functions. Examples are Lloyd and Lipow model and Wolman model (Lloyd and Lipow, 1963).

Other reliability growth models (software) are: Jelinski and Moranda model, Littlewood and Verrall model, Littlewood model, Musa model, Musa-Okumoto model, Brooks and Motley model, Shooman model (Murthy et al., 2008).

## 6.3 Strengths and Weaknesses of Various Models and Methods

### 6.3.1 Duane Model

The strengths of the Duane model are:

- The Duane model is used for reliability growth programme planning even by the proponents of the AMSAA model because it is very simple to use and presented graphically
- On a log-log plot, the graph of cumulative failure rate versus the cumulative test time is linear, this makes it easier to determine the parameters
- Above all, it is practical and it has been found that development test data from a wide variety of applications fit the model well
- Essentially, it relates failure rate or MTBF, to improvement effort and duration

The limitations of Duane model are ([Denning and Abbey Wood, 2012](#)):

- The Duane model model is often criticised theoretically because, when  $t=0$ ,  $MTBF_c(t)$  is also zero, which is clearly untrue in practice. The Duane model also implies that given sustained reliability effort, growth will go on indefinitely, albeit at a decreasing rate
- Since the model is essentially a graphical technique, it may tend to lack accuracy. In particular, the straightness of a series of plotted points and the fitting of a line to cumulative data points can be highly subjective. Also, being plotted on a logarithmic scale, the model is rather insensitive to changes in reliability occurring late in a test programme
- The model assumes a uniform level of testing and improvement effort throughout the test programme and the prompt introduction of modification to produce reliability growth. In practice, these conditions may not always be fulfilled.

### 6.3.2 AMSAA Model

The Duane model assumes that growth is a deterministic process, while the AMSAA model views the process of reliability growth as a probabilistic process. This has advantages because the

parameters of a non-homogenous Poisson process (NHPP) can be estimated on a statistically rigorous basis, confidence intervals can be obtained and goodness of fit applied.

The Duane plot uses a least squares estimate of which the plot will fall while the AMSAA model takes into account the exponential relationship between each data plot. Therefore in reliability growth plotting, the AMSAA model tends to give a more accurate representation of the reduction in failure rate with respect to time.

# Chapter 7

## Using reliability growth testing to reveal systematic faults

### 7.1 Introduction

Reliability growth testing in the design and development phases can be used to reveal systematic faults or weaknesses that have design related causes. It is of high importance to reduce the probability of failure due to these weaknesses to the greatest extent possible to prevent their later appearance in the field.

In order to use RGT to reveal systematic faults, illustrative examples of functions of fire and gas (F&G) detection and mitigations systems, mobile phone, and car air bag was carried out. Formal and informal ways are used to conduct the test. By formal ways here, we mean testing in the laboratory and informal ways of testing is by distributing the prototype to employees or limited number of customers..

### 7.2 Illustrative Examples

This section discusses how the formal and informal RGT are conducted. The examples are chosen because they are systems where RGT can be implemented on the functions to reveal systematic faults. Although mobile phone is not a SIS, it is used here as an example for a better un-

derstanding of the concept. Some of the information about the illustrative examples are from online sources. Accelerated Life Testing (ALT) is not the best form of RGT to be applied since systematic faults are non-physical faults. However, in cases where the component is operating in an environment where there is vibration, ALT can be used. For example, in the airbag example given in section 7.2.3., the sensors are assumed to be in an environment where there is vibration, and hence ALT is applied in the test

### **7.2.1 F&G Detection and Mitigation System**

F&G detection and mitigation systems are key to maintaining the overall safety and operation of industrial facilities. F&G systems are used in offshore petroleum exploration and production, onshore oil and gas facilities, refineries and chemical plants, marine operations, tank farms and terminals, pipelines, power plants, mining and paper mills. A F&G safety system continuously monitors for abnormal situations such as fire, or combustible or toxic gas release within the plant and provides early warning and mitigation actions to prevent escalation of the incident and protect the process or the environment.

A typical F&G safety system comprises detection, logic control and alarm and mitigation functions. The logic solver is the central control unit of the overall F&G detection and control system. The controller receives alarm and status or analogue signals from field monitoring devices required for F&G detection. The controller handles the required actions to initiate alarms and mitigate the hazard. F&G detections systems are generally programmable electronic systems (PES) type with high safety availability and mitigation effectiveness. As modern F&G systems are tightly integrated with the overall process safety strategy, mitigation either takes place via the emergency shutdown system (ESD) or directly from the F&G system itself.

Assuming that about 20 prototypes of F&G detection and mitigations systems from the same company were inspected for design errors, the system has sensors connected to logic solvers which both contain software developed by a system integrator. The system also enables redundant control of the loop. If, by any reason, the primary loop control fails, the secondary loop will take over, and fire detection is thus maintained. Redundancy is achieved without introducing

two set of detection loops and thus avoiding twice the amount of cabling and detectors. The inspection covers checks on software, diversity and cabling. The system is sufficiently checked when all the functions have been thoroughly tested.

Formal and informal methods are used in testing the functions of the F&G detection and mitigation systems. The functions tested are (Stanley, 2011);

- **Heat detectors.** A restorable heat detector and the restorable element of a combination detector is tested by exposing the detector to a safe heat source (such as hot water, a hair dryer, or a shielded heat lamp) until it responds. The detector should reset automatically after each heat test.
- **Smoke detectors.** The smoke detectors are visually inspected first, then smoke is introduced through an aerosol to ensure that smoke enters the chamber and initiate alarm. Bee smoker can also be used as a source of relatively safe smoke.
- **Remote annunciators.** The function of a remote annunciator is to assist the responding fire service personnel in locating the fire source. The labeling and clarity of the annunciator layout is visually inspected. Keeping the remote annunciator to simple graphics will often be more beneficial to emergency responders. It is important also to verify that the zone or point indications at the remote annunciator are identical to those at the fire alarm control unit.
- **Notification appliances** are bells, horns, speakers, sirens, strobes, and combination units. One of the weakest points in many fire alarm system designs has been the audibility of the notification appliances. The audibility is measured using a sound level meter. Never trust the inspectors "caliberated ear".

There can also be software errors such as configuration error that can make a F&G detection system to fail.

### 7.2.2 Mobile Phone (Smartphone)

Both formal and informal tests are conducted on the mobile phone. Prototypes of the mobile phone are sometimes given to employees of the manufacturing company who gets to test the

mobile phone applications and new features on the phone before they are actually released to the public. The employees are advised to use the phone as roughly as possible and report any fault back to the design team. Hardware features such as battery and processor are tested, because to keep a smartphone running smoothly you will need a long lasting battery and a powerful processor.

- **Battery.** Because smartphones have large screens and can perform lots of power hungry tasks, their batteries often last for less than 24 hours. This particular smartphone is meant to last for 18 hours in normal use, but the employees were asked to use many features continuously until the phone battery dies to check the number of hours the battery can last. The results show that the battery can only last for an average of 10 hours of continuous use of multiple features of the phone.
- **Processor.** The phone's processor is effectively the phone's brain, telling it what to do and how to do it. Its performance is measured according to the number of tasks it can complete per second, known as 'cycle' - a 1 GHz processor can process one billion cycles per second. Typically, a processor with a high speed will perform better and will give a faster, smoother performance - though memory cache and RAM (Random Access Memory) do also have an effect. The smartphone is marked 1 GHz processor and they were instructed to run different programs at the same time. The results show that the speed of the processor is much slower compared to other smartphones with the same processor speed specification.

Software features of the phone such as the operating system, app stores and maps are also checked. To test how effective the anti-virus software installed on the phone is, the employees were asked to go to online store and download a game app known to contain virus and see how efficiently the anti-virus software can protect the phone, Unfortunately the phone crashed as the anti-virus could not protect the phone. These results were reported back to the design team and the actions taken to deal with the faults are discussed in section 7.3

Other features of the phone tested are entertainment features such as camera and the quality of the pictures and videos are found to be good, the multi media player produced quality

sounds and the web browser were able to access the internet. Also, connectivity features such as frequency bands, Wi-Fi, and bluetooth were working perfectly.

Formal methods of testing mobile phones in the laboratory are provided by PC Magazine, a technology product review company. The mobile phones are subjected to conditions even far worst than that they will meet in operation. Most of the functions tested are:

- **Reception.** They identified locations throughout the laboratories where each major wireless carriers has very weak signal. In those locations, an attempt to connect, listen to, and record 2-minutes calls were made on each of the phones.
- **Call quality.** Calls were made to automated voice-recognition systems and landline answering machines, from indoor and outdoor areas with varying degrees of noisiness using both the microphone and speakerphone. Then the messages are listened to, to hear the quality of the sound. The sound quality shows that the phone has a systematic hardware fault with the speakers.
- **Bluetooth.** Each phone were connected to mono and stereo bluetooth headsets. Using the headsets, calls were made, activate voice dialing, and play control music. Also transfer files to and from Mac and Windows PCs (personal computers).
- **Battery life.** The battery life is measured with a continuous talk-time-test. The phone is placed in a location where it has full signal, then a "test line" is dialed with the phone's earpiece connected to a microphone and the microphone to a computer running audacity recording software, with its timeline, then we let the call go until the phone's battery dies.

### 7.2.3 Airbag

Airbags are passive safety devices. They are critical part of the supplemental restraint system in most vehicles. The objective of the airbag, which is activated when the vehicle suddenly decelerates (as in a collision), is to prevent the vehicle occupants from hitting any rigid surfaces and cushion the forces on their heads and upper or lower bodies. Airbags are typically made of nylon fabric and are hidden behind panels at various locations in the vehicle, including the steering



wheel. Depending on the crash severity, the rate at which airbags are activated is decided by the airbag control unit. In the event of a crash, the sensor (an accelerometer) sends a signal to the airbag control unit. This control unit triggers the inflation device, which generates nitrogen gas by igniting a mixture of sodium azide ( $NaN_3$ ) and potassium nitrate ( $KNO_3$ ). The time between crash detection and complete activation of the airbag is approximately 0.05 seconds.

Assuming that a customer requested for a design and development of a new airbag system, and the customer has specified the function and the performance requirements of the airbag system, i.e., what is to be protected (upper body, head, knees, etc. of driver and front-seat passenger), and the reaction time of the airbag. As the airbag designer, the number and location of airbags, the number and location of the crash sensors, and the type of airbag controller are among the decisions made to achieve high reliability.

An RGT must therefore address a wide range of issues/questions which include:

1. Do we have a sufficient number of crash sensors?
2. Are they located in the "optimal" places of the car?
3. Do they respond fast enough?
4. Do they deteriorate with time?
5. Can the sensors discriminate between real and "false" decelerations? Can heavy braking give sensor activation?
6. Is it likely that the sensors send false signals?
7. Do we have sufficient number of airbags?
8. Are they located in the "optimal" places to protect the driver/passenger?
9. Do they inflate fast enough?
10. Do they maintain pressure long enough?

11. Does the inflation gas have any negative side effects?
12. Do the airbags deteriorate with time?
13. Can the airbags be spuriously activated?
14. When there is no passenger in the passenger seat, the airbag controller should deactivate the airbags protecting the passenger. How can this system fail?

Prototypes of airbags are tested in a crash laboratory by a team of experienced crash engineers, safety integration engineers, restraint engineers, consultants and project managers. The vehicles are made to impact on a crash wall to test the capability of the airbag to protect the passengers. The results of the crash tests are given below ([Honda, 2013](#)):

#### **For Crash Sensors**

Crash sensors collect the data necessary to make decisions about airbag deployment. They measure how quickly a vehicle slows down in a frontal crash or accelerates to the side in a side impact crash. There are sufficient number of crash sensors which are located in the ideal areas to protect the driver and passenger. Frontal crash sensors are located in the front of the vehicle near the engine, in the passenger compartment or sometimes in the electric control unit (ECU). Side impact crash sensors may be located in the ECU, the door, the door sill, or between the front and rear doors. Rollover crash sensors may be located in the ECU or at the vehicles center of gravity. Sensors can discriminate between real and false decelerations. Severe or panic braking alone cannot cause an airbag to deploy; airbags deploy only in crashes.

#### **For Airbags**

There are sufficient number of airbags to protect driver and passenger and they are located in the right places. The driver's airbag is stored in the center of the steering wheel; the front passenger's airbag is stored in the dashboard. The two side airbags, one for the driver and one for the front passenger are stored in the outer edges of the seat-backs. The two side curtain airbags, one for each side of the vehicle are stored in the front, center and rear pillars. During a crash, the airbags normally inflate within a split of a second. After inflating, the airbags will immediately deflate, so they won't interfere with the driver's visibility or the ability to steer or operate

other controls. The total time for inflation and deflation is one-tenth of a second, so fast that most occupants are unaware that their airbags deployed until they see them lying on their laps. After a crash, you may see what looks like smoke. This is actually powder from airbags' surface. Although the powder is not harmful, people with respiratory problems may experience some temporary discomfort. If this occurs, the driver and passenger should get out of the car as soon as it is safe to do so. Airbags have long life spans. Many companies do not give recommendations on the life span of their airbags but Volvo, who have their vehicles recalled from customers and tested them, have changed their recommendation, firstly to fifteen years, then to twenty years as they have found the components and chemical to be stable. Mercedes-Benz originally stated their airbag should be replaced during normal service at the customers cost after fifteen years, but both now say the units should last the life of the vehicle. The passengers advanced front airbag system has weight sensors under the seat. If the sensors detect a total weight on the seat of about 30kg or less, the system will automatically turn the passengers front airbag off. The sensors can be negatively influenced if items are placed under the front seats as this could make the drivers seat position sensor and the passenger's weight sensor ineffective.

The airbag can also have a problem in the software that controls airbag deployment for the front seat passenger, the software installed on the vehicles incorrectly determines whether the passenger seat is empty when it is, in fact, occupied. If that were to happen, and the vehicle subsequently involves in an accident, the airbag would fail to deploy, increasing the possibility of injury or death.

### **7.3 Challenges and Pitfalls of Reliability Growth Testing in Relation to Systematic Faults**

The general objectives of RGT especially accelerated tests are to determine the components limits by applying stresses high enough to induce failures, to quickly discover the root causes of failures that will occur in field use, and thus improve the reliability (Suhir, 2002). This section discusses the challenges and pitfalls of RGT in relation to systematic faults.

RGT to assess product performance is usually done in controlled conditions and an environment that mimics the real world in a limited manner. The field performance depends on several factors like operating environment, usage intensity, load (or stress) on the product. It is important to point out here that this varies from customer to customer. SISs with rather cheap assemblies and components can be tested in the way described above. Some SISs are, however, so complex and so expensive that it is not possible to do any degradation testing apart from some minor components. An example is a high integrity pressure protection system (HIPPS) on a subsea oil and gas pipeline. The valves of such a system are so big and so expensive that it is not possible to do any reliability testing at the assembly (valve) level. We, therefore have to accept the testing of minor parts like seals, material samples, etc (Murthy et al., 2008).

Other pitfalls can be grouped into the following categories (Meeker et al., 2013):

- Pitfalls that occur during the planning of the test
- Pitfalls that occur during the execution of the test
- Pitfalls that occur during the analysis and interpretation of the test data

### 7.3.1 Pitfalls During RGT Planning

This section deals with the pitfalls that are associated with the early stages of planning an accelerated test.

1. **Testing at too severe stresses (requiring extreme extrapolation).** Testing at high levels of the accelerating variable, far away from the use level of the accelerating variable implies a large amount of extrapolation. Statistical models used in the analysis of accelerated test data are most times idealized approximations to the truth and will always contain some degree of error. Extrapolation to use conditions will greatly amplify these errors that are always present. Also, testing at levels of the accelerating variables that are too extreme
  - Can result in activating new failure modes that would not be experienced at use conditions. Might make the assumed model inadequate at the high stress levels.

- Will result in greater statistical uncertainty in estimates at the use conditions (compared to running at more moderate stresses that still induce failures).
2. **Using unnecessarily complicated testing and data analysis.** Simultaneous application of several different accelerating variables and test plans where the level of the accelerating variables changes during the test (most commonly used in HALTs) are complicated approaches to reliability testing. This complicated approach will not necessarily lead to better results. Simpler test designs (e.g., the commonly used constant-stress tests) provide cleaner data and require fewer assumptions.

Other pitfalls during test planning are: Choosing the wrong accelerating variable or having a test plan that will not provide useful information and focusing only on the obvious failure mode(s).

### 7.3.2 Pitfalls During the Execution of Test

This section discusses pitfalls that are associated with the execution stage of an accelerated test.

1. **Improper use of system level ALT.** The purpose of most ALTs is to focus on a particular failure mode, usually at the material or component level, and then accelerate failures by testing at higher-than-usual levels of one or more accelerating variables, such as temperature, voltage, load etc. Increasing temperature or stress on a full system (or even a subsystem) is generally less useful. This is because the maximum allowable temperature at higher levels of assembly (system or subsystem) will generally be much lower than the maximum allowable level at the material/component level.
2. **Inadequate monitoring of failures during ALT.** There have been ALT applications where no useful lifetime data were available at the end of the test. For example, the test to find the lifetime of smartphone battery. The test units were placed in the chamber, the desired levels of stresses were set and the units left unmonitored, without checking the status of the units at regular intervals. After 14 hours, all of the units have failed. If possible, one should

obtain time-to-failure information by continuous monitoring so that exact times to failure can be determined. When continuous monitoring is impossible, inspection should be carried out at regular intervals.

3. ***Not inspecting survivors at the end of an accelerated test (or not fully using the information gained from such inspections)***. An online source provided an example where ALT results were not interpreted properly because inspection information, although collected, was not properly used. In particular, a sample of refrigerator compressors was run for one year in an accelerated test with no failures. Because there were no failures in the test, a decision was made to launch the product. What was not reported to higher-level management was that the unfailed compressors, when disassembled for inspection, exhibited unexpected discolouration, providing evidence of a lubrication issue and suggesting that the unfailed were on their way to failure. After problems were discovered from field failures, it was determined that all of the refrigerators that had been sold would have premature failures.
4. ***RGTs that generate failures that will not arise in the field***. We have seen a number of ALT and HALT programs that led to the discovery of a failure mode followed by an expensive redesign effort to eliminate the failure mode, only to learn later that the failure mode would never have occurred at use conditions.

### 7.3.3 Pitfalls During Analysis of Test Data

This section treats pitfalls that are associated with the data analysis stage of accelerated test.

1. ***Attempting to estimate a life distribution from HALT***. HALTs serve a useful but different purpose than ALTs. HALTs are importantly useful tests that are often conducted in product development processes, especially in electronic products, to discover product design weaknesses, screen for failure modes and understand design and destruct limits (as opposed to estimating reliability). HALTs often lead to modifications in the design of the product whenever there is a change in design (which could be triggered by either a HALT or an ALT), the data would be no longer relevant for estimating the product's reliability.

2. **Using an inadequate acceleration model.** Example is using Arrhenius relationship instead of inverse power relationship and vice versa.
3. *Ignoring or inadequately treating interactions*
4. *Not comparing failure modes in the field with failure modes in the laboratory*

Other pitfalls as discussed in [Meeker and Escobar \(1998\)](#) are:

- Multiple (unrecognized) failure modes
- Failure to properly quantify statistical uncertainty
- Multiple time-scales and multiple factors affecting degradation
- Masked failure modes
- Faulty comparison
- Accelerating variables can cause deceleration!
- Beware of untested design/production changes
- It is difficult to use ALT to predict field reliability

## 7.4 Handling of Systematic Faults Revealed During the Test

Systematic faults are events that can result from one or more systematic failures. Going by this definition, it can be argued that the methods used to handle systematic failures is covering systematic faults as well. There will still be systematic faults that are not taken care of through these methods. To what extent they constitute a threat to the systematic safety integrity is unknown. However, since systematic faults are not a physical fault and will always occur under the same conditions, it is reasonable to believe that they are reduced or eliminated by the same measures as for systematic failures.

The standards IEC 61508 and IEC 61511 are not prescriptive and gives room for different interpretations, and hence opens up for new methods, approaches and technology. They focus on

qualitative measures for handling systematic failures by using *requirements* and *checklists*. Documents and specific guidelines are therefore provided to address various aspects of how the IEC 61508 and IEC 61511 requirements should be adopted (Lundteigen and Rausand, 2006). One such guideline is the NOG-070, presented in section 4.3.1.

The consequences of systematic faults can have great impact on SISs. In section 4.3.2., the importance of systematic faults is discussed. They are not only the main contributor to CCFs but also a dominant contributor towards the overall failure probability (Hauge et al., 2013).

IEC 61508 part 2 and 3 provide standard sets of requirements for the avoidance and control of systematic failures, which are based on expert judgement from practical experiences gained from the industry. IEC 61511 mainly focuses on handling systematic failures and faults in the operational phase, as the standard provides guidance to end-users on the application of SISs. The standard states that systematic faults are best addressed through the implementation of a protective management system, which overlays a quality management system with a project development process.

Systematic faults are undesired - not only in the design and development phases, but in the entire SIS life cycle. Already there are a number of tools that deal with errors in production and operation phases. The difficulties in working with the design process stem from the fact that mistakes made then are usually latent errors, which are those errors that cause adverse consequences within the system but lie dormant for a long time. They are harder to recognize or detect, unlike the active errors whose effects are usually felt immediately.

In preventing systematic failures and faults revealed in the design and development phases of the SIS life cycle, the following measures are taken in addition to those discussed in section 5.5.

1. **Design reviews.** The design team having received feedback from the employees on the faults in the battery life and processor speed of the smartphone, went back to the drawing board to redesign them. They came out with a larger capacity battery of 3100mAh that



can last for 24 hours of continuous usage. They also came out with dual core processors. Having two chips means that the phone is better at multi-tasking as one can handle background tasks while another can work on your active task. The extra power also means faster interfaces and enables new functionality such as high definition video recording, also each core works less hard to accomplish a task, and the phone should use less battery power. Also the quality of the phone speaker was redesigned to a better one with superb sound quality. The airbags was redesigned to a more safer one that can offer the needed protection to drivers and passengers.

2. **Redundancy.** Redundant components offer protection against failures. The probability of multiple failure is less than that of a single failure provided they are independent. Also if the components are identical, systematic failures (design flaws) are common source of common mode failure. Examples can be seen in the phone processor, where dual core processors were introduced. Also we now see phones that run on quad core processors which are faster in performance and battery life is longer. Also redundancy can be applied in the case of F&G detectors.

This report discusses four types of redundancy ([Braunl, 2014](#)) and ([Abd-El-Barr, 2006](#)): They are;

- Hardware redundancy
- Software redundancy
- Information redundancy
- Time redundancy

- (a) **Hardware Redundancy.** This method uses additional hardware to overcome faults. The inclusion of some extra hardware ensures that concurrent computations can be voted upon, errors can be masked out or duplicate (spare) hardware can be switched automatically to replace failed components. Three techniques of hardware redundancy can be identified. These are:

- **Passive (Static).** In this form of hardware redundancy, the effects of faults are essentially masked with no specific indication of their occurrence, i.e., these effects are hidden from the rest of the system. A representative of this technique can be seen in *N-Modular Redundancy (NMR)*.
  - **Triple-Modular Redundancy (TMR)** uses three times hardware plus majority voter, if one system becomes faulty, the two other correct one mask the fault. This technique is simple but expensive and provide uninterrupted service in the presence of faults.
  - **Active (Dynamic).** This technique detect the existence of fault, then perform some actions to remove faulty hardware from system (reconfiguration).
  - **Hybrid approach (Passive + Active).** This technique combines the advantages of the passive and active redundancy. The advantage of this system is that it hides the effect of faults (fault masking) while replacing faulty units with spares (reconfiguration). This method is far more expensive that the static hardware redundancy technique, and therefore be applied in critical conditions, such as space applications.
- (b) **Software Redundancy.** This technique uses extra code, small routines or possibly complete programs, in order to check the correctness or the consistency of the results produced by a given software. Some of the software techniques used have their hardware counterparts techniques:
- Static Software Redundancy Techniques
    - **N-Version Programming (NVP).**The idea behind this technique is to generate N independent and different versions of the program. The versions need to be independent so that a fault will not occur in all modules. This model can tolerate  $(N - 1)/2$  faults. NVP is similar to TMR hardware redundancy. It is expensive, difficult to maintain, and its repair is not trivial.
    - **Consistency checks.** For many applications, a certain data range and conditions for the result are known beforehand. Check the data range, report error if out of bounds and restart.

- ***Ad-Hoc Techniques.*** These techniques are application dependent. For example, a consistency checking on a bank withdrawal can be made by checking that the amount of money withdrawn from any bank machine does not exceed a certain known a priori amount, and that the aggregate amount withdrawn by any customer during the 24 hours period from all bank machines does not exceed his/her allowed maximum amount per day.
- Dynamic Software Redundancy Techniques
  - ***Forward Error Recovery.*** The system will continue operation with the current system state even though it may be faulty.
  - ***Backward Error Recovery.*** In this case, the system use previously saved correct state information at the starting point after failure.
  - ***Use of Recovery Blocks.*** In this case, write  $N$  unique versions of program plus a single acceptance test. If the current program version fails, switch to the next version, the system continues with the next acceptable version. The system fails if no version passes the test. This technique can tolerate  $(N - 1)$  faults.
- (c) ***Information Redundancy.*** This refers to the addition of redundant information to data in order to allow fault detection, fault masking and fault tolerance. Examples of added information include error-detecting and error-correcting codes that are usually added to logic circuits, memories, and data communicated over networks.
  - ***Error Detecting Code (EDC).*** This is used to indicate a code that has the ability to expose error(s) in any given data word. Exposure of an erroneous data word is achieved by showing the invalidity of the decoded data word.
  - ***Error Correcting Code (ERC).*** This is used to indicate a code that has the ability that if an error has been detected, then it is possible to determine what the correct data would have been.
- (d) ***Time Redundancy.*** This refers to the repetition of a given computation a number of times and a comparison of the results to determine if a discrepancy exists. The existence of a discrepancy between subsequent computations indicates the existence of

transient or intermittent faults. Hardware redundancy and information redundancy require extra hardware for calculation and storing. These leads to higher production cost. Instead of using extra hardware, time redundancy is used which means longer processing time.

3. **Diversity.** Diverse (dissimilar) implementations offer (some) protections against CCF. Even if all variants contain flaws, they are less likely to fail simultaneously. A processor-based hardware module is duplicated to provide redundancy in a large-scale digital control system, particularly in safety-critical systems. Programs within the module must also be duplicated. Duplication of the hardware module provides protection against failures as in the case of the F&G detection systems. However, a problem within identically duplicated software is likely to affect all identical modules at the same time. Therefore, software within each hardware module should be diversified in order to protect the system from software faults. This measure is applied in the antivirus software example. Diversity in software is an essential factor for software fault tolerance. Diversity here refers to using different means to perform a required function or solve the same problem. This means developing more than one algorithm to implement a solution for software. The results from each algorithm are compared, and if they agree, then appropriate action is taken.

As an example to show how the effects of systematic faults can be controlled through the use of diversity, consider a software that contain error that will manifest when the same specific piece of code is executed in the same circumstances. Consider that diverse redundant elements are designed, configured, and manufactured through different methodologies and using different components, design processes and by different teams. Diverse channels have different features, behaviour and reliability data. In this situation, diversity prevents systematic faults from affecting the redundant channels in the system (Ramirez, 2008). The problems with diversity is that it provides limited protection against systematic faults because of the following reasons:

- Specifications for all versions will be the same
- People (are trained to) think alike; given identical (correct) specifications, similar er-

rors in product are likely. This has been shown in several studies such as [Knight and Leveson \(1986\)](#) on n-version programming.

However, diversity can avoid systematic faults introduced by tools.

4. ***Simplicity of design.*** A simple design is easier to understand and maintain and reduces the number of intervention errors.
5. Use of certified tools and formal methods (for software) - proof and refinement helps to prevent systematic failure.
6. ***Competence and training.*** Designers, operators and maintainers can help to reduce systematic faults and CCFs by understanding the root causes. In all the examples given, the management ensured that design teams are competent in their various fields. Also they undergo training to sharpen their skills ([Rausand, 2014](#)).
7. ***Analysis.*** FMECA and other reliability analyses can identify causes of faults and propose measures to reduce of faults.
8. ***Procedures and human interface.*** Clear procedures and an adequate human-machine interface reduce the likelihood of human errors.

# Chapter 8

## Summary and Recommendations for Further Work

### 8.1 Summary and Conclusions

The main objective of this thesis is to study, evaluate and discuss to what extent RGT of SISs is a suitable approach for identifying and avoiding systematic faults. If the conclusion is affirmative, another objective is to establish guidelines for RGT for this purpose and identify challenges and pitfalls related to such testing. This is an extensive task and is covered in chapters 4 and 7. It was further split into sub-objectives that the thesis attempts to give suitable answers to.

When a SIS fails to perform its intended function due to a non-physical fault in specification, design, operation or maintenance/testing of the system (specification and design phase is only covered in this work), the term systematic failure is more commonly used than systematic faults to describe the event. A failure is the loss of the ability to perform as required, which is an event, while a fault is the state when an item, component, system has failed. The standards IEC 61508 and IEC 61511 do not provide a specific definition of systematic faults, as they do for systematic failures. All systematic failures are systematic faults and not the other way around. All about systematic faults and its classification, difference between a failure and a fault, and the relationship between the two terms are discussed in chapter 4. This answers sub-objectives 1,2 and 4.

A failure will only occur when a design fault is challenged by a triggering mechanism. In order for such a failure to occur; a systematic fault must exist in multiple components in the integrated system, and a triggering event must occur to challenge the systematic fault. A CCF is a failure that occurs when two or more separate channels in a multiple channel system are in a fault state simultaneously, leading to a system fault. CCFs and its relationship with systematic failure is sub-objective 3 and is discussed in chapter 3.

RGT is the same as TAAF (Test, Analyze, and Fix) testing of a product early in the development cycle when design changes can be made readily in response to observed failures. The main objective of reliability RGT is early identification of faults in the design phase, so that design reviews can be made to improve reliability. The aim of reliability growth models is to monitor the progress of the development programme and the improvements in reliability. Both discrete and continuous models have been developed to assess reliability growth. Continuous models are used in the context of continuous variables and attempts to describe the reliability improvement as a function of the total test time. Examples of continuous models are the Duane Model and the Army Material Systems Analysis Activity (AMSAA) model. Discrete models involve discrete data and are concerned with incremental improvements in reliability as a result of design changes. Examples of discrete models are the Lloyd-Lipow model and the Wolman model. The strengths of the Duane model lies in its simplicity, possibility to study as a linear model (after having taken the log) and this makes it easy to determine the parameters. The Duane model is often criticised theoretically because, when  $t=0$ ,  $MTBF_c(t)$  is also zero, which is not so in practice. Also, the model may tend to lack accuracy since it is a graphical technique. Since AMSAA model views the process of reliability growth as a probabilistic process, it has the advantage of making it easier for the parameters of a non-homogenous Poisson process to be estimated statistically, confidence intervals can be obtained and goodness of fit applied. This solves the problem of sub-objectives 6 and 7, and is discussed in chapter 6.

RGT can be used to identify systematic faults. In order to demonstrate how RGT is used to reveal faults, illustrative examples of fire and gas detection and mitigation systems, car airbag, and

mobile phones were used. A mobile phone, though, not a SIS, is used for a better understanding of the concept. RGT as used in the thesis, can be formal, semi-formal or informal. Formal testing means testing in the laboratory in controlled conditions and an environment that mimics the real world in a limited manner. Informal testing is releasing the product to a limited number of customers or to employees of the manufacturer to use and keep appropriate information relating usage mode and intensity, operating environment, failure modes and report back to the design team for necessary corrections in design. The functions tested in the crash sensors and airbags show that there is a software error in the logic solver that controls the activation of the front seat passenger airbag, this makes it impossible to determine whether the seat is occupied or not. This can be dangerous, if accident happen to occur, the airbag will fail to activate, thereby increasing the possibility of injury or death. After a crash, the inflation gas is released, this gas is not harmful, but may cause discomfort to people with respiratory problems. Also, it was found out that, severe or panic breaking alone cannot cause an airbag to activate; airbags only activate in crashes. The mobile phone was found to have design errors in the battery life and processor speed. The design errors revealed are handled and avoided as the consequences can have great impact on SIS. The two main methods used to minimize the number of faults in SIS are fault avoidance/prevention and fault detection/removal. These two methods applies measures like; design reviews, use of redundant and diverse channels, use of simple design, use of competent designers, training of designers to overcome and prevent these systematic faults. This answers the sub-objectives 8, 9 and 10 and is discussed in chapter 7.

## **8.2 Recommendations for Further Work**

Since this thesis report is on "Using RGT to reveal systematic faults in SISs" and is restricted to the design and development phases of the SIS lifecycle; therefore, research should be performed on this topic in other phases of the SIS lifecycle.

In this work, all the RGTs - both formal and informal are assumed, a step should be taken further, if possible to carry out practically such tests both in the laboratory and through customers.



# Appendix A

## Acronyms

**ALARP** As low as reasonably practicable

**ALT** Accelerated life testing

**AMSAA** Army materials systems analysis activity

**CCF** Common-cause failure

**DU** Dangerous undetected

**E/E/PE** Electrical, electronic, or programmable electronic

**ECU** Electronic control unit

**EDC** Error detecting code

**ERC** Error correcting code

**ESD** Emergency shutdown

**EUC** Equipment under control

**FMECA** Failure modes effects and criticality analysis

**FRACAS** Failure reporting and corrective action system

**FTA** Fault tree analysis

**F&G** Fire and gas

**HALT** Highly accelerated testing

**HSE** Health and safety executive

**HIPPS** High integrity pressure protection system

**IC** Integrated circuit

**I&C** Instrumentation and control

**ICDE** International common cause data exchange

**IEC** International electrotechnical commission

**LOPA** Layer of protection analysis

**MFSC** Multiple failure with a shared cause

**MTBF** Mean time between failures

**MTTF** Mean time to failure

**NOG** Norwegian oil and gas

**NEA** Nuclear energy agency

**NHPP** Non homogenous poison process

**NMR** N-modular redundancy

**NRC** Nuclear regulatory commission

**NVP** N-version programming

**PES** Programmable electronic systems

**PDF** Probability of failure on demand

**PFH** Probability of failure per hour

**QFD** Quality function deployment

**RAM** Random access memory

**RAMS** Reliability, availability, maintainability, and safety

**ROCOF** Rate of occurrence of failures

**RGT** Reliability growth testing

**SDV** Shutdown valve

**SIF** Safety-instrumented function

**SIL** Safety integrity level

**SIS** Safety-instrumented system

**SLC** Safety lifecycle

**SRS** Safety requirements specification

**SSC** Structures systems or components

**TAAF** Test-analyze-and-fix

**TMR** Triple-modular redundancy

# Bibliography

Abd-El-Barr, M. (2006). *Design and analysis of reliable and fault-tolerant computer systems*. World Scientific Pub Inc, Singapore.

Ali, R. (2007). How to implement a safety life-cycle. *Valve Magazine*, 19(3):1–6. Valve Manufacturers Association, Washington DC, USA

Amstadter, B. L. (1971). *Reliability mathematics: fundamentals, practices, procedures*. McGraw-Hill Inc., New York, US.

Bell, R. (2006). Introduction to IEC 61508. pages 3–12. Australian Computer Society, Inc. Level 11, 50 Carrington Street, Sydney NSW 2000.

Braunl, T. (2014). Fault tolerant computing systems (elec4422) lecture notes. Department of Electrical and Electronics Engineering, University of West Australia

Crow, L. H. (1975). Reliability analysis for complex, repairable systems. Technical report, DTIC Document. U.S ARMY MATERIAL SYSTEMS ANALYSIS ACTIVITY. Aberdeen Proving Ground Maryland.

Denning, R. and Abbey Wood, M. (2012). *Applied R& M manual for defence systems (GR-77 Part D - Supporting theory)*.

Dhillon, B. S. (1983). Reliability engineering in systems design and operation. Technical report, Univ. of Ottawa.

DOE-NE-STD-1004-92 (1992). DOE. Root cause analysis guidance document. Technical Report DOE-NE-STD-1004-92, U.S. Department of Energy Washington DC.

- Duane, J. (1964). Learning curve approach to reliability monitoring. *Aerospace, IEEE Transactions on*, 2(2):563–566.
- Gentile, M. and Summers, A. E. (2006). Random, systematic, and common cause failure: How do you manage them? *Process safety progress*, 25(4):331–338.
- Handbook (1998). Mil-hdbk-338b. military handbook electronic reliability design. *US Department of Defense*.
- Hauge, S., Hokstad, P., Herrera, I., and Onshus, T. (2004). The impact of common cause failures in safety systems. Technical report, SINTEF REPORT, Trondheim.
- Hauge, S., Hokstad, P., Kråkenes, T., Håbrekke, S., and Jin, H. (2013). Reliability prediction method for safety instrumented systems. PDS method handbook. 2013 edition. *SINTEF report A24442*, 60S051.
- Hauge, S., Lundteigen, M. A., Hokstad, P., and Håbrekke, S. (2010). Reliability prediction method for safety instrumented systems–pds method handbook, 2010 edition. *SINTEF report STF50 A*, 6031.
- Herrmann, D. S. (1999). *Software safety and reliability*. IEEE Computer Soc.
- Hoem, A. S. (2013). Systematic faults in safety-instrumented systems. Master's thesis, NTNU Trondheim: Department of Quality and Production Engineering.
- Hokstad, P. and Rausand, M. (2008). Common cause failure modeling: status and trends. In *Handbook of performability engineering*, pages 621–640. Springer.
- Honda (2013). *Honda Owner's manual*. Honda Motor Co. Ltd, Tokyo Japan.
- HSE (2003). Out of control (why control systems go wrong and how to prevent failure). Health and Safety Executive, UK
- IEC 61014 (2003). Programmes for reliability growth. Norm IEC 61014, International Electrotechnical Commission, Geneva.

- IEC 61508 (2010). Functional safety of electrical/electronic/programmable electronic safety-related systems. Norm IEC 61508, International Electrotechnical Commission, Geneva.
- IEC 61511 (2003). Functional safety: Safety instrumented systems. part 1: Framework, definitions, systems, hardware and software requirements. Norm IEC 61511, International Electrotechnical Commission, Geneva.
- ISA-TR 84.00.04. Part 1: Guidelines for the implementation of ANSI/ISA-84.00.01.2004. ISA-SP84 Working group 2. Norm, The Instrumentation, Systems, and Automation Society. North Carolina.
- ISA-TR84.00.02 (2002). Safety Instrumented Functions (SIF), Safety Integrity Level (SIL) Evaluation Techniques. Part 1: Introduction. Safety Instrumented Functions (SIF), Safety Integrity Level (SIL) Evaluation Techniques. Norm ISA-TR84.00.02, The Instrumentation, Systems, and Automation Society. North Carolina.
- ISO/TR 12489 (2013). Petroleum, petrochemical and natural gas industries - Reliability modelling and calculation of safety systems. Norm ISO/TR 12489, International Organisation for Standardization, Geneva.
- Jigar, A. A. (2013). Quantification of reliability performance: Analysis methods for safety instrumented system. Master's thesis, Department of Mathematical Sciences, NTNU.
- Jin, H. *A contribution to reliability assessment of safety-instrumented systems*. PhD thesis, Department of Quality and Production Engineering, NTNU, Trondheim.
- Knight, J. C. and Leveson, N. G. (1986). An experimental evaluation of the assumption of independence in multiversion programming. *Software Engineering, IEEE Transactions on*, (1):96–109.
- Krasich, M., Quigley, J., and Walls, L. (2004). Modeling reliability growth in the product design process. In *Reliability And Maintainability, 2004 Annual Symposium-RAMS*, pages 424–430. IEEE.
- Lloyd, D. K. and Lipow, M. (1963). Reliability: management, methods and mathematics.

- Lundteigen, M. A. (2009). *Safety instrumented systems in the oil and gas industry*. PhD thesis, NTNU Trondheim: Department of Production and Quality Engineering.
- Lundteigen, M. A. and Rausand, M. (2006). Assessment of hardware safety integrity requirements. In *Proceedings of 30th ESReDA European Safety, Reliability and Data Association Seminar on Reliability of Safety Critical Systems*.
- Lundteigen, M. A. and Rausand, M. (2009). Reliability assessment of safety instrumented systems in the oil and gas industry: A practical approach and a case study. *International Journal of Reliability, Quality and Safety Engineering*, 16(02):187–212.
- Meeker, W. Q. and Escobar, L. A. (1998). Pitfalls of accelerated testing. *Reliability, IEEE Transactions on*, 47(2):114–118.
- Meeker, W. Q., Sarakakis, G., and Gerokostopoulos, A. (2013). More pitfalls of accelerated tests. *Journal of Quality Technology*, 45(3).
- Miller, A., Kaufer, B., and Carlsson, L. (2000). Activities on component reliability under the OECD nuclear energy agency. *Nuclear engineering and design*, 198(3):325–334.
- Murthy, D. P., Rausand, M., and Østerås, T. (2008). *Product reliability: specification and performance*. Springer, Verlag London Limited.
- NOG 070 (2004). Guidelines for the application of IEC 61508 and IEC 61511 in the petroleum activities on the continental shelf. Technical Report NOG 070, Norwegian Oil and Gas Association.
- NP-T-1.5 (2009). Protecting against Common Cause Failures in Digital I&C Systems of Nuclear Power Plants. Technical Report NP-T-1.5, IAEA Nuclear Energy Series, Vienna, Austria.
- NUREG/CR 6268 (1998). Common-Cause Failure Database and Analysis System: Event Data Collection, Classification, and Coding (NUREG/CR-6268, INEL/EXT-07-12969, Revision 1). Technical Report NUREG/CR 6268, U.S. Nuclear Regulatory Commission.
- O'Connor, P. D. (2002). *Practical Reliability Engineering*. Wiley, Chichester, UK.

- Paula, H. M., Campbell, D. J., and Rasmuson, D. M. (1991). Qualitative cause-defense matrices: Engineering tools to support the analysis and prevention of common cause failures. *Reliability Engineering & System Safety*, 34(3):389–415.
- Priest, J. W. (1988). *Engineering design for producibility and reliability*. M. Dekker, University of Michigan.
- Quigley, J. and Walls, L. (1999). Measuring the effectiveness of reliability growth testing. *Quality and reliability engineering international*, 15(2):87–93.
- Ramirez, E. C. (2008). Diverse redundancy used in SIS technology to achieve higher safety integrity. ABB value paper
- Rausand, M. (2011). *Risk assessment: theory, methods, and applications*. Wiley, Hoboken, NJ.
- Rausand, M. (2014). *Reliability of Safety-Critical Systems, Theory and Applications*. Wiley, Hoboken, NJ, 1st edition.
- Rausand, M. and Høyland, A. (2004). *System Reliability Theory: Models, Statistical Methods, and Applications*. Wiley, Hoboken, NJ, 2nd edition.
- Son, H. S. and Kim, M. C. (2009). Software faults and reliability. In *Reliability and Risk Issues in Large Scale Safety-critical Digital Control Systems*, pages 81–103. Springer.
- Stanley, O. (2011). *A Practical Guide to Fire Alarm Systems*. Central Station Alarm Association, 8150 Leesburg Pike Vienna, VA 22182.
- Suhir, E. (2002). Accelerated life testing ALT in microelectronics and photonics: its role, attributes, challenges, pitfalls, and interaction with qualification tests. *Journal of Electronic Packaging*, 124(3):281–291.
- Walls, L. and Quigley, J. (2001). Building prior distributions to support bayesian reliability growth modelling using expert judgement. *Reliability Engineering & System Safety*, 74(2):117–128.
- Walls, L., Quigley, J., and Krasich, M. (2005). Comparison of two models for managing reliability growth during product design. *IMA Journal of Management Mathematics*, 16(1):12–22.



Yang, G. (2007). *Life cycle reliability engineering*. Wiley, 1st edition.

Yiliu, L. (2014). Rams engineering and management lecture notes. TPK 5165, Department of Quality and Production Engineering, NTNU, Trondheim